ЈАНИ СЕРВИНИ
ЖАНЕТА СЕРВИНИ

ДИГИТАЛНА ЕЛЕКТРОНИКА И МИКРОПРОЦЕСОРИ

II (втора) година – електротехничка струка
електротехничар за електроника и телекомуникации

Битола, 2011 год.

Дигитална електроника и микропроцесори
II (втора) година, електротехничка струка,
електротехничар за електроника и телекомуникации

Автори: Дипл. ел. инж. Јани Сервини
М-р Жанета Сервини, дипл. ел. инж.

Рецензенти: Проф. Д-р Мирка Попниколова Радевска
Дипл. ел. инж. Пеце Петров
Дипл. ел. инж. Илче Ацевски

Лектор: Д-р Трајко Огненовски

Уредник: Јани Сервини

Компјутерска обработка: Јани Сервини

Изработка на илустрации и графичко уредување: Јани Сервини

Идејно решение на корица: Јани Сервини

Дизајн на корица: Никола Сотировски

Издавач: Министерство за образование и наука на РМ,
ул. Мито Хаџи-Василев Јасмин бб, Скопје

Печати: Графички центар дооел, Скопје

Тираж: 450

Со решение на Министерот за образование и наука на Република Македонија
бр. 22-293/3 од 15.03.2011 година се одобрува употребата на овој учебник.

CIP - Каталогизација во публикација
Национална и универзитетска библиотека "Св.Климент Охридски" , Скопје
АВТОР: Сервини, Јани - автор
ОДГОВОРНОСТ: Сервини, Жанета - автор
НАСЛОВ: Дигитална електроника и микропроцесори : II(втора) година - електротехничка струка : електротехничар за електроника и телекомуникации
ИМПРЕСУМ: Скопје : Министерство за образование и наука на Република Македонија, 2011
ФИЗИЧКИ ОПИС: 266 стр. : илустр. ; 29 см
ISBN: 978-608-226-317-5
УДК: 621.38.049.77(075.3), 004.31(075.3)
ВИД ГРАЃА: монографска публикација, текстуална граѓа,печатена
ИЗДАВАЊЕТО СЕ ПРЕДВИДУВА: 08.11.2011
COBISS.MK-ID: 89132298

Најважната придобивка од решавањето на зададен проблем не е добивањето на неговото решение, туку енергијата што се добива при откривањето на тоа решение.

Непознат автор

СОДРЖИНА

ПРЕДГОВОР

Учебникот "Дигитална електроника и микропроцесори" е пишуван во согласност со постоечките наставни планови и програми за истоимениот предмет, кој се изучува во втора година во паралелките од електротехничка струка за образовниот профил електротехничар за електроника и телекомуникации. Ракописот во целост ги опфаќа предвидените наставни содржини, поделени во осум тематски целини. Презентираниот материјал е фундаментален, бидејќи ги анализира елементите и компонентите, кои се основни и неизоставни функционални делови на електронските инструменти, уреди и апарати, како и компјутерските или микропроцесорски базираните системи.

(1) Првата тематска целина БРОЈНИ СИСТЕМИ И КОДОВИ започнува со претставување на начинот на кодирање на информациите во дигитален облик, мерењето на количеството на информација и поделбата на дигиталните кола и мрежи според различни критериуми. Во понатамошниот текст е ставен акцент на бинарниот и хексадецималниот броен систем, што се базични системи за дигиталната електроника и компјутерската техника. Поконкретно, се обработува конверзијата на броевите помеѓу бинарниот, хексадецималниот и декадниот броен систем, од еден во друг, потоа правилата за извршување на основните операции во бинарната аритметика: собирање, одземање, множење и делење, означувањето на негативните броеви во бинарен облик, како и бинарните кодови за бројчани (нумерички) и текстуални (алфа-нумерички) податоци.

(2) Во втората тематска целина, БУЛОВА АЛГЕБРА, им е посветено внимание на појдовните аксиоми, логички операции, закони и теореми на логичката алгебра. Потоа се објаснети основните прекинувачки функции, нивниот аналитички, табличен и графички облик, како и преминувањето од еден во друг облик. Значаен дел од оваа тема е минимизацијата на прекинувачките функции со методата на Карноови карти, за чие полесно совладување се дадени поголем број на решени примери. Покрај претходно наведеното, во оваа тематска целина се претставени и симболите на стандардните логички кола: И, ИЛИ, НЕ, НИ, НИЛИ, ЕКСИЛИ, ЕКС НИЛИ и баферското коло со три состојби. Посебен акцент е ставен на анализата на поедноставни логички дијаграми заради одредување на функцијата што тие ја извршуваат, како и на синтезата на логичките мрежи во две нивоа од И-ИЛИ (НИ) и ИЛИ-И (НИЛИ) тип.

(3) Во третата тематска целина, КОМБИНАЦИСКИ МРЕЖИ, се анализира логичката структура и принципот на работа на дигиталниот компаратор, потоа на колата за собирање, комплементирање, одземање, како и на прекинувачките матрици кодер, декодер, мултиплексер и демултиплексер. На крај е дадена поделбата на програмабилните логички структури, а се опишува и начинот на функционирање на PROM мемориската структура.

(4) Во четвртата тематска целина, ФЛИП-ФЛОПОВИ, се изучува функционирањето на елементарните асинхрони и синхрони (тактирани) секвенцијални кола со стандардна и master-slave конфигурација, кои реагираат на нивото на такт сигналот, и тоа поконкретно на SR, JK, T и D флип-флоповите, но воедно и на прекинувачките флип-флопови кои се активираат со појавата на растечкиот или опаѓачкиот раб на такт сигналот.

Овде се обработени и колото за заклучување (лечот) и елементарната RAM мемориска ќелија. При тоа се користат таблици на вистинитост, логички равенки и временски дијаграми. Дополнително е даден осврт на меѓусебната трансформација на флип-флоповите, како и на можноста за нивна примена во реализирањето на посложени секвенцијални компоненти.

(5) Во петтата тематска целина РЕГИСТРИ се обработува логичката структура на регистрите, базичните принципи на нивната работа, како и нивната поделба според начинот на внесување (полнење) и читање на податоците. Стационарниот регистер, поместувачкиот регистер, регистерот со комбиниран влез, со комбиниран излез, како и универзалниот регистер, според функцијата и примената се разликуваат едни од други и во дигиталните системи се користат како стандардни компоненти.

(6) Шестата тематска целина БРОЈАЧИ нè запознава со поделбата на бројачите на асинхрони и синхрони, со нивната логичка структура и принцип на работа. Во оваа тематска целина преку временски дијаграми во поголеми детали е објаснето однесувањето на бинарните бројачи, бројачите наназад и двонасочните бројачи. Овде се дадени и два примери за проектирање на поедноставни асинхрони и синхрони бројачи со произволна основа (модул) на броење. На крајот се анализирани и кружните бројачи со основа 5 и 10 (квинарниот и декадниот бројач).

(7) Седмата тематска целина МЕМОРИСКИ КОМПОНЕНТИ нè воведува во основните поими и концепти кои се однесуваат на мемориските компоненти и на организацијата на меморијата. Поделбата на мемориите овозможува споредување на различните типови на мемории: ROM, PROM, EPROM, EEPROM, RAM, а со тоа и разбирање на сличностите и разликите помеѓу мемориските интегрирани кола. Посебно внимание им е посветено на објаснување на логичката структура на статичката RAM (SRAM) мемориската ќелија, улогата на контролните сигнали, а со тоа и на принципот на работа користејќи ја нејзината функционална таблица. Овде станува збор и за начинот на адресирање на меморијата, а преку анализата на временските дијаграми на едноставен начин се презентира одвивањето на процесите на читање и запишување во меморијата.

(8) Во осмата тема АНАЛОГНО-ДИГИТАЛНА И ДИГИТАЛНО-АНАЛОГНА КОНВЕРЗИЈА (АДК и ДАК) се запознаваме со базичните поими, кои се однесуваат на процесите на АДК и ДАК, како и со принципите на различните методи на АДК и ДАК. На крајот од оваа последна тематска целина е даден принципот на работа на: Д/А конверторите со тежинска и скалеста отпорничка мрежа, како и на паралелниот А/Д конвертор, А/Д конверторите со бројачка рампа, со последователно приближување, А/ДК конверторите базирани на интеграторско коло и делта-сигма А/Д конверторите.

Автор на поголемиот дел од материјалот обработен во тематските целини е Јани Сервини, дипл. елтех. кта инжинер, наставник во средното општинско техничко училиште (СОТУ) "Ѓорѓи Наумов" од Битола, додека автор на прашањата и задачите за повторување кои се нивен составен дел е М-р Жанета Сервини, дипл. елтех. кта инжинер, исто така наставник во СОТУ "Ѓорѓи Наумов" од Битола.

Пишаниот материјал е подготвен врз база на стручна литература која е актуелна последниве неколку години. Како автори, во текстот настојувавме објаснувањата да бидат јасни, разбирливи, сèопфатни и исцрпни, соодветни на возраста на учениците, но воедно насочени кон исполнување на целите од наставната програма за овој предмет. Во таа насока вложивме значајни напори и внимававме да користиме соодветен стил на пишување, без да се намали квантитетот и квалитетот на презентираните наставни содржини во ширина и во длабочина, како од стручен, така и од педагошки и од методолошки аспект.

При објаснувањето на начинот на функционирање на колата користевме конзистентни системи за означување, соодветни таблици на вистинитост и функционални таблици, логички равенки и формули со соодветна нумерација, принципиелни логички и електрични шеми, при тоа применувајќи стандардни симболи за логичките кола и електронските елементи, како и временски дијаграми на напоните во карактеристичните точки на шемите.

Имајќи во вид дека станува збор за стручен предмет во втора година и учениците, за кои е наменета оваа книга ги имаат потребните базични познавања од математика, акцентот го ставивме на поопширно и подетално објаснување на принципот на работа и анализата на колата, додека математичкиот апарат се грижевме да биде сведен на минимум.

Во текстот на учебникот, за секоја тематска целина посебно, се избрани и решени повеќе карактеристични примери со цел полесно да се разбере суштината на методските единици што се обработуваат. Дополнително, на крајот на секоја тематска целина се дадени голем број на прашања и задачи за повторување со различна тежина. Со нивно одговарање и решавање ученикот ќе може да си го проверува и утврдува стекнатото знаење и воедно значително да го зголеми нивото на квалитет, заради што сметаме дека прашањата и задачите се доста важен дел од учебникот.

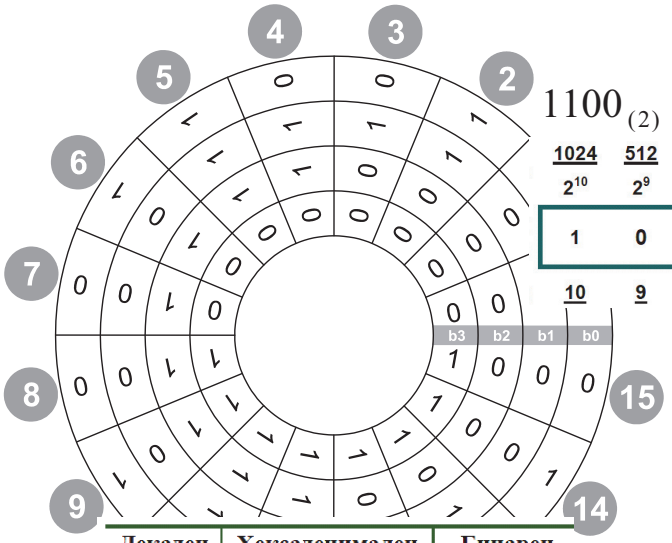
Имајќи во вид дека обработениот материјал е доволно голем по обем и длабочина наставниците имаат можност да вршат селекција и да стават поголем акцент на одредени наставни содржини, зависно од потребите за реализација на наставната програма по овој предмет, како и од капацитетот на учениците во класовите каде предметот се предава. Покрај тоа, воведувањето на прашањата и задачите им остава простор на наставниците за примена на различни наставни методи, со кои учениците ќе можат дополнително да ја развиваат својата креативност.

Искрено се надеваме дека ваквиот пристап значително ќе им помогне на колегите наставници кои го предаваат овој предмет квалитетно да го реализираат наставниот процес во функција на соодветен трансфер на знаење и исполнување на секојдневните работни задачи.

На крај им искажуваме благодарност на рецензентите кои со своите конструктивни и добранамерни сугестии и забелешки придонесоа кон значително подобрување на квалитетот на финалната верзија на учебникот..

Битола, јуни 2010 год.

Од авторите



$$1100_{(2)} = 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 8+4+1 = 12_{(10)}$$

1024	512	256	128	64	32	16	8	4	2	1
2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	0	1	1	0	1	0	0	0	1
10	9	8	7	6	5	4	3	2	1	0

Декаден	Хексадецимален	Бинарен
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
1	1	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

1.

БРОЈНИ СИСТЕМИ И КОДОВИ

По изучувањето на оваа тематска целина

- ќе ги познавате предностите на дигиталниот пренос на сигнали;
- ќе го познавате начиниот на кодирањето на информациите во дигитален облик;
- ќе можете да го објаснувате мерењето на количеството на информација;
- ќе ја познавате поделбата на дигиталните кола и мрежи и ќе ги опишувате по различни критериуми;
- ќе ги познавате и разликувате бројните системи;
- ќе ја објаснувате и применувате конверзијата на броевите од еден во друг броен систем;
- ќе ја применувате бинарната аритметика;
- ќе го разбирате означувањето на негативните броеви во бинарен облик и ќе решавате задачи во врска со нив;
- ќе ги објаснувате бинарните кодови за нумерички и алфа-нумерички податоци;

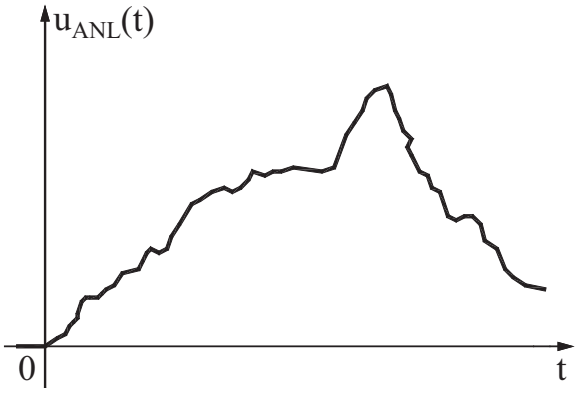
I) ВОВЕД ВО ДИГИТАЛНАТА ЕЛЕКТРОНИКА

1.1. ОСНОВНИ ПОИМИ

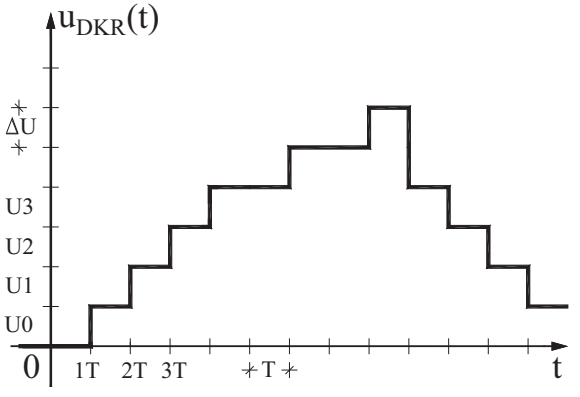
Сите природни појави и процеси се менуваат со текот на времето, заради што тие, аналитички или графички, се опишуваат со временски зависни функции. Во зависност од величината што се следи и испитува овие функции можат да бидат *континуирани* (непрекинати) или *дисконтинуирани* (со прекини).

Макроскопски гледано, скоро сите природни појави се од континуален карактер, бидејќи нивните промени со текот на времето се без моментални (брзи) скокови. Примери за вакви физички големини има многу: температурата, притисокот, брзината, природната светлина, должината итн. Ваквите појави се опишуваат со временски функции кои во било кој временски интервал зафаќаат бесконечно многу различни вредности. На сл. 1-1 е прикажан графикот на една континуирана временска функција. Очигледно е дека амплитудата на оваа функција се менува постепено, бидејќи во било кој конечен временски интервал нејзината амплитуда прима бесконечно многу вредности.

Меѓутоа, во природата постојат и дисконтинуирани појави. За нив е карактеристична појавата на брзи промени во амплитудата бидејќи во еден или во повеќе моменти временски зависната функција, која ја опишува дисконтинуираната појава, нагло преминува од една вредност на друга.



Сл. 1-1. Временски дијаграм на континуирана функција

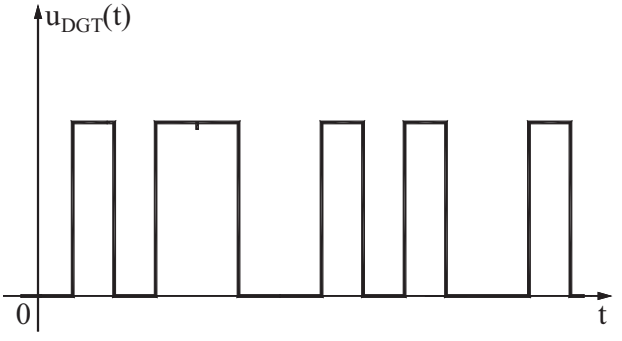


Сл. 1-2. Временски дијаграм на дискретна функција

За човекот од посебен интерес се оние појави што имаат *дискретен* карактер. Тоа се такви дисконтинуирани функции кои можат да примат одреден опсег на вредности од едно конечно множество. Со нив на пример, може да се претстават енергијата на електроните во атомот, буквите од азбуката на говорните јазици, цифрите од бројните системи, итн. Во било кој конечен временски интервал дискретната функција има конечен број на вредности кои може да се нумерираат – секоја вредност со еднозначно определен број. На сл. 1-2 даден е еден пример на ваква функција од каде се гледа дека нејзината амплитуда добива дискретни нивоа.

Човечката желба да ги осознае и користи појавите предизвикала тој да изработува разни уреди со кои ќе ги мери и испитува тие појави, да ги презентира и пренесува добиените резултати. Уредите што ги изучуваме се електрични. Тие работат со електрични сигнали: струи и напони. *Техничките уреди чии влезни и излезни величини се сигнали кои се менуваат по аналогија (сличност) со природните континуирани појави се викаат* **аналогни (линеарни) уреди**. Бидејќи секое ниво на дискретната величина може да се претстави со одреден број, *уредите што работат со дискретни сигнали се нарекуваат* **дигитални уреди**. Терминот *дигитални* потекнува од латискиот збор digitus, кој означува прст или поадекватно значење би било "броење со прсти". Тоа всушност е првиот начин на прикажување на броевите во човечкото општество, но денес се поврзува со англискиот збор digit што значи цифра или број. Како пример за илустрација на аналоген и дигитален начин на работа може да се земе примерот за мерење на времето со часовник. Имено, ако стрелките на часовникот се движат континуирано, тоа е аналогно сметање на времето, а таквиот часовник работи како аналоген уред. Меѓутоа ако часовникот го покажува времето преку броеви кои се менуваат секоја секунда или секоја минута, тоа е дигитално мерење на времето, а таквиот часовник е дигитален уред. Слично на ова, ако некој инструмент ја покажува измерената вредност на електричната величина преку отклон на стрелката тоа е аналоген инструмент, додека ако вредноста се отчитува во облик на број, тоа е дигитален инструмент.

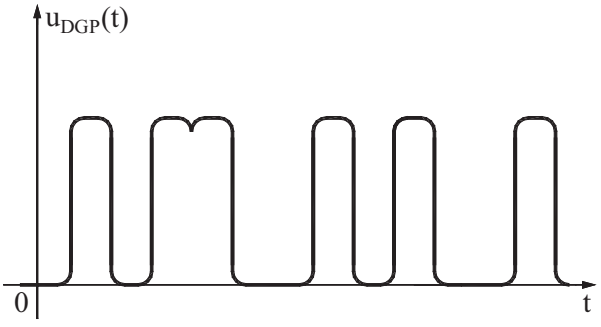
Физичката претстава на броевите во дигиталниот уред оди со посредство на посебен тип на сигнали што имаат специфичен бранов облик. Тоа се *дигитални* сигнали, а пример на еден таков сигнал е даден на сл. 1-3. Од сликата се гледа дека дигиталниот сигнал има бинарен облик, т.е. две нивоа: високо и ниско и всушност претставува една низа од напонски, многу поретко струјни, импулси и паузи. Имено, во продолжение ќе покажеме дека секој број може да се претстави (кодира) со комбинација од само две цифри: 1-а и 0-а, за што е потребен токму сигнал со две различни нивоа.



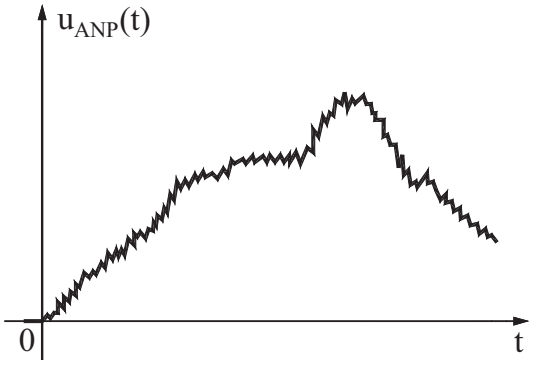
Сл. 1-3. Временски дијаграм на дигитален напонски сигнал

Причината за употреба на дигиталните сигнали е едноставната и евтина изработка на електронските елементи и кола кои ги генерираат тие сигнали. Дигиталните кола можат да се најдат во една од две можни состојби. Поради тоа, основни градбени елементи на секој дигитален уред се електронските кола со две состојби. Покрај едноставната изведба и цената на чинење на дигиталните кола што ги реализираат електричните сигнали во дигитална форма, многу важен фактор е и нивната мала осетливост на влијанието на шумовите и пречките, поточно нивната отпорност на нив, со што нивната трансмисија на далечина е со многу поголема сигурност. Да претпоставиме дека аналогниот сигнал претставен на сл. 1-1 и дигиталниот сигнал прикажан на сл. 1-3, независно еден од друг, се испраќаат до некој приемник по преносен пат на кој дејствуваат непожелни паразитни сигнали.

Сигналите што се примаат после извршениот пренос се дадени на сл. 1-4 и сл. 1-5 од каде се гледа дека тие се изобличиле заради дејството на шумот. Обновувањето на оригиналниот сигнал е многу поедноставно за дигиталниот сигнал од сл. 1-4, отколку за аналогниот сигнал од сл. 1-5. Имено, многу е потешко да се отстранат сите непожелни промени на амплитудата кај добиениот аналоген сигнал, отколку едноставно да се констатира дали бил испратен импулс или пауза, за применият дигитален сигнал.



Сл. 1-4. Временски облик на дигитален сигнал добиен по негово пренесување



Сл. 1-5. Временски облик на аналоген сигнал добиен по негово пренесување

Примери за различни аналогни уреди се сите аналогни мерни инструменти, аналогните сметачки машини, итн., а примери за дигитални уреди се калкулаторите, дигиталните инструменти и како најсложени уреди, дигиталните сметачки машини - компјутерите.

Иако аналогниот начин на работа изгледа поточен од дигиталниот, прецизноста на континуираните сигнали ретко може целосно да се искористи затоа што инструментите и опремата со која тие се мерат не можат да ги мерат, отчитуваат, обработуваат, пренесуваат или на било кој друг начин да ги интерпретираат добиените резултати со многу висока точност. Од друга страна, дигиталните сигнали се присутни во нумеричка форма, па полесно квантитативно се изразуваат, обработуваат, пренесуваат, меморираат и отчитуваат. Со еден збор, тие на човекот му се многу поблиски, бидејќи тој лесно може да манипулира и да работи со нив. Во прилог на оваа констатација оди и фактот што во практиката голем број аналогни величини се претставуваат како збир на конечен број дискретни вредности, односно се претвоуваат во дискретни величини, т.е. се дискретизираат. Многу едноставен и вообичаен пример за ова би било мерењето на тежината кога таа се изразува како сума на тегови со различна единечна тежина: килограм, хектограм, декаграм и грам.

Покрај аналогните и дигиталните уреди, сè почесто се применуваат и уреди кои ги користат добрите страни и на аналогниот и на дигиталниот начин на работа, т.е. што функционираат на *хибриден принцип*. Имено, влезните реални аналогни сигнали се претворуваат во „вештачки" дискретни сигнали над кои се извршуваат потребните операции, а потоа на излезот обработените дигитални сигнали повторно да се конвертираат во аналоген облик. Ваквото претворување се изведува со посредство на склопови што се викаат **аналогно-дигитални конвертори** (АДК), и **дигитално-аналогни конвертори** (ДАК). Благодарение на нив, овозможена е многу широка примена на дигиталните уреди со сите свои предности дури и во областите каде сигналот во суштина е од аналогна природа.

Меѓутоа, при изразувањето на континуалните величини преку дискретни, свесно се прави помала или поголема грешка која се нарекува *грешка на дискретизација* или *грешка на квантизација*. Да речеме, за претходно наведениот пример секоја тежина која има вредности на делови од грамот, или помали, не ќе може точно да се измери. За да се постигне поголема прецизност при мерењето, треба да се употребуваат се помали мерни единици, односно *кванти* или *нивоа на дискретизација* за континуалната величина. Значи кај **хибридните електронски уреди** проблемот на квантизација, т.е. дискретизација по ниво на аналогната величина ќе биде од суштинско значење од причина што со тоа се одредува и грешката, односно точноста при работата.

Дигиталната електроника *ги изучува дигиталните кола, како и колата за А/Д и Д/А конверзија од аспект на нивна анализа, синтеза, проектирање и развој*.

1.2. ИНФОРМАЦИЈА И НЕЈЗИНО КОДИРАЊЕ

Во меѓусебната комуникација, како и со примената на различните уреди луѓето стекнуваат нови сознанија, добиваат соопштенија, новости за светот што ги опкружува, разменуваат и пренесуваат вести. Зборот **информација** во секојдневниот живот значи исто што и *известување* заради што овој поим најмногу се користел во системите за пренос, во телекомуникационите системи. Меѓутоа, брзиот напредок на науката и техниката поставил барања не само за брз и точен пренос и доставување на информациите, туку и потреба од нивна обработка и чување (меморирање).

За олеснување на работата, и преносот и обработката на информациите човекот сакал да ги направи автоматизирано, со машина. Се барале различни начини за претставување на информациите во облик едноставен за пренос, чување и обработка. При ова носител на информацијата е електричниот сигнал, што се содржи во промената на еден негов параметар, а најчесто тоа е неговата промена на амплитудата. Така е овозможено информациите забележани на „апстрактен начин" со симболи, „физички" да се презентираат со електрични сигнали. Големиот број на испитувања покажале дека најевтин, најсигурен, најпогоден и најквалитетен облик за работа е нивното претставување со дигитални електрични сигнали - сигнали со само две различни нивоа. На овој начин информацијата се прикажува во бинарна форма со низи составени само од два симболи. Значи при преносот, обработката и чувањето на информациите, нивното претставување е во поинаква форма од начинот што им е близок на луѓето, т.е. не се користат алфабетски и нумерички симболи (букви и цифри), туку бинарни симболи.

Претставувањето на информациите со помош на симболи, кои се елементи од некое конечно множество се вика **кодирање на информациите**. *Множеството на сите симболи што стојат на располагање за кодирање на информациите се вика* **кодна азбука**, *а секој поединечен симбол од кодната азбука се вика* **коден симбол**. *Секоја група од кодни симболи што претставува информација, или некој нејзин дел, се вика* **коден збор**. Бинарно кодираните информации ќе бидат составени од кодни зборови што претставуваат групи на бинарни симболи, симболи што можат да примат само две различни вредности и му припаѓаат на бинарното множество B. Вообичаено е за овие два симболи да се прифатат ознаките 1 и 0, така што важи $B = \{1,0\}$. Сега станува јасно зошто бинарните кодови и бинарниот броен систем имаат суштинско значење во дигиталната електроника и дигиталната обработка на информации.

1.3. ЕДИНИЦИ ЗА МЕРЕЊЕ КОЛИЧЕСТВО НА ИНФОРМАЦИЈА

Кога станува збор за пренос на информации, субјективно гледано од страна на човекот што го прима соопштението, најважна е неговата содржина, односно колкаво изненадување има во тоа известување, а скоро воопшто не е важна формата и начинот на пренос. Според ова, примената информација дека настан што има голема веројатност на остварување, навистина се случил, содржи мало количество информација. Обратно, соопштувањето за настанување на неочекувани случки кои имале многу мала веројатност дека ќе се остварат, би содржело многу поголемо количество на информација. Од овде произлегува и врската помеѓу количеството на информацијата и веројатноста за случување на настаните.

Меѓутоа, кога станува збор за единици за мерење на количество информација во дигиталната техника, веројатноста нема директно влијание. Имено, како единица за мерење количество на информација се дефинира еден бит, т.е. *појавата на еден симбол од бинарното множество* B = {1,0}. Овај поим произлегува како кратенка од англискиот термин *Binary digit* **(bit)** што значи **бинарна цифра** и се означува со **b**. Вредноста на битот може да биде или 1 или 0, односно кај дигиталниот сигнал тоа ќе биде појавувањето на високо или ниско напонско ниво со кое тој бит е претставен. Ваквиот приод доаѓа оттаму што во дигиталната електроника најважен е коректниот пренос, предвидената обработка и точното меморирање на бинарните симболи, а не е важна веројатноста со која тие се појавуваат. Во системите каде што се обработуваат информациите се цени можноста за различни начини и брзини на обработка, како и големината на капацитетот за меморирање на информациите. Од тука станува јасно дека ако една порака е составена од поголем бројот на кодни зборови, и ако секој збор е составен од повеќе битови, поголемо ќе биде и количеството на информација што се содржи во таа порака.

Во врска со ова како поголема единица од битот се дефинира 1 **бајт** (byte), кој содржи осум бита и претставува случаен распоред на 1-и и/или 0-и и обично се означува со [B]. На следната слика (сл. 1-6) се прикажани примери на два бајти.

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	0	1	0	1	1	1	0

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
0	1	0	0	0	0	1	1

Сл. 1-6. Примери на два мемориски збора со должина по еден бајт.

