Лекция 3

Логически основи на ЕИМ. Логически електронни схеми

1. Логически основи на ЕИМ

Теоретическите основи за създаването на ЕИМ са изградени върху фундамента на някои специални математически дисциплини. Една от най-съществените от тях е логическата алгебра (булева алгебра), разработена от Джордж Бул - английски математик от 19 век. Апаратът на булевата алгебра се използва за синтезиране и оптимизация на електронните схеми, използвани в компютърните системи.

Информацията в ЕИМ се представя чрез използване на двоичната бройна система. Логическата алгебра от своя страна е дисциплина, която се занимава с обекти, близки по характер с елементите на двоичната бройна система. Поради тази причина, теоретичните постижения в логическата алгебра се използват като основа за съставяне на различни логически схеми за преобразуване и обработка на двоична информация. Така в ЕИМ информацията се третира по различен начин в зависимост от действията, които се извършват. Съдържанието на елементите, които съхраняват информацията в ЕИМ, може да се разглежда като двоични цифри и да се интерпретира като информация (числова, текстова и други). Когато трябва да се обработва информацията, съдържанието на тези елементи се разглежда като логически обекти и посредством апарата на логическата алгебра се извършва съответното преобразуване на информацията.

Логическата алгебра се занимава със специални обекти, наричани логически величини или променливи. Тези обекти могат да се дефинират като абстрактни величини, които имат стойност истина (**true**) или лъжа (**false**). Например, това могат да бъдат обикновени изречения от разговорния език, към които може да бъде поставен въпросът дали е истина или лъжа твърдението в тях. Нека разгледаме следните изречения (означени като p, q, r, s):

1. България е държава в Европа. (**p**)
2. Желязото е метал. (**q**)
3. Къде се намира ЮЗУ? (**r**)
4. Числото 8 е по-голямо от 23. (**s**)

Към три от тези изречения (p, q и s) може да се поставим въпроса “истина или лъжа е?” твърдението в него. Такива изречения се наричат съждения. Ако се абстрахираме от конкретното съдържание на съжденията и разглеждаме само какъв е отговора на въпроса “истина или лъжа?”, тези изречения може да се разглеждат като абстрактни обекти, които имат стойности истина (**true**) или лъжа (**false**). По-нататък те могат да се третират като величини, без да се свързват непосредствено със съжденията. Това могат да бъдат и други обекти, които се идентифицират със стойности истина (**true**) или лъжа (**false**). Те могат да бъдат именовани (както променливите в математиката) и тогава те се разглеждат като логически (булеви) величини - променливи или константи. Стойностите на логическите величини са **истина** или **лъжа**, като за удобство при представянето се записват с двоичните цифри 0 и 1 (0 - лъжа и 1 - истина). Тогава стойността **ν** на логическа величина ***x***, може да се представи по следния начин:

$$\nu(x) = \begin{cases} 0 \longrightarrow ako \quad x \quad true \\ 1 \longrightarrow ako \quad x \quad false \end{cases}$$

За горните примери можем да запишем: ν(p) = 1 ; ν(q) = 1; ν(s) = 0.

Разгледаните дотук съждения са прости. Простите съждения могат да се използват за образуване на съставни (сложни) съждения. За съставяне на сложни съждения се използват специални свързващи елементи (съюзи). Тези съюзи (логически връзки) дефинират логически операции, които се използват за образуване на логически изрази в пространството на логическите величини (както аритметичните операции в пространството на числата).

Основните логически операции са: конюнкция, дизюнкция, отрицание, изключваща дизюнкция, импликация и равнозначност.

Конюнкция. Операцията, при която едно съждение се получава при свързване на две съждения посредством логическата връзка **И** (**and**), се нарича конюнкция. Пример за конюнкция на две съждения е изразът: "Азотът е газ **и** е съставна част на въздуха". Това е съставно съждение, получено от две прости съждения и се нарича конюнкция.

Ако **p** и **q** са две прости съждения, то тяхната конюнкция се означава с **p & q**. Приема се, че съждението **p & q** има стойност истина (1), само когато и двете съждения **p** и **q** имат стойности истина (1). Във всички други случаи стойността на конюнкцията е неистина (0).

Дизюнкция Операцията, при която едно съждение се получава от две съждения посредством логическата връзка **ИЛИ (or),** се нарича дизюнкция Пример на дезюнкиця е изразът: "Орелът е птица **или** живее в морето". Това съставно съждение реализира дизюнкция на две прости съждения, като за благозвучие в говоримия език се пропуска повторението на подлога в простите изрешения.

