«Интегралдық және микропроцессорлық схемотехника»

Дәуренов ӘД

Дәріс №2. Буль алгебрасы: аксиомалар, тепе-теңдіктер, негізгі логикалық функциялар, базис ұғымы

Дәрістің мақсаты

- Цифрлық логиканың математикалық негіздерін,
- негізгі логикалық амалдарды,
- Буль алгебрасының заңдары мен тепе-теңдіктерін,
- логикалық функцияларды сипаттау тәсілдерін және
- функционалды толық логикалық базис ұғымын меңгеру.

Дәріс соңында студент келесі байланысты түсінуі тиіс:
Физикалық сигнал → логикалық айнымалы → логикалық функция → логикалық элементтер → цифрлық схема

Алдыңғы дәрісте келесі тақырыптар қарастырылды:

- позициялық санау жүйелері;
- екілік, сегіздік, ондық және он алтылық санау жүйелері;
- сандарды бір санау жүйесінен екіншісіне түрлендіру;
- екілік арифметика;
- оң және теріс сандарды көрсету;
- тура, кері және қосымша кодтар.

Цифрлық құрылғылар ақпаратты екілік түрде өңдейтінін анықтадық: 0, 1

Алайда цифрлық құрылғыларды құру үшін сандарды екілік түрде көрсетуді білу жеткіліксіз.

Екілік сигналдармен **қандай логикалық амалдар орындалатынын** анықтау қажет.

Ол үшін **Буль алгебрасы** қолданылады.

Логикалық функция

Қарапайым тілмен айтсақ:

Логикалық функция — белгілі бір шарттарға қарап, қандай нәтиже шығатынын көрсететін ереже.

Мысалы, бөлмедегі шамды алайық: Егер **қараңғы болса** ЖӘНЕ **бөлмеде адам болса** → шам жанады.

Мұнда екі кіріс шарты бар:

- AA — қараңғы ма? **Иә = 1, Жоқ = 0**
- BB — адам бар ма? **Иә = 1, Жоқ = 0**
- YY — шам жана ма? **Иә = 1, Жоқ = 0**

Оны былай көрсетуге болады: $Y = A \wedge B$

Яғни **логикалық функция кірістегі шарттарды тексеріп, шығыста 0 немесе 1 нәтижесін береді.**

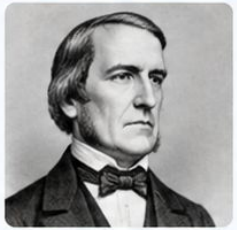
бір сөйлеммен: **Логикалық функция — «Егер осындай жағдай болса, нәтиже қандай болады?» деген сұраққа жауап беретін ереже.** Осы анықтама Boolean алгебрасын алғаш түсіндіру үшін өте қолайлы.

3. Буль алгебрасы дегеніміз не?

Анықтама

Буль алгебрасы — логикалық функцияларды сипаттауға, талдауға және түрлендіруге арналған математикалық аппарат.

Атының шығуы



Ол ағылшын математигі **Джордж Бульдің** атымен аталған.

(George Boole, 1815–1864)

Айнымалының мүмкін мәндері

Кәдімгі алгебрада айнымалы әртүрлі сандық мәндерді қабылдай алады:

$$x = 2, \quad x = -5, \quad x = 3.14$$

Буль алгебрасындағы мәндер

Ал Буль алгебрасында айнымалы тек екі мәннің бірін қабылдайды:

$$X \in \{0, 1\}$$

Мәндердің мағынасы

0 $= FALSE =$ **ЖАЛҒАН**

1 $= TRUE =$ **АҚИҚАТ**

0101

Цифрлық электроникада бұл мәндер сигналдың екі логикалық күйіне сәйкес келеді.

4. Логикалық 0 және логикалық 1

Негізгі идея

Физикалық тұрғыдан микросхема тікелей 0 және 1 сандарымен емес, **электрлік кернеу деңгейлерімен** жұмыс істейді.

Мысал

Төмен кернеу деңгейі

0 V

↓

Логикалық 0
(0)

Жоғары кернеу деңгейі

5 V

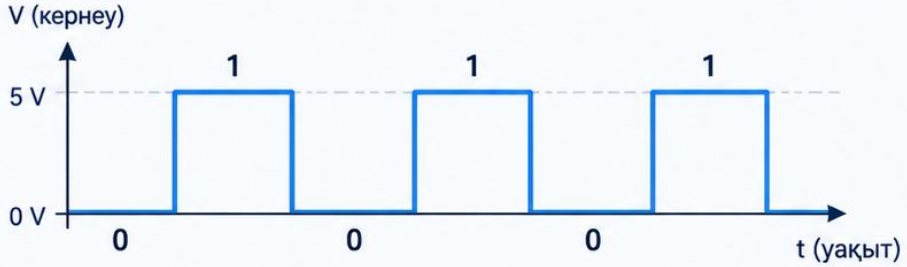
↓

Логикалық 1
(1)

Кернеу → Логикалық деңгей

Кернеу деңгейі	Логикалық деңгей
Төмен (≈ 0 V)	**0**
Жоғары (≈ 5 V)	**1**

Цифрлық сигналдың мысалы



Жалпы түрде

Кернеу → Логикалық деңгей → 0/1

Қолданылуы

Кернеудің нақты диапазондары қолданылатын цифрлық микросхема технологиясына байланысты.



Кейінірек **ТТЛ және КМОП** элементтік базасын қарастырған кезде бұл мәселені толығырақ зерттейміз.

Қорытынды

Микросхемалар **логикалық 0 және 1-ді электрлік кернеу деңгейлері** арқылы ажыратады.

5..Логикалық айнымалылар

Логикалық айнымалы дегеніміз не?

Логикалық айнымалылар әдетте әріптерімен белгіленеді:

$$A,\quad B,\quad C,\quad X_1,\quad X_2\ \ldots$$

Шығыс айнымалы

Схеманың шығуы әдетте келесі әріптермен белгіленеді:

$$Y,\ F,\ Q\ldots$$

түрінде белгіленуі мүмкін.