Во минатото како поголема единица од битот се користел 1 **нибл** (анг. nibble) и тоа за група (низа) од четири бита, но денес ниблот поретко се користи или се заменува со терминот тетрада.

Бидејќи и бајтот е мала единица, во практиката се воведени поголеми единици од бајтот, а тоа се: 1 **килобајт**, тоа е група од 2^{10}=1024 бајта кој се означува со [KB], потоа 1 **мегабајт** [MB] кој содржи 2^{10} [KB]= 2^{20} [B], 1 **гигабајт** [GB] кој се формира од 2^{10} [MB]= 2^{20} [KB]= 2^{30} [B] и 1 **терабајт** [TB] кој се формира од 2^{10} [GB]= 2^{20} [MB]= 2^{30} [KB] = 2^{40} [B]. Од изнесеното произлегува дека во жаргонот на дигиталната обработка на податоци префиксот „кило" има малку неконвенционална употреба, бидејќи означува 2^{10}=1024 единици заради што се бележи со [K] и по тоа се разликува од вообичаеното означување со префиксот [k] кој се однесува на 1000 единици. Според ова 1 [MB] = 1024 x 1024 [KB], итн. множителот ќе биде 1024, а не 1000, како што сме навикнати до сега.

Како пример ќе го решиме следниов едноставен проблем. Да земеме дека еден, било кој, текстуален симбол (буква, цифра или интерпункциски знак) може да се кодира со 1 бајт [B] и дека треба да пресметаме колкав мемориски простор е потребен за да запамтиме една книга од 200 страници, со претпоставка дека една страница содржи по 3.000 симболи.

Решение: 200 стр. x 3.000 симб. = 200 стр. x 3.000 [B] = 600.000 [B]. Бидејќи 1 [KB] $=2^{10}$ [B] = 1024 [B] ≈ 1000 [B], за меморирање на книгата ќе ни треба мемориска компонента со капацитет од 600 [KB].

Овде е многу важно да напоменеме и тоа дека во зависност од бројот на битови n во еден коден збор, може да се пресмета и вкупниот број на различни комбинации од 0-и и 1-ци N кои можат да се појават, според следнава равенка:

$$N = 2^n \qquad (1\text{-}1)$$

Така на пример, ако на располагање имаме 1 [B] бајт, тогаш тој може да се прикаже во 2^8= 256 различни комбинации, што значи дека со 1 бајт можеме да претставиме 256 различни информации, како на пр. целите броеви од 0 до 255, или сите мали и големи букви на англиската абецеда, декадните цифри, знаците на интерпункција, малите и големите букви на македонската азбука, итн. сите вкупно 256 различни симболи.

1.4. ВИДОВИ НА ИНФОРМАЦИИ

Во процесот на работата на компјутерите доаѓа до размена и проток на податоци кои носат различни и специфични информации што се важни за континуирано одржување на правилна функција на сметачот. Овде припаѓаат различните видови на **наредби** (*команди, инструкции*), **податоци** и **адреси.**

Инструкциите содржат информација за сметачот што тој треба да работи. Низата наредби што образува една логичка целина, претставува програма по која компјутерот работи.

Податоците содржат информација за одредена појава од надворешниот свет, или за вредност која е добиена како резултат на некоја обработка внатре во компјутерот. Како податоци може да се сметаат вредностите на различни временски променливи дискретни или континуирани величини. Податоците се внесуваат во сметачот, се обработуваат зависно од инструкциите на програмата, при што се генерираат некои нови податоци.

Покрај податоците и наредбите, во компјутерот се разменуваат уште еден посебен вид информации, а тоа се адресите. **Адресите** содржат информации за тоа која е точната местоположба на податоците над кои треба да се изврши зададената инструкција.

Во сметачот сите информации се внесуваат во бинарен облик за тој да може да ги обработи. На апстрактно ниво тоа се низи од **0**-и и **1**-и, но физички најчесто тоа се електрични сигнали во облик на низи или групи од напонски импулси или паузи.

Бидејќи во компјутерот се извршуваат одредени програми кои покрај нумеричките податоци како влезни податоци може да прифаќаат и други програми или адреси, може да се земе дека и програмите и податоците и адресите**, во поширока смисла на зборот, претставуваат податоци.**

Заклучокот е дека и инструкциите и податоците и адресите, всушност, се информации, кодирани во бинарен облик и претставени со соодветни електрични сигнали. Благодарение на интерната организација на компјутерот и програмата што се извршува, тој ги разликува едните од другите.

1.5. ПОДЕЛБА НА ДИГИТАЛНИТЕ КОЛА И МРЕЖИ

Споменавме дека дигиталните кола оперираат со електрични сигнали кои можат да имаат само две различни нивоа. Ваквите сигнали претставуваат определени податоци кои се кодирани во бинарен облик, а секој од тие податоци во себе содржи одредена информација за реалниот свет.

Основни, базични градбени елементи во дигиталната техника се *логичките кола* (*врати*, *порти*). Тие се реализирани со употреба на полупроводнички прекинувачки елементи, а секоја друга дигитална компонента, мрежа, или склоп може да се изведе со соодветно спојување на одреден број логички кола. Зависно од работата, сите елементарни компоненти од кои се изведуваат дигиталните уреди можат да се распределат во две групи: *компоненти на комбинационата логика (логички компоненти)* и *мемориски (секвенцијални) компоненти*. Логичките склопови ги реализираат логичките состојби 1 или 0, односно нивните комбинации, но без можност за помнење на претходните нивни состојби. *Сите електрични мрежи кои ги имаат овие особини се викаат* **комбинациони логичко-прекинувачки мрежи**, *или скратено само* **комбинациони мрежи**. Заради непостоењето на повратна врска од излезот кон влезот, кај нив излезните сигнали постојат само додека постојат влезни сигнали. Ако влезните сигнали се изгубат, тогаш се губат и излезните сигнали. Меѓутоа, во дигиталната обработка на податоци се јавува потреба да се меморираат одредени податоци, кои порано или подоцна повторно ќе се користат. За таа цел се употребуваат мемориските елементи што служат за помнење на претходните логички состојби. *Бистабилниот мултивибратор, кој популарно се вика* **флип-флоп** *е електронско коло со две стабилни состојби, кое може да се користи за меморирање на еднобитен податок*. Бидејќи флип-флопот може да помни најмало количество информација само од еден бит, тој претставува основен мемориски елемент, т.е. елементарна мемориска ќелија во дигиталната електроника. Флип-флоповите се реализираат со примена на елементарни логички кола кај кои е изведена позитивна повратна врска.

Електричните мрежи кои имаат ваква карактеристика на помнење се викаат **секвенцијално логички-прекинувачки мрежи**, *или скратено само* **секвенцијални мрежи**, *или* **секвенцијални автомати**. Кај овие елементи и мрежи, постојат сигнали на излезот и тогаш кога престанало дејството на влезните побудни сигнали. Токму заради тоа, следните излезни состојби ќе зависат и од сегашните влезни сигнали, но и од редоследот, т.е. секвенцата на претходните состојби низ кои поминало колото.

Некако интуитивно ни станува јасно дека комбинационите мрежи вршат различни операции со податоците кои доаѓаат од секвенцијалните мрежи, каде се запамтени. За реализација на комбинационите мрежи се користат логички кола, додека кај секвенцијалните мрежи основни градбени елементи се флип-флоповите, а покрај нив се употребуваат и различни типови логички кола. Во понатамошното излагање ќе ги набројме најмногу употребуваните комбинациони и секвенцијални компоненти со кои се изведуваат сложени дигитални склопови.

Кола за реализација на аритметичко-логички функции: Сите основни аритметички операции, па дури и диференцирањето и интегрирањето можат да се извршат со различни постапки на собирање од каде што произлегува фактот дека *бинарниот собирач* е од основно значење во дигиталните аритметички уреди. Инаку, тој припаѓа во групата на комбинациони мрежи. Покрај него овде спаѓаат и *колото за одземање*, *колата за комплементирање*, *колото за компарирање* итн.

Овде посебно важно место завземаат и т.н. *аритметичко-логички единици*. Тоа се интегрирани компоненти што реализираат најразлични аритметички и логички функции.

Прекинувачки матрици: *Прекинувачките матрици* се сложени комбинациони мрежи, кај кои прекинувачките елементи се подредени во редици и колони, формирајќи матрични структури. Секоја од овие матрици функционира на различен начин, па нив ги разликуваме по нивните функционални имиња:

1. ***Кодер:*** Со оваа логичка мрежа се реализира постапката на кодирање. На нејзините влезови се јавуваат сигнали кои презентираат цифри од некој броен систем или букви и специјални знаци (симболи, карактери) од некоја азбука, а на излезот се добиваат кодни зборови во бинарниот броен систем или во некој од бинарните кодови.

2. ***Декодер:*** При дигиталната обработка на податоците се јавува потребата за претворување на бинарно-кодираните податоци во некој друг облик, како што е, на пример, декадниот броен систем, или во некој друг систем со основа различна од два. Оваа постапка е инверзна на кодирањето, а се извршува со користење на декодерска комбинациона мрежа.

3. ***Мултиплексер (селектор):*** Тоа е таква комбинациона мрежа која избира еден податок од поголемиот број податоци, присутни на влезовите од мрежата, и истиот го пренесува до единствениот излез. Кој податок ќе биде проследен до излезот, се одредува преку посебните селекциони (адресни) влезови. Постојат и такви мултиплексери кои вршат избор на една група влезни податоци, од повеќето расположливи влезни групи, и неа ја проследуваат до единствените излезни линии.

4. ***Демултиплексер:*** Оваа прекинувачка мрежа функционира на инверзен начин од мултиплексерот. Имено, демултиплексерот го прима податокот кој доаѓа на единствениот влез, и потоа врши негово пренесување до еден од повеќето излези. И во овој случај, како и кај мултиплексерот, постојат селекциони линии за адресирање на соодветниот излез. Се сретнуваат и демултиплексери кои единствената група на влезни сигнали, ја проследува до една од повеќето излезни групи на линии.

Програмабилни логички структури: Ваквите комбинациони мрежи имаат сложена матрична структура, а нивната најважна карактеристика е фактот што даваат можност за *програмирање на врските помеѓу прекинувачките елементи*, т.е. за поврзување на прекинувачките елементи на начин кој ќе биде зададен од страна на корисникот. Тоа се интегрирани компоненти кои наоѓаат сè поширока примена во дигиталните системи.

Регистри: Ова се најчесто употребувани елементи во дигиталните уреди, и тоа посебно во комбинација со аритметичко-логичките единици. Регистрите се составени од одреден број на флип-флопови. Секој бит од информацијата се помни во посебен флип-флоп, заради што регистрите припаѓаат во групата на секвенцијални мрежи. Постојат различни типови на регистри, но најпознати се стационарните и поместувачките. Во стационарниот регистер, како негова содржина, привремено може да се меморира бинарен податок со ограничена должина, како на пр. 1 нибл или 1 бајт. Во поместувачките регистри може да се врши поместување бит по бит надесно, или налево на содржината на регистерот, т.е. на податокот кој се наоѓа во него.

Бројачи: И бројачите спаѓаат во групата на секвенцијални мрежи, затоа што како основен градбен елемент употребуваат мемориски елемент, флип-флоп. Со секоја појава на импулс на влезот од бројачот, доаѓа до последователно менување на состојбата на бројачот. Секоја состојба на бројачот претставува низа од неколку бита, т.е. бинарно кодиран децимален број, што значи дека овој склоп ги брои влезните импулси. По одреден број имулси, бројачот се враќа во почетната состојба и броењето циклично се повторува.

Мемории: Мемориите чуваат бинарни податоци во неизменет облик, така што по определен временски период тие податоци можат повторно да се користат. Меморијата може да се сфати како едно уредено множество на регистри, при што секој регистер претставува една *мемориска локација* во која може да се чува еден податок. Податокот кој се сместува во мемориската локација претставува нејзина *содржина*, при што се вели дека во една локација може да се смести еден *мемориски збор*. Мемориските зборови имаат фиксна *должина* во битови, која обично се изразува во бајти. Секоја мемориска локација има своја позиција во меморијата која се карактеризира со соодветен број: *адресата на локацијата*. Адресата служи за идентификација на положбата на локацијата во меморијата. Со задавање (специфицирање) на адресата може да се пристапи до секоја мемориска локација и да се манипулира со нејзината содржина, т.е. запомнатиот податок: да се прочита меморираниот податок или да се запише нов. Вкупниот број на сите мемориски локации го дава *капацитетот* на меморијата, и тој се дава во килобајти или во мегабајти.

Критериуми за оценка на квалитетот на мемориите се: брзината на работа, која најмногу зависи од организацијата на меморијата, потоа средното време на пристап до мемориските локации, па моќноста на дисипација и цената на чинење.

Дигитални кола за А/Д и Д/А конверзија: Постојат огромен број информации со континуирана природа кои треба да се обработат со дигитални уреди, како на пример: температурата, притисокот, брзината, времето итн. Трансформацијата на континуираните појави од реалниот свет во дигитални сигнали станува сè позначајно со развојот на дигиталната техника. Континуираните појави се трансформираат во електрични сигнали кои се аналогни на нив со помош на *сензори*. Добиените аналогни сигнали се претвораат во дигитални сигнали со електронски кола кои се викаат *аналогно-дигитални (А/Д) конвертори* и како такви може да се обработуваат во дигиталните уреди. Инверзната постапка се реализира со *дигитално-аналогни (Д/А) конвертори*. На влезот во Д/А конверторот доаѓа дигитален сигнал, а на излезот се добива соодветен аналоген сигнал. Ваквиот аналоген сигнал преку *претворувач* се трансформира во сигнал со континуирана природа.

При изучувањето на колата од дигиталната електроника се јавуваат два основни проблеми: едниот проблем е вршење *анализа* на дигиталните кола, а другиот проблем е проблемот на *синтеза*, т.е. *развој*, *проектирање* или *дизајнирање* на дигиталните кола со однапред зададени особини.

Анализата се однесува на конструктивен и функционален план. Нејзината задача е да ја одреди конструкцијата и начинот на работа на зададен дигитален склоп. При ова е позната логичката, електричната или монтажната шема на уредот и влезните променливи (побудата, екситацијата), а треба да се определи одзивот, т.е. излезите од склопот. Анализата е особено потребна при експлоатација, поправка и одржување на дигиталните уреди.

При *синтезата*, познати се особините на склопот кој треба да се добие, како на пример функциите кои тој треба да ги извршува и нивната зависност од влезните сигнали. Како прво се дефинира блок-шемата на склопот, улогата и задачите на секој блок, влезовите и излезите од секој блок, како и капацитетот на меморијата кога постои потреба за помнење на податоците или резултатите. Потоа се врши избор на комбинационите и мемориските елементи со кои уредот може технички да се реализира. Овој проблем практично се сведува на реализација, поточно на проектирање и развој на дигитални уреди.

II) БРОЈНИ СИСТЕМИ И КОДОВИ

1.6. ОСНОВНИ ПОИМИ

Во дигиталната техника информациите се претставуваат и обработуваат нумерички, како броеви, па нормално е да се постави прашањето која форма на претставување на броевите е најпогодна за да бидат тие „разбирливи" за дигиталните склопови. *Во оваа глава ќе се задржиме на претставувањето на информациите на начин кој е најблизок до реалниот начин на претставување на податоците во дигиталните системи*. Овој проблем е многу важен и тој во себе го вклучува изучувањето на процесот на кодирање и тоа посебно т.н. бинарни кодови, како и анализата на различните бројни системи од кои најзначаен е бинарниот броен систем.

Бинарното претставување на информациите е тешко замисливо како дел од човековото изразување затоа што бинарно запишаните податоци, всушност, се *низи од само два различни симболи*. Затоа *бинарната нотација* (*бинарното означување*) е „природно" за дигиталните уреди кај кои сите елементарни кола се прекинувачки и чии излези, како што веќе знаеме, може да се најдат само во една од две можни состојби. *Под поимот* ***кодирање*** *се подразбира начин на претставување на информациите со помош на симболи кои се елементи на некое множество* како што се, на пример, буквите од некоја азбука, или, пак цифрите од некој броен систем. Во натамошното излагање ќе се обработат оние кодови што наоѓаат соодветна примена при анализа на работата на дигиталните уреди, и тоа посебно кај дигиталните сметачки машини, т.е. компјутерите.

Да направиме една аналогија со *националните јазици*. Во човековото општество секој национален јазик претставува средство со кое се остварува комуникација меѓу луѓето. Имено, со зборувањето ние, практично, вршиме „кодирање" на информациите во зборови, па тие стануваат разбирливи за секој член на соодветниот народ. Пишаниот текст, всушност, претставува еден начин со кој се помнат (меморираат) „кодираните" поими. На секој поим одговара еден збор кој може да се подели на гласови, а за секој глас постои соодветен симбол, т.е. буква од националната азбука. За да се пишуваат (кодираат) или читаат (декодираат) зборовите, мора да се познава синтаксата на соодветниот национален јазик. Тоа е една група на правила и прописи што треба децидно да се применуваат и при читањето и при пишувањето.

Бројните (нумерички) системи *претставуваат системи на симболи, кои се викаат цифри, и со кои се означуваат броевите*. Постојат *тежински* (*позициски*) и *нетежински бројни системи*, а ние ќе се задржиме само на тежинските бројни системи. *Секој тежински броен систем има своја* **основа (база, радикс)** *што се означува со b. Тоа е всушност вкупниот број на различни цифри во системот*. Во општ случај, за основа на бројниот систем може да се земе било кој број еднаков или поголем од еден. *Декадниот броен систем* е познат од порано и ние секојдневно го користиме: тој има 10 различни цифри и основа b = 10. Меѓутоа во оваа тематска целина во куси црти ќе се разгледаат и *хексадецималниот броен систем* кој има 16 различни цифри и основа b = 16, потоа *окталниот броен систем* кој има 8 цифри така што неговата основа е b = 8, и на крај како најважен, *бинарниот броен систем* кој има само две цифри и основа b = 2.

1.7. БРОЈНИ СИСТЕМИ

Со воведувањето на различните бројни (нумерички) системи ќе треба да се дефинира запишувањето и означувањето на броевите. За таа цел прво ќе го дефинираме обликот на цифрите за секој броен систем. Имено, декадниот броен систем располага со десет различни цифри **0,1,2,3,4,5,6,7,8,9**, кои се земаат како симболи за цифрите и на другите бројни системи, ако системот има помалку од десет цифри. Така на пр. октални от броен систем располага со осум цифри: **0, 1, ..., 6, 7,** а бинарниот само со две: **0** и **1**. Доколку основата на бројниот систем е поголема од десет тогаш се додаваат и првите букви од англиската абецеда: A, B, C, ..., така што хексадецималниот броен систем располага со 16 цифри и тоа: **0, 1, ..., 8, 9, A, B, C, D, E, F.**

Броевите ќе ги запишуваме така што прво ќе го напишеме самиот број со употреба на цифри, а потоа во заграда или како индекс ќе ја запишиме основата на бројниот систем во кој тој број е запишан. За децималните броеви тоа е 10, DEC или D, за хексадецималните броеви се додава 16, HEX или само X, за октални те 8, OCT или само Q, а за бинарните 2, BIN или само B.

Често пати се поставува како проблем следното прашање: Колку различни броеви *N* можат да се напишат во некој броен систем со основа b ако се зададени n цифри. Бројот *N* се пресметува така што основата на бројниот систем b се степенува на степен n:

$$N=b^n \tag{1-2}$$

Од ова станува јасно дека со еднаков број на цифри *n,* во различни бројни системи, повеќе броеви можат да се напишат во оној систем кој е со поголема основа. На пример, со три цифри во декадниот броен систем може да се напишат вкупно $10^3 = 1000$ различни броеви, во хексадецималниот броен систем тој број е поголем и изнесува $16^3 = 4096$ различни броеви, во октални от броен систем може вкупно да се напишат $8^3 = 512$ броја, а во бинарниот броен систем само $2^3 = 8$.

Кај тежинските бројни системи секоја цифра во бројот има одредена *тежина*, (*тежинска вредност*) која зависи од позицијата (местоположбата) на цифрата во бројот во однос на позиционата (раздвојната) точка (до сега велевме децимална запирка). **Тежината** *всушност е некој цел степен (потенција) од основата b на бројниот систем*. Тежината на првата цифра лево од точката (нултата позиција) (тежинската вредност на цифрата на нултото бројно место) изнесува b^0, на втората цифра (првата позиција) тежината и е b^1 итн. Најголема тежина (најголем степен на основата, највисоко ниво) има онаа цифра од бројот која се наоѓа најлево од точката и истата се означува со *MSD (анг. Most Significant Digit)* т.е. *најзначајна (најмногу значајна) цифра*. Првата цифра десно од точката (минус првата позиција) има тежина b^{-1}, втората b^{-2} итн. Значи, најмала тежина (најмал степен на основата) ќе има најдесната цифра во бројот и таа се означува со LSD (анг. Last Significant Digit), т.е. *најнезначајна (најмалку значајна) цифра.*

Одредувањето на вредноста на некој број *X* кој е зададен во било кој тежински броен систем со основа b, и има *n*-целобројни цифри и *m*-дробно-рационални, може да се врши преку следната т.н. *тежинска формула.*

$$X = X_{(10)} = \sum_{i=-m}^{i=n-1} c_i t_i = \sum_{i=-m}^{i=n-1} c_i b^i = c_{n-1}b^{n-1} + c_{n-2}b^{n-2} + \ldots + c_1 b^1 + c_0 b^0 + c_{-1}b^{-1} + \ldots + c_{-m}b^{-m} \tag{1-3}$$

Во формулата (1-3) со c_i е означена цифрата што се наоѓа на *i*-тото место сметајќи од позиционата точка, при што 0-то место е првото лево, потоа е 1-то, па 2-то, итн., додека

првото место десно од точката е (–1)-то, потоа е (–2)-то, па (–3)-то, итн. Со t_i е претставена позиционата вредност (тежината) на соодветната цифра. Кај природните тежински системи секогаш важи следната равенка:

$$t_i=b^i \tag{1-4}$$

каде што b е константа која ја претставува основата на бројниот систем. Со формулата (1-3) практично се врши конверзија од било кој броен систем во децимален.

При анализата на целите броеви во различните бројни системи, слично како и за целите декадни броеви, не се запишува раздвојна точка или запирка, туку се подразбира дека таа се наоѓа веднаш до најдесната цифра, т.е. до цифрата со најмала тежина.

За секој броен систем, покрај *вистинските* (*директните*, *номиналните*) *вредности* на броевите, се дефинираат и нивните *комплементи*. Комплементот на некој број X се означува со $\overline{X}$, а се одредува според следнава дефиниција:

$$\overline{X} = K - X \tag{1-5}$$

каде што K е константа која може да има вредност b^n или $b^n - 1$, при што b е основата на бројниот систем, а n бројот на цифрите во даденият број.

Кога $K = b^n - 1$, се добива *комплемент до најголемиот број* во применетиот броен систем. Така на пример, за четирицифрени декадни броеви (b=10, n=4) константата K ќе има вредност $K = 10^4 - 1 = 9999$. Ако е даден бројот X=1234, неговиот комплемент до девет ќе биде $\overline{X^9} = 9999 - 1234 = 8765$. Ако станува збор за бинарниот броен систем, тогаш ќе зборуваме за комплемент до еден, т.е. за единечен (прв) комплемент кој се означува со $\overline{X^1}$, $X^{'}$ или само $\overline{X}$.

Кога константата K има вредност $K = b^n$, тогаш станува збор за *комплемент до опсегот на броеви во* бројниот систем. Така на пример, за четвороцифрени децимални броеви (b=10 , n=4), константата K ќе има вредност $K = 10^4 = 10000$, па комплементот до 10 на бројот X=1234 ќе биде $\overline{X^{10}} = 10000 - 1234 \ = 10000 - 1234 = 8766$. Ако се работи за бинарниот броен систем, тогаш комплементот е до два, т.е. втор (двоен) комплемент кој се означува со $\overline{X^2}, X^{''}$.

Споредувајќи ги вредностите на константите K за комплемент до најголемиот број $K = b^n - 1$ и комплемент до опсегот на броеви $K = b^n$, за втората константа можеме да напишеме дека $K = (b^n - 1) + 1$, што значи дека вториот комплемент на бројот може да се добие од првиот ако на првиот комплемент му се додаде 1-ца.

Комплементарните вредности на броевите, како што ќе видиме подоцна, се од фундаментално значење во дигиталната обработка на податоци. Имено, со нив е овозможено претставувањето на броевите со знак (негативни и позитивни броеви), а со тоа и извршувањето на основните аритметички операции собирање и одземање.

1.7.1. КОНВЕРЗИЈА НА БРОЕВИ ОД БИЛО КОЈ ВО ДЕКАДЕН БРОЕН СИСТЕМ

Најнапред ќе стане збор за декадниот броен систем на кој луѓето се навикнати и кој е во секојдневна употреба. Овој броен систем има десет различни цифри, а тоа се: c={0,1,2,3,4,1,6,7,8,9} и основа b=10. Примената на тежинската формула ќе ја објасниме на бројот $6013_{(10)}$.

Применувајќи ја формулата добиваме:

$$5387_{(10)} = 5 \cdot 10^3 + 3 \cdot 10^2 + 8 \cdot 10^1 + 7 \cdot 10^0 = 5000 + 300 + 80 + 7 = 5387_{(10)}$$

Од решението се забележува дека бројот запишан во декаден броен систем има иста вредност и според тежинската формула (1-3) затоа што таа формула ја дава вредноста на бројот токму во декадниот систем кој е разбирлив за човекот.

Хексадецималниот броен систем располага со 16 различни цифри кои се означуваат со: c={0,1,2,3,4,1,6,7,8,9,A,B,C,D,E,F} и основа b=16. Овој систем има на располагање шест дополнителни цифри за означување на шест броеви од декадниот систем. Имено, со цифрата A се означува бројот $10_{(10)}$, со B бројот $11_{(10)}$, со C бројот $12_{(10)}$, со D е $13_{(10)}$, E е $14_{(10)}$ и со цифрата F бројот $15_{(10)}$.

Пресметувањето на вредноста, или конверзијата во декаден броен систем - што е исто, на еден хексадецимален број е даден со примерот што следува. Повторно се применува тежинската формула така што се добива:

$$A2B_{(16)} = A \cdot 16^2 + 2 \cdot 16^1 + B \cdot 16^0 = 10 \cdot 16^2 + 2 \cdot 16^1 + 11 \cdot 16^0 = 2560 + 32 + 11 = 2603_{(10)}$$

Окталниот броен систем има 8 цифри: c={0,1,2,3,4,1,6,7} и основа b = 8. Сега тежината на секоја цифра ќе биде степен на бројот 8, бидејќи тоа е неговата основа. Претворувањето на октален број во декаден повторно ќе го разгледаме со еден пример, а тоа е бројот $157_{(8)}$.

$$157_{(8)} = 1 \cdot 8^2 + 5 \cdot 8^1 + 7 \cdot 8^0 = 64 + 40 + 7 = 111_{(10)}.$$

Декаден	Хексадецимален	Бинарен	Октален
0	**0**	**0 0 0 0**	**0**
1	**1**	**0 0 0 1**	**1**
2	**2**	**0 0 1 0**	**2**
3	**3**	**0 0 1 1**	**3**
4	**4**	**0 1 0 0**	**4**
1	**1**	**0 1 0 1**	**1**
6	**6**	**0 1 1 0**	**6**
7	**7**	**0 1 1 1**	**7**
8	**8**	**1 0 0 0**	**10**
9	**9**	**1 0 0 1**	**11**
10	**A**	**1 0 1 0**	**12**
11	**B**	**1 0 1 1**	**13**
12	**C**	**1 1 0 0**	**14**
13	**D**	**1 1 0 1**	**11**
14	**E**	**1 1 1 0**	**16**
15	**F**	**1 1 1 1**	**17**

Таб. 1-1. Преглед на цифрите на различни бројни системи

Природниот бинарен броен систем има само две цифри: 0 и 1. Било која од овие две цифри се нарекува *бит бидејќи* претставува кратенка од англискиот збор *BInary digiT*, што во превод значи **бинарна цифра**. Во бинарниот систем цифрата со најголема тежина се означува како *MSB (Most Significant Bit)*, што значи *најзначаен* или *најмногу значаен бит*, а битот со најмала тежина се означува со *LSB (Last Significant Bit)*, што значи *најнезначаен* или *најмалку значаен бит*. Преминот од бинарниот во декадниот систем, повторно оди преку тежинската формула (1-3), што е прикажано со конверзијата на бинарниот број $10010010_{(2)}$:

$$10010011_{(2)} = 1\cdot 2^7 + 0\cdot 2^6 + 0\cdot 2^5 + 1\cdot 2^4 + 0\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 128 + 16 + 2 + 1 = 147_{(10)}$$

Во таб. 1-1 се дадени ознаките и еквивалентните вредности на сите цифри од различните бројни системи.