Ако **p** и **q** са две прости съждения, то тяхната дизюнкция се означава чрез **p** V **q**. Приема се, че съждението **p** V **q** има стойност истина (1), когато поне едно от двете съждения **p** или **q** имат стойности истина (1). Когато и двете съждения имат стойност неистина, стойността на дизюнкцията е неистина (0).

Отрицание. Някои съждения се образуват чрез отричане на други съждения. За тази цел най-често се използва частичката **не (not)**. Операцията, при която се получава ново съждение чрез отричане на съществуващо съждение, се нарича логическо отрицание. Следното съждение е образувано чрез логическата операция отрицание: "Водата **не** е газ". Ако **p** е съждение, то неговото отрицание се означава с ⌉ p. Стойността на логическата променлива, която се получава чрез операцията отрицание се нарича още инвертирана стойност. В някои литературни източници инвертираната стойност се означава с горно подчертаване – $\overline{p}$.

Изключваща дизюнкция (изключващо или). Ако **p** и **q** са две съждения, то тяхната изключваща дизюнкция се означава чрез **p** $\overset{o}{V}$ **q**. Изключващата дизюнкция има стойност истина, само ако едното от съжденията е истина, а другото е лъжа. Във всички други случаи стойността е неистина.

Другите логически операции са импликация, която се записва като **p -> q** и равнозначност - **p <->q** . Тези операции не са основни, тъй като могат да се получат посредством предишните операции. Техните стойности са дадени в следната таблица на логическите операции:

p	q	p&q	p V q	⌉p	⌉q	p Ý q	p -> q	p <->q
1	1	1	1	0	0	0	0	1
1	0	0	1	0	1	1	1	0
0	1	0	1	1	0	1	0	0
0	0	0	0	1	1	0	0	1

Дефинираните логически операции могат да се използват за получаване на логически изрази. В тези изрази могат да се използват скоби, за да се зададат определени приоритети при изпълнение на логическите операции. По този начин логическите величини и операциите, които могат да се извършват с тях, дефинират логическата алгебра. Ето няколко примера на логически изрази:

(p V **q)** & **(r** Ý **q)**

(s V ⌉**p)** V **(r** & **s)** **<->** **(q** V **s)**

¬ (p & r)

Тези изрази задават сложни съждения и се наричат логически изрази. Стойността на тези изрази се определя от стойностна на съдържащите се в изразите логически величини и използваните логически операции. Ето едно примерно изчисляване на логически израз:

(s V ¬ p) V (r & s) <-> (q V s)

ако стойностите на логическите променливи са: s = 1; p = 0; r = 1 ; q = 1, то

((1 V ¬ 0) V (1 & 1)) <-> (1 V 1) =(1 V 1) <-> 1 = 1 <->1 = 1.

Както в математиката, величините в един логически израз могат да се разглеждат като аргументи, а резултатът от израза като функция на логически променливи. Логическите функции задават определена изходяща стойност (0 или 1) при зададена комбинация от стойности на аргументите $\mathbf{y = f(x_1,x_2,x_3,x_4,\ldots.)}$. Ако се разгледа семейство от логически функции, действащи върху определена група аргументи, се получава съответствие между входящи величини (аргументи) и изходящи (резултантни) стойности.

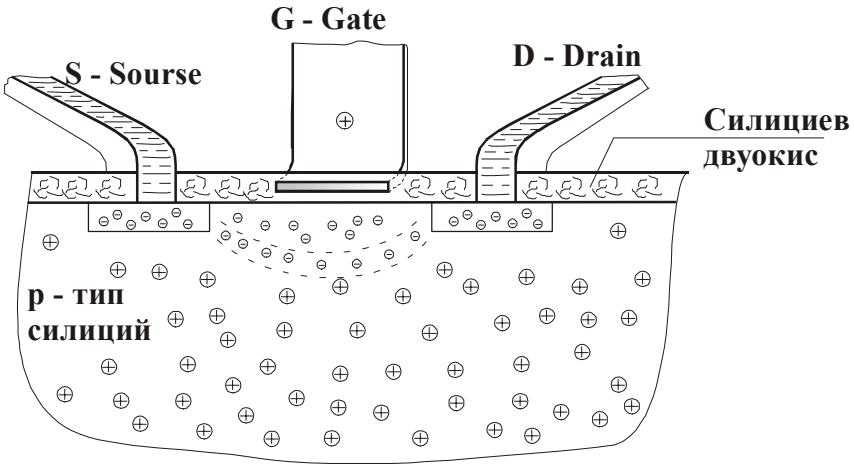
Посредством логическите функции се проектират различни логически схеми, използвани в електронно-изчислителните устройства. От тази гледна точка, проблемът за преобразуване и минимизация на логическите функции има важно значение, тъй като се получават по-прости електронни схеми за реализация. Съществуват различни методи за минимизация на логически функции.