Логикалық функция



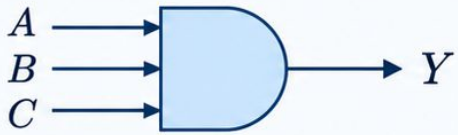
Мысалы:

$$Y = f(A, B)$$

Бұл Y шығысының күйі A және B кірістерінің күйіне тәуелді екенін білдіреді.

Үш кіріс болған жағдайда

$$Y = f(A, B, C)$$



Кіріс комбинацияларының саны

n кіріс болған жағдаіда мүмкін комбинациялардың саны:

$$N = 2^n$$

Мысалы, $n = 3$:

$$2^3 = 8$$

3 кіріс үшін (8 комбинация)

A	B	C
0	0	0
0	0	1
0	1	0
...	...	...
1	1	1

Негізгі қорытынды

- Логикалық айнымалылар әріптерімен белгіленеді $(A, B, C, X_1, X_2\ \ldots)$.
- Схеманың шығысы әдетте $Y, F, Q\ \ldots$ түрінде белгіленеді.
- Шығыс кірістердің күйіне тәуелді: $Y = f(A, B, \ldots)$.
- n кіріс үшін комбинациялар саны: $N = 2^n$.

6. Логикалық функция

Анықтама

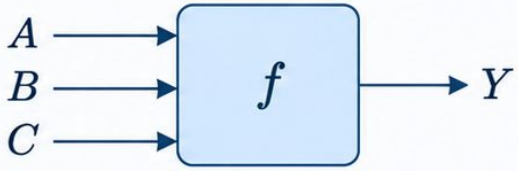
Логикалық функция — бір немесе бірнеше шығыс логикалық айнымалыларының кіріс логикалық айнымалыларына тәуелділігі.

Жалпы түрде

$$Y = f(A, B, C, \ldots)$$

Y — шығыс айнымалы.
$A, B, C, \ldots$ — кіріс айнымалылары.

Логикалық функцияның көрінісі



Мысал 1. ЖӘНЕ (AND)

$$Y = A \cdot B$$

Y шығысы A және B кірістеріне тәуелді.

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

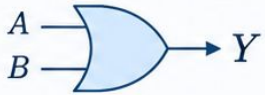


ЖӘНЕ
(AND)

Мысал 2. НЕМЕСЕ (OR)

$$Y = A + B$$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

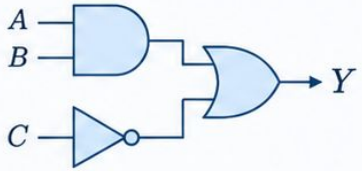


НЕМЕСЕ
(OR)

Мысал 3. Күрделірек функция

$$Y = A \cdot B + \overline{C}$$

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1



Кез келген комбинациялық цифрлық схеманың жұмысын логикалық функция арқылы көрсетуге болады.

7. Логикалық функцияны көрсету тәсілдері

Логикалық функцияны бірнеше тәсілмен көрсетуге болады. Барлық тәсілдер бір логикалық функцияны сипаттайды.

1 Сөзбен сипаттау

A және B кірістері бір уақытта қосылған жағдайда шығыс қосылады.

Түсіндірме:

- $A = 1$ және $B = 1$ болғанда ғана $Y = 1$.
- Басқа барлық жағдайларда $Y = 0$.

2 Логикалық өрнек

$$Y = A \cdot B$$

Мұнда:

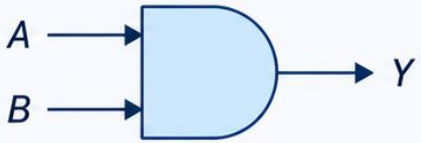
- Y — шығыс айнымалысы
- A, B — кіріс айнымалылары
- $\cdot$ — логикалық ЖӘНЕ (AND) операциясы

3 Ақиқаттық кестесі

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

$Y = 1$ тек қана $A = 1$ және $B = 1$ болғанда.

4 Логикалық схема



ЖӘНЕ (AND) қақпасы

$$A, B \rightarrow AND \rightarrow Y$$

Кірістер A және B логикалық ЖӘНЕ қақпасына беріледі, нәтижесінде шығыс Y алынады.

Қорытынды: Сөздік сипаттама, логикалық өрнек, ақиқаттық кесте және логикалық схема — логикалық функцияны көрсетудің эквивалентті тәсілдері.

8. Негізгі логикалық амалдар

Буль алгебрасының негізгі үш амалы: **NOT, AND, OR**. Осы амалдардың көмегімен күрделі логикалық функцияларды құруға болады.

Амал	Атауы	Символы	Жазылуы	Ақиқаттық кестесі	Түсіндірме
NOT	**ЕМЕС** (терістеу)		$Y = \overline{A}$	A | Y 0 | 1 1 | 0	Кірістегі логикалық мәнді терістейді (инверсиялайды).
AND	**ЖӘНЕ** (конъюнкция)		$Y = A \cdot B$	A | B | Y 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	Шығыс 1-ге тең тек барлық кірістер 1 болғанда.
OR	**НЕМЕСЕ** (дизъюнкция)		$Y = A + B$	A | B | Y 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1	Шығыс 1-ге тең кем дегенде бір кіріс 1 болғанда.

Қорытынды: NOT, AND және OR — логикалық схемалар мен күрделі функцияларды құрудың негізгі құрылыс блоктары.

9. NOT — ЕМЕС логикалық амалы

NOT амалы **логикалық терістеу** немесе **инверсия** деп аталады.

1. Анықтама

NOT (ЕМЕС) — кірістегі логикалық мәнді қарама-қарсы мәнге өзгертетін логикалық амал.