1.7.2. КОНВЕРЗИЈА ОД БИНАРЕН ВО ХЕКСАДЕЦИМАЛЕН И ОКТАЛЕН БРОЕН СИСТЕМ И ОБРАТНО

Претворувањето од хекса во бинарен броен систем, и обратно, многу лесно се извршува бидејќи $16 = 2^4$ и тоа со директна примена на таблицата т.1-1. При конверзијата од хексадецимален во бинарен систем секоја хекса-цифра наједноставно се заменува со соодветниот нибл (четворка од битови), како што е дадено со следниов пример, при што ако се појават водечки нули на најлевите позиции, тие наједноставно се занемаруваат бидејќи немаат тежини.

$$5C_{(16)} = 0101\ 1100 = 01011100_{(2)} = 1011100_{(2)}$$

При обратна задача, кога треба да се направи конверзија од бинарен во хекса-систем зададениот бинарен број се дели на групи од по четири бита и тоа лево и десно од позиционата точка. Ако при оваа поделба најлево и најдесно не се добијат четворки тогаш, на најлевата група однапред, а на најдесната група битови одназад им се дополнуваат онолку 0 колку што е потребно за да се добијат четворки. Бидејќи разгледуваме само *цели броеви* формираме четворки од лево на десно. Ако на крај не се добие *нибл*, тогаш напред се додаваат 0-и колку што треба за да се формира четворка од битови. Потоа секој нибл се заменува со соодветната хекса-цифра според таблицата т.1-1. Во продолжение е дадено претворување на бинарниот број $101010_{(2)}$ во хексадецимален:

$$101010_{(2)} = 0010\cdot 1010 = 2A_{(16)}$$

Конверзијата од октален во бинарен, и обратно, е идентична на постапката која се користеше при конверзија хекса-бинарен, само што сега се работи со тројки од битови бидејќи $8 = 2^3$. Ова ќе биде илустрирано со следниве примери.

$$421_{(8)} = 100\cdot 010\cdot 001 = 110010001_{(2)}$$

$$10110011_{(2)} = 010\cdot 110\cdot 011 = 263_{(8)}$$

Конверзијата од октален во хексадецимален броен систем, и обратно, најлесно се изведува ако бројот зададен во едниот броен систем прво се конвертира во бинарниот броен систем, а потоа од бинарниот броен систем се конвертира во другиот броен систем, како што е дадено во примерите што следуваат.

$$BC_{(16)} = 1011\cdot 1100 = 10111100_{(2)} = 010\cdot 111\cdot 100 = 274_{(8)}$$

$$762_{(8)} = 111\cdot 110\cdot 010 = 111110010_{(2)} = 0001\cdot 1111\cdot 0010 = 1F2_{(16)}$$

Заради едноставната конверзија „долгите“ бинарни броеви обично скратено се пишуваат во хексадецимална нотација. Ова доаѓа оттаму што во компјутерот се разменуваат податоци, составени од еден, два, четири или повеќе бајти. Накратко да се потсетиме дека еден бајт претставува една осморка, т.е. низа - група од осум бита. Ова значи дека податоците најчесто ќе се со должина од 8, 16, 32, 64 итн. бита кои должини се деливи со четири, така што е најлесно да се прави конверзија од бинарен во хексадецимален броен систем, и обратно.

Окталното означување поретко се користи. Тоа би било згодно за употреба кај тие сметачки машини што користат податоци претставени со битови, чија должина е делива со три, како на пример, групи со должина од 12 или 24 бита. Но, и овие должини се деливи со 4, па јасно е дека и во овој случај може да се работи со хексадецимално означување.

1.7.3. КОНВЕРЗИЈА НА БРОЕВИ ОД ДЕКАДЕН ВО БИЛО КОЈ БРОЕН СИСТЕМ

Претворувањето од декаден броен систем во било кој друг броен систем се врши на тој начин што зададениот декаден број го делиме со основата на бројниот систем во кој сакаме да извршиме конверзија. При ова делење се добива некој количник што се запишува десно од зададениот број и одреден остаток кој се запишува под количникот. Сега овој прв количник понатаму се дели со основата и пак се добива некој друг количник што повторно се запишува во десно и втор остаток кој пак се пишува под добиениот количник итн. Процедурата продолжува сé додека не се добие количник 0, без оглед на остатокот што ќе се добие со него.

Цифрите од бројот, всушност, се остатоците од секое поединечно делење, што значи дека последниот остаток ќе биде најзначајната цифра од бројот (*MSD)*, додека првиот добиен остаток ќе биде цифрата со најмала тежина (*LSD)*. Поедноставно кажано цифрите на бројот се добиваат кога остатоците ќе се запишат обратно, т.е. оддесно налево.

Кажаното ќе го илустрираме со разгледување на примерите за конверзија од декаден во хексадецимален, октален и бинарен броен систем, последователно и тоа: $15797_{(10)}$ во хекса ќе биде $3DB1_{(16)}$, $3336_{(10)}$ во октален ќе се напише како $6410_{(8)}$ и $155_{(10)}$ во бинарен ќе биде $10011011_{(2)}$.

Декаден бр./16	11797	987	61	3	0
Остаток:		1	11	13	3
			B	D	
Хексадецимален број:		3	D	B	1

Декаден бр./8	3336	417	12	6	0
Остаток:		0	1	4	6
Октален број:		6	4	1	0

Декаден бр./2	155	77	38	19	9	4	2	1	0
Остаток:		1	1	0	1	1	0	0	1

Бинарен број: 1 0 0 1 1 0 1 1

Бидејќи претворувањето од декаден во бинарен систем ќе треба да го правиме прилично често, во продолжение ќе презентираме и една побрза постапка која ќе ја примениме на претходно зададениот пример. Најнапред ќе претпоставиме дека на располагање имаме мемориски збор со должина од повеќе битови и дека над секој бит е запишана неговата тежина. Сега гледаме дали дадениот број одговара на некоја од напишаните тежини, или од која е помал. За конкретниов пример 155 < 256 што значи дека тежината 128 е најголема и ја маркираме со 1. Ова е битот b7 кој за бројот 155 во бинарен облик ќе има најголема тежина (MSB) што укажува дека при конверзијата од десетте бита ќе искористиме само 8 бита (1 бајт), додека другите битови со поголеми тежини не се потребни; тие ќе бидат 0-и. Сега, почнувајќи од таа позиција па надолу, вршиме собирање на 128 со следната тежина, а тоа е 64. Бидејќи (128+64) > 155, оваа тежина не ја земаме предвид, односно на таа позиција запишуваме бит 0. Продолжуваме понатаму со тежината 32. Бидејќи и збирот (128+32=)160 > 155, и овде запишуваме 0. Продолжуваме со 16. Бидеќи (128+16=)144 < 155 под тежината 16 запишуваме 1 и одиме понатаму со тестирање, но сега имаме вредност 144 и следна тежина 8. Бидејќи (144+8=)152 < 155, го земаме и тој бит. Сега тестираме 152+4 што е поголемо од 155 и битот под тежината 4 не го земеме, но затоа точно ни фалат уште 3 за што ќе ги земеме последните два бита кои имаат тежина 2 и 1 со што конечно ќе добиеме 152+2+1=155. Кога пред нас се наоѓа ваква таблица со бинарните тежини претворувањето од декаден во бинарен систем е доста побрзо отколку со претходната метода кога требаше да делиме со остатоци и потоа тие да ги завртиме за да го добиеме бараниот бинарен број.

Тежини:	**1024** 2^{10}	**512** 2^{9}	**256** 2^{8}	**128** 2^{7}	**64** 2^{6}	**32** 2^{5}	**16** 2^{4}	**8** 2^{3}	**4** 2^{2}	**2** 2^{1}	**1** 2^{0}
Бинарен број:				**1**	**0**	**0**	**1**	**1**	**0**	**1**	**1**
Позиции:	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Битови:	11	10	9	8	7	6	5	4	3	2	1

1.7.4. АРИТМЕТИКА ВО БИНАРНИОТ БРОЕН СИСТЕМ

Бинарната аритметика ги дефинира правилата според кои се извршуваат операциите собирање, одземање, множење и делење во бинарниот броен систем.

Собирање. Правилата според кои се врши *собирањето* се следниве:

1. збир на две 0-и дава 0;
2. збир на 0 и 1, или 1 и 0 дава 1;
3. збир на две 1-и како резултат дава 0, но и 1 пренос (*carry*) кон битот со поголема тежина (кон повисокото ниво, класа).

Ова најдобро ќе се разбере од следниот пример за собирање на броевите $101111_{(2)}$ и $111_{(2)}$.

Преноси:		1	1	1	1	
Прв собирок:	1	0	1	1	1	1
Втор собирок:	0	0	0	1	1	1
Збир:	1	1	0	1	1	0

Одземање. Правилата за *одземање во бинарниот систем* се следниве:

1. Ако од 0 се одзема 0, или од 1 се одзема 1, резултатот е 0;
2. Ако од 1 се одзема 0, како резултат се добива 1;
3. Од 0 може да се одземе 1 така што ќе се позајми 1 од битот што има поголема тежина. Таму ќе остане 0, а во битот со помала тежина се префрлува $2_{(10)}$ т.е $10_{(2)}$. Сега од $2_{(10)}$ т.е од $10_{(2)}$ што е префрлена на пониското ниво се одзема 1-та, и една 1-а останува, и како резултат се добива 1.

Повторно ќе разгледаме еден пример. Станува збор за одземање на бројот $11_{(2)}$ од $10110_{(2)}$

Позајмувања:				10	
			0	~~0~~	10
Намаленик:	1	0	~~1~~	~~1~~	~~0~~
Намалител:				1	1
Разлика:	1	0	0	1	1

Множење. Правилата по кои се врши бинарното *множење* се следниве:

1. Кога еден од множителите е 0 резултатот е 0;
2. Само ако и двата множители се 1, резултатот ќе биде 1.

При множењето на два бинарни броја се користи принципот од декадното множење: прво се множи LSB од множителот со дадениот број (множеникот) и се добива првиот парцијален производ, потоа множеникот се множи со првиот бит лево од LSB на множителот и добиениот втор парцијален производ се запишува под првиот, но при тоа поместен во лево за едно место итн. Постапката продолжува сè додека не се добие последниот парцијален производ како резултат од множењето на MSB од множителот со зададениот број (множеникот). Еден пример за множење е даден во продолжение.

Множеник:				1	1	0	1
Множител:					1	0	1
Прв парц. производ:				1	1	0	1
Втор парц. производ:			0	0	0	0	
Трет парц. производ:		1	1	0	1		
Производ:	1	0	0	0	0	0	1

Делење. За *делењето* важат следниве правила:

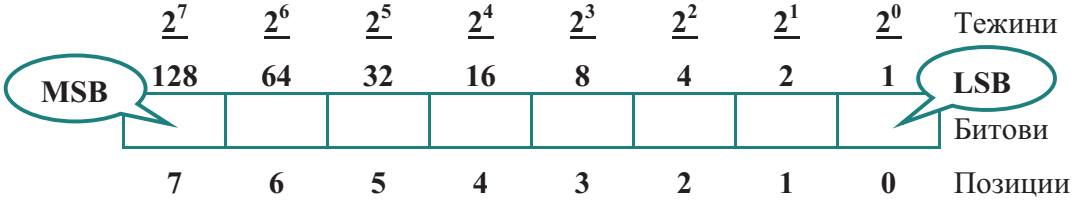
1. делењето со 0 не е дефинирано,
2. ако 0 се подели со 1 количникот е 0, и
3. ако 1 се подели со 1 се добива 1.

Бинарното делење е слично на декадното со тоа што меѓу операциите множење и одземање се полесни бидејќи се изведуваат во бинарниот систем, а цифрите од количникот се добиваат многу лесно затоа што тие можат да се или 1 или 0. Следува еден пример за делење на два бинарни броеви: деленикот е 11110, додека делителот 110.

```
  11110 : 110 = 101
- 110
-----------------
 ==110
   - 110
   -------------
     = = =
```

1.7.5. ОЗНАЧУВАЊЕ НА ПОЗИТИВНИ И НЕГАТИВНИ БРОЕВИ

Во досегашното излагање претпоставувавме дека податоците внесени во компјутерот можеа да бидат само позитивни цели броеви и евентуално нула. Имено, ако претпоставиме дека податоците се претставуваат како зборови чија должина е 1 бајт (8 бита) како што е прикажано на сл.1-6, седмиот бит ќе има најголема тежина 2^7=128 и тој ќе биде MSB, додека нултиот бит ќе има најмала тежина 2^0=1 и тој е LSB. Ако сите 8 бита од зборот бидат употребени за кодирање во природниот бинарен броен систем тогаш применувајќи ја равенката (1-1), со нив можеме да означиме вкупно $2^8 = 256$ броеви, и тоа во бинарен облик почнувајќи од најмалиот број $00000000_{(2)}$ до најголемиот $11111111_{(2)}$, односно во децималната нотација од $0_{(10)}$ до $255_{(10)}$.



Сл. 1-7. Мемориски збор со должина од 1 бајт според природниот бинарен броен систем

Меѓутоа, при обработката на информации, покрај работата со целите позитивни броеви секако треба да се имаат во вид и негативните броеви, така што нивните вредности мора да бидат посебно означени. Во секојдневната работа, во вообичаената аритметика, позитивните броеви се означуваат така што пред нив стои знакот „+”, или истиот се испушта, додека пред секој негативен број стои знакот „-” . Затоа што сметачите работат само со броеви што се запишани во бинарниот облик, се воведува посебен бит за знак. Овој бит вообичаено се наоѓа на најлевата позиција во бинарниот вектор при што знакот „+” се заменува со битот „0”, додека знакот „–” со битот „1”.

Позитивните броеви се означуваат на ист начин во сите системи. Кај нив предзнакот се означува со запишување на битот 0 на највисокото позиционо место, исто како во природниот бинарен броен систем.

Пр. 1. $69_{(10)}=1000101_{(2)} \Rightarrow (+\,69) = 0\ 1000101 = 01000101$

Негативните бројни вредности се претставуваат на три различни начини: со SM систем (анг. *sign and magnitude*) во кој се менува само битот за знак, со DC систем (анг. *digit complement*) или т.н. прв комплемент (комплементот до единица, 1's) и со RC систем (анг. *range complement)* или т.н. втор комплемент (комплемент до двојка, 2's).

SM систем: Кај овој систем на *запишување со предзнак, или т.н. систем на знак и вредност* првиот бит од бројот го дава знакот на бројот, додека останатите битови се вредносни (тежински). Имено, овие тежински битови ја претставуваат апсолутната вредност на негативниот број во природниот бинарен броен систем. Оваа нотација најлесно ќе ја сфатиме ако разгледаме два примери за тоа на кој начин се пишуваат броевите +6, -9, +13 и -13 во SM системот:

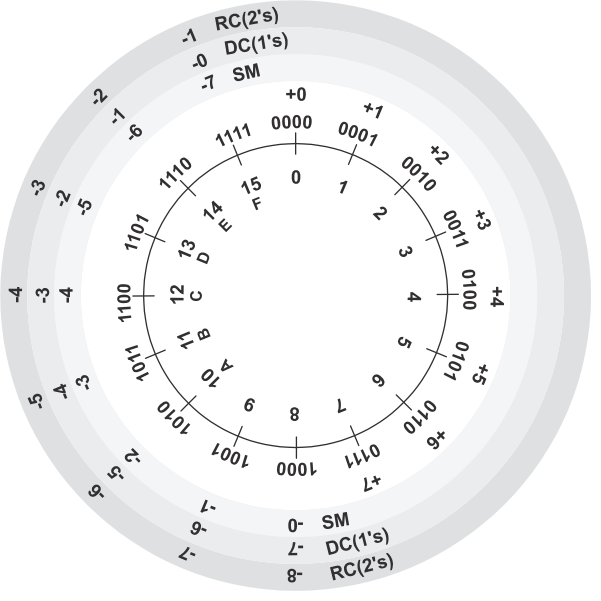
Пр. 2. Бидејќи $4_{(10)}=100_{(2)} \Rightarrow (+4) = 0\ 100 = 0100_{(SM)}$, $(-4) = 1\ 100 = 1100_{(SM)}$;

Пр. 3. Бидејќи $5_{(10)}=101_{(2)} \Rightarrow (+5) = 0\ 101 = 0101_{(SM)}$, $(-5) = 1\ 101 = 1101_{(SM)}$;

Од примерите гледаме дека негативниот предзнак наједноставно се заменува со 1 и декадната вредност со бинарна.

ТС системи: Изразувањето на негативните вредности на броевите најчесто се претставува со примена на комплементарни бинарни броеви. Ова е од причини што со ваквото комплементарно означување на негативните броеви кога се извршуваат аритметички операции со битот за предзнак се манипулира на ист начин како и со вредносните (тежинските) битови. Постојат два комплементарни системи: DC систем или т.н. прв комплемент (1's) и RC систем или т.н. втор комплемент (2's).

Со кружната презентација од сл.1-8 и таб.1-2 се претставени начините на прикажување на бројните вредности кодирани со по четири бита.



Сл. 1-8. Презентација на броеви со предзнак кодирани со 4 бита во различни системи

Позитивни броеви					Негативни броеви													
SM, DC, RC				Вр	SM				Вр	DC (1's)				Вр	RC (2's)				Вр
0	0	0	0	+0	1	0	0	0	-0	1	0	0	0	-7	1	0	0	0	-8
0	0	0	1	+1	1	0	0	1	-1	1	0	0	1	-6	1	0	0	1	-7
0	0	1	0	+2	1	0	1	0	-2	1	0	1	0	-5	1	0	1	0	-6
0	0	1	1	+3	1	0	1	1	-3	1	0	1	1	-4	1	0	1	1	-5
0	1	0	0	+4	1	1	0	0	-4	1	1	0	0	-3	1	1	0	0	-4
0	1	0	1	+5	1	1	0	1	-5	1	1	0	1	-2	1	1	0	1	-3
0	1	1	0	+6	1	1	1	0	-6	1	1	1	0	-1	1	1	1	0	-2
0	1	1	1	+7	1	1	1	1	-7	1	1	1	1	-0	1	1	1	1	-1

Таб. 1-2. Презентација на броеви со предзнак кодирани со 4 бита во различни системи

Од прикажаната табела забележуваме дека позитивните броеви се пишуваат на ист начин во било која нотација, потоа дека битот со најголема тежина (MSB) за негативните броеви има вредност 1 без оглед на тоа кое означување се користи. Понатаму, од табелата се гледа и тоа дека при пишувањето на броевите со предзнак или со првиот комплемент, нулата може да се изрази на два начина: како позитивна и како негативна, при што бројот на претставените негативни и позитивни броеви е еднаков. Кај SM и DC нотацијата постојат две нули: позитивна и негативна, додека RC означувањето се разликува од нив бидејќи постои само една нулта вредност која се третира како позитивна со што се добива еден негативен број повеќе од позитивните.

DC систем: Како еден пример ќе го разгледаме начинот на изразување на негативната вредност на бројот $6_{(10)} = 110_{(2)}$, т.е. $-6_{(10)}$. Според DC системот претставувањето е прилично лесно, бидејќи прво се одредува апсолутната вредност на бројот, потоа истата се запишува како бинарен број и на крај поединечно се комплементира бит по бит од добиениот бинарен број, како што е презентирано со следниот пример.

Даден негативен број: – 6
Апсолутна вредност: 6
Бинарен еквивалент: 0110
Комплементирање: $1001_{(1's)}$ т.е. – 6 кодиран во DC систем.

Конверзијата на негативните броеви од прв комплемент во декаден систем се изведува на истиот начин. Најнапред поединечно се комплементира секој бит на даденият бинарен вектор, потоа се одредува декадната вредност на новодобиената бинарна комбинација и на крај се додава знакот "–". Во примерот што следи е претпоставено дека треба да се одреди декадната вредност на бинарниот вектор $11010001_{(1's)}$ со претпоставка дека станува збор за DC систем.

Даден бинарен збор: 11010001
Комплементирање (1's): 00101110
Апсолутна вредност: 00101110 = 46
Декаден број: – 46

Конверзијата на негативните броеви од прв комплемент во декаден систем може да се изврши побрзо и поедноставно. Имено, се собираат тежините на оние позиции (места) во бројот каде што се наоѓаат 0-те и се додава знакот "–". За претходниот пример на бројот 11010001 ќе добиеме:

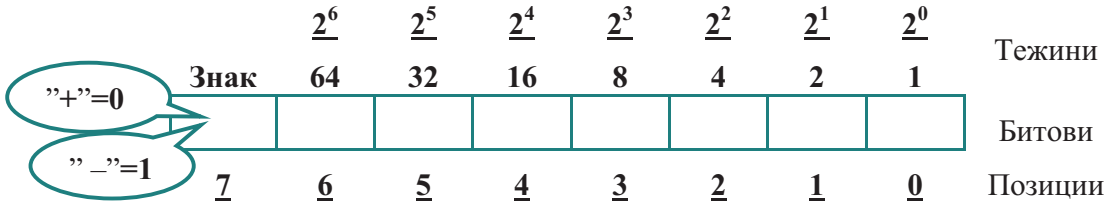
$2^5 + 2^3 + 2^2 + 2^1 = 32 + 8 + 4 + 2 = 46$, т.е. $11010001_{(1's)} = -46$.

Во практиката се користат сите три начина, меѓутоа најважно е означувањето во двоен комплемент, бидејќи со него на наједноставен начин може да се манипулира со аритметичките операции собирање и одземање на цели броеви со знак, поради што во продолжение ќе му биде посветено посебно внимание.

1.7.6. ОЗНАЧУВАЊЕ СО ДВОЕН КОМПЛЕМЕНТ

Ова означување изворно на анг. се вика *two's compliment notation*, а скратено се означува со 2's *complement*, или покрај бројот како индекс се додава (2's). Оваа нотација најчесто се користи во оние случаи кога се работи со цели броеви со знак.

И при претставувањето на броевите во двоен комплемент најважно е тоа што се применува бит за знак, а тоа е битот на највисоката седма позиција, т.е. осмиот бит. Ако вредноста на овој бит е 0, тогаш бројот е позитивен, но ако овој бит е 1, тогаш бројот е негативен, како што може и да се види на сл.1-9.



Сл. 1-9. Бајт за претставување на двоен комплемент

И во овој случај, според рав. (1-1) повторно можат да се напишат $2^8 = 256$ различни броеви, но 128 од нив ќе бидат позитивни, а 128 негативни. Најголем позитивен број во двоен комплемент е 01111111, што одговара на +127 декадно, додека најмал број е 00000000 што одговара на 0 декадно што покажува дека нулата се третира како позитивен број. Земајќи го предвид кажаното и последната констатација, може интуитивно да се претпостави дека најголемиот негативен број ќе биде -1, а најмалиот -128.

За претставувањето на негативните броеви ќе продискутираме во продолжение. Конверзијата на било кој зададен цел негативен декаден број во двоен комплемент облик може да се изврши со примена на следниве чекори:

1. Декадно се запишува апсолутната вредност на бројот;

2. Се врши конверзија од декаден во бинарен број;

3. Се комплементира секој бит поединечно, со што се добива единечен комплемент на бинарниот број (1's), и

4. Се додава (се зголемува) за 1 добиениот прв комплемент на бројот.

Добиениот резултат претставува нотација во двоен комплемент (2's) на зададениот негативен број. Во продолжение е илустрирана конверзијата на негативниот број $-6_{(10)}$ во 2's-комплемент облик со претпоставка дека компјутерот работи со податоци долги 1 бајт.

Декаден број:	– 6
Апсолутна вредност:	6
Бинарен број:	00000110
Прв комплемент 1's:	11111001
Зголемување за 1:	+ 1

Втор комплемент (2's):	11111010

Ако се постави обратен проблем, а тоа е наоѓање на еквивалент на децимална вредност на некој зададен негативен број во двоен комплемент, тогаш треба да се изведе истата постапка како и претходно:

1. Дадениот бинарен 2's број се комплементира бит по бит со што се добива неговата единечна комплемент нотација;

2. Добиениот единечен комплемент на бројот се зголемува за 1;

3. Се врши конверзија на добиениот бинарен број во декаден броен систем и се додава знакот „–" (минус).

Со следниов пример се изведува конверзија на бинарниот вектор $11110001_{(2's)}$ во декаден броен систем.

Втор комплемент на бројот (2's):	11110001
Прв комплемент (1's):	00001110
Зголемување за 1:	+ 1

Апсолутна вредност:	00001111 = 15
Декаден број:	– 15

Претворувањето на негативните броеви од двоен комплемент во декаден систем може да се изврши и поедноставно. Имено, се собираат тежините на оние позиции (места) во бројот каде што се наоѓаат 0-те, потоа се додава 1-а, и на крај знакот "–". За претходниот пример, за бројот 11110001 би имале:

$(2^3 + 2^2 + 2^1) + 1 = (8 + 4 + 2) + 1 = 15 = 16$, т.е. $11110000_{(2's)} = -16$.

Со броевите претставени во двоен комплемент многу лесно се изведува собирањето, а најважно е тоа што и одземањето се сведува на собирање. Постапката е прилично едноставна:

1. Дадените броеви претставени во двоен комплемент се собираат како да се најобични бинарни броеви (битот за знак се третира исто како и другите битови);

2. Ако при тоа се појави префрлување (пречекорување, *owerflow*), т.е. пренос по осмиот бит (по седмата позиција) тој пренос едноставно се занемарува, а останатите осум бита од резултатот го даваат решението и

3. Ако не постои префрлување добиениот збир е бараното решение.

Примерите што следуваат ја илустрираат постапката за собирање и одземање на броеви со знак. Првиот пример е за собирање на 5 со 3, додека вториот е за одземање на 6 од 2, т.е. (+2) + (–6) што резултира со – 4.

00000101	5	00000010	2
+ 00000011	+ 3	+ 11111010	– 6
---	---	---	---
00001000	+ 8	11111100	– 4

1.8. БИНАРНИ КОДОВИ

Под поимот **кодирање** *се подразбира начин на претставување на информациите со помош на симболи што се елементи на некое множество*. Дигиталните системи содржат електронски прекинувачки елементи кои може да се наоѓаат само во две состојби, така што за нив најпогоден начин на претставување е *бинарниот облик*. Причината за ова е едноставна. Имено, било кој податок, напишан во бинарен облик претставува една низа од битовите 0 и 1.

Аналогно на националната азбука, множеството на сите различни симболи кои стојат на располагање за писмено изразување на информациите во одреден код се вика **кодна азбука**. Кодната азбука за бинарните кодови е **бинарното множество** кое има само два елементи: *логичка нула* (*бит 0*) и *логичка единица* (*бит 1*). Групата на симболи со кои се претставува, т.е. кодира некој поим или информација се вика **коден збор**.

Кодниот збор има своја **должина**, а тоа е вкупниот број на симболи со кои тој е напишан. Ние најчесто ќе употребуваме кодови кај кои сите кодни зборови имаат иста должина. Ваквите кодови се викаат *рамномерни кодови*. Бинарните кодови обично оперираат со зборови чија должина е 8 бита, односно 1 бајт.

За разлика од рамномерните кодови, постојат и *нерамномерни кодови* кај кои кодните зборови имаат различна должина. Пишувањето во било кој национален јазик практично е кодирање во нерамномерен код, а истото тоа важи и за запишувањето на броевите во било кој броен систем.

Ако некој код содржи барем еден коден збор кој не означува никаква, или никаква нова информација, тогаш за него се вели дека е *редундантен код*.

Начинот според кој се кодираат и декодираат кодните зборови се изразува преку математички равенки или со помош на група правила и прописи (како што е, на пример, синтаксата за еден национален јазик), но најчесто се применуваат т.н. кодни таблици. Кодната таблица има две колони и повеќе редици. Во левата колона по редици последователно се пишуваат симболите кои треба да се кодираат. Десната колона, исто така, се пополнува по редици, но со кодните зборови што мора еднозначно да одговараат на секој изворен симбол. Кога станува збор за бинарно кодирање лево се пишуваат буквите или децималните цифри, а десно бинарните кодни зборови.

Бинарните кодови можат да бидат *тежински* или *редоследни*. Кај *тежинските кодови* комбинирањето на битовите се врши така што на секој бит од кодниот збор му се припишува одредена тежина. Еден пример за тежински код би бил веќе опишаниот природен бинарен броен систем. Меѓутоа, кодните комбинации, т.е. зборови, можат да се формираат и според некои други тежински законитости. Така се развиени голем број бинарни кодови за специјални намени. Сите кодови што не се тежински спаѓаат во групата на *редоследни кодови*. Кај нив битовите немаат соодветна тежина во кодниот збор, туку е важен само нивниот распоред во него. Кај овие кодови врската меѓу декадниот број и бинарниот коден збор најчесто се дава преку специјални кодни таблици.

1.8.1. НУМЕРИЧКИ КОДОВИ

Заради едноставната конверзија бинарните броеви најчесто се претставуваат во хексадецимална нотација. Од друга страна во калкулаторите, дигиталните инструменти итн. се јавува потреба за внесување на податоци или добивање на резултати во декаден облик. Но, веќе видовме дека конверзијата од бинарен во декаден броен систем е доста комплицирана. За да се соединат добрите особини на бинарниот и декадниот броен систем и за да се излезе во пресрет на навиката на луѓето да размислуваат декадно, развиени се различни *бинарни (BCD) кодови*. Оваа кратенка доаѓа од англискиот израз *Binary Coded Decimal*, што значи бинарно кодирани декадни броеви. Кај овие кодови секоја цифра од декадниот број одвоено се кодира со користење еднозначно определен коден збор. За да може да се кодираат сите десет декадни цифри, мора да се употребат најмалку 4 бита, бидејќи 3 не се доволни. Имено, со три бита може да се кодираат $2^3 = 8$ кодни зборови, т.е. цифри што е помало од 10, додека со 4 бита може да се добијат 2^4=16 различни комбинации што е доволно бидејќи 16 > 10. Ова укажува дека за секоја декадна цифра ќе се користи по еден нибл (група од 4 бита, тетрада). Распоредот на битовите во овие групи ќе се одвива според некоја таблица или закономерност. Бидејќи со 4 бита можат да се кодираат 16 различни тетради, очигледно е дека секогаш 6 кодни зборови ќе останат неискористени за кодирање. Од изнесеното произлегува дека овие кодови се редундантни. Токму заради тоа постои можност за кодирање на декадните броеви со различни бинарни кодови (теоретски со 16!/6! или приближно околу $29x10^9$).

Декад. цифра	8421 (NBCD)	2421	Ајкенов	Вишок_3	5421	Декад. цифра
0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 1 1	0 0 0 0	0
1	0 0 0 1	0 0 0 1	0 0 0 1	0 1 0 0	0 0 0 1	1
2	0 0 1 0	0 0 1 0	0 0 1 0	0 1 0 1	0 0 1 0	2
3	0 0 1 1	0 0 1 1	0 0 1 1	0 1 1 0	0 0 1 1	3
4	0 1 0 0	0 1 0 0	0 1 0 0	0 1 1 1	0 1 0 0	4
5	0 1 0 1	1 0 1 1	1 0 1 1	1 0 0 0	1 0 0 0	5
6	0 1 1 0	1 1 0 0	1 1 0 0	1 0 0 1	1 0 0 1	6
7	0 1 1 1	1 1 0 1	1 1 0 1	1 0 1 0	1 0 1 0	7
8	1 0 0 0	1 1 1 0	1 1 1 0	1 0 1 1	1 0 1 1	8
9	1 0 0 1	1 1 1 1	1 1 1 1	1 1 0 0	1 1 0 0	9

Таб. 1-3. Кодни таблици на различни бинарни (BCD) кодови

Табелата таб.1-3 прикажува неколку бинарни (BCD) кодови кои често се применуваат во праксата. Најмногу користен е т.н. 8421 BCD код, што е познат и како *природен BCD или NBCD код*. Секоја декадна цифра се кодира со еднозначно одреден коден збор што има должина еден нибл. Почетната ознака 8421 во името на кодот се однесува на вредностите на тежините за секој од четирите битови во кодниот збор.