2. Логически електронни схеми.

Транзистори. Видове. Полеви транзистори.

Основни градивни елементи на електронните схеми се явяват транзисторите. В първите модели електронно-изчислителни устройства се използваха електронни лампи, но по-късно те бяха заменени с транзистори. В съвременните компютърни системи се използват огромно количество транзистори, които се вграждат в малки силициеви пластини с много голяма степен на интеграция. Съществуват два основни типа транзистори - биполярни и униполярни (полеви). Биполярните транзистори са от типа **p-n-p** или **n-p-n** тип и се използват широко в електрониката. Най-важната особеност на този тип електронни елементи е, че при тях има инжекция на токоносители през **p-n** прехода и работният им ток се определя от два вида токоносители – електрони и дупки. По тази причина те се наричат биполярни. Управлението на транзистора се извършва чрез изменение на тока в управляващата верига. Обикновено биполярните транзистори са с малко съпротивление.

За разлика от биполярните транзистори, при полевите транзистори липсва инжекция на токоносители през прехода и работният им ток се обуславя или само от електрони или само от дупки. Затова те се наричат униполярни Управлението им се извършва посредством напрежението на управляващ електрод и обикновено имат голямо съпротивление. Използват се в различни логически и цифрови схеми. Те са известни още като МОП (метал - оксид - полупроводник) транзистори или МОС (метал - оксид - силиций) транзистори.



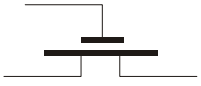
Фиг. 1.1 Схема на полеви транзистор

На фиг. 1.1 е показана принципна схема на полеви транзистор. Той се изгражда върху силициева подложка обикновено от тип **p** (с дупчеста проводимост). Повърхността на силициевата пластина е покрита със силициев двуокис, който играе роля на изолатор. Върху повърхността на пластината са вградени две области от силиций с **n** – проводимост (електронна проводимост), към които са свързани алуминиеви изводи (електроди). Единият от изводите се нарича **Source (S),** а другият - **Drain (D)**. Третият електрод се изработва от поликристален силиций и се разполага в изолационния слой. Той се свързва с метален извод, който се нарича **Gate (G**).

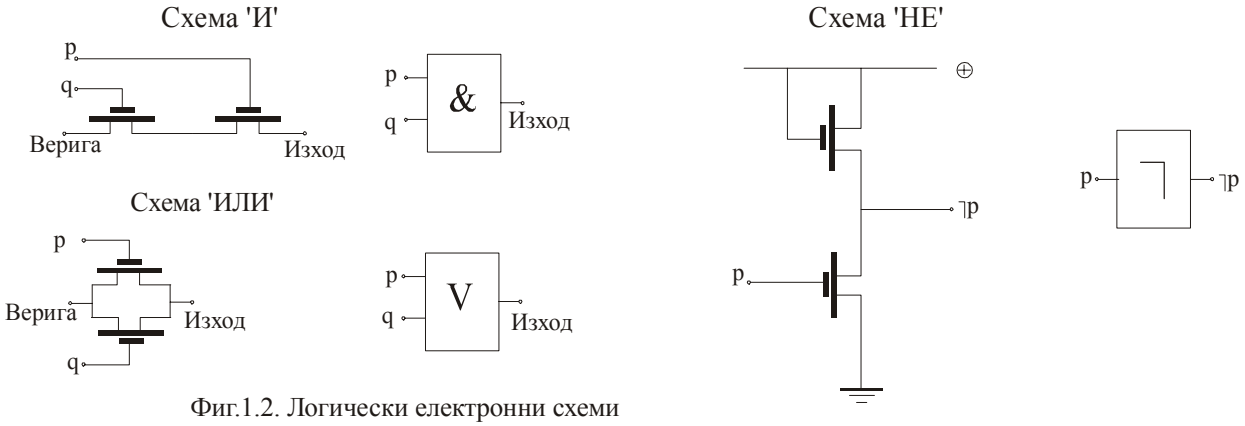
Посредством изводите S и D транзисторът се свързва в електрическа верига, а извода G се явява управляващ и към него може да се подава положителен или отрицателен потенциал. Когато към G не е подадено напрежение, транзисторът е запушен, тъй като в **p** областта няма токови носители, които да осъществят връзка между **n** - областите на изводите S и D. Когато към G се подаде слаб положителен потенциал, между **n** - областите се натрупват отрицателно заредени електрони (от p - областта), които образуват проводящ канал между двата електрода (индуциран проводящ канал). Транзисторът се отпушва и във веригата протича слаб ток. Когато потенциалът на G стане нула или отрицателен, то p - областта се възстановява в този участък и транзисторът се запушва.