2. Жазылуы

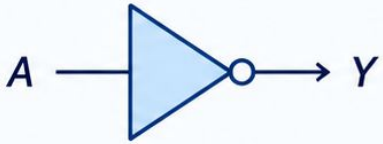
Белгіленуі:

$$Y = \overline{A}$$

немесе:

$$Y = \neg A$$

3. Логикалық схема



NOT (инвертор)

Шығыстағы шеңбер («көпіршік») — терістеуді білдіреді.

4. Ақиқаттық кестесі

A	$Y = \overline{A}$
0	1
1	0

5. Жұмыс принципі

$$A = 0 \longrightarrow Y = 1$$

(жалған → ақиқат)

$$A = 1 \longrightarrow Y = 0$$

(ақиқат → жалған)

6. Яғни

$$\overline{0} = 1$$

$$\overline{1} = 0$$

NOT амалын орындайтын физикалық логикалық элемент инвертор деп аталады.

Қорытынды: NOT кірістегі логикалық күйді қарама-қарсы күйге өзгертеді.

11. AND — ЖӘНЕ логикалық амалы

AND амалы **логикалық көбейту** немесе **конъюнкция** деп аталады.

1. Анықтама

AND (ЖӘНЕ) —
логикалық көбейту немесе конъюнкция.
Шығыс тек **барлық** кірістер 1 болған жағдайда ғана 1-ге тең болады.

2. Жазылуы

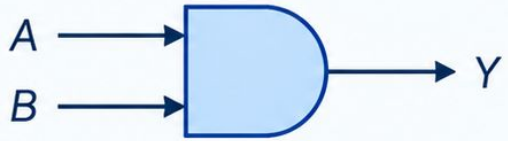
Нүктемен:

$$Y = A \cdot B$$

Кейде көбейту белгісі жазылмайды:

$$Y = AB$$

3. Логикалық схема



ЖӘНЕ (AND) қақпасы

4. Ақиқаттық кестесі

A	B	$Y = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	**1**

5. Жұмыс принципі

- Шығыс $Y = 1$ тек барлық кірістер 1 болғанда.
- Егер кем дегенде бір кіріс 0 болса, шығыс 0 болады.
- Халықаралық атауы: AND (шарт: “және”).
- Логикалық көбейту ретінде қарастырылады:
  $1 \cdot 1 = 1$, ал $1 \cdot 0 = 0, 0 \cdot 1 = 0, 0 \cdot 0 = 0$.

Қорытынды: AND амалы бірнеше шарттың бәрі орындалған кезде ғана ақиқат нәтиже береді.

12. AND амалының қолданылу мысалы

Электрозғалтқыш тек екі шарт орындалғанда іске қосылсын: қорғаныс есігі жабық және START батырмасы басылған.

1 Практикалық жағдай



2 Логикалық өрнек

$$Motor = A \cdot B$$

$A = 1$ – қорғаныс есігі жабық

$B = 1$ – START батырмасы басылған

$Motor = 1$ – электрозғалтқыш іске қосылады

3 Ақиқаттық кестесі

A (есік)	B (START)	Motor (шығыс)
0	0	0
0	1	0
1	0	0
1	**1**	**1**

4 Жұмыс принципі

Егер $A = 0$ немесе $B = 0$ болса, $Motor = 0$ (электрозғалтқыш қосылмайды).

Тек $A = 1$ және $B = 1$ болған кезде, $Motor = 1$ (электрозғалтқыш іске қосылады).

5 Қорытынды

AND бір уақытта орындалатын екі немесе бірнеше шарттың бірлесіп жұмыс істеуін тексеруге мүмкіндік береді.

Мысалдың мағынасы: Электрозғалтқыш тек қорғаныс есігі жабық және START батырмасы басылған кезде ғана іске қосылады.

13. OR — НЕМЕСЕ логикалық амалы

OR амалы **логикалық қосу** немесе **дизъюнкция** деп аталады.

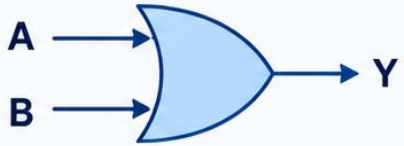
1 Анықтама

OR (НЕМЕСЕ) — кірістердің кем дегенде біреуі 1 болса, шығыс 1 болатын логикалық амал.

$$Y = A + B$$

Қосу белгісі: +

2 Логикалық схема



НЕМЕСЕ (OR) қақпасы

Кірістердің кем дегенде біреуі 1 болғанда шығыс 1.

3 Ақиқаттық кестесі

A	B	$Y = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

4 Логикалық өрнек

$$Y = A + B$$

Оқу түрлері:

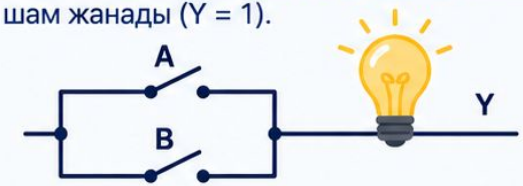
- A НЕМЕСЕ B
- A немесе B

Шығыс кемінде бір кіріс 1 болған жағдайда 1-ге тең болады.

5 Практикалық мысал

Бөлмедегі жарықты екі жерден басқару

Екі қосқыштың (A немесе B) кез келгені қосылса, шам жанады (Y = 1).



Кез келген қосқыш жабық болса, шам жанады.

6 Қорытынды

OR амалы — кірістердің кем дегенде біреуі 1 болғанда 1-ге тең болатын логикалық қосу (дизъюнкция) амалы.

Ол «НЕМЕСЕ» шартының логикалық орындалуын сипаттайды.

14. OR амалының қолданылу мысалы

Апаттық сигнализация екі жағдайда іске қосылсын: температура немесе қысым шектен жоғары болса.