Со примерите што се дадени во продолжение е илустриран начинот на кодирање и декодирање во *NBCD*-кодот.

Пр. 1. $7694_{(10)} = 0111 \cdot 0110 \cdot 1001 \cdot 0100_{(NBCD)} = 0111011010010100_{(NBCD)}$

Пр. 2. $001101010010_{(NBCD)} = 0011 \cdot 0101 \cdot 0010_{(NBCD)} = 352_{(10)}$

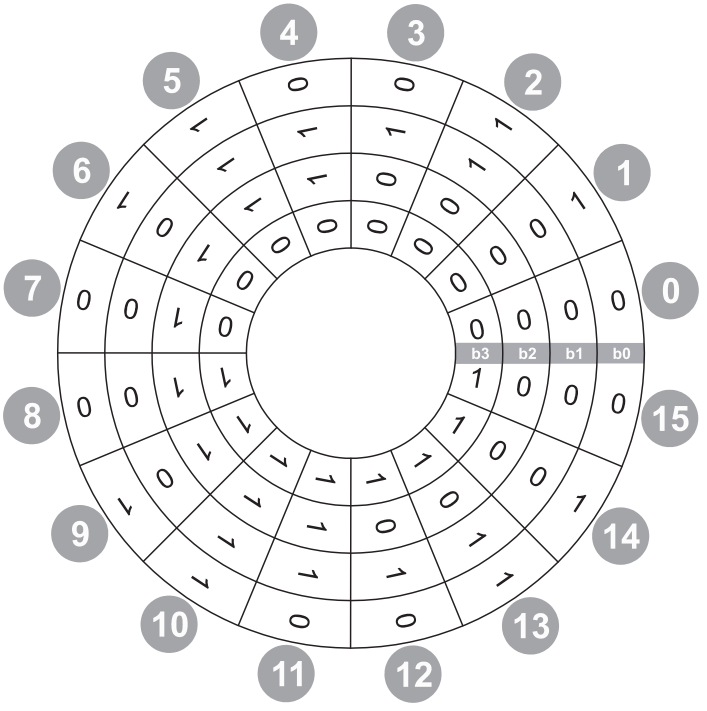
Од примерите се забележува дека принципот на работа се сведува на принципот според кој се врши конверзија од хексадецимален во бинарен броен систем, и обратно.

Покрај 8421 BCD кодот, постојат и други BCD кодови. Попознати се: *Грејовиот код*, *Ајкеновиот код*, *Вишок-3 кодот*, потоа 5421, *поместувачкиот код*, *2421,* итн. Некои од нив се дадени во таб. 1-3. Кодирањето во овие кодови се врши според истиот принцип како и за 8421 (*NBCD)* BCD кодот, при што разликата се јавува во редоследот на битовите во секоја од кодните комбинации.

Грејов код. На овој код ќе посветиме малку повеќе внимание бидејќи има голема примена во различни домени, како на пр. во преносот на дигитални сигнали за минимизирање на појавата на грешки, потоа кај уредите за кои е важно прикажување на аголното отстапување во бинарен облик, како што се дисковите, потоа за минимизирање на логичките функции со методот на Карноови карти (за што ќе зборуваме во следната тема) итн. Главна карактеристика на Грејовиот код е во тоа што соседните кодни зборови се разликуваат само во еден бит. Таб. 1-4 и сл. 1-10 го прикажуваат начинот на кодирање според Грејовиот код со четири бита. Заради лесно воочување на кодните комбинации на Грејовиот код и за два и за три бита во таб. 1-4 тие се означени со сенка.

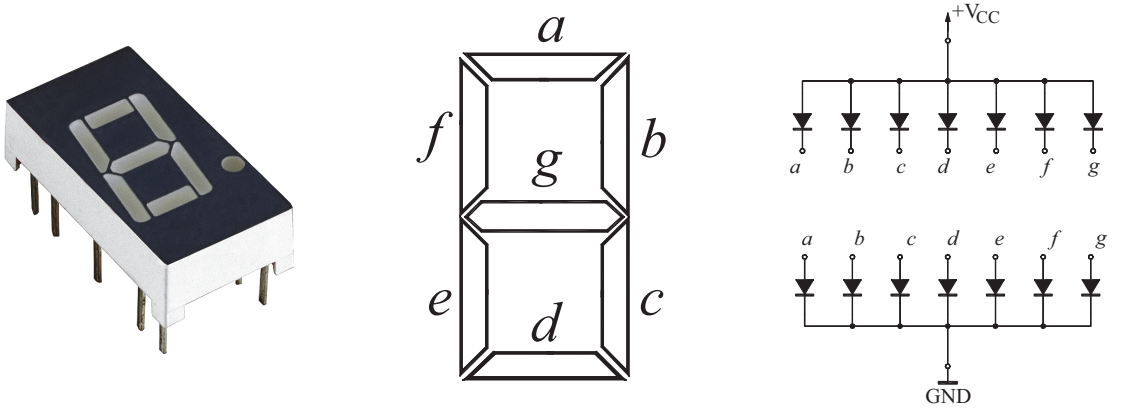
ДЦ	b_3	b_2	b_1	b_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1
10	1	1	1	1
11	1	1	1	0
12	1	0	1	0
13	1	0	1	1
14	1	0	0	1
15	1	0	0	0

Таб. 1-4. Грејова таблица



Сл. 1-10. Грејов коден круг

Седумсегментен код. На крај ќе го споменеме *седумсегментниот нумерички систем* кој е создаден од чисто практични причини заради човековата потреба лесно да може да ги чита бројните вредности. Овој код се однесува на дигиталните уреди кои резултатите ги покажуваат преку индикатори (мали екрани) со светлечки (LED) диоди. На сл. 1-11 прикажан е еден таков екран. Тој е составен од седум сегменти означени со буквите a, b, c, d, e, f, g. Секој поединечен сегмент од индикаторот може да свети или да не свети, што значи дека секој сегмент може да се претстави со по еден бит. Секоја декадна цифра може да се формира со комбинација од поединечни сегменти кои светат. Така седумсегментниот систем користи десет кодни збора од по седум бита, при што секоја поединечна комбинација претставува една декадна цифра.



Сл. 1-11. Реален седумсегментен индикатор со светлечки (LED) диоди и негов симбол

Седумсегментните индикатори се произведуваат со заедничка анода (ЗА, анг. CA) или со заедничка катода (ЗК, анг. CC). Кај индикаторите со ЗК сите катоди се споени во

единствена точка која треба да се спои на земја, додека анодите се одвоени. За сегментот да светне на соодветната анода треба да се доведе високо ниво, ниво на логичка 1. Кај екраните со ЗА сите аноди се споени во заеднички пин кој треба да се врзи на напојувањето бидејќи тоа е високо ниво, додека катодите се одвоени и секоја се појавува на посебен пин. Било кој сегмент ќе светне ако на соодветната катода се доведе ниско напонско ниво, ниво на логичка 0, т.е. заземјување, маса.

Во таб. 1-5 се претставени и двата седумсегментни кодови кои меѓусебно се комплементарни. Првиот се однесува на индикатор со заедничка катода, а другиот на екран со заедничка анода.

Дек. циф.	Екран со заедничка катода						
	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

а) Екран со заедничка катода

Екран со заедничка анода							Дек. циф.
a	b	c	d	e	f	g	
0	0	0	0	0	0	1	0
1	0	0	1	1	1	1	1
0	0	1	0	0	1	0	2
0	0	0	0	1	1	0	3
1	0	0	1	1	0	0	4
0	1	0	0	1	0	0	5
0	1	0	0	0	0	0	6
0	0	0	1	1	1	1	7
0	0	0	0	0	0	0	8
0	0	0	0	1	0	0	9

а) Екран со заедничка анода

Таб. 1-5. Таблица на кодовите за седумсегментен екран со LED диоди

1.8.2. АЛФАНУМЕРИЧКИ КОДОВИ

Комуникацијата помеѓу човекот и компјутерот се изведува со помош на монитор (екран) или со посредство на печатач. При ова нормално е да се користат секакви алфанумерички (текстуални) симболи како на пр. алфабетските знаци, т.е. малите и големите букви, потоа знаците на интерпункција, одредени нумерички податоци, т.е. броеви кои што не се користат заради извршување на математички операции над нив, како што се на пр. телефонските броеви или броевите во адресите, некои посебни графички знаци, итн. Имајќи во вид дека станува збор за голем број симболи, се наметнала потребата од кодови со кодни зборови чија должина е поголема од 4 бита. Сите претходно наброени знаци се кодираат со посебни кодови кои се викаат *алфанумерички кодови*.

Најмногу се користи алфанумеричкиот код кој носи ознака *ASCII и с*е чита АСКИ. Кратенката доаѓа од анг. израз *American Standard Code for Information Interchange* што значи американски стандарден код за информациска размена. На почетокот овој код беше стандард во САД, а потоа е усвоен и како меѓународен стандард со ознака ISO-7. Табелата на стандарниот *ASCII код е* означена со таб. 1-6. Од табелата се гледа дека овој код ги опфаќа следниве симболи (анг. characters): некои посебни контролни симболи, па декадните цифри, потоа сите големи и мали букви од англиската азбука, знаците за интерпункција, некои специјални графички знаци и на крај некои математички знаци. Стандардниот *ASCII код* за кодирање користи 7 битови што значи дека со него може да се кодираат 2^7=128 различни знаци чии кодови се дадени во ASCII табела таб. 1-6.

Иако кодните зборови на стандардниот *ASCII код изворно имаат 7 бита, за нивно меморирање се применуваат мемориски зборови со должина од* 1 бајт, т.е. 8 бита кај кои крајниот лев бит е слободен, поточно во процесот на де/кодирање се зема дека неговата вредност е 0.

Битови $b_3b_2b_1b_0$	Битови $b_6b_5b_4$							
	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	1	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	`	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Таб. 1-6. Таблица на стандарден ASCII код

Кодниот збор за секој знак може да се добие ако се има предвид таблицата таб. 1-6 и ако се примени начинот на формирање на кодните зборови илустриран на сл. 1-12.

Сл. 1-12. Принцип на формирање на ASCII кодните зборови.

Прикажаната таблица таб. 1-6 на ASCII кодот ја користи само латиничната азбука која се однесува на англискиот јазик и не ги содржи буквите на нашата азбука, како на пр. Ч, ч, Ќ, ќ, Ш, ш, Ѓ, ѓ, Ж, ж, ниту пак знаците на латиничните или кириличните азбуки на другите европски земји, како на пр. Ä, ü, ö, ř, Ł, é, ... итн.

Заради оваа причина ваквата таблица е проширена со осми бит со што се добија дополнителни места за нови 128 симболи, или вкупно 256 знаци кога се работи за ASCII-8 кодот, кој целосно користи 1 бајт. Бидејќи и овие знаци не се доволни за сите знаци на азбуките на различните држави, се формираат различни таблици од една до друга земја. Така на пр. специјални знаци од македонската азбука се наоѓаат во ASCII таблицата со ознака 1211 za Windows (Cyrillic Code Page 1211 Alphabet). Во таа таблица се наоѓаат и буквите од Руската, Српската и Бугарската азбука. Земјите од централна Европа, меѓу кои спаѓаат и Словенија, Хрватска, Србија, Чешка. Полска, и сл. ја користат кодната таблица Windows Code Page Latin 1210.

Од минатото постои уште еден алфанумерички кој во сегашноста има широка примена. Тоа е *EBCDIC кодот* (*анг. Extended Binary Coded Decimal Interchange Code*), т.е. проширен бинарно-кодиран децимален код за размена, со должина на кодните зборови од 8 бита кој е воведен од страна на познатата американска компанија за производство на компјутери, IBM.

1.9. ЕКСПЛИЦИТНА И ИМПЛИЦИТНИ ВРЕДНОСТИ

Бидејќи информациите се претставуваат во бинарна форма како бинарни вектори, т.е. во облик на групи (низи) од битови со различна должина, а запишани по различен принцип: како некој бинарен број со, или без знак, или како некој коден збор според некој бинарен код, се воведува единствениот термин **збор** (*word*), со кој ќе се означува било која *група на битови со одредена должина*. Во врска со изнесеното ќе ги воведеме термините *експлицитна* и *имплицитна вредност* на зборот (податокот). Станува збор за вредности кои се разбирливи за човекот, а се добиваат со декодирање на зададен бинарен вектор. Имено, експлицитната вредност на зборот претставува цел позитивен декаден број поголем или еднаков на 0, што се добива со конверзија на битовите од зборот, ако сите битови се третираат како тежински битови запишани според природниот бинарен броен систем. Од друга страна, имплицитната вредност на зборот е онаа вредност кога битовите на зборот се декодираат или конвертираат според одреден бинарен код или бинарен систем.

За полесно разбирање на овие два поими ќе разгледаме два примери. Ќе претпоставиме дека во два бајти на меморијата на компјутерот се запамтени следниве бинарни зборови (вектори): (а) 01010100 и (б) 11010100.

Пр.1. (а) 01010100.

Експлицитна вредност = 64+16+4 = 84.

Имплицитни вредности:

- според SM системот = +(64+16+4) = + 84.
- според DC системот = +(64+16+4) = + 84.
- според RC системот = +(64+16+4) = + 84.
- според NBCD кодот = 54.
- според Вишок-3 кодот = 21.
- според ASCII кодот = T.

Пр. 2. (б) 11010100.

Експлицитна вредност = 128+64+16+4 = 212.

Имплицитни вредности:

- според SM системот = - (64+16+4) = - 84.
- според DC системот = - (32+8+2+1) = - 43.
- според RC системот = - [(32+8+2+1)+1] = - 44.
- според NBCD кодот = ?4.
- според Вишок-3 кодот = ?1.
- според ASCII кодот = ?.

Во примерите на оние места каде се појавува симболот "?" означува дека ќе се јави грешка бидејќи не постои таков коден збор.

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

1-1. Во што се разликуваат континуираните и дискретните функции?

1-2. Каков е обликот на дигиталните сигнали?

1-3. Со какви сигнали работат а) аналогните б) дигиталните уреди?

1-4. Наведи ги и објасни ги предностите, односно слабостите, на аналогниот и дигиталниот начин на работа.

1-5. Што се подразбира под поимот информација?

1-6. Со што се претставуваат информациите на апстрактно ниво и реално каде се "втиснуваат"?

1-7. Што се подразбира под поимот кодирање? Што е коден збор?

1-8. Која е основната единица за мерење количество на информација и како се дефинира во рамките на дигиталната електроника?

1-9. Покрај битот [B] во дигиталната обработка на информации се употребуваат и следниве поголеми единици: (а) Еден бајт [B]; (б) Еден килобајт [KB]; (в) Еден мегабајт [MB]; (г) Еден гигабајт [GB]; (д) Еден tera бајт [TB]. Наведи ја должината на бајтот, а потоа одговори колку бајти содржи килобајтот, мегабајтот, гигабајтот и терабајтот.

1-10. Што се подразбира под поимот податоци во поширока смисла?

1-11. Какви информации носат а) инструкциите б) податоците во потесна смисла на зборот в) адресите?

1-12. Кои се базичните градбени елементи на дигиталните уреди?

1-13. Кои се двете големи групи на кои се делат дигиталните уреди?

1-14. Што е карактеристично за а) кобинациските б) секвенцијалните мрежи?

1-15. Бинарните собирачи се ... мрежи со кои се извршува ...

1-16. Прекинувачките матрици се ... мрежи чии прекинувачки елементи ...

1-17. Кодерот спаѓа во групата на ... и негова намена е ...

1-18. Декодерот спаѓа во групата на ... и неговата намена е ...

1-19. Мултиплексерот спаѓа во групата на ... и неговата намена е ...

1-20. Демултиплексерот спаѓа во групата на ... и неговата намена е ...

1-21. Програмабилните логички структури се ... мрежи чии прекинувачки елементи формираат ... структура кај која постои можност за ...

1-22. Флип-флопот претставува ..., бидејќи во него ...

1-23. Кои се основни градбени елементи на регистрите и бројачите?

1-24. Која е основната намена на а) регистрите б) бројачите?

1-25. Мемориите се дигитални компоненти или уреди во ...

1-26. Меморијата претставува ... множество на ...

1-27. Местоположбата на било која мемориска локација се определува со ...

1-28. Секоја мемориска локација има ... должина и во неа се сместува ...

1-29. Капацитет на меморијата е ...

1-30. Колата за аналогно-дигитална конверзија вршат ...

1-31. Колата за дигитално-аналогна конверзија вршат ...

1-32. Проблемот на анализа на дигиталните уреди опфаќа ...

1-33. Проблемот на синтеза (проектирање) на дигиталните уреди опфаќа ...

1-34. Бројните системи се системи на ...

1-35. Секој тежински броен систем има своја ..., а тоа е ...

1-36. Која е основата и кои се цифрите на следните бројни системи (а) декадниот; (б) хексадецималниот; (в) окталниот; (г) бинарниот.

1-37. Нека е зададена основата на бројниот систем b и бројот на цифрите n, што стојат на располагање. Колку различни броеви N можат да се напишат во него?

1-39. Ако станува збор за (а) декадниот; (б) хексадецималниот; (в) окталниот; (г) бинарниот; броен систем при што на располагање стојат n=4 цифри, одговори (1) колку различни броеви можат да се напишат? (2) Кој е најмалиот број? (3) Кој е најголемиот број?

1-40. Како се одредува тежинската вредност на секоја цифра во бројот?

1-41. Според која формула се пресметува вредноста на некој број X, кој има n целобројни и m дробно-рационални места, запишан во било кој броен систем? Во кој броен систем е претставен даденият број?

1-42. Определи ги (а) комплементот до девет; (б) комплементот до десет, на следните декадни броеви (1) 7531; (2) 9862; (3) 41.

1-43. Определи ја вредноста, т.е. изврши конверзија во декаден броен систем, на секој од следниве броеви (а) $EE_{(16)}$; (б) $F0_{(16)}$; (в) $10_{(16)}$; (г) $CDA_{(16)}$; (д) $10_{(8)}$; (ѓ) $100_{(8)}$; (е) $77_{(8)}$; (ж) $1000_{(2)}$; (з) $1111_{(2)}$; (ѕ) $1011_{(2)}$.

1-44. Изврши конверзија на декадните броеви (а) 123; (б) 69; (в) 127; (г) 128; (д) 255; во (1) бинарен; (2) хексадецимален; (3) октален броен систем.

1-45. Изврши ги следниве конверзии помеѓу бројните системи (а) од хексадецимален и октален $14_{(16)}$ и $57_{(8)}$ во бинарен; (б) од бинарен $1010111_{(2)}$ во хексадецимален и во октален; (в) од хекса $24_{(16)}$ во октален; (г) од октален $346_{(8)}$ во хекса.

1-46. Собери ги следниве парови на бинарни броеви (а) 1111 со 1011; (б) 1011 со 1011; (в) 10111011 со 11110111.

1-47. Помножи го бинарниот број 1101 со бројот (а) 1110; (б) 1011; (в) 1101.

1-48. Изврши ги следниве одземања во бинарниот броен систем (а) од 1110 извади 1011; (б) од 10100000 извади 10000111; (в) од 10100001 извади 10001111.

1-49. Подели ги следниве парови бинарни броеви (а) 1010 со 100; (б) 10110110 со 1011; (в) 10011110 со 1100; (г) 10000011 со 1001.

1-50. Претстави ги следниве декадни броеви (а) +37; (б) 0; (в) - 37; (г) - 41 и (д) - 99 како (1) цели броеви со предзнак (SM систем); (2) броеви во нотација со единечен комплемент (DC систем, 1's). Претпостави дека податокот се запамтува како збор со должина 1 бајт (8 бита) од кои првиот бит е за предзнак.

1-51. Изврши конверзија на следниве (а) декадни броеви + 128, + 15, - 1, - 7 и - 127 во нотација со двоен комплемент (RC систем, 2's).); (б) броеви дадени во двоен комплемент 01110011, 01011101, 11000101 и 10111101 претвори ги во декадни броеви ако податоците се запишуваат во облик на бајти со еден бит за предзнак.

1-52. Изведи ги следниве операции во 2's-комплемент нотација, а потоа провери дали добиените резултати се точни (а) 14 + 23; (б) 9 - 6; (в) 8 - 1 (в) 5 - 7 (г) 14 - 35; (д) - 12 + 19; (ѓ) - 48 - 5. Претпостави дека податоците се запишуваат во облик на бајти со еден бит за предзнак.

1-53. Што се подразбира под поимот кодирање?

1-54. Што претставува кодната азбука?

1-55. Која е кодната азбука за бинарните кодови? Кои се нејзините симболи?

1-56. Што претставува кодниот збор? Во што се изразува неговата должина?

1-57. Што е карактеристично за а) рамномерните б) нерамномерните кодови?

1-58. Редундантните кодови содржат кодни зборови кои ...

1-59. За што се користи кодната таблица? Опиши го нејзиниот изглед.

1-60. Кај тежинските кодови секој бит од кодниот збор...

1-61. Кај редоследните кодови е битен ...

1-62. За бинарните (BCD) кодови карактеристично е тоа што ...

1-63. Напиши ја NBCD формата на декадните броеви (а) 18367; (б) 42509.

1-64. Кои броеви се кодирани со следниве NBCD зборови а) 10000011 б) 10011100.

1-65. Кодирај ги следниве декадни броеви 132, 645 и 7890 во (а) Грејовиот код; (б) Ајкеновиот код; (в) вишок_3 кодот; (г) BCD 5421 кодот.

1-66. Колку вкупно различни симболи може да се добијат со примена на кодот за седум-сегментниот дисплеј? Колку од нив се користат?

1-67. Каков ќе биде кодниот збор abcdefg за екран со LED диоди и заедничка (а) анода; (б) катода, ако на него треба да се прикаже цифрата (1) 7; (2) 4; (3) 9.

1-68. Што се прикажува на LED екранот со заедничка (а) катода; (б) анода, ако кодниот збор abcdefg го има следниов облик (1) 1001111; (2) 1100011.

1-69. Што е карактеристично за алфанумеричките кодови?

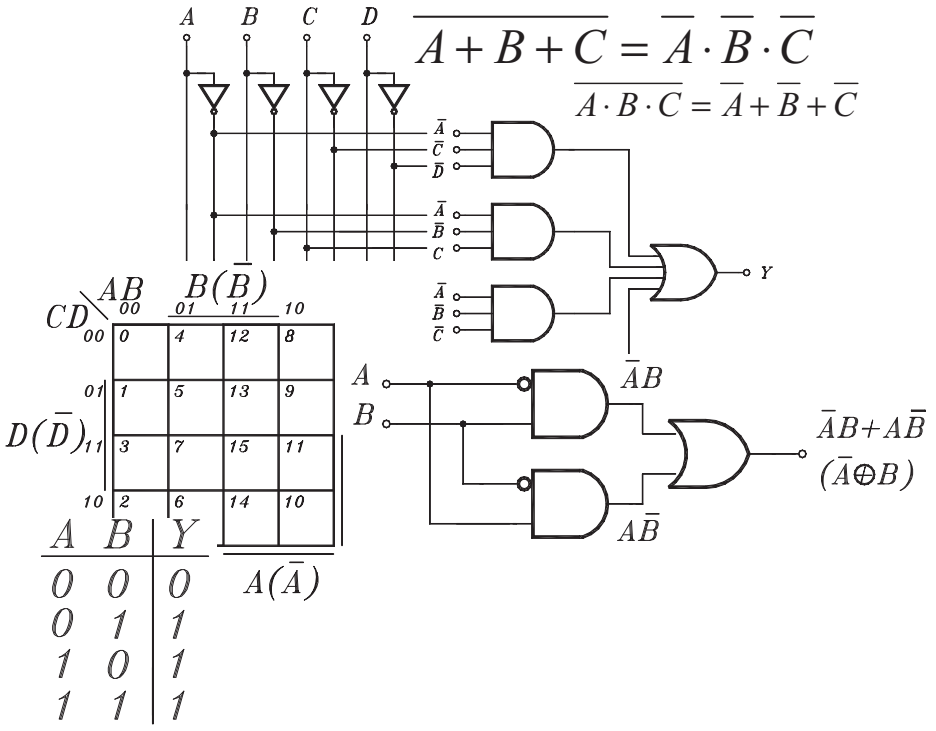
1-70. Податоците (а) ZAba; (б) UB40; (в) Dete, кодирај ги во стандардниот ASCII код. Секој коден збор напиши го и во хексадецимална нотација.

1-71. Дадени се следниве ASCII кодни зборови (а) 66 73 84 79 71 61; (б) 83 71 79 80 74 69; во декадно означување. Кои се кодираните податоци?

1-72. Која е разликата помеѓу (а) тежинските и редоследните; (б) рамномерните и нерамномерните, (в) редундантните и нередундантните кодови. Каде припаѓа (1) 8421 (NBCD) кодот; (2) ASCII кодот; (3) Седумсегментиот код?

1-73. Со терминот збор се означува ...

1-74. Даден е податокот (а) 10101011; (б) 11001100; (в) 01010001; (г) 00111001. Одреди ја неговата (1) експлицитна вредност; (2) имплицитна вредност, ако секој податок посебно се разгледува како (1) SM; (2) DC; (3) RC; (4) 8421 NBCD број; (5) податок запишан во ASCII код.



2.

БУЛОВА АЛГЕБРА

По изучувањето на оваа тематска целина

- ќе ги познавате аксиомите, законите и теоремите од Буловата алгебра;
- ќе можете да ги претставувате прекинувачките функции во алгебарски, табличен и графички облик;
- ќе решавате задачи за премин од еден во друг облик на претставување на прекинувачките функции;
- ќе решавате задачи од минимизација на прекинувачки функции од четири променливи по аналитички пат и со методата на Карноови карти;
- ќе ги познавате симболите на стандардните логички кола, ќе ги разликувате според функцијата што ја извршуваат и ќе ги применувате во логички дијаграми;
- ќе решавате поедноставни задачи од анализа и синтеза на логички мрежи;
- ќе умеете да претставувате поедноставни прекинувачки мрежи во две нивоа.

2.1. ВОВЕД

Бинарното претставување на броевите тешко може да се замисли како дел од човечкото изразување затоа што ние сме научени да размислуваме декадно. Меѓутоа, елементарните составни делови на дигиталните уреди се електронски кола кои се карактери-зираат само со две состојби, така што за нив „природен јазик" е бинарното означување (нотација). Токму заради ова во дигиталната техника се применува бинарниот броен систем и соодветна алгебра која оперира со бинарни броеви.

Буловата алгебра своите корени ги влече од средината на *XIX*-от век кога се појавила како нова математичка дисциплина. Нејзин основоположник е англискиот математичар Џорџ Бул според кој и го добила името. Бидејќи оваа алгебра се базира на законите на формалното-логичко мислење и заклучување, за неа се користи и терминот *логичка алгебра*. Овие закони се темелат на тврдења кои можат да бидат само вистинити, или невистинити, т.е. примаат само две вредности, а нив прв ги запишал големиот грчки филозоф Аристотел. Џорџ Бул предложил законите на формалното-логичко заклучување да се опишат со алгебарски релации и операции. Токму со ова се овозможи процесот на формалното-логичко расудување и заклучување едноставно квантитативно да се претстави и технички да се реализира и автоматизира со примена на компоненти кои имаат само две состојби. Бидејќи вака се однесуваат прекинувачките елементи и логичките кола, оваа алгебра се нарекува и *прекинувачка алгебра*.

2.2. АКСИОМИ И ЛОГИЧКИ ОПЕРАЦИИ

Буловата алгебра *е дедуктивен математички систем што се дефинира на бинарното множество B кое содржи само два меѓусебно различни елементи*. За нив во литературата можат да се сретнат различни симболи, но ние ќе ги употребуваме симболите *„1" (логичка единица) и „0" (логичка нула)*, така што B={1,0}. Според ова сите константи и променливи во Буловата алгебра можат да имаат само една од вредностите 1 или 0, па заради тоа тие и се викаат *логички или прекинувачки променливи*. Вообичаено е независно променливите да се означуваат со големите букви од англиската абецеда и тоа: *A,B,C,D,E,...* или X_0, X_1, X_2, X_3, …, додека зависно променливите, т.е. функциите кои сега се викаат *логички, прекинувачки, или комутациони функции*, да се означуваат со букви Y, Y_0, Y_1, Y_2, …, или F, F_1, F_2.

Во множеството *B* се дефинираат *две интерни бинарни операции* „+" и „·", кои ги задоволуваат следните три аксиоми, познати како аксиоми на Хантингтон:

A.1. Бинарните интерни операции се комутативни и дистрибутивни една кон друга, т.е. за било која променлива *A,B,C* од {*B*} важи:

$$A + B = B + A\,,\ A\ B = B\ A$$

$$A(B + C) = (AB) + (AC),\ A + (BC) = (A + B)\ (A + C)$$

A.2. Бинарните интерни операции поседуваат различни неутрални елементи 1 и 0, така што за било која логичка променлива A постои елемент 0 за кој важи $A+0=A$, и постои елемент 1 за кој важи $A \cdot 1=A$.

A.3. За било која логичка променлива A постои единствена инверзна променлива $\overline{A}$ таква што важи $A + \overline{A} = 1,\ A \cdot \overline{A} = 0$.

Една важна особина што произлегува директно од наведените аксиоми е принципот на дуалност (симетричност). Тоа значи дека сите аксиоми се дадени во парови, и тоа посебно за операцијата „+”, и посебно за операцијата „.”. Според овој принцип може да се изврши меѓусебна замена на операцијата „+” со операцијата “.” и на елементите 1 со 0, па така тргнувајќи од аксиомите за операцијата „+” се добиваат дуални аксиоми за операцијата „.”, и обратно.