Характерно за полевите транзистори е, че те имат голямо съпротивление и през тях протича малък ток. Посредством потенциала на G може да се регулира тока през транзистора, но в цифровите електронни схеми това свойство почти не се използва. В тях полевият транзистор играе роля на обикновен вентил (две състояния - отпушено и запушено).

Когато на изводът G има постоянно подаден положителен потенциал, полевият транзистор играе роля на обикновено съпротивление, чиято стойност може да бъде регулирана посредством конструктивните характеристики на транзистора (разстояние между електродите, разположение на G извода и други). При подходящо оформление на елементите на транзистора той може да изпълнява роля и на кондензатор, тъй като притежава известни възможности за съхраняване на електрически заряди за определено време (електрически капацитет). Тези характеристики правят полевият (МОС) транзистор основен градивен елемент на цифровите електронни схеми.

Разглеждан като електронен вентил МОС транзисторът се изобразява в електронните схеми като:  .

С помощта на МОС транзистори лесно могат да бъдат реализирани основните логически операции в Булевата алгебра. На фиг. 1.2 са показани електронни схеми,



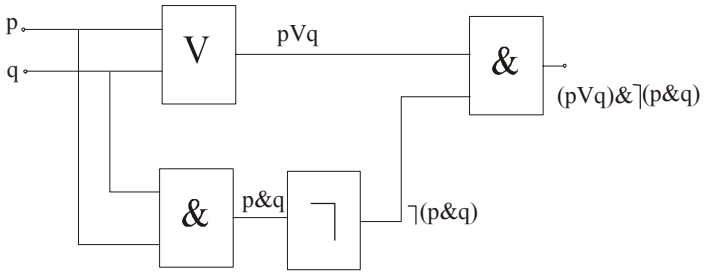
Фиг.1.2. Логически електронни схеми

реализиращи логическите операции ‘И’, ‘ИЛИ’ и ‘НЕ’. За реализация на логическата операция ‘И’ са необходими два полеви транзистора, които се свързват последователно в

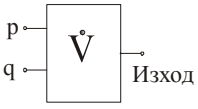
електрическа верига. Когато веригата е затворена (транзисторите са отпушени), се приема, че резултатът е ‘**истина**’ (**true**), а когато е прекъсната - **лъжа** (**false**). За входните величини p и q се приема, че имат стойност ‘истина’, когато на входа се подава положителен потенциал и ‘лъжа’ - когато няма потенциал. Тогава при логическата схема ‘И’ веригата ще бъде затворена, когато и двата управляващи входа на транзисторите (p и q) имат положителен потенциал - стойност ‘истина’. При логическата схема ‘ИЛИ’ транзисторите се свързват паралелно и веригата ще бъде затворена, когато поне на единия от входовете има положителен потенциал.

При логическата схема ‘НЕ’ единият от транзисторите работи като съпротивление (горния транзистор) в електрическа верига, в единия край на която има постоянен положителен потенциал, а другият край е свързан към нулев потенциал (земя). Това означава, че горният транзистор е отпушен постоянно и играе роля на съпротивление. За да се осъществи веригата, трябва да се отпуши долният транзистор. Резултатът се определя от потенциала на изходящата линия ¬**p**. Входящата линия **p** определя дали долният транзистор е отпушен или не. Когато на този вход се подаде положителен потенциал, транзисторът се отпушва и по веригата протича ток. Тогава на горния транзистор се появява пад на напрежение и на изхода ¬**p** се получава по-ниско напрежение (резултат ‘лъжа’). Когато на входа няма положителен потенциал (‘лъжа’), транзисторът е запушен и във веригата не протича ток. Тогава на изхода ¬**p** се поддържа висок потенциал, което съответства на стойност ‘истина’.

Всяка от горните логически схеми е представена и с условно означение, което улеснява използването им в други по-сложни логически електронни схеми. Тези схеми са основни, защото чрез тях могат да бъдат получени (синтезирани) много електронни схеми, използвани в цифровата схемотехника. Като илюстрация може да се представи електронната схема, реализираща логическата операция ‘Изключващо ИЛИ’. За тази схема се въвежда условно означение, както за останалите логически схеми.