1 Практикалық жағдай



2 Логикалық өрнек

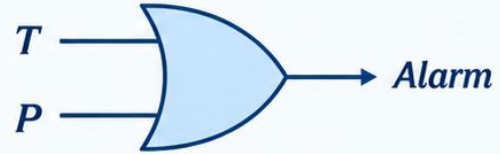
$$Alarm = T + P$$

$T = 1$ – температура шектен жоғары

$P = 1$ – қысым шектен жоғары

$+$ – логикалық қосу (НЕМЕСЕ, OR)

3 Логикалық схема



НЕМЕСЕ (OR) қақпасы

4 Ақиқаттық кестесі

T (температура)	P (қысым)	Alarm (шығыс)
0	0	0
0	1	**1**
1	0	**1**
1	1	**1**

5 Жұмыс принципі

- Температура немесе қысым (немесе екеуі де) шектен жоғары болса, сигнализация іске қосылады.
- Тек екеуі де қалыпты болғанда сигнализация сөнулі (0) болады.
- Бұл OR амалының негізгі қасиеті: кірістердің кем дегенде бірі 1 болса, шығыс 1.

6 Қорытынды

Қорытынды: OR амалы екі немесе бірнеше апаттық жағдайдың кез келгені орындалғанда дабыл беру сияқты қолданбаларда кеңінен қолданылады.

15. NAND — ЖӘНЕ-ЕМЕС

NAND екі амалдың комбинациясы: AND + NOT (ЖӘНЕ + ЕМЕС).

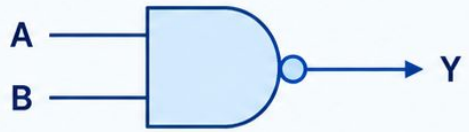
1 Анықтама

NAND (ЖӘНЕ-ЕМЕС) — AND амалынан кейін инверсия жасайтын логикалық амал.

$$Y = \overline{A \cdot B}$$

Яғни, шығыс тек барлық кірістер 1 болғанда ғана 0-ге тең, қалған жағдайларда 1 болады.

2 Логикалық схема



NAND қақпасы
(ЖӘНЕ + ЕМЕС)

Бұл — AND қақпасының шығысында инвертор (шағын шеңбер) бар.

3 Ақиқаттық кестесі

A	B	$A \cdot B$	$Y = \overline{A \cdot B}$
0	0	0	**1**
0	1	0	**1**
1	0	0	**1**
1	1	1	**0**

4 Жұмыс принципі

- Егер $A = 1$ және $B = 1$ болса, алдымен $A \cdot B = 1$, содан кейін инверсия → $Y = 0$.
- Қалған барлық жағдайларда $A \cdot B = 0$, инверсия → $Y = 1$.
- NAND қақпасы — **универсал қақпа**, яғни тек NAND қақпасының көмегімен кез келген логикалық функцияны құруға болады.

5 Басқа белгіленуі



NAND (IEC белгісі)

6 Қорытынды

NAND — AND амалын инверсиялау.
Шығыс тек барлық кірістер 1 болғанда ғана 0-ге тең, ал қалған жағдайларда 1 болады.

Маңызды: NAND — ең әмбебап логикалық қақпалардың бірі, цифрлық схемаларда кеңінен қолданылады.

16. NOR — НЕМЕСЕ-ЕМЕС

NOR — OR амалынан кейін инверсия жасайтын логикалық амал (НЕМЕСЕ-ЕМЕС).

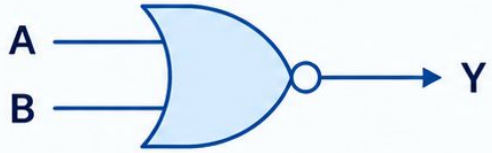
1 Анықтама

NOR (НЕМЕСЕ-ЕМЕС) — кірістердің кем дегенде біреуі 1 болса, шығыс 0 болатын логикалық амал.

$$Y = \overline{A + B}$$

Яғни, OR амалынан кейін инверсия жасалады.

2 Логикалық схема



НЕМЕСЕ-ЕМЕС (NOR) қақпасы

Бұл — OR қақпасының шығысында инвертор (шағын шеңбер) бар.

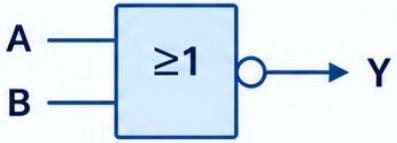
3 Ақиқаттық кестесі

A	B	$A + B$	$Y = \overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

4 Жұмыс принципі

- Егер $A = 0$ және $B = 0$ болса, алдымен $A + B = 0$, сонда $Y = 1$.
- Егер $A = 1$ немесе $B = 1$ болса, $A + B = 1$, сонда $Y = 0$.
- Шығыс тек екі кіріс те 0 болған жағдайда ғана 1-ге тең.

5 Басқа белгіленуі



NOR (IEC белгісі)

6 Қорытынды

NOR — кірістердің кем дегенде біреуі 1 болғанда 0 шығатын логикалық амал.

Шығыс тек $A = 0$ және $B = 0$ болғанда ғана 1-ге тең.

Маңызды: NOR — ең әмбебап логикалық қақпалардың бірі, күрделі логикалық схемаларды құруда кеңінен қолданылады.

17. XOR — айрықша НЕМЕСЕ

XOR (айрықша НЕМЕСЕ) — кірістердің мәндері **бір-бірінен өзгеше** болғанда 1 шығатын логикалық амал.

1 Анықтама

XOR (айрықша НЕМЕСЕ) — екі кірісінің мәндері бір-бірінен өзгеше болғанда шығыс 1 болатын логикалық амал.

$$Y = A \oplus B$$

$\oplus$ — айрықша НЕМЕСЕ белгісі.

2 Логикалық схема



XOR қақпасы
(айрықша НЕМЕСЕ)

XOR қақпасы OR қақпасына ұқсайды, бірақ кіріс жағында қосымша иілген сызық бар.