Во Буловата алгебра постојат три основни (елементарни) операции: две операции кои оперираат со два или повеќе операнди: логичко собирање (+) и логичко множење (·) и една унарна операција која работи со еден операнд: логичка негација($\bar{\ }$).

Логичкото собирање се вика уште операција ***ИЛИ*** (анг. OR) и логичка дисјункција, а нејзиниот оператор покрај ознаката „+”, може да биде и „$\cup$“. Логичкото множење се вика и операција ***И*** (анг. AND), или логичка конјункција, а нејзиниот оператор покрај ознаката „.”, може да биде и „$\cap$“, или „&”. Вообичаено е овој оператор да се испушта при пишувањето на логичките изрази. Логичката негација се вика уште операција ***НЕ*** (анг. NOT), или ***КОМПЛЕМЕНТИРАЊЕ***, а покрај ознаката „ $\bar{\ }$ “, се означува и со „$\lceil$“ или „$\sim$”.

Основните логички операции се дефинираат на начинот прикажан во табелите таб. 2-1, таб. 2-2 и таб. 2-3.

ИЛИ (+)
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 1

Таб. 2-1. ИЛИ.

И (·)
0 · 0 = 0
0 · 1 = 0
1 · 0 = 0
1 · 1 = 1

Таб. 2-2. И.

НЕ ($\bar{\ }$)
$\bar{0} = 1$
$\bar{1} = 0$

Таб. 2-3. НЕ.

Основни логички операции

Од табелите се гледа дека за операцијата ИЛИ (логичко собирање) неутрален елемент е 0, додека за операцијата И (логичко множење) е 1, така што може да се заклучи следново:

1. Ако се соберат два операнди, резултатот ќе биде 0 само ако едновремено и двата операнди се на вредност 0, а инаку се добива 1, т.е. барем една 1 дава 1;

2. Ако се помножат два операнди, резултатот ќе биде 1, само ако и двата операнди се на вредност 1, инаку се добива 0, т.е.барем една 0 дава 0;

3. Ако вредноста на било кој операнд не е 0,тогаш таа е 1, и обратно, ако вредноста на операндот не е 1, тогаш таа е 0.

Од наведените операции највисок ранг на извршување има комплементирањето (операцијата НЕ, логичката негација), потоа е логичкото множење (операцијата И), и на крај е логичкото собирање (операцијата ИЛИ). Редоследот на извршување на операциите може да се измени со употреба на загради.

Со комбинирање на основните логички операции можат да се изведат други, нешто посложени операции: ***НИ,*** (анг. *NAND)* која се добива со комплементирање на множењето (И, а потоа НЕ) и ***НИЛИ***, (анг. *NOR)* која се добива со комплементирање на собирањето (ИЛИ, а потоа НЕ). Потоа, операциите: **исклучиво *ИЛИ***, т.е**. ексклузивно *ИЛИ***, ***ИСКИЛИ*** т.е.***ЕКСИЛИ*** (анг. *XOR)* што се означува со “$\oplus$“, и ***ИСКНИЛИ*** т.е. ***ЕКСНИЛИ*** (анг. *XNOR)* која се добива со комплемент од ЕКСИЛИ (ЕКСИЛИ, а потоа НЕ).

Сите претходно дефинирани операции се претставени со табелите таб. 2-4, таб. 2-5, таб. 2-6 и таб.2-7.

НИЛИ ($\overline{+}$)
$\overline{0+0}=1$
$\overline{0+1}=0$
$\overline{1+0}=0$
$\overline{1+1}=0$

Таб. 2-4. НИЛИ

НИ ($\overline{\cdot}$)
$\overline{0\cdot 0}=1$
$\overline{0\cdot 1}=1$
$\overline{1\cdot 0}=1$
$\overline{1\cdot 1}=0$

Таб. 2-5. НИ

ЕКСИЛИ ($\oplus$)
$0\oplus 0=0$
$0\oplus 1=1$
$1\oplus 0=1$
$1\oplus 1=0$

Таб. 2-6. ЕКСИЛИ

ЕКСНИЛИ ($\overline{\oplus}$)
$\overline{0\oplus 0}=1$
$\overline{0\oplus 1}=0$
$\overline{1\oplus 0}=0$
$\overline{1\oplus 1}=1$

Таб. 2-7. ЕКСНИЛИ

Изведени логички операции

Од дефинициите може да се заклучи дека:

1. Резултатот од НИЛИ ќе биде 1 само ако и двата операнда се 0, инаку се добива 0,

2. Резултатот од НИ ќе биде 0 само ако и двата операнда се 1, инаку се добива 1;

3. Резултатот од ЕКСИЛИ е 0 секогаш кога операндите се со иста вредност, т.е. две 0-и даваат резултат 0, но и две 1-и даваат 0. Ако операндите имаат спротивни вредности, тогаш се добива 1;

4. Резултатот од ЕКСНИЛИ ќе биде обратен во однос на ЕКСИЛИ. Оваа операција всушност ги споредува, ги компарира вредностите на операндите. Имено, ако двата операнда се исти, тогаш резултатот е 1, но ако се различни тогаш се добива 0.

2.3. ТЕОРЕМИ И ЗАКОНИ

Од аксиомите на Хантингтон може да се изведат различни теореми во Буловата алгебра кои имаат своја соодветна примена. Ние ќе се задржиме само на оние теореми кои ги вклучуваат операциите И, ИЛИ и НЕ. Некои од овие теореми ги изразуваат законите на Буловата алгебра, а сите заедно се користат како правила при решавањето и поедноставувањето на логичките равенки и логичките изрази. Теоремите се дадени во симетрични парови со примена на принципот на дуалност. Сите променливи што се користат во нив се логички променливи, што значи дека нивните вредности можат да бидат само 0 или 1. Од овде произлегува дека ако било која променлива A има вредност 1, т.е. ако е исполнето A = 1, тогаш $\overline{A}=0$, и обратно: ако A = 0, тогаш $\overline{A}=1$.

Најнапред ќе ги наведеме оние теореми кои вклучуваат само една променлива.

Тоа се:

$$\overline{\overline{A}}=A \qquad \text{(т.2-1)}$$

$$A+0=A \qquad A\cdot 1=A \qquad \text{(т.2-2)}$$

$$A+1=1 \qquad A\cdot 0=0 \qquad \text{(т.2-3)}$$

$$A+A=A \qquad A\cdot A=A \qquad \text{(т.2-4)}$$

$$\overline{A}+\overline{A}=\overline{A} \qquad \overline{A}\cdot\overline{A}=\overline{A} \qquad \text{(т.2-5)}$$

$$A+\overline{A}=1 \qquad A\cdot\overline{A}=0 \qquad \text{(т.2-6)}$$

Теоремите со кои се искажуваат асоцијативниот, комутативниот и дистрибутивниот закон се дадени последователно во продолжение:

$$A+(B+C)=(A+B)+C \qquad A\cdot(B\cdot C)=(A\cdot B)\cdot C \qquad \text{(т.2-7)}$$

$$A+B=B+A \qquad A\cdot B=B\cdot A \qquad \text{(т.2-8)}$$

$$A\ (B+C)=A\ B+A\ C \qquad A+B\cdot C=(A+B)\cdot(A+C) \qquad \text{(т.2-9)}$$

Докажувањето на теоремите се базира на трите аксиоми, меѓутоа за нас тоа не е од суштинско значење, па затоа ќе докажеме за илустрација само една од нив. Имено, равенките со кои се искажуваат асоцијативниот и комутативниот закон, како и првата равенка со која се изразува дистрибутивниот закон многу лесно интуитивно ги разбираме затоа што тие се многу слични со истите овие закони кои важат за вообичаената алгебра. Меѓутоа, втората равенка за дистрибутивниот закон изгледа малку чудно, и некако не се вклопува во нашите сфаќања. Токму затоа неа и ќе ја докажеме, и тоа на два начини.

(1) Првиот доказ ќе го изведеме *аналитички* (*алгебарски*) со примена на аксиомите и претходно наведените теореми, при што ќе тргнеме од десниот дел од равенката и ќе го добиеме нејзиниот лев дел:

$$(A+B)(A+C)=AA+AC+AB+BC=A+AC+AB+BC=A+AB+AC+BC=$$
$$=A(1+B)+AC+BC=A+AC+BC=A(1+C)+BC=A+BC$$

(2) Вториот доказ ќе биде со примена на *методот на совршена индукција*. Според овој метод теоремата се докажува така што се проверува дали левата страна на равенката има идентични вредности со изразот од десната страна за сите комбинации на вредности што можат да ги земат променливите. Бидејќи во конкретниов случај фигурираат три променливи, следува дека ќе се појават вкупно $2^3=8$ можни комбинации. За секоја од нив ќе ја пресметаме вредноста на логичкиот израз кој се наоѓа на левата страна од знакот за еднаквост: *(A+B) (A+C)*, и вредноста на изразот кој се наоѓа од десната страна: *(A+BC)*, а добиените резултати ќе ги запишуваме во таблица. Ако за секоја комбинација на променливите се добие ист резултат, тоа значи дека теоремата е докажана. Од таблицата таб. 2-8 се забележува дека изразот *(A+B)(A+C)* има иста вредност како и изразот *(A+BC)* за било која комбинација на вредности што ги примаат променливите *A, B* и *C* така што доказот е завршен.

A	B	C	(A+B)·(A+C)	(A+B·C)
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Таб. 2-8. Метод на совршена индукција

Во Буловата алгебра исклучително значајно место заземаат и Де Морган-овите закони или теореми. Тие можат да се напишат во следниов облик:

$$\overline{A+B+C+...}=\overline{A}\cdot\overline{B}\cdot\overline{C}...,\qquad \overline{A\cdot B\cdot C...}=\overline{A}+\overline{B}+\overline{C}+... \qquad \text{(т. 2-10)}$$

Од наведените изрази може да се констатира дека:

1. комплемент од логички збир на повеќе променливи може да се замени со логичкиот производ од комплементите на секоја поединечна променлива и

2. комплемент од логички производ на повеќе променливи може да се замени со логичкиот збир од комплементите на секоја поединечна променлива.

Покрај претходно наведените теореми, важни се и следниве:

$$A+AB=A, \qquad A(A+B)=A \qquad \text{(т.2-11)}$$

$$A+\overline{A}B=A+B, \qquad A(\overline{A}+B)=AB \qquad \text{(т.2-12)}$$

$$AB+A\overline{B}=A, \qquad (A+B)(A+\overline{B})=A \qquad \text{(т.2-13)}$$

$$AB+\overline{A}C=(A+C)(\overline{A}B), \qquad (A+B)(\overline{A}+C)=AC+\overline{A}B \qquad \text{(т.2-14)}$$

$$AB+\overline{A}C+BC=AB+\overline{A}C, \qquad (A+B)(\overline{A}+C)(B+C)=(A+B)(\overline{A}+C) \qquad \text{(т.2-15)}$$

$$AB+BC+\overline{B}C=AB+C, \qquad (A+B)(B+C)(\overline{B}+C)=(A+B)C \qquad \text{(т.2-16)}$$

Теоремите (т. 2-11), (т. 2-12) и (т. 2-13) се познати и како *теореми за апсорпција*.

На крај ќе ја наведеме и теоремата за развивање *(експанзија)*:

$$Y(A,B,C,...)=[A\cdot Y(1,B,C,...)]+[\overline{A}\cdot Y(0,B,C,...)] \qquad \text{(т. 2-17)}$$

При упростувањето и решавањето на посложените логички изрази се употребуваат сите до сега наведени аксиоми, закони и теореми. Да разгледаме неколку примери:

Пр. 1. $ABC+AB\overline{C}+A\overline{B}C=A(BC+B\overline{C}+\overline{B}C)=A[B(C+\overline{C})+\overline{B}C]=A(B+\overline{B}C)=$
$=A(B+C)=AB+AC$

Пр. 2. $(A+B)(A+\overline{B})(\overline{A}+C)=(AA+A\overline{B}\_AB+B\overline{B})(\overline{A}+C)=(A+A\overline{B}+AB)(\overline{A}+C)=$
$=[A(1+B)+A\overline{B}](\overline{A}+C)=(A+A\overline{B})(\overline{A}+C)=A(1+\overline{B})(\overline{A}+C)=AC$

Пр. 3. $Y(A,B,C)=(A+B)[A(B+C)+AB+AC]=A(1+B)[1(B+C)+1B+1C]+$
$+A(0+B)[0(B+C)+0B+0C]=A(B+C+B+C)+AB(0+0+0)=A(B+C)$

2.4. ПРЕКИНУВАЧКИ ФУНКЦИИ И НИВНО ПРИКАЖУВАЊЕ

Секоја логичка променлива чија вредност зависи од вредноста на други логички променливи претставува логичка (прекинувачка) функција. Прекинувачките функции се прикажуваат на три начини: табеларно со користење на т.н. **комбинациони таблици (таблици на вистинитост)**, потоа аналитички (алгебарски) со помош на логички равенки, и графички со примена на **логички симболи** (стандаризирани блок-шеми)**.** Во продолжение ќе биде обработено табеларното и аналитичкото претставување на функциите. Исто така и на графичката презентација ќе и биде посветено посебно внимание и простор во понатамошниот текст, бидејќи таа има исклучителна важност затоа што води кон шематско прикажување на прекинувачките функции.

2.4.1. ТАБЕЛАРНО ПРЕТСТАВУВАЊЕ

При табеларната (табличната) презентација најпрво се црта комбинационата таблица или таблица на вистинитост во која се запишуваат имињата на сите независно променливи во левиот дел, и името на функцијата, или функциите ако ги има повеќе, во десниот дел од табелата. Така се добиваат онолку колони колку што има вкупно независни и зависни променливи. Потоа по редици се запишуваат сите возможни комбинации на вредности што можат да ги примат независните променливи, и на крај за секоја комбинација се внесува вредноста на функцијата во соодветната колона.

Ако се претпостави дека е дадена функција што зависи од n променливи, тогаш во комбинационата таблица ќе има n колони за независно променливите и една колона за функцијата. Бидејќи постојат вкупно $N=2^n$ возможни влезни комбинации, јасно е дека во таблицата на вистинитост ќе се појават вкупно $N=2^n$ редици. Секоја редица може да се означи во декаден облик со соодветен индекс *"i"*, и тоа така што на првата редица и се придружува индексот 0, а на последната редица индексот *(N-1)*, т.е. (2^n-1). Комбинационите таблици на било која функција од 2, 3 и 4 променливи, се означени како таб. 2-9, таб. 2-10 и таб. 2-11, последователно.

i	AB	Y
0	00	
1	01	
2	10	
3	11	

Таб. 2-9. Функција од две променливи

i	ABC	Y
0	000	
1	001	
2	010	
3	011	
4	100	
5	101	
6	110	
7	111	

Таб. 2-10. Функција од три променливи

i	$ABCD$	Y
0	0000	
1	0001	
2	0010	
3	0011	
4	0100	
5	0101	
6	0110	
7	0111	
8	1000	
9	1001	
10	1010	
11	1011	
12	1100	
13	1101	
14	1110	
15	1111	

Таб. 2-11. Функција од четири променливи

Комбинациони таблици на логички функции

Вкупниот број на функции N_F што можат да произлезат ако на располагање се n независни променливи изнесува:

$$N_F = 2^{2^n} \tag{2-18}$$

2.4.2. АНАЛИТИЧКА ПРЕТСТАВА

Запишувањето во аналитички облик е познато од конвенционалната алгебра. Слично како таму, и во Буловата алгебра се формира одредена равенка која се вика **логичка, булова** или **прекинувачка равенка**. Имено, од левата страна на знакот за еднаквост „=“ се наведува функцијата (зависно променливата), а од десната страна независно променливите поврзани со знаците на логичките операции. Општо, секоја прекинувачка функција може да се напише на различни начини, така што некогаш се добива поедноставна, а некогаш посложена форма. Ние ќе се запознаеме со нормирани (стандардни, канонички) форми за пишување на логичките функции. Тоа се такви облици чија структура е точно пропишана, а се викаат **нормални форми (НФ).** Станува збор за аналитичко претставување на функциите во **дисјунктивна нормална форма (ДНФ) и конјунктивна нормална форма (КНФ).**

ДНФ ја прикажува функцијата во облик на збир, т.е. сума (Σ) од производи (p) на независно променливите (Σp). Парцијалниот производ се вика **минтерм** (m) (елементарен производ, полна или целосна конјункција), ако во него учествуваат сите независно променливи, независно од тоа дали тие се јавуваат во директен или комплементарен облик. Ако сите производи што влегуваат во сумата на ДНФ се минтерми, тогаш станува збор за **совршена ДНФ (**Σm**) (СДНФ)**. КНФ, од друга страна, ја прикажува функцијата како производ (П) од збировите, т.е. сумите (s) на независно променливите (Пs). Парцијалната сума се вика макстерм (М) (елементарна сума, полна или целосна дисјункција) ако таа претставува збир од сите независни променливи, при што тие можат да се јават во директен или комплементарен облик. Кога сите суми што влегуваат во производот на КНФ се макстерми, тогаш се добива **совршена КНФ (**ПМ**) (СКНФ).**

Заради појаснување, ќе разгледаме неколку примери на функции кои зависат од три и четири променливи: $Y = Y(A,B,C), Z = Z(D,G,H), F = F(X_1, X_2, X_3, X_4)$. Кај ДНФ потцртаните членови се минтерми (m), а кај КНФ тоа се макстерми (М).

СДНФ: $Y = \underline{ABC} + \underline{\overline{A}B\overline{C}} + \underline{\overline{A}\overline{B}\overline{C}}$ ДНФ: $Y = \underline{A\overline{B}C} + \underline{\overline{A}BC} + \underline{A\overline{B}C} + \underline{\overline{B}\overline{C}}$

КНФ: $Y = (A + \overline{B})(\overline{A} + \overline{B})(\overline{A} + C)$ СКНФ: $Y = \underline{(A + \overline{B} + C)}\underline{(A + B + \overline{C})}$

КНФ: $Z = \underline{(\overline{D} + \overline{G} + H)}(G + \overline{H})$ ДНФ: $Z = \underline{\overline{D}GH} + \underline{D\overline{G}H} + GH + \overline{D}$

СДНФ: $\underline{X_1X_2\overline{X}_3\overline{X}_4} + \underline{\overline{X}_1X_2X_3X_4} + \underline{\overline{X}_1\overline{X}_2X_3\overline{X}_4}$

СКНФ: $\underline{(X_1 + \overline{X}_2 + X_3 + \overline{X}_4)}\underline{(X_1 + \overline{X}_2 + \overline{X}_3 + X_4)}$

Нормалните форми се изведуваат многу брзо и едноставно, но најважна причина што токму нив ќе ги применуваме е фактот што со нив функцијата се добива по две нивоа. Имено, кај ДНФ прво променливите логички се множат, а потоа сите резултати логички се собираат (И-ИЛИ). Кај КНФ е обратно: на првото ниво променливите логички се собираат, додека на второто ниво резултатите логички се множат (ИЛИ-И). Како што ќе видиме подоцна, ова е една многу битна особина.

Меѓутоа, за нормалните форми треба да знаеме и тоа дека тие во општ случај се *редундантни форми*, т. е. форми кои не ја претставуваат логичката функција во наједноставен и најкраток облик бидејќи содржат поголем број членови од минимално потребниот број со кои се дефинира истата функција. Нормалните форми со најмал број членови (суми, односно производи), и воедно секој од тие членови да вклучува најмалку променливи, се дефинираат како минимални нормални форми: **МДНФ и МКНФ.** Овие форми треба да ја претставуваат логичката функција во најкраток и наједноставен облик.

2.4.2.1. ЦЕЛОСНО ЗАДАДЕНИ ФУНКЦИИ

СДНФ и СКНФ вообичаено се означуваат во еден поедноставен аналитички облик со декадна нотација преку т.н. множество на индекси. Имено, наместо елементарните членови (производите, односно сумите) се користат ознаки за минтерм: „m", односно макстерм „M", а покрај нив се запишува соодветен индекс i_{1j}, односно i_{0k}. Индексот покрај минтермот i_{1j} одговара на редниот број на оние редици во кои вредноста на функцијата е 1, а индексот покрај макстермот i_{0k} одговара на редниот број на оние редици за кои вредноста на функцијата е 0, така што секогаш ќе важи $j+k=N$, каде што $N=2^n$, што значи дека и двата индекса припаѓаат во опсегот од 0 до $N-1$: [0,1,2…,2^n-1]. Според ова СДНФ на функцијата се означува како сума од минтерми: $Y = m_{i11} + m_{12} + ... + m_{i1j}$, додека СКНФ на функцијата се означува како производ од макстерми: $Y = M_{i01} M_{i02} ... M_{i0k}$. Почесто се користи скратено претставување преку множествата на индекси на следниов начин. За СДНФ ќе имаме $Y = \sum m(i_{11}, i_{12}, ..., i_{1j})$ или $f^{(1)}=(i_{11}, i_{12}, ..., i_{1j})$ додека за СКНФ ќе се добие $Y = \prod M(i_{01}, i_{01}, ..., i_{0k})$ или $f^{(0)}=(i_{10}, i_{10}, ..., i_{1k})$. *Оние индекси што не се појавуваат во СДНФ ќе фигурираат во СКНФ и обратно, затоа што ако функцијата нема вредност 1, тогаш таа има вредност 0.*

Комбинационата таблица за било која функција од три променливи е дадена како таб. 2-12, при што во посебни колони се означени сите минтерми, односно макстерми.

i	A B C	Y
0	0 0 0	
1	0 0 1	
2	0 1 0	
3	0 1 1	
4	1 0 0	
5	1 0 1	
6	1 1 0	
7	1 1 1	

m_i	Минтерми
m_0	$\overline{A} \cdot \overline{B} \cdot \overline{C}$
m_1	$\overline{A} \cdot \overline{B} \cdot C$
m_2	$\overline{A} \cdot B \cdot \overline{C}$
m_3	$\overline{A} \cdot B \cdot C$
m_4	$A \cdot \overline{B} \cdot \overline{C}$
m_5	$A \cdot \overline{B} \cdot C$
m_6	$A \cdot B \cdot \overline{C}$
m_7	$A \cdot B \cdot C$

M_i	Максетерми
M_0	$A + B + C$
M_1	$A + B + \overline{C}$
M_2	$A + \overline{B} + C$
M_3	$A + \overline{B} + \overline{C}$
M_4	$\overline{A} + B + C$
M_5	$\overline{A} + B + \overline{C}$
M_6	$\overline{A} + \overline{B} + C$
M_7	$\overline{A} + \overline{B} + \overline{C}$

Таб. 2-12. Минтерми и макстерми на функција од три променливи

Од таблиците се гледа дека секој макстерм претставува комплементарна вредност на соодветниот минтерм, и обратно, т.е. дека за секое $i = (0, 1, 2, ..., 2^n-1)$ важи:

$$M_i = \overline{m_i} \qquad (2\text{-}19)$$

Со следната комбинациона таблица, која е означена како таб. 2-13 се зададени три различни функции што зависат од три исти променливи *A, B,* и *C.* Тоа се функциите: $Y=Y(A,B,C)$, $Z=Z(A,B,C)$, $W=W(A,B,C)$ за кои се наведени различни нормални форми за некои од зададените функции кои нив ги опишуваат преку множества на индекси.

i	$A\,B\,C$	Y	Z	W
0	0 0 0	1	0	1
1	0 0 1	0	0	1
2	0 1 0	1	0	0
3	0 1 1	0	0	1
4	1 0 0	0	1	0
5	1 0 1	1	1	0
6	1 1 0	0	0	1
7	1 1 1	1	1	1

Таб. 2-13. Комбинациони таблици на функциите Y, Z и W од три променливи

СКНФ: $Y = \prod M(1,3,4,6) = (A + B + \overline{C})(A + \overline{B} + \overline{C})(\overline{A} + B + C)(\overline{A} + \overline{B} + C)$

СДНФ: $Y = \sum m(0,2,5,7) = (\overline{A}\,\overline{B}\,\overline{C}) + (\overline{A}B\overline{C}) + (A\overline{B}C) + (ABC)$

СДНФ: $W = \sum m(0,1,3,6,7) = (\overline{A}\,\overline{B}\,\overline{C}) + (\overline{A}\,\overline{B}C) + (\overline{A}BC) + (AB\overline{C}) + (ABC)$

СКНФ: $Z = \prod M(0,1,2,3,6) = (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(A + \overline{B} + \overline{C})(\overline{A} + \overline{B} + C)$

2.4.2.2. НЕЦЕЛОСНО ЗАДАДЕНИ ФУНКЦИИ

Досега секоја логичка функција ја дефиниравме со задавање на вредноста на функцијата за секоја комбинација на независно променливите, при што функцијата имаше вредност или 0 или 1.

i	$A\,B\,C\,D$	F
0	0 0 0 0	x
1	0 0 0 1	1
2	0 0 1 0	1
3	0 0 1 1	x
4	0 1 0 0	x
5	0 1 0 1	x
6	0 1 1 0	0
7	0 1 1 1	0
8	1 0 0 0	1
9	1 0 0 1	0
10	1 0 1 0	1
11	1 0 1 1	1
12	1 1 0 0	0
13	1 1 0 1	0
14	1 1 1 0	0
15	1 1 1 1	x

Меѓутоа, во праксата често пати можат да се сретнат и *некомплетно (нецелосно) зададени (дефинирани) функции.* За да се случи ова постојат две различни причини, кои практично се сведуваат на едно исто. Како прво, понекогаш не е важно каква вредност има функцијата за една или повеќе комбинации, од влезните променливи. Од друга страна, може да се случи некои комбинации на независно променливите да не можат никогаш да се појават.

И во двете ситуации може да се земе дека не е важно каква ќе биде вредноста на функцијата за одредени влезни комбинации. Ваквите вредности на функцијата се нарекуваат "неважни" (анг. "don't care") и во литературата се означуваат со различни симболи, како на пр. „/", „\", „-", „ѣ", „x" или „X". Ние во понатамошното излагање ќе го користиме симболот „x".

Таблицата на вистинитост за една некомплетно зададена функција од четири променливи F=*F(D,C,B,A)* е претставена како таб. 2-14.

Таб. 2-14. Комбинациона таблица на некомплетно дефинирана функција *F(D,C,B,A)*

Нејзиното претставување со множество на индекси во СДНФ и СКНФ облик ќе биде:

$$F=\sum m(1,2,8,10,11)+\sum_{xm} x(0,3,4,5,15) \text{ и } F=\prod M(6,7,9,12,13,14)\prod_{xM} x(0,3,4,5,15).$$

2.4.3. ПРЕМИНУВАЊЕ ОД ЕДЕН ОБЛИК ВО ДРУГ

Без оглед на тоа дали функцијата е позната преку таблицата на вистинитост или аналитички, релативно лесно може да се премине од едниот во другиот облик.

Кога функцијата е дадена таблично, можат да се изведат и двете форми на нејзиното аналитичко претставување: совршената дисјунктивна нормална форма (СДНФ) и совршената конјуктивна нормална форма (СКНФ). *СДНФ се добива така што се пишува збирот на онолку минтерми, колку што во табелата постојат редици за кои вредноста на функцијата изнесува 1*. Во минтермите независно променливите се појавуваат во директен облик (номинален, некомплементирани) ако во соодветната редица имаат вредност 1, а комплементирани ако нивната вредност е 0, како што може да се види и од примерот што следува. Станува збор за комбинационите таблици таб. 2-15 а) б) на функциите Y(A,B,C), Z(A,B,C) од каде што произлегува нивниот СДНФ облик:

$$Y=\sum m(0,3,7)=(\overline{A}\,\overline{B}\,\overline{C})+(\overline{A}BC)+(ABC)$$

$$Z=\sum m(1,2,3,4)=(\overline{A}\,\overline{B}C)+(\overline{A}B\overline{C})+(\overline{A}BC)+(A\overline{B}\,\overline{C})$$

i	$A\,B\,C$	Y
0	0 0 0	1
1	0 0 1	0
2	0 1 0	0
3	0 1 1	1
4	1 0 0	0
5	1 0 1	0
6	1 1 0	0
7	1 1 1	1

а) Y=Σm(0,3,7)

i	$A\,B\,C$	Z
0	0 0 0	0
1	0 0 1	1
2	0 1 0	1
3	0 1 1	1
4	1 0 0	1
5	1 0 1	0
6	1 1 0	0
7	1 1 1	0

б) Z= Σm(1,2,3,4)

Таб. 2-15. Комбинациони таблица на прекинувачки функции од три променливи

СКНФ се добива така што се пишува производ од онолку макстерми колку што во таблицата има редици во кои вредноста на функцијата е 0. Сега во макстермот се комплементира онаа променлива чија вредност во соодветната редица е 1, додека онаа променлива чија вредност е 0 се пишува во директен облик. Добивањето на СКНФ формата илустрирана е на претходниот пример на функциите *Y* и *Z* од таб. 2-15:

$$Y=\prod M(1,2,4,5,6)=(A+B+\overline{C})(A+\overline{B}+C)(\overline{A}+B+C)(\overline{A}+B+\overline{C})(\overline{A}+\overline{B}+C)$$

$$Z=\prod M(0,5,6,7)=(A+B+C)(\overline{A}+B+\overline{C})(\overline{A}+\overline{B}+C)(\overline{A}+\overline{B}+\overline{C})$$

Правилно е да се употреби онаа форма која дава помал број минтерми, односно макстерми затоа што е позгодна за понатамошно упростување.

Кога функцијата е дадена аналитички, со логичка равенка, преминот во табличен облик се прави на следниов начин. Прво се црта комбинационата таблица во која по колони се наведуваат независно променливите, па функцијата, а потоа по редици се запишуваат сите можни комбинации на независно променливите. Понатаму, во зададената равенка последователно се заменува секоја влезна комбинација и се пресметува вредноста на функцијата. Оваа вредност се запишува во таблицата во колоната на функцијата, кај соодветната редица.

Да разгледаме еден пример на преминување од аналитички во табличен облик. Зададена е функцијата W што зависи од три променливи *A,B,C:* $W = A\overline{B}C + ABC + A\overline{B}$. Вредноста на функцијата за секоја влезна комбинација ќе ја пресметаме подолу, почнувајќи од влезната комбинација *ABC=0*, па се до *ABC=111*.