Фиг. 1.3. Логическа схема 'Изключващо ИЛИ'



Ако се проследи работата на схемата, ще се установи, че тя наистина реализира логическата операция ‘изключващо ИЛИ’ – на изхода да се получава резултат ‘истина’ само ако на входовете има различни стойности – 1 и 0 или 0 и 1. Така например, ако приемем, че p и q имат стойности 1, то на входа на последната схема & ще има 1 от изход на схема V и 0 от схемите & и ¬. Това ще генерира 0 на изхода. Същият резултат се получава и когато за p и q се зададат стойности 0. Само когато за p и q се зададат различни стойности, на изхода ще се реализира стойност 1.

Електронни логически схеми.

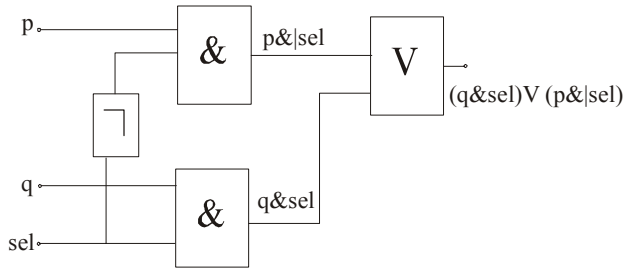
Съществуват два основни типа електронни схеми, използвани в цифровата схемотехника - комбинационни електронни схеми и схеми с обратна връзка (памет). При комбинационните схеми, резултат от изпълнение на логическа операция се получава едновременно с подаването на входящи стойности (сигнали). Когато тези сигнали изчезнат от входовете на електронната схема, изчезва и резултатът от логическата операция. При схемите

с обратна връзка резултатът от логическа операция се запазва (съхранява) до постъпване на други сигнали, които променят изходния сигнал.

Тук ще бъдат разгледани някои елементарни схеми от двата типа.

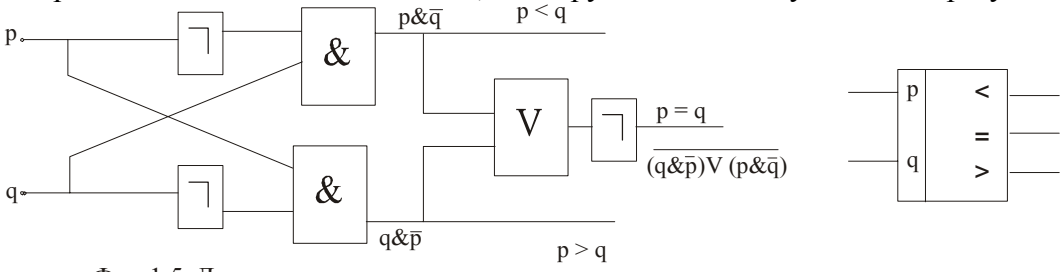
А. Комбинационни електронни схеми.

- Електронна схема селектор. Тази електронна схема се използва за избор между две логически (двоични) стойности. Тя има два логически входа, на които се подават сигналите (стойностите) на величините **p** и **q** и един вход за сигнал (**sel**), който определя коя от двете стойности на входните величини ще бъде изведена на изхода. Когато стойността на **sel** е 1, на изхода се извежда стойността на **q**, а ако е 0 - се извежда стойността на **p**. Това се проверява лесно чрез задаване на различни стойности на **p** и **q** и изчисляване на стойностите, които се получават като резултат на изхода на схемата.



Фиг. 1.4. Логическа схема селектор

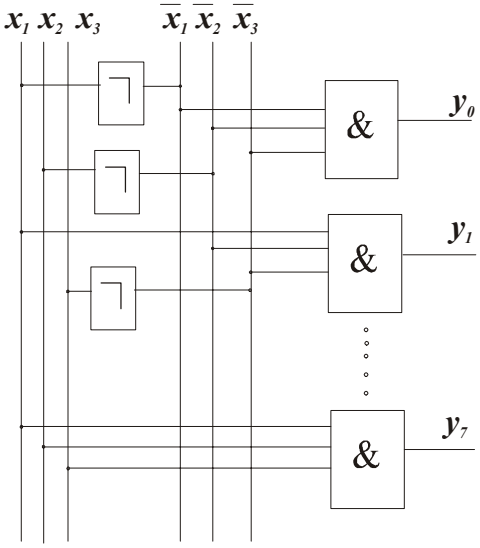
- Компаратор. Компараторът има два логически входа **p** и **q** и три изхода. Когато двете входни величини са еднакви (единици или нули), то на горния и долния изход се получава логическа нула, а на средния изход – логическа единица. Този резултат отговаря на равенство на сравняваните входни величини - **p = q**. Когато входната величина **p** е единица, а **q** - нула, се появява единица на долния изход, а другите изходи са нули. Това съотношение на изходните сигнали определя резултат от сравнението **p>q**. Накрая, когато **q** е единица, а **p** - нула, на горния изход се появява единица, а на другите изходи нула. Това е резултат **p<q**.