3 Ақиқаттық кестесі

A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

4 Жұмыс принципі

- Егер кіріс мәндері бір-бірінен өзгеше болса, шығыс 1-ге тең.
- Егер кіріс мәндері бірдей болса (екеуі де 0 немесе екеуі де 1), шығыс 0-ге тең.
- XOR — «тең емес» жағдайын анықтайды.

$$Y = 1 \Leftrightarrow A \neq B$$

(А мен В әртүрлі)

5 Практикалық мысал

Екі қосқышпен бір шамды басқару
(саты жарығы)



Екі қосқыштың жағдайы әртүрлі болғанда (біреуі қосулы, екіншісі өшірулі) шам жанады. Екеуі бірдей жағдайда болса — шам өшеді.

6 Қорытынды

- XOR — кірістердің мәндері әртүрлі екенін тексеретін логикалық амал.
- Ол көбіне салыстыру, қателікті анықтау және басқару жүйелерінде қолданылады.
- XOR қақпасы математикалық жағынан $A \neq B$ шартын жүзеге асырады.

18. XOR не үшін маңызды?

XOR — кірістер әртүрлі болғанда 1 шығаратын ерекше логикалық амал.

1 Негізгі идея

XOR (айрықша НЕМЕСЕ) — екі кірістің мәндері бір-бірінен өзгеше болғанда шығыс 1 болады.

$$Y = A \oplus B$$

$\oplus$ — айрықша НЕМЕСЕ белгісі.

2 Ақиқаттық кестесі

A	B	$Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

3 Двоичлік қосу мысалы

Бір разрядты қосқышта (жартылай сумматор) XOR жиынтық (қосынды) битін анықтайды.

$$0 + 0 = 0 \quad (S = 0)$$
$$0 + 1 = 1 \quad (S = 1)$$
$$1 + 0 = 1 \quad (S = 1)$$
$$1 + 1 = 10_2 \quad (S = 0,\ C = 1)$$

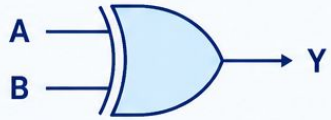
Жиынтық (қосынды) биты:

$$S = A \oplus B$$

Тасымал (көшіру) биты:

$$C = A \cdot B$$

4 Логикалық қақпа



XOR қақпасы
(айрықша НЕМЕСЕ)

Қасиеті:

- Кірістер әртүрлі болса — $Y = 1$
- Кірістер бірдей болса — $Y = 0$
- Коммутативті: $A \oplus B = B \oplus A$
- Ассоциативті: $(A \oplus B) \oplus C = A \oplus (B \oplus C)$

5 Салыстыру схемалары

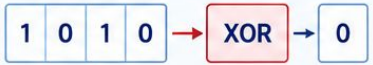
Екі санның разрядтары әртүрлі ме жоқ па — соны анықтау үшін XOR қолданылады.



$$A \oplus B = 1 \ \Leftrightarrow \ A \neq B$$
(кірістер әртүрлі)

6 Жұптықты бақылау

Деректерді беру кезінде қателікті анықтау үшін паритет (жұптық) есептеуде XOR қолданылады.



Барлық битті XOR арқылы қосқанда нәтиже 0 болса — қате жоқ, 1 болса — қате бар.

7 Цифрлық ақпаратты өңдеу

XOR әртүрлі операцияларда кеңінен қолданылады:

- шифрлау (мәліметтерді кодтау);
- мәліметтерді салыстыру;
- ақпаратты маскілеу/инверсиялау;
- циклік бақылау жиынтығы (CRC);
- қателікті анықтау және түзету кодтарында.

8 Қорытынды

XOR — цифрлық жүйелердің ең маңызды қақпаларының бірі, өйткені ол:

- қосу операциясында;
- салыстыруда;
- қателікті анықтауда;
- ақпаратты қорғауда

кеңінен қолданылады.

19. XNOR — эквиваленттілік

XNOR — XOR амалының инверсиясы. XNOR екі логикалық мәннің теңдігін тексереді.

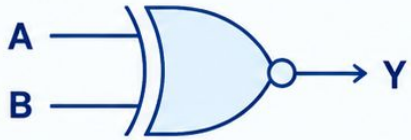
1 Анықтама

XNOR (эквиваленттілік) — кірістер бірдей болғанда 1 шығаратын логикалық амал.

$$Y = \overline{A \oplus B}$$

Яғни, XNOR — XOR амалының инверсиясы.

2 Логикалық схема



XNOR қақпасы
(XOR + NOT)

Бұл XOR қақпасының шығысында инвертор (шағын шеңбер) бар.

3 Ақиқаттық кестесі

A	B	$A \oplus B$	$Y = \overline{A \oplus B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

4 Эквиваленттілік шарты

XNOR шығысында 1 пайда болады, егер:

$$A = B$$

Яғни, кірістердің мәндері тең болғанда XNOR = 1.

$$A \neq B \Rightarrow Y = 0$$
$$A = B \Rightarrow Y = 1$$

5 XOR және XNOR салыстыру

XOR (айырмашылық)

$$Y = A \oplus B$$

$$A \neq B \Rightarrow Y = 1$$
$$A = B \Rightarrow Y = 0$$

XNOR (эквиваленттілік)

$$Y = \overline{A \oplus B}$$

$$A = B \Rightarrow Y = 1$$
$$A \neq B \Rightarrow Y = 0$$

6 Практикалық мысалдар

Құпия сөзді тексеру

Енгізілген код эталонмен дәл сәйкес келгенде рұқсат беріледі (XNOR = 1).

Деректердің тұтастығын бақылау

Жіберілген және қабылданған биттер бірдей болса — қате жоқ (XNOR = 1).

Басқару жүйелері

Екі сенсордың жағдайы бірдей болса, жүйе қалыпты күйде (XNOR = 1).

7 Қорытынды

XNOR — екі логикалық мәннің **теңдігін** тексеретін маңызды қақпа. Ол деректерді салыстыру, қателікті анықтау және қауіпсіздік жүйелерінде кеңінен қолданылады.

20. Логикалық функциялардың жалпы кестесі

Екі кірісті негізгі логикалық функциялардың ақиқаттық кестесі.