Таблицата на вистинитост за оваа функција е означена со таб. 2-16.

i	*A B C*	*W*
0	0 0 0	0
1	0 0 1	0
2	0 1 0	0
3	0 1 1	0
4	1 0 0	1
5	1 0 1	1
6	1 1 0	0
7	1 1 1	1

Кога A=0, B=0, C=0 тогаш W=010+000+01=0;

Кога A=0, B=0, C=1 тогаш W=011+001+01=0;

Кога A=0, B=1, C=0 тогаш W=000+010+00=0;

Кога A=0, B=1,C=1 тогаш W=001+011+00=0;

Кога A=1, B=0, C=0 тогаш W=110+100+11=1;

Кога A=1, B=0, C=1 тогаш W=111+101+11=1;

Кога A=1, B=1, C=0 тогаш W=100+110+10=0;

Кога A=1, B=1, C=1 тогаш W=101+111+10=1.

Таб. 2-16. Комбинациона таблица на прекинувачка функција од три променливи W(A,B,C)

Преминувањето од еден аналитички облик во друг може да се изврши на различни начини, што зависи од појдовниот (зададениот) облик на функцијата, кој треба да биде нејзиниот конечен облик.

Ќе наведеме два примери за тоа како од НФ може да се премине во СНФ. Нека се дадени две функции: $Z = Z(A,B,C)$ во следните форми: $Y = A\overline{B} + C$, $Z = (A + \overline{B} + C)B$. Првата функција е дадена во ДНФ и од неа треба да се добие СДНФ, а од втората која е дадена во КНФ треба да се добие СКНФ.

Пр. 1.
$$Y = A\overline{B} + C = A\overline{B}1 + 11C = A\overline{B}(C + \overline{C}) + 1(B + \overline{B})C = A\overline{B}C + A\overline{B}\overline{C} + 1(BC + \overline{B}C) =$$
$$= A\overline{B}C + A\overline{B}\overline{C} + (A + \overline{A})(BC + \overline{B}C) = A\overline{B}C + A\overline{B}\overline{C} + ABC + A\overline{B}C + \overline{A}BC + \overline{A}\overline{B}C$$

Пр. 2.
$$Z = (A + \overline{B} + C)B = (A + \overline{B} + C)(0 + B + 0) = (A + \overline{B} + C)(A\overline{A} + B + 0) = (A + \overline{B} + C)$$
$$[(A + B)(\overline{A} + B) + 0] = (A + \overline{B} + C)[(A + B)(\overline{A} + B) + C\overline{C}] = (A + \overline{B} + C)$$
$$\{[(A + B)(\overline{A} + B) + C][(A + B)(\overline{A} + B) + C]\} = (A + \overline{B} + C)(A + B + C)(\overline{A} + B + C)$$
$$(A + B + \overline{C})(\overline{A} + B + \overline{C})$$

Преминувањето во обратна насока од СНФ во НФ, практично претставува одредено упростување на зададената функција, а може да се изведе со примена на претходно наведените теореми. Следниве примери покажуваат како може да се упрости функцијата *U(X,Y,Z)* дадена во СДНФ облик и *V=V(X,Y,Z)* зададена во СКНФ облик.

Пр. 3. $$U(X,Y,Z) = X\overline{Y}Z + X\overline{Y}\overline{Z} + XYZ + \overline{X}YZ + \overline{X}\overline{Y}Z = X\overline{Y}(Z+\overline{Z}) + YZ(X+\overline{X}) + \overline{X}\overline{Y}Z =$$
$$= X\overline{Y} + YZ + \overline{X}\overline{Y}Z = X\overline{Y} + Z(Y + \overline{X}\overline{Y}) = X\overline{Y} + Z(\overline{X}Y)$$

Пр. 4. $$V(X,Y,Z) = (X+\overline{Y}+Z)(X+Y+Z)(X+Y+\overline{Z})(\overline{X}+Y+\overline{Z}) = [(X+Z)+Y\overline{Y}]$$
$$[(Y+\overline{Z})X + \overline{X}] = (X+Z)(Y+\overline{Z})$$

Од примериве е очигледно дека добиените резултати се минимални нормални форми (МНФ), што значи дека веќе се навлегува во проблематиката на минимизација на прекинувачките функции. Тоа е едно комплексно прашање кое во поголеми детали ќе биде обработено понатаму.

Преминувањето од една НФ во друга може да се изврши со примена на дистрибутивниот закон. Меѓутоа, тоа може да биде доста макотрпно, па ние ќе го користиме преминот преку множеството на индекси. Преминувањето од ДНФ во КНФ, или обратно, се врши редоследно така што се преминува преку СНФ. Така, функцијата која е зададена во КНФ прво се проширува во аналитичка СКНФ, па потоа тој СКНФ облик се запишува во форма на множество на индекси. Понатаму се изведува СДНФ формата на функцијата преку множеството на индекси, која се запишува во СДНФ аналитички облик, што конечно се упростува со примена на соодветни теореми. За премин од ДНФ во КНФ се постапува обратно, т.е. се почнува од ДНФ во СДНФ, па во СКНФ и на крај се добива КНФ. И за двата случаи во продолжение е даден по еден пример.

Пр. 5. $$F_1(A,B,C) = A + B\overline{C} + \overline{A}\overline{B}C = A11 + 1B\overline{C} + \overline{A}\overline{B}C = A(B+\overline{B})(C+\overline{C}) +$$
$$+ (A+\overline{A})B\overline{C} + \overline{A}\overline{B}C = (AB + A\overline{B})(C+\overline{C}) + AB\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}C = ABC + AB\overline{C} +$$
$$+ A\overline{B}C + A\overline{B}\overline{C} + ABC + \overline{A}B\overline{C} + \overline{A}\overline{B}C = \sum m(7,6,5,4,6,2,1) = \sum m(1,2,4,5,6,7)$$

Пр. 6. $$F_2(A,B,C) = \prod M(0,3) = (A+B+C)(A+\overline{B}+\overline{C}) = A + (B+C)(\overline{B}+\overline{C}) =$$
$$= A + B\overline{B} + B\overline{C} + \overline{B}C + C\overline{C} = A + B\overline{C} + \overline{B}C$$

Пр. 7. $$F_3(A,B,C) = A(\overline{B}+C)(A+B+\overline{C}) = (A+0+0)(0+\overline{B}+C)(A+B+\overline{C}) =$$
$$= (A + B\overline{B} + C\overline{C}) + (A\overline{A} + \overline{B} + C)(A+B+\overline{C}) = [(A+B)(A+\overline{B}) + C\overline{C}](A+\overline{B}+C)$$
$$(\overline{A}+\overline{B}+C)(A+B+\overline{C}) = (A+B+C)(A+\overline{B}+\overline{C})(A+\overline{B}+C)(\overline{A}+\overline{B}+C)(A+B+\overline{C}) =$$
$$= \prod M(0,3,2,6,1) = \prod M(0,1,2,3,6)$$

Пр. 8. $$F_4(A,B,C) = \sum m(4,5,7) = A\overline{B}\overline{C} + A\overline{B}C + ABC = A\overline{B}1 + ABC = A\overline{B} + ABC$$

2.5. СТАНДАРДНИ ЛОГИЧКИ ФУНКЦИИ

Логичките функции кои ги извршуваат основните логички операции И, ИЛИ и НЕ (комплементирање, инвертирање), како и функциите кои ги извршуваат операциите НИ и НИЛИ, потоа ЕКСИЛИ (ИСКИЛИ) и ЕКСНИЛИ (ИСКНИЛИ), се од особена важност заради што сите нив заедно ќе ги прикажеме уште еднаш со нивните таблици на вистинитост таб. 2-17 а), б), в), г), д), ѓ) и е), вклучувајќи ги и нивните аналитички форми.

A	B	$A+B$
0	0	0
0	1	1
1	0	1
1	1	1

а) ИЛИ

A	B	$A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

б) И

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

в) НЕ

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

г) НИЛИ

A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

д) НИ

A	B	$A \oplus B$
0	0	1
0	1	0
1	0	0
1	1	0

ѓ) ЕКС ИЛИ

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

е) ЕКС НИЛИ

Таб. 2-17. Стандардни логички функции

Гледајќи ги таблиците на функциите ЕКСИЛИ и ЕКСНИЛИ можеме да заклучиме дека овие две логички функции можат да се применат за детектирање на нееднаквост, односно на еднаквост. Имено, функцијата ЕКСИЛИ има вредност 1 само ако променливите А и В меѓусебно се разликуваат, т.е. ако А=0 и В=1 или ако А=1 и В=0, додека кога А и В се еднакви функцијата ЕКСИЛИ дава резултат 0. Од друга страна функцијата ЕКСНИЛИ се однесува спротивно бидејќи нејзината вредност е 1 само ако променливите А и В меѓусебно се еднакви, т.е. ако А=0 и В=0 или А=1 и В=1, додека ако се различни, вредноста на ЕКСНИЛИ е 0. Покрај ова, од таблицата на функцијата ЕКСИЛИ може да се забележи и тоа дека таа може да се користи за аритметичко собирање во бинарниот броен систем, бидејќи ги исполнува правилата за собирање на бит со бит. Двете претходно посочени функции можат да се претстават и во аналитички облик со логички равенки, применувајќи ги на основните логички функции ИЛИ, И и НЕ (Комплементирање).

$$Y_{\text{ЕКСИЛИ}} = A \oplus B = A \cdot \overline{B} + \overline{A} \cdot B \qquad (2\text{-}20)$$

$$Y_{\text{ЕКСНИЛИ}} = \overline{A \oplus B} = A \cdot B + \overline{A} \cdot \overline{B} \qquad (2\text{-}21)$$

Множеството на оние прекинувачки функции со чија комбинација може да се реализира било која сложена функција се вика **функционално потполн систем на логички функции.**

Таков систем, на пример е множеството од елементарните функции И,ИЛИ и НЕ, бидејќи со нив може да се изрази било која друга сложена прекинувачка функција. Во изведувањето што следи покажано е на кој начин може НЕ, И и ИЛИ функциите да се изразат само со помош на НИ функцијата:

$$\overline{A} = \overline{A} + \overline{A} = \left(\overline{A \cdot A}\right) \tag{2-22}$$

$$AB = \left(\overline{\overline{A \cdot B}}\right) \tag{2-23}$$

$$A + B = \left(\overline{\overline{A + B}}\right) = \left(\overline{\overline{A} \cdot \overline{B}}\right) \tag{2-24}$$

На сличен начин ќе покажеме дека основните логички функции можат да се изведат и само со НИЛИ функцијата:

$$\overline{A} = \overline{A} \cdot \overline{A} = \left(\overline{A + A}\right) \tag{2-25}$$

$$AB = \left(\overline{\overline{A \cdot B}}\right) = \left(\overline{\overline{A} + \overline{B}}\right) \tag{2-26}$$

$$A + B = \left(\overline{\overline{A + B}}\right) \tag{2-27}$$

Според ова, секоја сложена прекинувачка функција може да се реализира и само со примена на функцијата НИ, или само со примена на функцијата НИЛИ. Ова значи дека и функцијата НИ, т.е. функциите НЕ и И формираат функционално потполн систем. Истото важи и за функцијата НИЛИ, т.е. функциите НЕ и ИЛИ. Групата функции кои го сочинуваат функционално потполниот систем се нарекуваат **универзални функции.**

Конечно може да извлечеме еден многу важен заклучок, а тоа е дека со употребата на универзалните логички функции: И,ИЛИ и НЕ, потоа само со НИ, или само со НИЛИ може да се претстави било која прекинувачка функција. Ваквата констатација одигра клучна улога во производството на електронски елементи и компоненти кои практично реализираат најразлични логички функции.

2.6. МИНИМИЗАЦИЈА НА ПРЕКИНУВАЧКИ ФУНКЦИИ

Веќе видовме дека постојат различни *нормални форми* (НФ) на една иста прекинувачка функција. Овие НФ не содржат подеднаков број на променливи и операции, па природно е да ја избереме онаа НФ која содржи минимален број членови (суми, односно производи) и минимален број на променливи по член. Постапката со која некоја зададена функција се доведува во минимална форма се вика минимизација на прекинувачките функции. Јасно, како последица од минимизацијата треба да се добие МДНФ, односно МКНФ на дадената функција.

Еден начин за да се изврши минимизацијата е аналитички, со директна примена на алгебарските трансформации, при тоа користејќи ги правилата на Буловата алгебра со кои можат да се упростат логичките изрази. При овој начин на минимизирање на прекинувачките функции фактички се преминува од еден аналитички облик во друг. Во процесот на минимизирање се тргнува од некоја НФ на функцијата, која најчесто е СНФ, па треба да се добие МНФ. Меѓутоа, овој метод е доста комплициран и не претставува сигурен пат кој ќе нè донесе до минималниот облик на прекинувачката функција.

Освен аналитичката минимизација постојат и други начини за минимизирање на логичките функции, од кои ќе го споменеме т.н. метод на Карноови карти кој ја минимизира функцијата по графички пат. Овој метод најчесто се користи за минимизирање на функции кои зависат најмногу од пет променливи, а вообичаено се користи за функции од три или четири променливи.

За функциите што треба да се минимизираат, а зависат од произволен број променливи се применува еден друг табеларен метод кој се вика *метод на Квајн Мек Класки (Quine McCluskey)*, или табличен метод. Овој метод се користи при минимизирање на функции со поголем број променливи и е доста згоден за употреба кога сакаме процесот на минимизација да го автоматизираме, користејќи компјутер.

2.6.1. АНАЛИТИЧКИ МЕТОД НА МИНИМИЗАЦИЈА

Аналитичкиот (алгебарскиот) метод за минимизација на прекинувачките функции всушност претставува упростување на зададен логички израз за што е веќе претходно пишувано во учебников. Во врска со ова, проблемот на аналитичка минимизација на логичките функции најдобро ќе го разбереме ако обработиме уште неколку конкретни примери. Само да се потсетиме дека при користењето на овој метод најважно е да се знаат теоремите на Буловата алгебра, кои овде треба да се користат како правила за поедноставување при процесот на минимизација.

Пр. 1.
$$F_1(A,B,C,D)=\overline{A+B+C+D}+ABC+\overline{A}\,\overline{B}D+A\overline{C}+ABC+AB\overline{D}+\overline{A}C=$$
$$=\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+\overline{A}\,\overline{B}D+\overline{A}C+ABC+AB\overline{D}+\overline{A}C=\overline{A}\left(\overline{B}\,\overline{C}\,\overline{D}+C\right)+\overline{A}\,\overline{B}D+$$
$$+A\left(BC+\overline{C}\right)+AB\overline{D}=\overline{A}\,\overline{B}\,\overline{D}+\overline{A}\,\overline{B}D+\overline{A}C+\overline{A}\,\overline{B}+A\overline{C}+AB\overline{D}=\overline{A}\,\overline{B}\left(D+\overline{D}\right)+$$
$$+\overline{A}C+AB\left(1+\overline{D}\right)+A\overline{C}=\overline{A}\,\overline{B}+\overline{A}C+AB+A\overline{C}=\overline{A}\left(\overline{B}+C\right)+A\left(B+\overline{C}\right)$$

Пр. 2.
$$F_2(A,B,C)=AB+\overline{A}C+BC=AB+\overline{A}C+\left(A+\overline{A}\right)BC=AB(1+C)+\overline{A}C(1+B)=$$
$$=AB+\overline{A}C$$

Функцијата од вториот пример $F_2(A,B,C,D)$ може да се минимизира и со примена на теоремата за експанзија:

Пр. 3.
$$F_2(A,B,C)=AB+\overline{A}C+BC=A\left(1B+1\overline{C}+BC\right)+\overline{A}\left(0B+0\overline{C}+BC\right)=$$
$$=A\left(B+0+BC\right)+\overline{A}\left(0+C+BC\right)=AB(1+C)+\overline{A}C(1+B)=AB+\overline{A}C$$

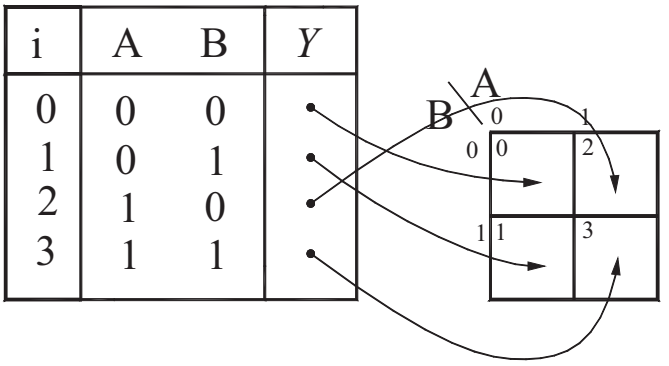
Со изведената минимизација практично се докажува теоремата (т. 2-15).

2.6.2. КАРНООВ МЕТОД НА МИНИМИЗАЦИЈА

Овој метод е прилично едноставен и доста практичен и заради тоа многу често се користи. *Минимизацијата се изведува по графички пат, така што функцијата се претставува со помош на една специјална таблица што се вика* **Карноова карта (КК).**

За да се почне со работа, неопходно е за функцијата да се формира соодветна КК која се изведува од таблицата на вистинитост на функцијата. КК претставува полигон со одреден број полиња (квадратчиња). За да се претстави било која редица од комбинационата таблица на зададената прекинувачка функција, доволно е само едно поле од КК. *Значи само со едно квадратче може да се претстави еден минтерм, односно еден макстерм од зададената функција.* КК овозможува упростување на логичките функции на многу едноставен начин: со помош на визуелно испитување на КК.

За да ја објасниме врската која постои меѓу КК и таблицата на вистинитост, најпрво ќе го прикажеме наједноставниот случај, а тоа е изгледот на КК за функција од две променливи Y(A,B) и нејзината таблица на вистинитост дадени на сл. 2-1.



Сл. 2-1. Карноова карта и таблица на вистинитост и на функција од две променливи

Ќе претпоставиме дека колоната во која треба да се внесат вредностите на функцијата е непополнета, затоа што сега сакаме да објасниме само како ќе изгледа КК. Од сликата се гледа дека за функција од две променливи КК има вкупно $2^2 = 4$ полиња. Понатаму во децимална нотација, преку индексите, се означува секоја редица од таблицата на вистинитост, и секое поле од прикажаната КК. Вака означената КК може да се користи како замена на комбинационата таблица.

Сега полињата од КК треба да се пополнат со вредностите на функцијата. Тоа пополнување оди на два различни начини:

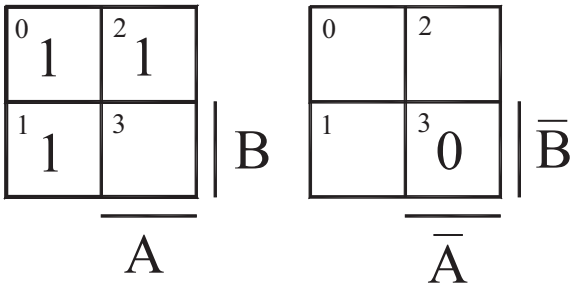
1. Во нацртаната КК може да се пополнат со 1 само оние полиња на коишто му одговара 1 во соодветната редица од вредноста на функцијата. На овој начин функцијата е прикажана во СДНФ облик, т.е. со сума од минтерми и

2. Функцијата може да се прикаже и во СКНФ облик, т.е. со производ од макстерми ако во КК се пишуваат 0 само во оние полиња кои одговараат на редиците за кои вредноста на функцијата е 0.

Како еден пример, табелата 2-18 ја претставува комбинационата таблица на функцијата НИ, додека сл. 2-2 а) ја прикажува нејзината КК во СДНФ, бидејќи се пополнети полињата со вредност 1. Сликата сл.2-2 б) ја претставува НИ функцијата во СКНФ облик, бидејќи сега во КК се пополнети полињата чија вредност е 0.

i	A	B	*Y*
0	0	0	1
1	0	1	1
2	1	0	1
3	1	1	0

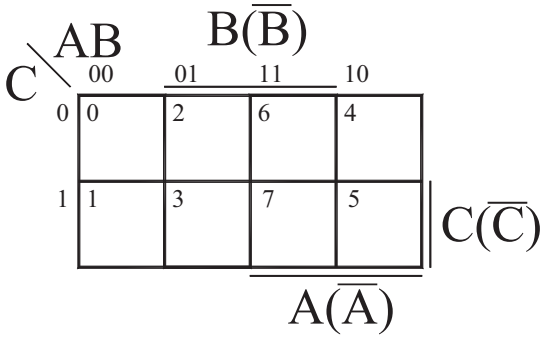
Таб. 2-18. Таблица на вистинитост на НИ функцијата



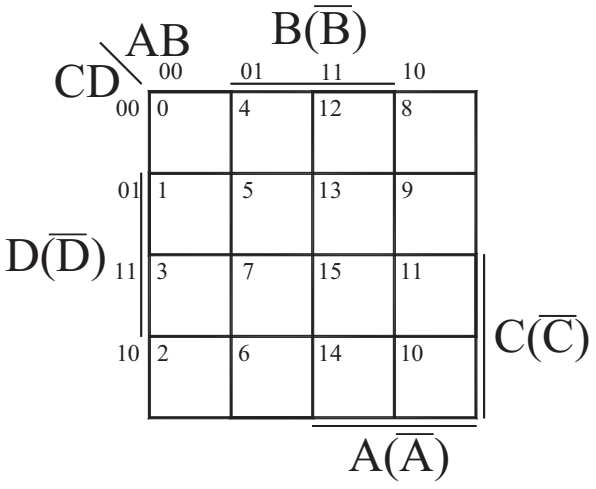
а) СДНФ облик б) СКНФ облик

Сл. 2-2. Карноови карти на НИ функцијата.

КК за функција од три променливи има $2^3 = 8$ полиња, а нејзиниот изглед е прикажан на сл. 2-3, додека КК за функција од четири променливи е даден на сл. 2-4 од каде се забележува дека таа има $2^4 = 16$ полиња.

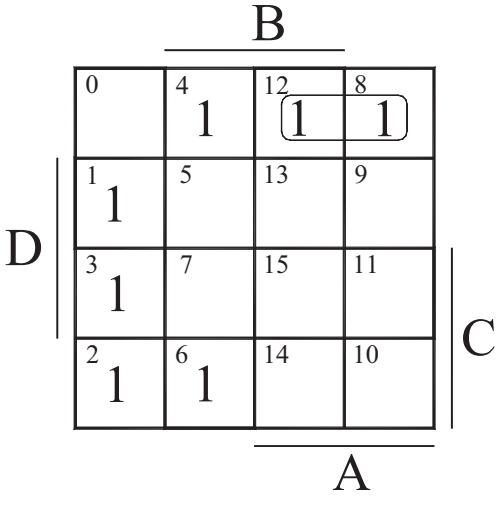


Сл. 2-3. Карноова карта на функција од три променливи



Сл. 2-4. Карноова карта на функција од четири променливи

За секоја КК треба да се запомни индексот на секое поле затоа што тој индекс одговара на соодветна редица од таблицата на вистинитост за дадената функција, но означувањето е прилично едноставно. Имено, ако КК се однесува на СДНФ обликот на функцијата, таа се пополнува со 1-и и тогаш ги наведуваме променливите во директен облик покрај страниците на КК, така што означувањето почнува од првата променлива долу десно, втората горе, третата десно и четвртата лево. За секоја страница, една половина од сите полиња ги покрива одредена променлива во директен облик што е означено со цртичка покрај страниците на КК, додека другата половина од полињата се покриени со комплементарниот облик на променливата. Ако во КК се внесат 0-те на функцијата таа се разгледува во СКНФ облик и тогаш променливите покрај цртичките се наведуваат во комплементиран облик, а за непокриените полиња нивниот облик ќе биде директен. Ваквото означување најмногу и ќе го употребуваме во текстот што следува.



Најважна особина кај КК е таа што *полињата кои меѓусебно се допираат*, хоризонтално или вертикално, одговараат на минтерми, односно макстерми, кои *се разликуваат само во една променлива,* врз основа на Грејовиот код. Оваа променлива се јавува во директен облик во членот кој одговара на едното поле, а во комплементирана форма во членот кој соодветствува на другото. Ваквите полиња ќе можат да се спојат (здружат) и затоа ќе ги викаме **споени полиња**. За да го видиме упростувањето што го нуди оваа особина ќе разгледаме еден пример за функцијата Y=*Y(A,B,C,D)* чија КК е прикажана на сл. 2-5.

Сл. 2-5. Пример на КК за функција од четири променливи $Y(A,B,C,D) = \Sigma m(1, 2, 3, 8, 12)$

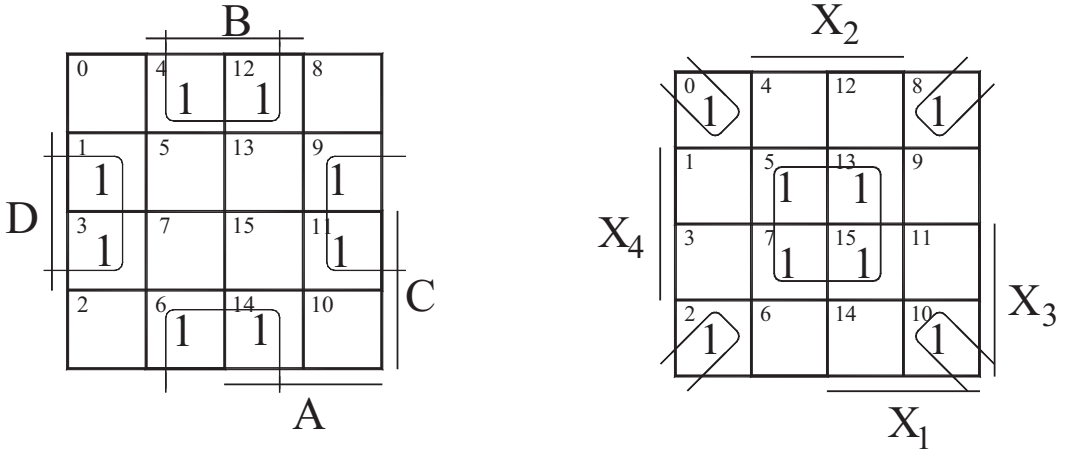
Од сл. 2-5 се гледа дека и двата минтерми $m_8 = A\overline{B}\,\overline{C}\,\overline{D}$, $m_{12} = AB\overline{C}\,\overline{D}$ се наоѓаат во споени полиња. Минтермите меѓусебно се разликуваат во тоа што променливата B во едниот член се јавува комплементирана, а во другиот член таа е во директен облик, т.е. некомплементирана. Ако овие два минтерми ги собереме ќе добиеме: $A\overline{B}\,\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} = A\overline{C}\,\overline{D}\left(B + \overline{B} = A\overline{C}\,\overline{D}\right)$.

Забележуваме дека и двата минтерми, кај кои фигурираа по четири променливи, можат да се заменат со единствен член, кој вклучува само три променливи, бидејќи се елиминирала променливата B. Овој принцип може да се применува за било кои други две полиња кои се пополнети со 1, а се споени (се наоѓаат едно до друго по хоризонтала или по вертикала).

Значи КК овозможува лесно препознавање на оние комбинации од минтерми, кои можат да се заменат со попрости изрази, и тоа со помош на геометриска визуализација. Општо земено секој пар од споени минтерми може да се комбинира во единствен член кој содржи една променлива помалку од самите минтерми. Овој член се добива така што од минтермите кои ги презентираат споените полиња се исфрла онаа променлива која се јавува некомплементирана во едниот минтерм, а комплементирана во другиот.

Треба да истакнеме и тоа дека споени полиња не се само оние што геометриски се допираат, туку такви полиња се и оние што ги претставуваат минтермите, кои се разликуваат само во појавниот облик (директен или комплементиран) на една променлива. Врз основа на ова произлегува фактот дека секое поле што се наоѓа во најлевата колона на било која редица е споено со полето што се наоѓа во најдесната колона од истата редица, и секое поле што се наоѓа во најгорната редица од било која колона е споено со полето што се наоѓа во најдолната редица и истата колона. Забележано накратко, ова значи дека горниот раб од КК се допира со долниот, а левиот со десниот. Крајните дијагонални полиња (четирите ќошиња) се споени полиња, но крајните две полиња од секоја дијагонала не се споени полиња.

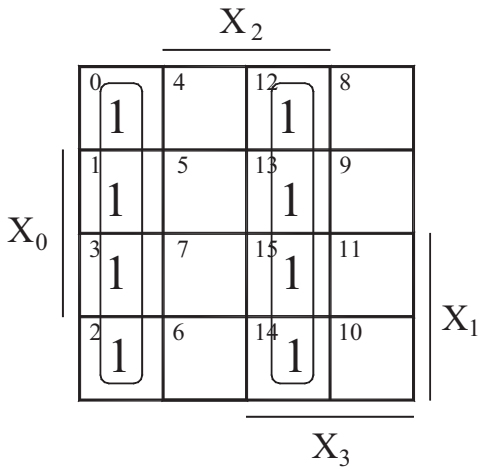
Видовме како две споени полиња од КК можат да дадат член кој елиминира една променлива. Слично може да се покаже дека *секогаш кога ќе се појават споени* $N=2^n$ полиња, од нив може да се добие само еден член кај кого се елиминирани *n* променливи, *и тоа оние што се јавуваат еднаш во комплементарна, а еднаш во директна (номинална) форма*. На сл. 2-6 а), б), в) и г) се прикажани различни начини за групирање на четири споени полиња.



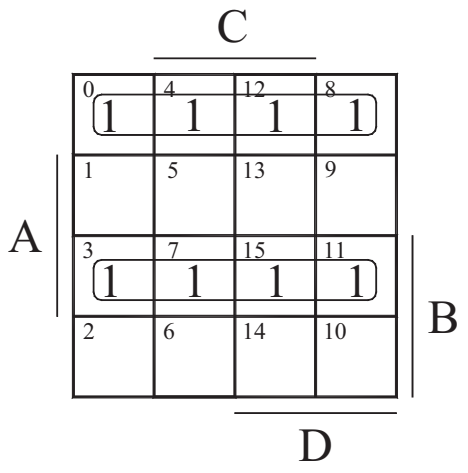
а) $F_1 = B\overline{D} + \overline{B}D$ б) $Z = X_2X_4 + \overline{X}_2\overline{X}_4$

Сл. 2-6. Карноови карти

На сл.2-6 а) е претставена КК на функцијата $F_1 = F_1\ (A,B,C,D)$, на сл. 2-6 б) на функцијата $Z=Z(X_1, X_2, X_3, X_4)$, на сл. 2-7 а) на функцијата $Y=Y(X_3, X_2, X_1, X_0)$ и на сл. 2-7 б) на функцијата $F_2 = F_2(D, C, B, A)$. На сл. 2-8 а) и б) последователно се претставени КК на функциите $W=W(X_4, X_3, X_2, X_1)$ и $F_3 = F_3(A, B, C, D)$ кај кои се спојуваат по осум полиња. Покрај секое заокружување е запишан производот кој ја претставува соодветната група на полиња.