Фиг. 1.5. Логическа схема компаратор

- Дешифратор. Дешифраторът е комбинационна схема с **n** входа и $\mathbf{m = 2^n}$ изхода. На изхода на дешифратора се формира комбинация от стойности, при които само на един от изходите има единичен сигнал, а на всички останали - нула. Единичният сигнал, формиран на един от **m** - те изхода, еднозначно определя зададена комбинация от стойности на входните сигнали. Следователно, броят на изходите на дешифратора трябва да е равен на броя на възможните комбинации от стойности на входните сигнали $\mathbf{m = 2^n.}$ На фиг. 1.6 е показана схема на дешифратор с три входни сигнала (x_1, x_2, x_3) и $m = 3^2 = 8$ изходни сигнала. В таблицата са показани стойностите на изходните сигнали на дешифратора при различни комбинации на входните сигнали.

- Полусуматор. Ако се анализира логическата схема ‘Изключващо ИЛИ’ (фиг.1.3) се установява, че тя изпълнява функциите на така наречения полусуматор. Това е така, защото при двоичното сумиране, когато двете цифри от съответния разряд на числото са нули или единици, резултатът е нула, а когато едното е единица, а другото нула – резултатът е единица. Нарича се полусуматор, защото тук не се отчита преносът от сумиране на цифрите от предишния разред на двоичното число.

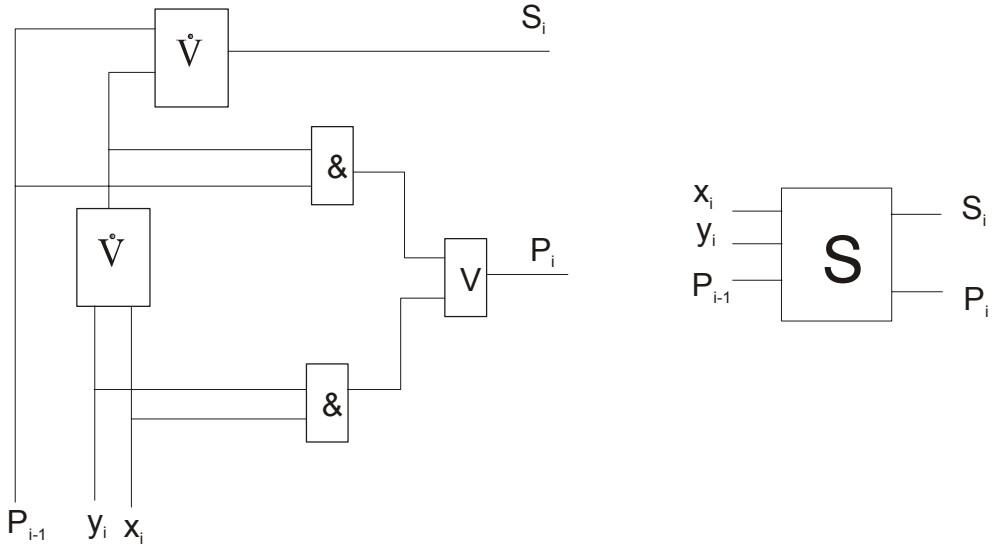


Фиг. 1.6. Логическа схема на дешифратор

Таблица 1.1. Таблица на истинност на дешифратора

Входове			Изходи							
x_1	x_2	x_3	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