A	B	AND $Y = A \cdot B$	OR $Y = A + B$	NAND $Y = \overline{A \cdot B}$	NOR $Y = \overline{A + B}$	XOR $Y = A \oplus B$	XNOR $Y = \overline{A \oplus B}$
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	1

1 AND (ЖӘНЕ)



$Y = A \cdot B$

Екі кіріс те 1 болғанда ғана 1 шығады.

2 OR (НЕМЕСЕ)



$Y = A + B$

Кірістердің кем дегенде бірі 1 болса, 1 шығады.

3 NAND (ЖӘНЕ-ЕМЕС)



$Y = \overline{A \cdot B}$

AND нәтижесінің инверсиясы. Тек $A = B = 1$ болғанда 0.

4 NOR (НЕМЕСЕ-ЕМЕС)



$Y = \overline{A + B}$

OR нәтижесінің инверсиясы. Тек $A = B = 0$ болғанда 1.

5 XOR (айырғыша НЕМЕСЕ)



$Y = A \oplus B$

Кірістер әртүрлі болғанда 1 шығады. $(A \neq B)$

6 XNOR (эквиваленттілік)



$Y = \overline{A \oplus B}$

Кірістер бірдей болғанда 1 шығады. $(A = B)$

Қорытынды: Бұл кесте барлық негізгі екі кірісті логикалық функциялардың жұмысын бір қарағанда салыстыруға мүмкіндік береді.

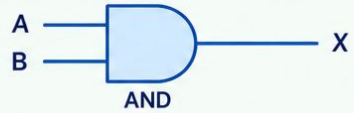
21. Құрама логикалық функциялар

Цифрлық схема бірнеше логикалық элементтен тұрады.

Мысал: $Y = A \cdot B + C$

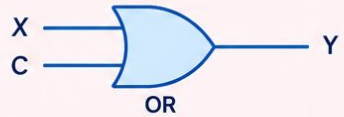
Бұл функция екі амалдан (AND, кейін OR) тұрады.

1 **Бірінші амал:** $X = A \cdot B$



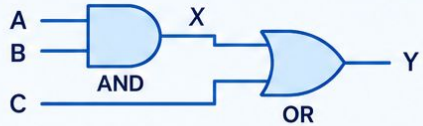
$X = A \cdot B$

2 **Екінші амал:** $Y = X + C$



$Y = X + C$

3 **Толық схема:** $Y = A \cdot B + C$



$Y = A \cdot B + C$

Маңызы: Күрделі логикалық функцияларды құрастыру үшін негізгі логикалық элементтер (AND, OR, NOT және т.б.) кезектей қолданылады.

22. Логикалық амалдардың орындалу реті

Логикалық өрнектерді есептеген кезде амалдардың орындалу ретін сақтау қажет.

Басымдық реті (жоғарыдан төмен):

1. Жақша (...)
2. **NOT** (инверсия, $\overline{A}$)
3. **AND** (және, $A \cdot B$)
4. **OR** (немесе, $A + B$)

Мысал: $Y = \overline{A} \cdot B + C$

1 **Алдымен: NOT**

А айнымалысының инверсиясын табамыз.

$\overline{A}$

2 **Содан кейін: AND**

Инверсияланған А мен В мәндерін және амалын орындаймыз.

$\overline{A} \cdot B$

3 **Соңында: OR**

Алынған нәтижені С-мен немесе амалында қосамыз.

$\overline{A} \cdot B + C$

Қорытынды: Логикалық өрнектерді дұрыс есептеу үшін амалдардың орындалу ретін міндетті түрде сақтау керек.

Буль алгебрасының негізгі заңдары

Буль алгебрасында логикалық өрнектерді түрлендіруге және ықшамдауға мүмкіндік беретін заңдар мен тепе-теңдіктер бар.

Олар не үшін қажет?

- **Қарапайым логикалық функция**
- **Логикалық элементтер саны аз**
- **Схема қарапайым**
- **Аппараттық жүзеге асыру жеңіл**

Енді негізгі заңдарды қарастырайық.

1 Коммутативтік заң

Айнымалылардың орнын ауыстыруға болады.

$$A + B = B + A$$
$$A \cdot B = B \cdot A$$

2 Ассоциативтік заң

Жақшаның орнын өзгертуге болады.

$$(A + B) + C = A + (B + C)$$
$$(A \cdot B) \cdot C = A \cdot (B \cdot C)$$

3 Дистрибутивтік заң

AND пен OR бір-біріне қатысты таратылады.

$$A \cdot (B + C) = A \cdot B + A \cdot C$$
$$A + (B \cdot C) = (A + B) \cdot (A + C)$$

4 Идемпотенттік заң

Бір айнымалыны қайта қосу нәтижені өзгертпейді.

$$A + A = A$$
$$A \cdot A = A$$

5 Нөл және бір заңы

0 және 1 элементтердің қасиеттері.

$$A + 0 = A \qquad A \cdot 1 = A$$
$$A + 1 = 1 \qquad A \cdot 0 = 0$$

6 Қарама-қарсылық (инверсия) заңы

Айнымалы мен оның инверсиясы.

$$A + \overline{A} = 1$$
$$A \cdot \overline{A} = 0$$

7 Қос теріcтеу заңы

Екі рет инверсиялау бастапқы мәнді қайтарады.

$$\overline{(\overline{A})} = A$$

32. Жұтылу заңы

Маңызды тепе-теңдіктердің бірі. Бұл заң логикалық өрнектерді қарапайымдауға көмектеседі.