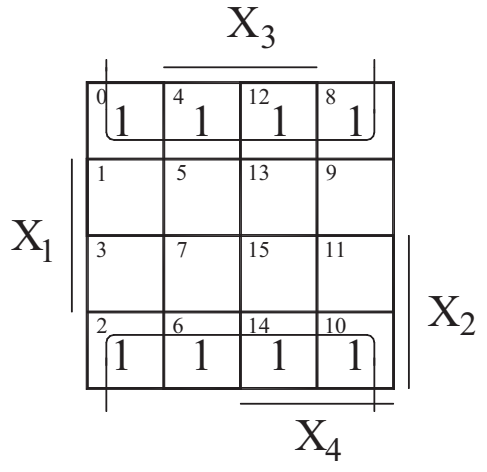


а) $Y = X_2 X_3 + \overline{X}_2 \overline{X}_3$

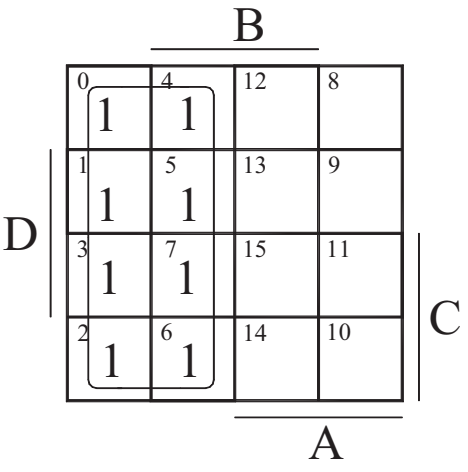


б) $F_2 = AB + \overline{A}\overline{B}$

Сл. 2-7. Примери за спојување на четири полиња кај КК за функции од четири променливи



а) $W = \overline{X}_1$



б) $F_3 = \overline{A}$

Сл. 2-8. Примери за спојување на осум полиња кај КК за функции од четири променливи

Што се однесува до логичките функции кои зависат од поголем број на променливи треба да се знае дека прегледноста ќе биде доста намалена. Така на пример за функции од 5 променливи КК би имала $2^5 = 32$ квадрати, но сèуште може да се применува. Во врска со ова, постојат два начини за прикажување на КК од 5 променливи. Според едниот се спојуваат две КК од по 16 полиња една до друга, а според вториот ваквите две КК од по 16 полиња се наоѓаат една до друга, но одвоени, при што се замислува дека тие се наоѓаат една над друга. За функција од 6 променливи ќе треба КК со $2^6 = 64$ полиња. Јасно е дека во овој случај КК ќе стане премногу голема и работењето со КК ќе стане посложено до таа мерка што таа практично ќе биде неупотреблива.

2.6.2.1. ПРИМЕНА НА КАРНООВИОТ МЕТОД

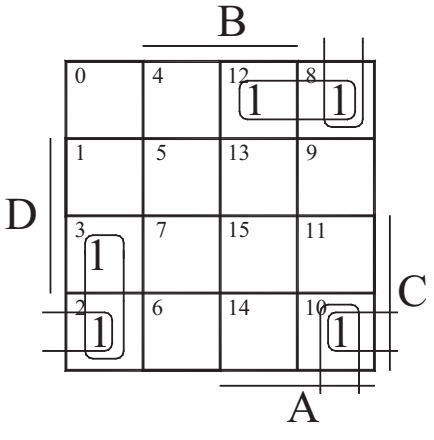
Во натамошното излагање ќе го објасниме начинот според кој треба да се примени Карноовиот метод за минимизација на зададена прекинувачка функција. Функцијата е дадена преку својата таблица на вистинитост или преку некоја СНФ: СДНФ или СКНФ. Од било кој од наведените облици може да се формира и да се пополни КК за дадената функција. Она што досега го зборувавме се однесуваше на функции кои се зададени во СДНФ облик, па затоа ќе продолжиме со објаснувањето за примената на КК на оние функции што се дадени во СДНФ, т.е. за КК пополнети со 1.

Минимизација на функции зададени во СДНФ. Од претходното објаснување интуитивно се наметнува заклучок за начинот според кој ќе се врши минимизација на дадена функција. Имено, ќе мора да се опфатат сите полиња пополнети со 1-и затоа што секое такво поле претставува некој минтерм на функцијата. При ова треба да се формираат што помалку групи на важечки (валидни) површини на споени полиња од што поголем ранг (r). Важечка површина од ранг r се формира со групирање само на 2^r број на полиња кои имаат (n-r) заеднички променливи каде r е позитивен цел број за кој важи $k \le n$, а n е вкупниот број на независно-променливите од кои зависи дадената функција. Со ова би добиле најмалку производи од кои секој би имал најмал број на променливи.

Значи секогаш кога ќе се упростува дадена функција со помош на КК треба да се следат следниве принципи:

1. Комбинациите од избраните полиња мора да се такви што секое поле мора да се појави најмалку еднаш во групите, т.е. барем во една група на полиња. При ова едно поле може да биде вклучено и во повеќе различни групи;
2. Поодделните групи треба да се избираат така што во секоја група да влегува што е можно поголем број полиња, но притоа да се добијат што е можно помал број различни групи.

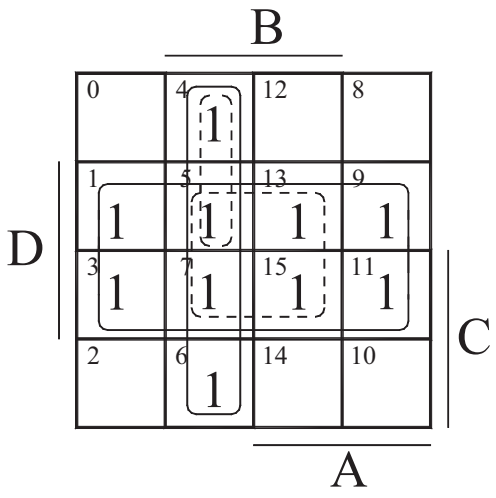
Било кој производ на променливите се вика импликанта. Импликанта се добива со заокружување на едно, две, четири итн, полиња во една група. **Примарна импликанта (ПИК)** е онаа импликанта која не е целосно вклучена во некоја друга импликанта, т.е. заокружената група на полиња да не е целосно покриена со некоја друга заокружена група. За да се претстави функцијата, во општ случај, не мора да се употребат сите ПИК. Кажаното подобро ќе го разбереме на следниов типичен пример.



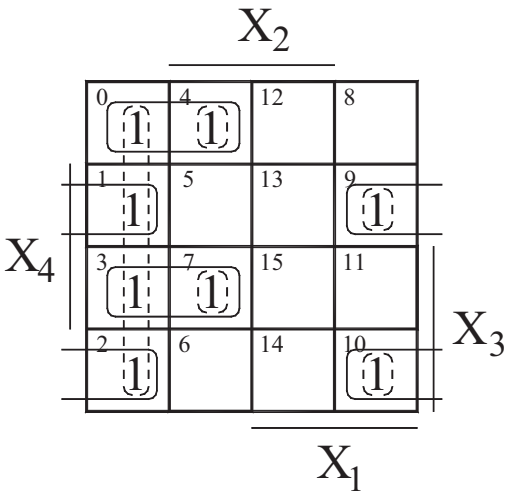
Сл. 2-9. КК на функцијата $Y_1 = Y_1(A, B, C, D)$

Станува збор за функцијата $Y_1 = Y_1(A, B, C, D)$ чија КК е прикажана на сл. 2-9. Овде ПИК се: $p_1 = m_2 + m_3, p_2 = m_8 + m_{12}, p_3 = m_2 + m_{10}, p_4 = m_8 + m_{10}$ Функцијата може да се претстави во два облици: или како $Y_1 = p_1 + p_2 + p_3$ или како $Y_1 = p_1 + p_2 + p_4$. И во двете прикажувања мораме да ја искористиме ПИК p_1, инаку минтермот m_3 нема да биде земен предвид. Токму поради ова p_1 се вика **суштинска (есенцијална) импликанта (СИК).** Значи *СИК* е *ПИК* која покрива барем еден минтерм (поле со 1), што не е покриен ниту со една друга ПИК. ПИК p_2 е, исто така, СИК, затоа што без неа ќе го нема минтермот m_{12}. ПИК p_3, p_4 не се СИК.

З*а претставување во МДНФ мора да се земат предвид сите СИК, и онолку ПИК за да се допокријат сите останати 1 на функцијата.* При ова, *минимизацијата е економично изведена ако се добијат што е можно помалку ПИК, а секоја ПИК да има што повеќе полиња.*



Сл. 2-10. КК на функцијата $Y_2= Y_2\ (A, B, C, D)$

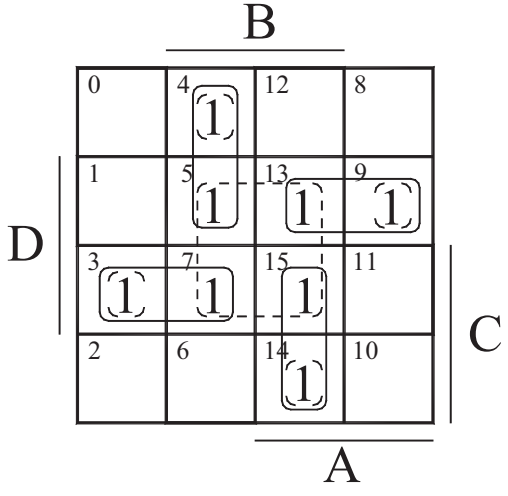


Сл. 2-11. КК на функцијата $Y_3= Y_3(X_1, X_2, X_3, X_4)$

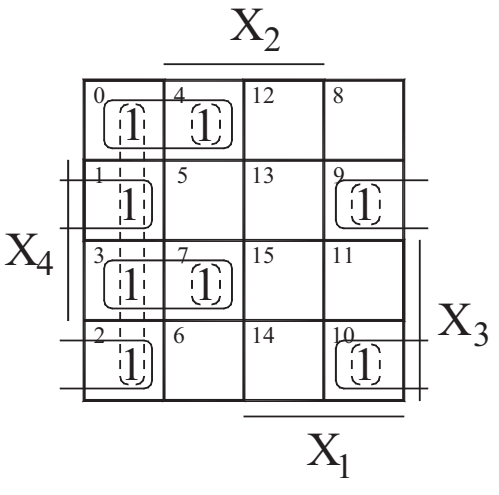
На сл. 2-10 е прикажана КК на функцијата $Y_2=Y(A,B,C,D)$, на која со испрекинати линии се означени две импликанти, а со полна линија две ПИК, кои во примеров се и СИК. На сл. 2-11 е прикажан еден пример за КК на функцијата $Y_3= Y_3(X_1, X_2, X_3, X_4)$, каде со испрекината линија се означени ПИК, а со полна линија СИК.

Тенденцијата најнапред да најдеме ПИК со што е можно поголем број полиња може да резултира со форма на функцијата која не е минимална. На пример, да ги разгледаме следниве случаи.

Функцијата $Z_1=Z_1(A, B, C, D)$ има КК која е дадена на сл. 2-12, а КК на функцијата $Z_1= Z_2(X_1, X_2, X_3, X_4)$ е прикажана на сл.2-13.



Сл. 2-12. Функцијата $Z_1= Z_1(A, B, C, D)$



Сл. 2-13. КК на функцијата $Z_2= Z_2(X_1, X_2, X_3, X_4)$

Следејќи го претходно искажаниот принцип би требало и на двете слики да извршиме заокружување на полињата според испрекинатите линии. Меѓутоа, се гледа дека таквото заокружување не дава наједноставно и најпросто решение. Дека навистина е така покажува вториот начин на заокружување со полна линија кој очигледно дава поедноставна претстава за функциите.

За да добиеме минимален облик на функцијата, а при тоа да избегнеме несакана грешка, треба да го користиме следниов алгоритам:

1. Да се заокружат и да се земат за СИК оние полиња кои не можат да се комбинираат со ниту едно друго поле;

2. Да се идентификуваат и да се групираат во двојки оние полиња што се соседни со некое друго поле, но на единствен начин;

3. Да се идентификуваат оние полиња со 1-ци, што можат да се групираат со други три полиња во четворка на единствен начин;

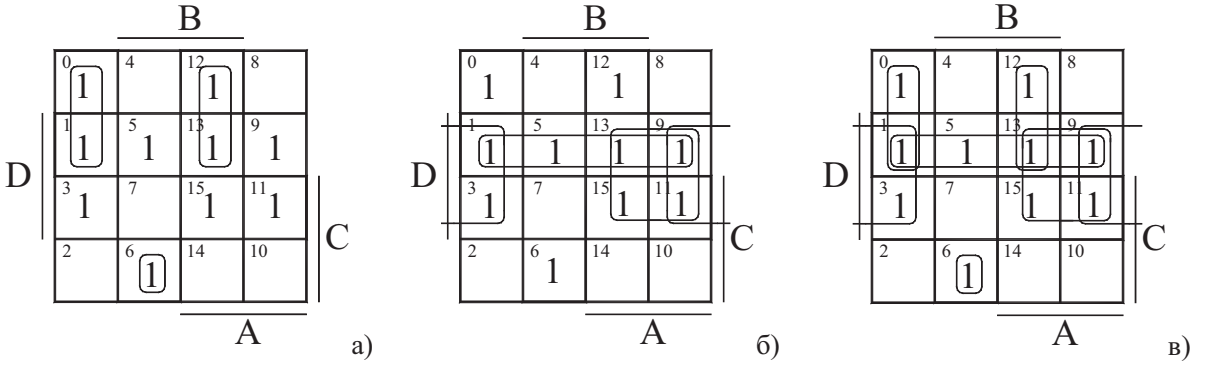
4. Оваа постапка продолжува и за групи од 8 полиња;

5. Ако по извршените 4 чекори останат непокриени полиња, тие можат да се групираат во што е можно помалку, но поголеми групи од 1-и.

Поинаку кажано, се проверуваат сите единици во КК една по една и тоа најнапред се земаат сите оние 1-и кои мора да бидат сами бидејќи не можат да формираат поголема група. Потоа се проверува секоја од преостанатите незаокружени 1-и дали формира група од две полиња во кои постои барем една незаокружена 1-ца која не може да се покрие со ниту една друга освен со онаа што се проверува. Ако има вакви двојки и тие се заокружуваат. Постапката продолжува со проверување на секоја останата и непокриена 1-а дали формира четворка во која има барем едно поле кое не формира четворка со други непокриени полиња. Ако постојат вакви групи од по четири 1-и и тие се заокружуваат, итн. тестирањето продложува со проверка за групи од по осум 1-и. Доколку останат непокриени 1-ци тие треба да се вклучат во што помалку групи кои покриваат што повеќе 1-и, и тоа што повеќе непокриени 1-и.

Со овој алгоритам практично прво се одредуваат сите СИК, а од преостанатите, се одбираат што помалку ПИК со што поголем број полиња, кои ќе ги опфатат сите 1-и на функцијата барем по еднаш.

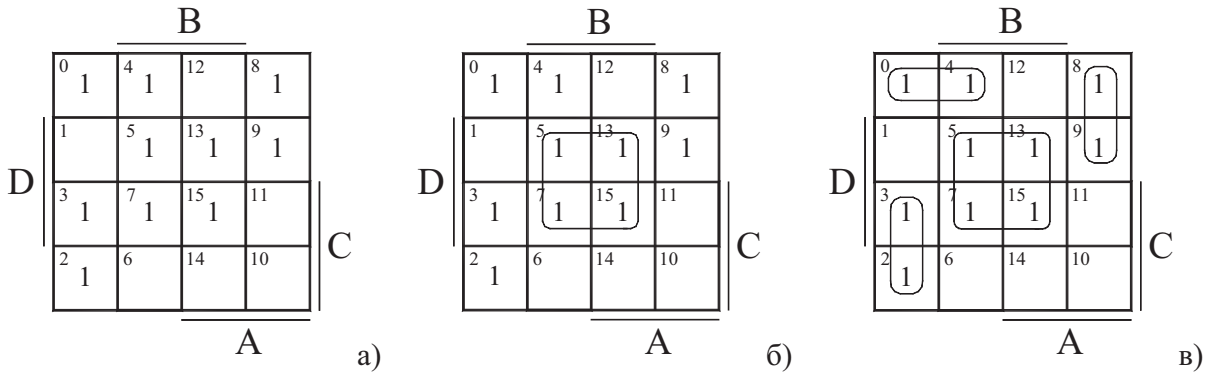
Наведениот алгоритам е во целост применет на следните два решени примери на функцијата $F_1 = F_1(A, B, C, D)$ чија КК е прикажана на сл.2-14 а),б),в) и на функцијата $F_2 = F_2(A, B, C, D)$ чија КК е претставена на сл. 2-15 а), б), в). Вториот пример е поспецифичен, бидејќи се јавува и петтиот чекор од алгоритмот, бидејќи треба сами да размислиме и да одлучиме на кој начин ќе се врши групирањето на преостанатите непокриени полиња.



Сл. 2-14. КК на функцијата $F_1 = F_1(A, B, C, D)$

За првиот пример на функцијата F_1 ја добиваме следнава минимална форма:

$F_1 = ABC\overline{D} + AB\overline{C} + \overline{A}\overline{B}\overline{C} + AD + \overline{B}D$.



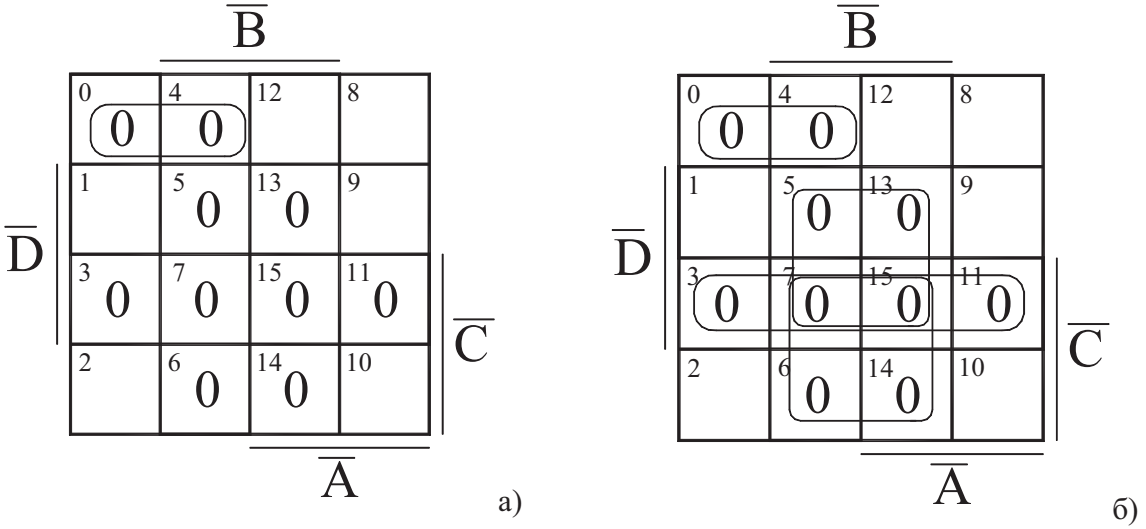
Сл. 2-15. КК на функцијата $F_2 = F_2(A, B, C, D)$

За вториот пример минималната форма на функцијата F_2 е:

$F_2 = BD + A\overline{B}\overline{C} + \overline{A}\overline{C}\overline{D} + \overline{A}\overline{B}C$.

Минимизација на функции зададени во СКНФ. Бидејќи секоја логичка функција може да биде зададена и со својот СКНФ станува јасно дека во овој случај ќе се разгледува КК која е пополнета со 0-и, а не со 1-и. Минимизацијата на вака зададените функции во принцип ќе биде идентична како и онаа кога функцијата беше дадена во СДНФ облик кога КК беше пополнета со 1, само што сега групирањето и заокружувањето ќе се однесува на полињата пополнети со 0-и. Конечно, функцијата ќе се добие во МКНФ, а не во МДНФ облик.

Во овој случај ќе го воведеме терминот **имплицента**. *Секоја имплицента презентира еден член, било која сума, во вкупниот производ*. **Примарната имплицента (ПИЦ)** е онаа што не е целосно вклучена во некоја друга имплицента. **Суштинската имплицента (СИЦ)** е ПИЦ која покрива барем еден макстерм (поле со 0), која не е опфатена ниту со една друга ПИЦ. Принципот на работа е идентичен со претходниот случај, а како илустрација за минимизирање на функција во СКНФ облик ќе го решиме примерот на функцијата $Z=Z(A, B, C, D)$ чија КК е прикажана на сл.2-16 а) и б).



Сл. 2-16. КК на функцијата $Z=Z(A, B, C, D)$

За последниов пример на функцијата Z која е минимизирана во МКНФ се добива:

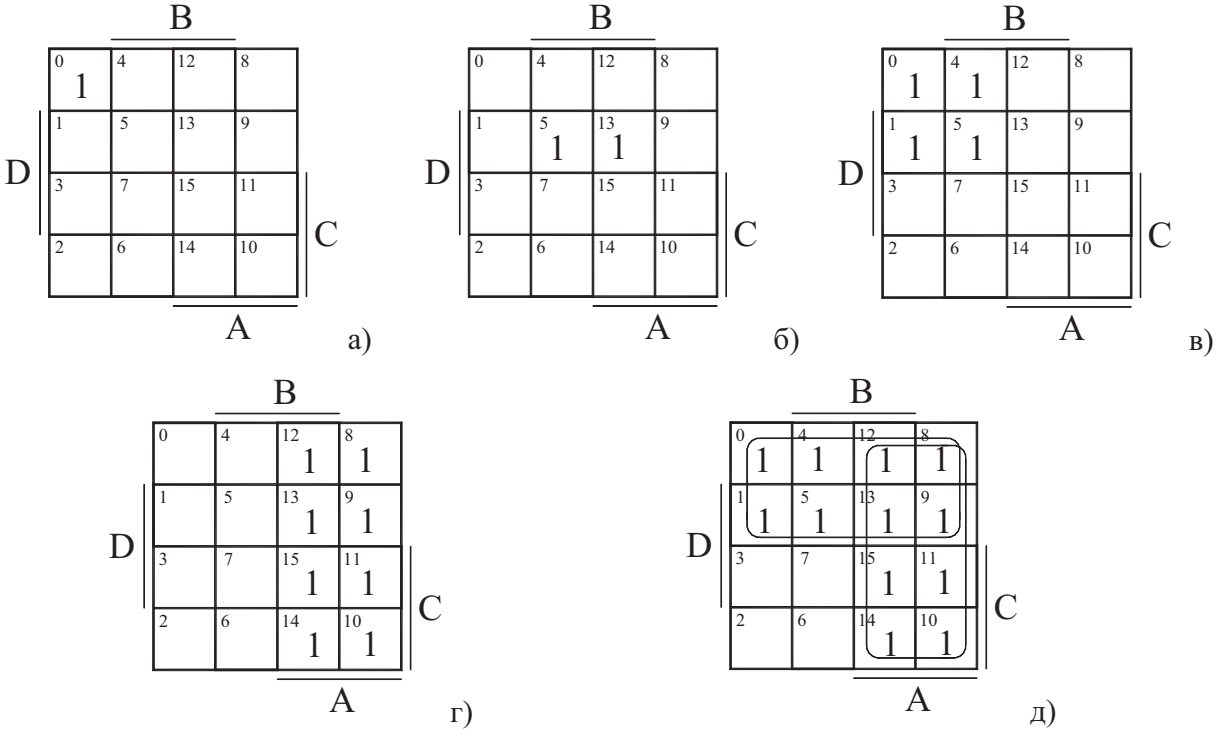
$Z = (A + C + D)(\overline{B} + \overline{C})(\overline{B} + \overline{D})(\overline{C} + \overline{D})$.

2.6.2.2. МИНИМИЗАЦИЈА НА ФУНКЦИИ ДАДЕНИ ВО ДНФ/КНФ

Во досегашната дискусија објаснивме на кој начин може да се користи методот на КК, но ако беше дадена таблицата на вистинитост на функцијата, што значи, нејзината СДНФ или СКНФ. Меѓутоа, понекогаш може да се сретнат функции кои се зададени во ДНФ или КНФ. За ваквите случаи покажавме како може да се изврши проширување на алгебарски начин на зададената ДНФ или КНФ до СДНФ, односно СКНФ, а потоа да се формира таблицата на вистинитост на функцијата, па дури потоа од неа и нејзината КК. Бидејќи ваквото аналитичко проширување може да биде доста комплицирано и макотрпно, а освен тоа потребен е уште еден чекор при решавањето: формирањето на комбинационата таблица, сега ќе покажеме како може директно од зададена функција во ДНФ или КНФ да се пополнува нејзината КК. Примерот се однесува на функцијата *Y=Y(A, B, C, D)* која е дадена со својот ДНФ облик: $Y = \overline{A}\overline{B}\overline{C}\overline{D} + B\overline{C}D + \overline{A}\overline{C} + A$.

Запишувањето на 1 во полињата од КК оди постапно и тоа почнувајќи од првиот производ, па сé до последниот. Првиот производ $\overline{A}\overline{B}\overline{C}\overline{D}$ е минтерм и тој може директно да се запише во КК на функцијата и тоа како m_0. Вториот производ е $B\overline{C}D$ и тој му одговара на она место од КК за кое важи *B=1, C=0, D=1* и не зависи од вредноста на променливата A, т.е. важи и за двете вредости на A. Така со овој член ќе се пополнат двете полиња за кои е исполнето *A=0, B=1, C=0, D=1* (m_5), *A=1, B=1, C=0, D=1*(m_{13}). Слично како во претходниот случај за третиот член $\overline{A}\overline{C}$ ќе се пополнат четири полиња: оние за кои важи *A=0, C=0* за двете вредности на B и за двете вредности на D. Конечно заради последниот член A ќе се пополнат осум полиња: оние за кои A=1, и за двете вредности на секоја од останатите три променливи B, C и D.

Целиот овој процес етапно е прикажан на сл. 2-17 а), б), в), г) и д) каде е претставен конечниот изглед на КК, кој претставува комбинација од пооделните пополнувања.



Сл. 2-17. Процес на минимизација на функција што е зададена во ДНФ облик

Од постапката се гледа дека неколку минтерми: m_0, m_5, m_{13} се пополнети со 1-и повеќе од еднаш. Меѓутоа ова не претставува додатен проблем затоа што ако минтермот се собере сам со себе се добива истиот тој минтерм само еднаш. На последната слика извршена е минимизацијата од која се добива дека МДНФ за функцијата е $Y = A + \overline{C}$.

2.6.2.3. МИНИМИЗАЦИЈА НА НЕЦЕЛОСНО ЗАДАДЕНИ ФУНКЦИИ

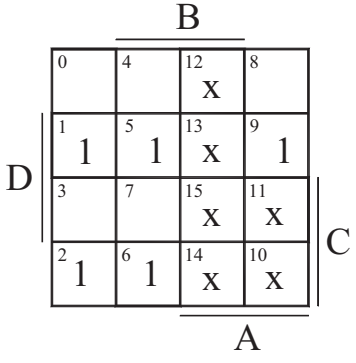
Досега видовме како се минимизираат комплетно дефинираните логички функции. Вредноста на ваквата функција секогаш беше 0 или 1, за секоја комбинација на независно променливите. Со вака зададената функција можеше веднаш да се пополнат полињата од КК на кои им одговараа минтерми или макстерми, и нормално потоа да се добие функцијата во нејзината минимална форма. Меѓутоа, во практиката се сретнуваат и некомплетно зададени функции.

За да ја илустрираме процедурата според која се врши минимизирањето на некомплетно дефинираните функции, ќе го разгледаме следниот пример, кој се однесува на функцијата *Y=Y(A,B,C,D),* што е зададена со следниов ДНФ облик:

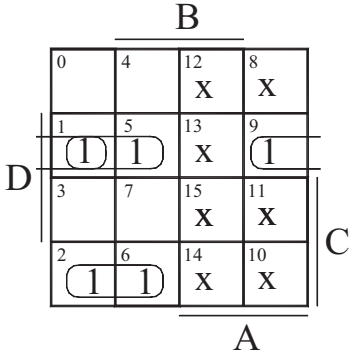
$$Y = \sum m(1,2,5,6,9) + \sum_{xm} m(10,11,12,13,14,15).$$

Оваа форма значи дека функцијата *Y=1* за секој минтерм: m_1, m_2, m_5, m_2, m_9, додека нејзината вредност не е битна за комбинациите на независно променливите на кои им одговараат минтермите: $m_{10}, m_{11}, m_{12}, m_{13}, m_{14}, m_{15}$. Затоа во полињата кои им одговараат на последно наведените минтерми ќе го запишеме симболот „*x*“.

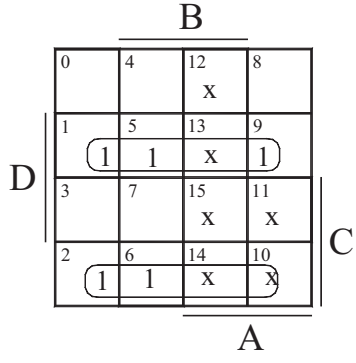
Изгледот на КК е прикажан на сл. 2-18. Во натамошната работа најважно е тоа што „*x*”-ите можеме да ги интерпретираме по наш избор: *како 1, ако со тоа се поедноставува минимизацијата, или како 0, т.е. едноставно да ги игнорираме ако со ништо не придонесуваат во понатамошното упростување на функцијата*.