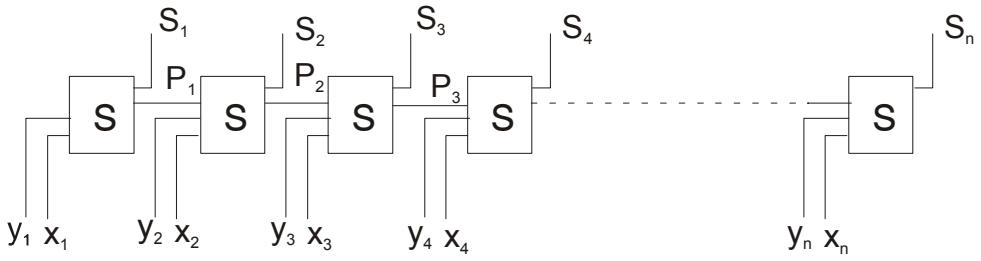
- **Комбинационен суматор**. За да се построи устройство за сумиране на двоични числа, е необходима по-сложна схема, която се нарича комбинационен суматор. Такава схема е показана на фиг. 1.7. Тя има три входни величини – двете двоични цифри x_i, y_i на i-тия разред на двоичното число и пренос от сумирането на цифрите от предишния разред P_{i-1}.



Фиг. 1.7. Логическа схема на комбинационен суматор

Комбинационният суматор формира две изходящи величини, които са резултат от сумирането на цифрите от съответния разряд на числото S_i и пренос за сумиране в следващия разряд P_i. Ако трябва да се синтезира логическо устройство, което да извършва сумиране на двоични числа с **n**-разряда (цифри), трябва да се комбинират **n** суматора, както е показано на фиг. 1.8. Двоичните цифри на едното събираемо ($x_1, x_2, \ldots x_n$) и на другото събираемо ($y_1, y_2, \ldots y_n$) се подават по двойки на входовете на отделните суматори на сумиращото

устройство, а преносите се подават последователно между суматорите. На изходите на отделните суматори се формира резултатът от сумиране на двете двоични числа.

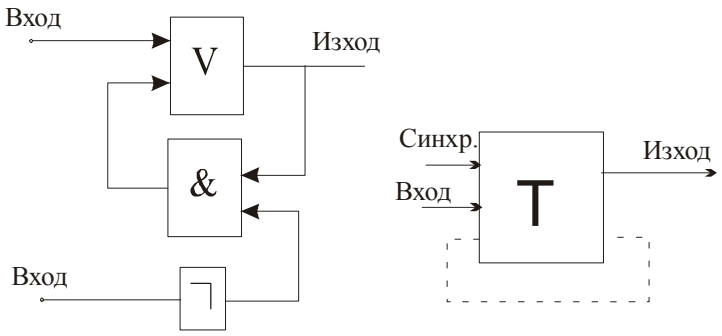


Фиг. 1.8. Сумиращо устройство

Логически схеми с обратна връзка.

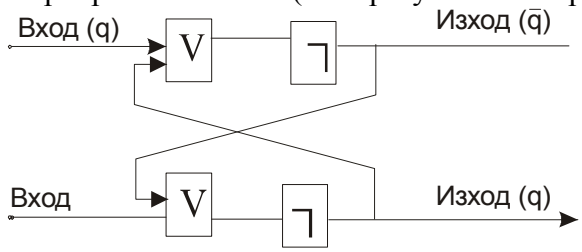
Разгледаните досега логически електронни схеми формират изходен резултат, когато на входа на схемата са подадени стойности на входните величини. Когато няма входни сигнали, не се формира изходен резултат. В компютърните системи се налага дадени състояния (логически стойности) да се съхраняват за дълго време (елементи на памет). За тази цел се използват специални електронни схеми, които включват обратна връзка. Наличието на обратна връзка в електронните схеми дава възможност тя да остане в зададено състояние, дори когато входният сигнал изчезне. Такава схема може да ‘запомни’ определено състояние и може да се използва като елемент от паметта. Общото название на такива схеми е тригери.

Най-простата електронна схема на тригер е показана на фиг.1.9. На единия вход на тригера (в случая на горния вход) се задава стойността, която трябва да се съхрани, а на другия вход се задава противоположна стойност. Тези стойности се задават само за момент, след което на изхода се запазва стойността, подадена на горния вход на тригера. Това може да се види, като последователно се анализират състоянията при задаване на различни стойности на входа на тригера. Като пример, нека разгледаме задаването на стойност 1 на тригера. Това означава, че на горния вход се подава 1, а на долния – нула. Независимо каква е била стойността на изхода преди подаване на входния сигнал, като резултат на изхода на логическата схема ‘ИЛИ’ (V) ще се получи стойност 1. Тази стойност се явява входен сигнал за схемата ‘И’ (&). Другият входен сигнал за тази схема се явява инвертираният сигнал от долния вход. Той е логическа нула (след инвертирането става 1) и остава такава и след прекъсване на подаването на входните сигнали. Следователно, на изхода на схемата ‘И’ ще има стойност 1, която като вход на схемата ‘ИЛИ’ поддържа съхраняването на изходна стойност 1. Когато трябва да се промени стойността, която съхранява тригера на горния вход, трябва да се подаде стойност 0, а на долния стойност 1. Тогава на входа на схемата ‘И’ ще се появи 0 (инвертираната стойност на долния вход) и на изхода й вече ще има 0. Това заедно с нулата от горния вход ще генерира на изхода на схемата ‘ИЛИ’ резултат 0 (който се съхранява). Когато се прекъснат входните сигнали, нулата на изхода се съхранява, тъй като на входа на схемата ‘И’ се подава резултатът (0) и тя има на изхода си нула.