Жұтылу заңы:

$$A + A \cdot B = A$$
$$A(A + B) = A$$

1 Бастапқы өрнек

Берілген өрнектен бастаймыз:

$$Y = A + A \cdot B$$

2 А-ны жақша сыртына шығарамыз

Дистрибутивтік заңды кері қолданамыз:

$$Y = A(1 + B)$$

Себебі:
$A \cdot 1 + A \cdot B = A(1 + B)$

3 Жақша ішін есептейміз

1 және B үшін:

$$1 + B = 1$$

Себебі OR үшін:
$1 + B = 1$
(бірлік үстем элемент)

4 Өрнекті түрлендіреміз

Алдыңғы қадамның нәтижесін қоямыз:

$$Y = A \cdot 1$$

Себебі:
$Y = A(1 + B) = A \cdot 1$

5 Соңғы түрлендіру

1 үшін AND қасиетін қолданамыз:

$$Y = A$$

Себебі AND үшін:
$A \cdot 1 = A$
(бірлік бейтарап элемент)

6 Нәтиже

Демек, бастапқы өрнек қарапайымдалады:

$$A + A \cdot B = A$$

Жұтылу заңының екінші түрі

Сол сияқты, келесі тепе-теңдік те орынды:

$$A(A + B) = A$$

Мағынасы

$A + A \cdot B$ өрнегінде $A \cdot B$ бөлігі A-мен «жұтылып» кетеді, өйткені $A = 1$ болғанда өрнек кез келген B үшін 1-ге тең болады. Сондықтан $A \cdot B$ бөлігі артық, оны алып тастауға болады.

Практикалық маңызы

Бұл заң логикалық схемаларды, функцияларды және теңдеулерді қарапайымдауда кеңінен қолданылады, элементтер санын азайтуға мүмкіндік береді.

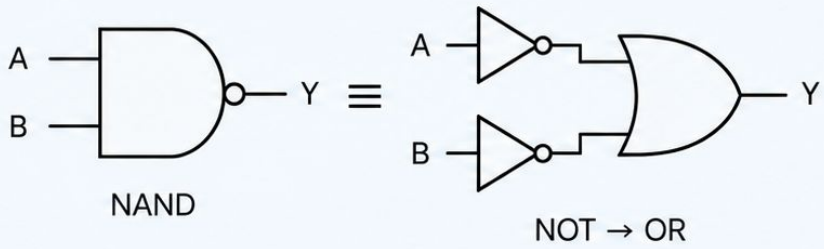
Қорытынды: Жұтылу заңы логикалық өрнектерді ықшамдауға және оларды қарапайым түрге келтіруге мүмкіндік береді.

Де Морган заңдары

Инверсияланған логикалық өрнекті эквивалентті түрде түрлендіруге мүмкіндік береді.

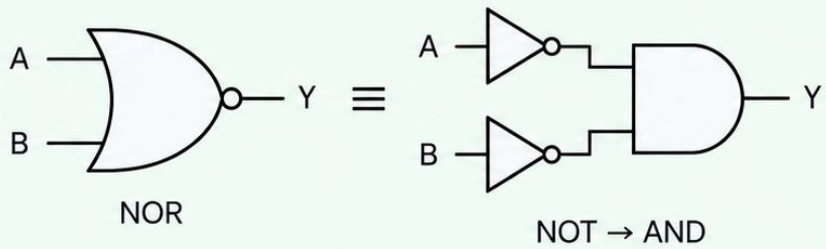
Бірінші заң
(AND → OR)

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$



Екінші заң
(OR → AND)

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$



Қорытынды: Де Морган заңдары логикалық схемаларды түрлендіруге, инверсияны таратуға және эквивалентті құрылымдарды алуға мүмкіндік береді.

Логикалық функцияларды не үшін ықшамдаймыз?

Жұтылу заңы арқылы артық элементтерді алып тастауға болады.

1 Бастапқы функция

$$Y = A + A \cdot B$$

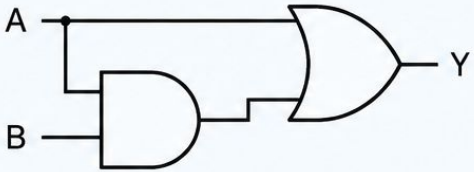
Осы функцияны тікелей жүзеге асыру үшін:

- бір AND элементі
- бір OR элементі

қажет болады.

2 Логикалық схема

Функцияның бастапқы схемасы:



3 Жұтылу заңы

Жұтылу заңы бойынша:

$$A + A \cdot B = A$$

Демек, A · B бөлігі артық, оны алып тастауға болады.

4 Ықшамдалған функция

$$Y = A$$

Бастапқы функциядағы екі логикалық элементті алып тастадық.

5 Ықшамдалған схема

Қорытынды схема:



Қорытынды

Логикалық функцияларды ықшамдау:

- логикалық элементтер санын азайтады;
- схеманы қарапайым етеді;
- аппараттық жүзеге асыруды жеңілдетеді.

Ықшамдау мысалдары

Логикалық заңдарды қолдана отырып, өрнектерді ықшамдаймыз.

1 Мысал №1

Берілген:

$$Y = A \cdot 1 + A \cdot 0$$

Заңдарды қолданамыз:

$$A \cdot 1 = A$$
$$A \cdot 0 = 0$$

Сонда:

$$Y = A + 0$$

Ал:

$$A + 0 = A$$

Демек:

$$Y = A$$

2 Мысал №2

Берілген:

$$Y = A + A \cdot B$$

Жұтылу заңы бойынша:

$$A + AB = A$$

Сондықтан:

$$Y = A$$

3 Мысал №3

Берілген:

$$Y = A \cdot B + A \cdot \overline{B}$$

Ортақ A-ны жақша сыртына шығарамыз:

$$Y = A(B + \overline{B})$$

Толықтыру заңы бойынша:

$$B + \overline{B} = 1$$

Сонда:

$$Y = A \cdot 1$$

Демек:

$$Y = A$$

Қорытынды: Логикалық заңдар арқылы күрделі функцияны бір ғана айнымалыға дейін ықшамдауға болады.

Логикалық базис дегеніміз не?

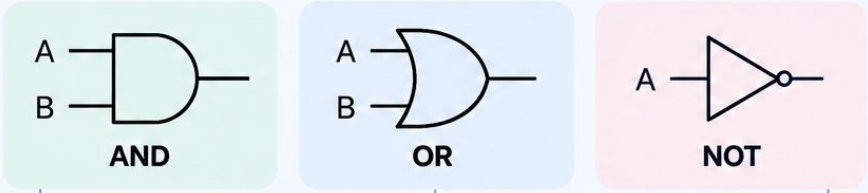
Кез келген цифрлық схеманы құру үшін қандай логикалық элементтер қажет?