Фиг. 1.9. Схема на тригер

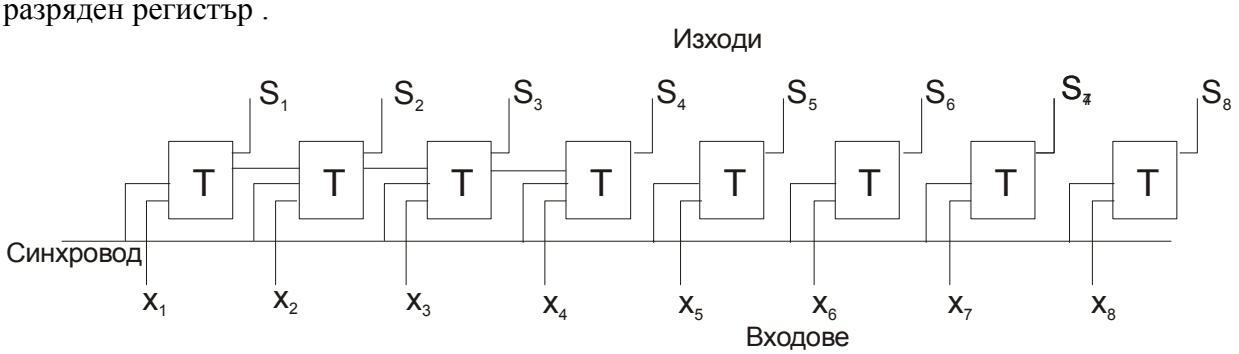
По-често в практиката се използват симетрични тригери. На фиг. 1.10 е показан най-простият симетричен тригер. Неговото действие е сходно с това на тригера от горната схема. И при симетричния тригер се подават два входни сигнала с противоположна стойност, но симетричният тригер може да има два изхода – за съхраняваната стойност и за нейната инвертирана стойност (за образуване на обратния код на числата).



Фиг. 1.10. Симетричен тригер

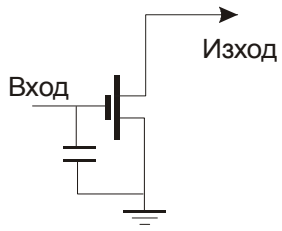
Тригерите са основни елементи на паметта в компютърните системи. Тези устройства работят в синхронен режим – синхронизират се с тактовите импулси на тактовия генератор на компютърните системи. Затова в условната схема на тригера е отбелязан вход за синхронизиращи сигнали.

Тригерите се свързват като последователност с общ синхровод и образуват елементи от паметта – регистри или клетки от паметта. Регистрите са памет, която се използва от микропроцесора за извършване на основните действия с информацията или управлението на системата. Броят на тригерите в регистрите определя разрядността на микропроцесорите. На фиг. 1.11 е показана схема на 8 разряден регистър .



Фиг. 1.11. 8 - разряден тригер

Масовата оперативна памет в компютърните системи се изгражда от по-прости и евтини електронни схеми. Като елемент за съхраняване на информацията се използва кондензатор. Пример на електронна схема за съхраняване на информация с кондензатор е показан на фиг.1.12.



Фиг. 1.12. Схема на тригер с кондензатор

Когато кондензаторът е зареден, съществува потенциална разлика между пластините му и на входа ***G*** на полевия транзистор има положителен потенциал. Това осигурява отпушено състояние и между неутралния край (земя) и изхода има верига, което съответства на съхранена стойност 1. Когато кондензаторът не е зареден, транзисторът е запушен и верига липсва. При подаване на входа на положителен потенциал (логическа единица), кондензаторът се зарежда, а когато се подаде нулев заряд (логическа нула) – той се разрежда.

Зарядът на кондензатора се съхранява много кратко време (няколко милисекунди) поради малките размери и ниската стойност на заряда. Това налага постоянно презареждане (опресняване) на паметта. В компютърните системи такова опресняване се извършва автоматично от специални логически устройства.