Логикалық базис — кез келген логикалық функцияны жүзеге асыруға мүмкіндік беретін логикалық амалдар жиынтығы.

Мұндай базис **функционалды толық базис** деп аталады.

AND–OR–NOT базисі

Классикалық функционалды толық базис:



ЖӘНЕ, НЕМЕСЕ, ЕМЕС

Осы үш операцияны пайдаланып, кез келген Буль функциясын жүзеге асыруға болады.

Мысалы:

$$Y = \overline{A} \cdot B + A \cdot C$$

Бұл функцияны жүзеге асыру үшін:

- **NOT** элементі (A-ны терістеу үшін);
- **AND** элементтері ($A \cdot B$ және $A \cdot C$ үшін);
- **OR** элементі (нәтижелерді қосу үшін).

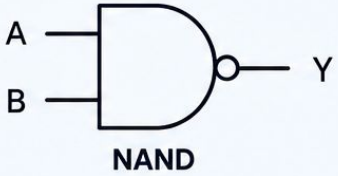
яғни AND–OR–NOT элементтері қажет.

Қорытынды: AND, OR, NOT — функционалды толық базис, себебі олар арқылы кез келген логикалық функцияны жүзеге асыруға болады.

Әмбебап логикалық элементтер

Кез келген логикалық функцияны құру үшін міндетті түрде үш түрлі логикалық элемент қолдану қажет емес.
NAND немесе NOR элементтерінің бірін ғана пайдаланып, кез келген логикалық функцияны жүзеге асыруға болады.

NAND элементі



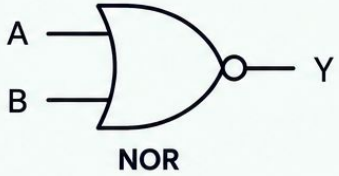
$$Y = \overline{A \cdot B}$$

Тек NAND элементтерін пайдаланып:

- NOT (инвертор)
- AND (ЖӘНЕ)
- OR (НЕМЕСЕ)

амалдарын жүзеге асыруға болады.

NOR элементі



$$Y = \overline{A + B}$$

Тек NOR элементтерін пайдаланып та:

- NOT (инвертор)
- OR (НЕМЕСЕ)
- AND (ЖӘНЕ)

амалдарын құруға болады.

Қорытынды:

$NAND$ — функционалды толық базис
NOR — функционалды толық базис

Логикалық есептен цифрлық схемаға дейін

Комбинациялық цифрлық схеманы жобалаудың жалпы кезеңдері келесі түрде көрсетуге болады:

1 Есептің шарты

«Шартта не берілген және не табу керек?»

Кіріс айнымалылар мен шығу функциясы анықталады.

2 Кірістер мен шығыстарды анықтау

Кірістер:
A, B, C, ...

Шығыс:
Y

Айнымалылар саны және олардың мағынасы анықталады.

3 Ақиқаттық кестесін құру

A	B	Y
0	0	0
0	1	0
1	0	
1	1	1

Барлық мүмкін мәндер үшін шығыс мәндері толтырылады.

4 Логикалық функцияны алу

$$Y = \bar{A} \cdot B + A \cdot C$$

Ақиқаттық кесте бойынша логикалық функция (алгебралық өрнек) алынады.

5 Логикалық функцияны минимизациялау

$$Y = A + A \cdot B = A$$

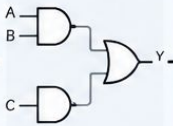
Буль алгебрасы заңдарын қолданып, өрнек қарапайым түрге келтіріледі.

6 Логикалық базисті таңдау

AND–OR–NOT
немесе
тек **NAND**
немесе
тек **NOR**

Қолжетімді логикалық элементтер жиынтығы таңдалады.

7 Логикалық схеманы құру



Алынған функция таңдалған элементтер арқылы цифрлық схема түрінде іске асырылады.

8 Физикалық жүзеге асыру

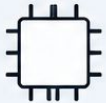


Схема нақты аппараттық құрылғыда (интегралдық микросхема, ПЛИС және т.б.) жүзеге асырылады.

Қорытынды: Бұл — цифрлық схемаларды жобалаудың негізгі логикасы.

Бақылау сұрақтары

1. Буль алгебрасы дегеніміз не?
2. Логикалық айнымалы қандай мәндерді қабылдай алады?
3. Логикалық функция дегеніміз не?
4. Логикалық функцияны қандай тәсілдермен көрсетуге болады?
5. NOT логикалық элементі қалай жұмыс істейді?
6. AND элементінің шығысында қандай жағдайда логикалық 1 пайда болады?
7. OR элементінің шығысында қандай жағдайда логикалық 0 пайда болады?
8. XOR операциясының OR операциясынан айырмашылығы қандай?
9. NAND элементі қалай жұмыс істейді?
10. NOR элементі қалай жұмыс істейді?
11. Ақиқаттық кестесі дегеніміз не?

12. Үш кірісі бар логикалық функцияда неше комбинация болады?
13. Төрт кірісі бар логикалық функцияда неше комбинация болады?
14. Қос терістеу заңының мәні неде?
15. Жұтылу заңын жазыңыз.
16. Де Морганның екі заңын жазыңыз.
17. Логикалық функцияны не үшін минимизациялайды?
18. Логикалық базис дегеніміз не?
19. NAND неліктен әмбебап логикалық элемент болып табылады?
20. NOR неліктен әмбебап логикалық элемент болып табылады?
21. AND–OR–NOT базисіне қандай логикалық амалдар кіреді?
22. NAND көмегімен NOT операциясын қалай алуға болады?
23. Нақты техникалық шарт логикалық схемаға қалай түрлендіріледі?