Сл. 2-18. КК на функцијата *Y= Y(A, B, C, D)*



Сл. 2-19. Минимизација без “x”



Сл. 2-20. Минимизација со “x”

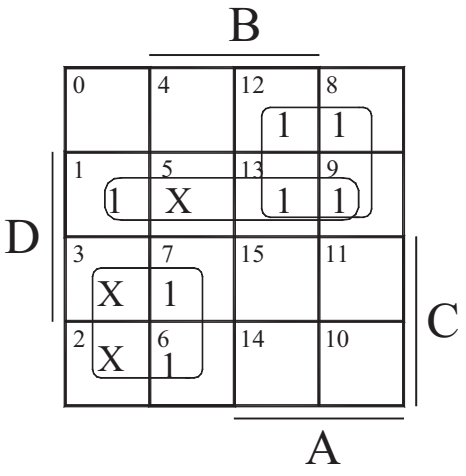
На сл. 2-19 е извршено минимизирање така што не е земено во вид ниту едно поле пополнето со „*x*”. Во овој случај за функцијата се добива изразот:

$$Y = (m_1 + m\ ) + (m_1 + m_9) + (m_2 + m_6) = \overline{A}\,\overline{C}D + \overline{B}\,\overline{C}D + \overline{A}C\overline{D}\ .$$

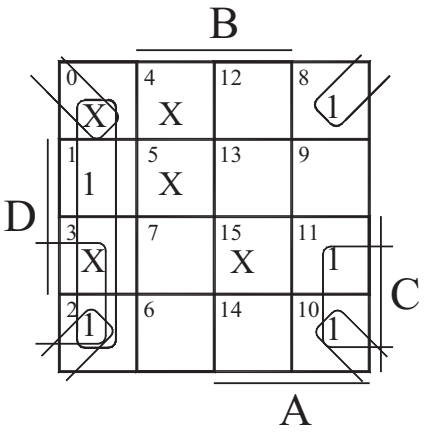
На сл. 2-20 „*x*”-ите што се наоѓаат на местата од m_{13}, m_{14}, m_{10} ги интерпретираме како 1-и со што добиваме една друга форма која е попроста и поедноставна од претходната:

$$Y = (m_1 + m_5 + m_9 + m_{13}) + (m_2 + m_6 + m_{10} + m_{14}) = C\overline{D} + \overline{C}D\,.$$

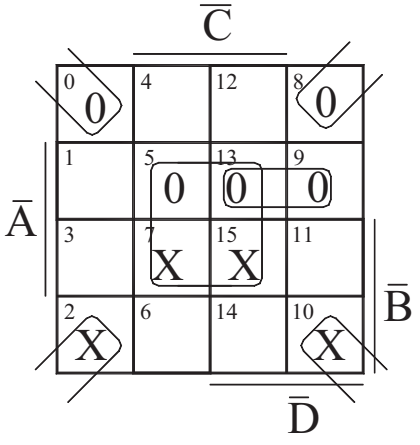
Останатите „*x*”-и кои се наоѓаат на местата од m_{11}, m_{12}, m_{13} не можат да ни послужат за редуцирање на бројот на членовите на функцијата, ниту пак за намалување на бројот на променливи во секој член, така што овие „*x*”-и едноставно ги третираме како да се 0. Заради дополнително појаснување на процесот за минимизација на некомплетно дефинирани функции со методот на Карноови карти, на сл. 2-21 а), б), в), г), д) и ѓ) се прикажани неколку нови примери на функции кои зависат од четири променливи.



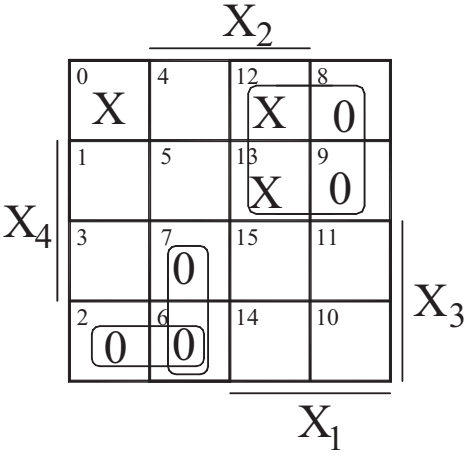
а) $Y_1 = AB + \overline{C}D + \overline{A}C$



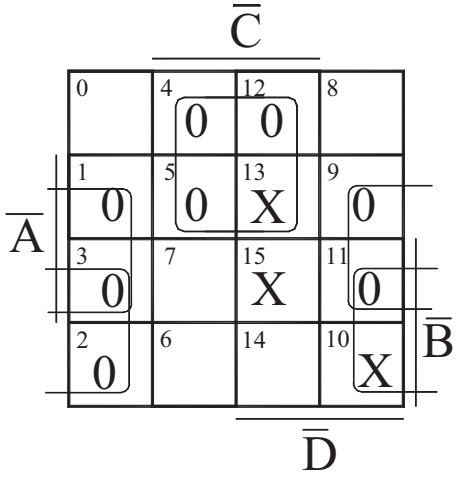
б) $Y_2 = \overline{B}\overline{D} + \overline{A}\overline{B} + \overline{B}C$



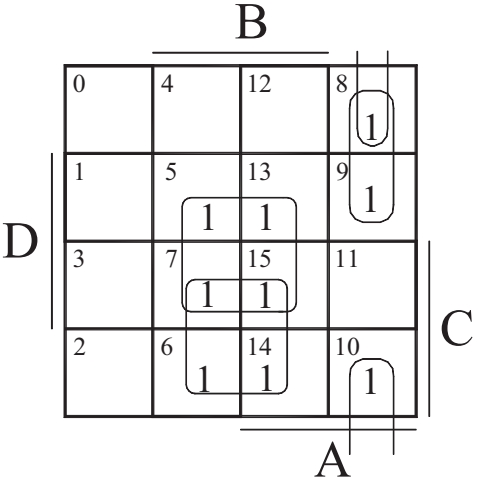
в) $Y_3 = (C + A)(\overline{C} + \overline{A})(\overline{D} + B + \overline{A})$



г) $Y_4 = (X_1 + \overline{X}_3)(\overline{X}_1 + X_2 + X_3)(\overline{X}_1 + X_3 + \overline{X}_4)$



д) $Y_5 = (\overline{C} + B)(C + \overline{A})(C + \overline{B})$



ѓ) $Y_6 = BD + BC + A\overline{B}\overline{C} + A\overline{B}\overline{D}$

Сл. 2-21. Минимизација на функции од четири променливи
со примена на методот на Карноови карти

2.7. ПРЕКИНУВАЧКИ МРЕЖИ

Логичките функции практично можат да се реализираат на различен начин применувајќи различни технички решенија, но секогаш користејќи елементи и/или компоненти кои имаат две состојби: механички прекинувачи, релеа, полупроводнички прекинувачи и сл. Во секој случај во основа се јавува потребата од нивно графичко прикажување со помош на соодветни шеми и симболи. Имајќи во вид дека нашата цел е реализација на логичките функции со електронски прекинувачи во продолжение фокусот ќе го ставиме врз објаснување на оваа проблематика, поточно врз графичкото прикажување на логичките функции со помош на логички кола и нивни симболички ознаки.

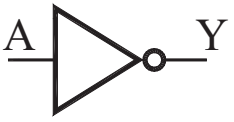
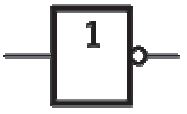
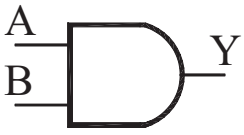
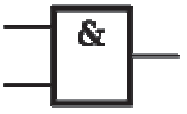
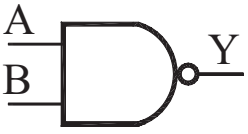
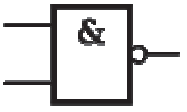
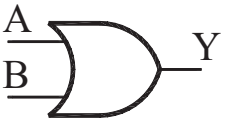
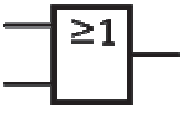
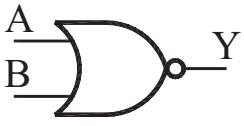
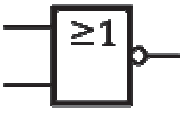
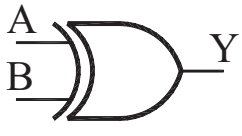
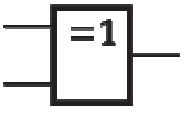
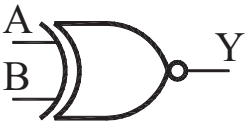
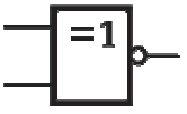
Секоја структура која е добиена со адекватно поврзување на одреден број различни логички кола и реализира некоја прекинувачка функција се вика **логичка, комбинациона, комутациона или прекинувачка мрежа**. Прекинувачката мрежа може да се прикаже со соодветна блок-шема (логички блок-дијаграм). Тоа е графички облик на претставување, а се добива со примена на симболите на логичките кола. Кај комбинационите мрежи не постои повратна врска од излезот на некое логичко коло до било кој влез во мрежата. Заради ова секој излез на мрежата постои и зависи само од моменталните вредности на влезните променливи.

2.7.1. ОСНОВНИ ЛОГИЧКИ КОЛА

Сите стандардни прекинувачки функции кои досега ги анализиравме технички се реализирааат со помош на посебни компоненти кои се викаат **логички кола, врати или порти.** Секое логичко коло има еден излез, кој одговара на функцијата која тоа коло ја извршува, и еден или повеќе влезови преку кои се доведуваат променливите од кои зависи функцијата на излезот.

За секое логичко коло постои соодветен графички симбол - елементарен блок-дијаграм, т.е. **логички симбол,** со кој тоа коло се прикажува во логичките дијаграми. Во литературата можат да се сретнат различни симболи за означување на логичките кола. IEEE (Институтот на инжинери од електротехничка струка) го прифати и го стандардизира означувањето на логичките кола воведено од страна на IEC (Интернационалната комисија за електротехника). Според овој стандард постојат во основа два различни типа на симболи. Во едната група припаѓаат симболите кои имаат различен облик зависно од нивната логичка функција, додека во втората група се користат симболите со правоаголен облик. Американскиот институт за национални стандарди (ANSI) ги применува токму симболите со различен облик кои вообичаено се користат и на светско ниво за едукативни потреби. Од друга страна, симболите со правоаголен облик се употребуваат од страна на призведувачите на дигитални компоненти, уреди и апарати за водење на документација. Имајќи го во предвид претходно кажаното, во понатамошното излагање ќе го користиме означувањето според ANSI стандардот, т.е. симболите кои имаат различен облик. Сепак, во табелата таб. 2-19 е даден целосен и споредбен преглед на симболите на основните (елементарните) логички кола И, ИЛИ и НЕ, т.е. на инверторот, како и на универзалните НИ и НИЛИ колата и нивното означување според двата стандарда. Дополнително, покрај секое логичко коло е наведена неговата таблица на вистинитост и функцијата што ја извршува во аналитички облик со кусо објаснување.

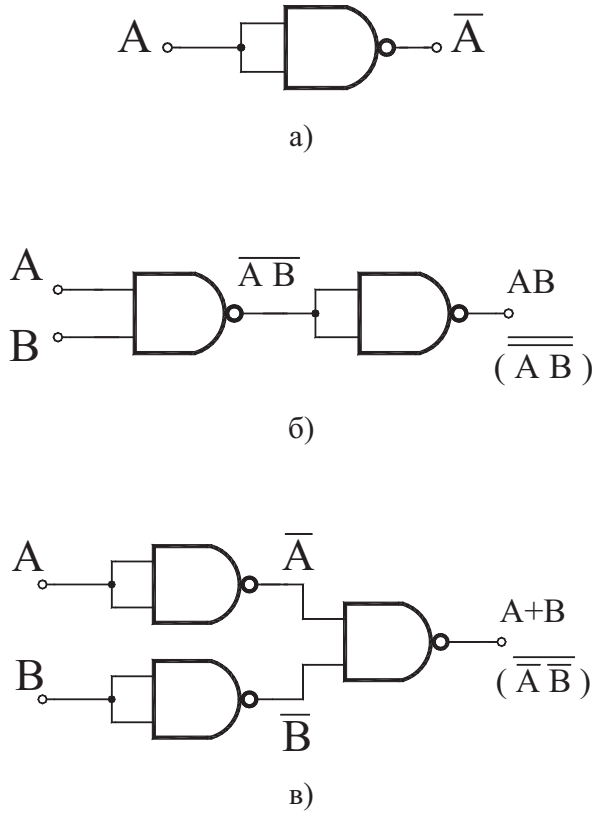
Од симболите на логичките функции на ИНВЕРТОРОТ (НЕ), НИ и НИЛИ колото се воочува дека комплементирањето на променливата се означува со малечок круг (○).

Логичко коло	Логички симбол		Логичка равенка	Таблица на вистинитост	Опис на функцијата
	ANSI	IEC/IEEE			
Инвертор (НЕ)			$Y = \overline{A}$	A | Y 0 | 1 1 | 0	Комплементирање (Инвертирање)
И			$Y = A \cdot B$	A B | Y 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1	Логичко множење
НИ			$Y = \overline{A \cdot B}$	A B | Y 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0	Комплемент на логичко множење
ИЛИ			$Y = A + B$	A B | Y 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1	Логичко собирање
НИЛИ			$Y = \overline{A + B}$	A B | Y 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0	Комплемент на логичко собирање
ЕКС-ИЛИ			$Y = A \oplus B$ $Y = A\overline{B} + \overline{A}B$	A B | Y 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0	Ексклузивно (Исклучиво) логичко собирање
ЕКС-НИЛИ			$Y = \overline{A \oplus B}$ $Y = AB + \overline{A}\overline{B}$	A B | Y 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1	Комплемент на Ексклузивно (Исклучиво) логичко собирање

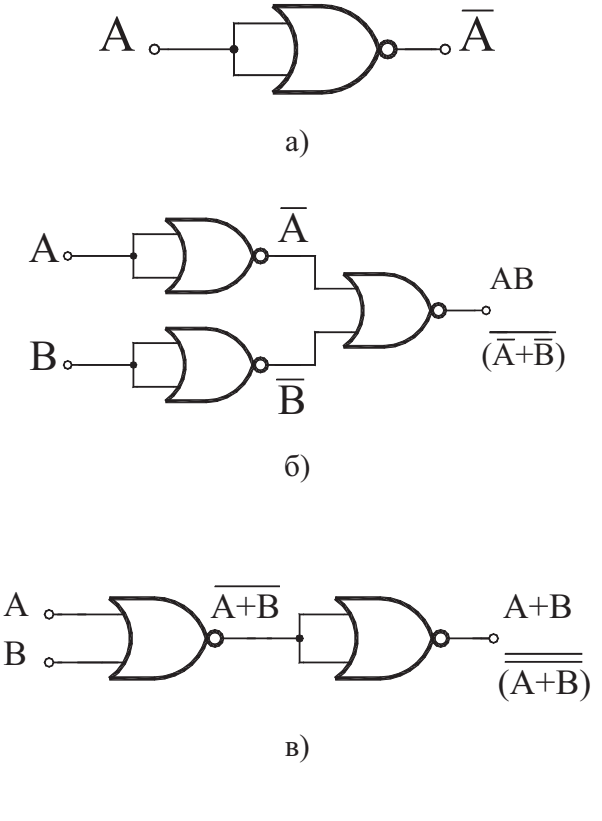
Таб. 2-19. Преглед на симболите на стандардните логички кола

Во последните две редици од табелата 2-19 се дадени и логичките кола кои ги реализираат логичките функции ИСКЛУЧИВО (ЕКСКЛУЗИВНО) ИЛИ и НИЛИ, т.е. ЕКСИЛИ и ЕКСНИЛИ (ИСКИЛИ и ИСКНИЛИ), бидејќи и тие многу често се сретнуваат и во теоријата и во практиката.

Како посебно значајни ќе ги истакнеме НИ и НИЛИ логичките кола, кои ги извршуваат универзалните функции НИ и НИЛИ, бидејќи со нив може да се реализира било која прекинувачка функција. Поаѓајќи од равенките (2-22, 2-23 и 2-24) со кои елементарните функции НЕ, И и ИЛИ се изразуваат само преку функцијата НИ, на сл. 2-22 а), б) и в) е прикажан начинот, според кој може основните логички кола НЕ, И и ИЛИ да се реализираат само со примена на НИ кола. Од друга страна, применувајќи ги равенките (2-25, 2-26 и 2-27), на сл.2-23 а), б) и в) се претставени основните логички врати само со соодветно поврзување на НИЛИ кола.



Сл. 2-22. Само со НИ порти



Сл. 2-23. Само со НИЛИ порти

Реализација на основните логички кола НЕ (Инвертор), И и ИЛИ

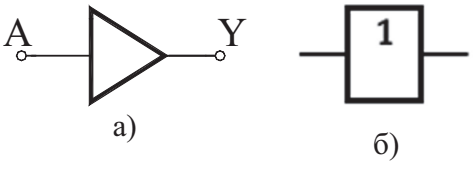
2.7.2. ДРУГИ БАЗИЧНИ ЛОГИЧКИ КОЛА

Покрај основните и универзалните логички кола во праксата многу често се сретнуваат и други базични логички кола кои имаат посебна намена. Станува збор за баферското коло, колото со три состојби и билатералната порта кои ќе ги анализираме во продолжение бидејќи овие кола имаат исклучително важна улога во проектирањето и изведбата на реални дигитални компоненти, а со тоа и голема практична примена.

Воведувањето на овие кола произлезе од реалните проблеми во практичната работа. Поконкретно, од потребата за поврзување на излезот од некое логичко коло со друг елемент или компонента која има потреба од поголема струја од онаа што може реално да му ја обезбеди тоа логичко коло, како и од потребата за поврзување на повеќе излези од различни логички кола во една точка.

2.7.2.1. БАФЕРСКО КОЛО

Најнапред ќе го претставиме колото за прилагодување (баферот, анг. *buffer*) кој има еден влез и еден излез исто како инверторот, само што кај баферското коло излезот ја следи логичката состојба на влезот, т.е. на излезот се добива исто логичко ниво со она што е присутно на влезот. На сл. 2-24 а) и б) се прикажани логичките симболи на баферот и тоа според двата стандарди: ANSI и IEC/IEEE. Имајќи го предвид однесувањето на баферот, неговата логичка равенка ќе биде Y = A, додека таблицата на вистинитост табела таб.2-20 ќе има облик соодветен на неа.



Сл. 2-24. Симболички ознаки

$$Y = A.$$

Логичка равенка

A	Y
0	0
1	1

Таб. 2-20. Комбинациона таблица

Баферско коло (бафер)

Главна карактеристика на баферското коло е неговата можност да дава поголема излезна струја при исто логичко ниво. Заради ваквата особина баферот се поврзува на излезот на она логичко коло кое не може директно да се поврзи на потрошувач со мала отпорност бидејќи постои опасност потрошувачот да повлече поголема струја од логичкото коло и истото да го преоптовари или оштети. Баферот служи како склоп за прилагодување и индиректно поврзување на излезот од дадено логичко коло со потрошувач кој може да повлече поголема струја од максимално дозволената. Овие кола се користат таму каде што е потребна поголема моќност од моќноста што некое логичко коло може да ја даде и затоа баферското коло уште се нарекува и погонско логичко коло или коло за побудување, логичко коло со моќен излез или драјвер (анг. *driver*).

2.7.2.2. БАФЕРСКО КОЛО СО ТРИ СОСТОЈБИ

Ова коло уште се вика и три-статички бафер, а својот назив го носи од англиската терминологија каде се сретнува под поимот three-state или tri-state buffer бидејќи покрај двете вообичаени логички состојби: 1 или 0, излезот од колото *Y* може да се најде и во т.н. *трета состојба*, или *состојба на висока импен*данса која вообичаено се означува со *HiZ, Hi-Z* или само со *Z.* Кога колото се наоѓа во трета состојба, тогаш тоа не троши никаква струја ($I_Y \to 0$), што произлегува од фактот дека во оваа состојба излезот *Y* се прекинува и се однесува како отпорник со бесконечно голема вредност ($Z_Y \to \infty, R_Y \to \infty$).

Нормалната работа на колото се овозможува преку логичкото ниво на новододадениот контролен влез означен со E (анг. Enable). Ако на влезот за контрола E се донесе 1 (E=1), тогаш колото функционира како обичен бафер, т.е. логичката состојба од влезот се пренесува на излезот (Y=A). Меѓутоа, ако на влезот E се донесе 0 (E=0), излезот на колото ќе оди во трета состојба (Y=HiZ), кога $R_Y \to \infty$ и $I_Y \to 0$ со што колото практично се исклучува. Ова значи дека активно ниво на контролниот сигнал E е 1, бидејќи само кога E=1, баферот функционира на вообичаен начин.

Однесувањето на три-статичкото баферско коло потсетува на славина за вода.

На сл. 2-25 а), б) се прикажани логичките симболи на три-статичкиот бафер и тоа според двата стандарди: ANSI и IEC/IEEE, додека таблицата на вистинитост таб. 2-21 дополнително го рефлектира неговиот начин на работа.

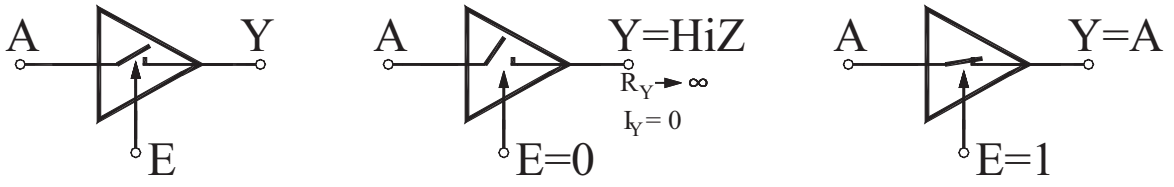
а) ANSI. б) IEC/IEEE.

E	A	Y
0	x	Hi-Z
1	0	0
	1	1

Сл. 2-25. Симболички ознаки Логичка равенка Таб. 2-21. Комбинациона таблица

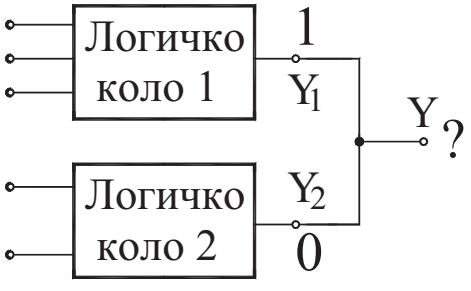
Баферско коло со три состојби (тристатички бафер) и контролен сигнал активен на 1

Заради појаснување на принципот на функционирање на колото со три состојби истото ќе го прикажеме и како надворешно контролиран механички прекинувач (сл. 2-26).

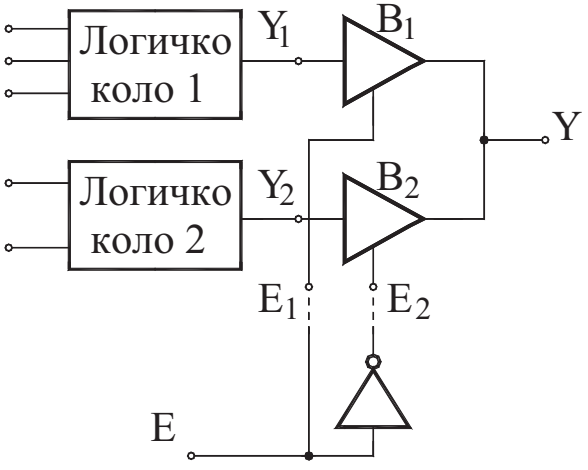


Сл. 2-26. Баферско коло како контролиран механички прекинувач во една насока

Тристатичкиот бафер се појави од чисто практични причини и тоа од потребата во една точка да се поврзат најмалку два излези од различни логички кола како што е прикажано на сл. 2-27. Едноставното поврзување на излезите од вратите може да прави проблем затоа што состојбата во заедничката точка не може да се контролира. Ова ќе се случи ако состојбите на излезите од колата меѓусебно се разликуваат. Во овој случај логичката состојба во спојната точка не може да се дефинира што резултира со појава на конфликт (колизија).



Сл. 2-27. Директно поврзување на излези од логички кола и појава на конфликт



Сл. 2-28. Поврзување на излези од логички кола преку бафери со три состојби

Конфликтот може да се избегне токму со воведување на бафери со три состојби според сликата сл. 2-28. Од неа се гледа дека заради присуството на баферите B_1 и B_2 излезите на логичките кола по потреба можат и да се откачат од спојот. Имено, логичката состојба во заедничката точка на спојување Y ќе зависи само од баферот што пропушта, т.е. што е активен, ако при тоа баферот на второто коло се држи пасивен и неговиот излез се форсира да оди во трета состојба со што ќе се откачи од заедничката точка. Ова е овозможено на тој начин што на контролниот влез од баферот на колото што треба да ја одреди логичката состојба во заедничката точка се носи 1, а на влезот за контрола кај баферот на пасивното коло се доведува 0, како на пр. E_1=1, а E_2=0. Сега е јасно дека логичката состојба во заедничката точка ќе биде дефинирана само од логичкото ниво на излезот од првото коло, т.е. ќе важи $Y=Y_1$. Функционалната таблица 2-22 дополнително го објаснува презентираниот начин на работа.

E1	E2	Y
0	0	Hi-Z
0	1	Y_2
1	0	Y_1
1	1	?

Таб. 2-22. Управување со два контролни влеза

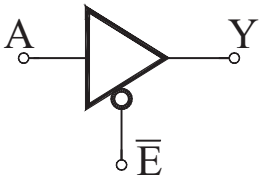
E	E1	E2	Y
0	1	0	Y_1
1	0	1	Y_2

Таб. 2-23. Управување со еден контролен влез и инвертор

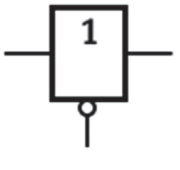
За да не дојде до грешка, двата влеза за контрола можат да се поврзат во единствен контролен влез E, при што контролниот влез од едниот бафер ќе се спои директно на овој заеднички влез, а контролниот влез на другиот бафер ќе оди преку инвертор како што е прикажано на сл. 2-27 со испрекинати линии. Во овој случај ќе важи комбинационата таблица 2-23, која покажува дека кога E=0 тогаш $Y=Y_1$, но ако E=1 тогаш $Y=Y_2$.

Поврзувањето во заедничка точка може да се изведе и со излези од повеќе логички кола, при што треба да се внимава на управувањето со контролните влезови. Имено, секогаш треба да е активно само едно од логичките кола, чии излези се поврзани, додека сите останати кола треба да се пасивни.

Во праксата често се сретнуваат и кола со три состојби кај кои активното ниво на контролниот сигнал е ниско. Во овој случај на симболот на колото му се додава кругче на неговиот контролен влез, според сликите сл. 2-29 а) и б). Ваквото три-статичко коло ќе биде активно ако E=0, додека тоа ќе стане пасивно и на излезот ќе се јави трета состојба ако E=1, соодветно на прикажаната таблица на вистинитост таб. 2-24.



а) ANSI.



б) IEC/IEEE.

$$Y=\begin{cases} A & \text{за } E=0 \\ \text{Hi-Z} & \text{за } E=1 \end{cases}$$

Логичка равенка

E	A	Y
0	0	0
	1	1
1	x	Hi-Z

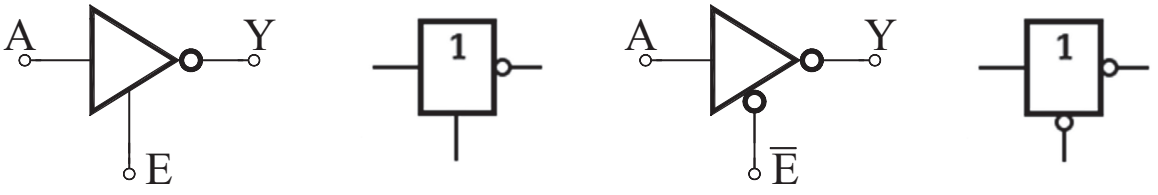
Таб. 2-24. Комбинациона таблица

Сл. 2-29. Симболички ознаки

Баферско коло со три состојби (тристатички бафер) и контролен влез активен на 0

Постојат и реализации на три-статички бафери со контролен влез за **оневозможување** D (од ang. disable) на работата. Во овој случај со доведување на активно ниво на контролниот влез D излезот оди во трета состојба, а ако на D се донесе пасивно логичко ниво, излезот нормално функционира и на него се пренесува состојбата од влезот.

Друг тип на бафери што исто така многу често се сретнуваат се т.н. бафер-инвертори кои во принцип функционираат на ист начин како и баферските кола што претходно ги наведовме, само што бафер-инверторите дополнително вршат инвертирање на влезниот сигнал. Кај нив, кога на контролниот влез се донесе соодветен активен сигнал, на излезот се јавува комплементарна вредност на влезната променлива ($Y = \overline{A}$), заради што на логичкиот симбол на колото на излезот му се додава кругчето според сл. 2-30 а), б), в), г). Спротивно, ако контролниот влез е пасивен, баферот оди во трета состојба.



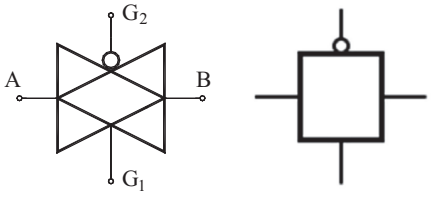
а) ANSI б) IEC/IEEE в) ANSI г) IEC/IEEE

Контролен влез активен на 1 Контролен влез активен на 0

Сл. 2-30. Логички симболи на бафер-инвертори со три состојби

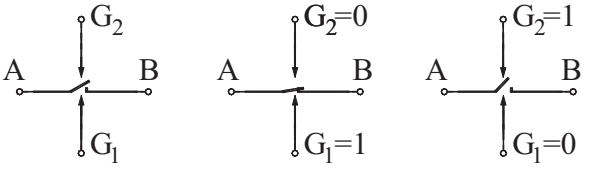
2.7.2.3. БИЛАТЕРАЛНА (ТРАНСМИСИОНА) ПОРТА

Од посебен интерес во практиката е и колото чиј логички симбол е прикажан на сл. 2-31. На сл. 2-31 а) е даден симболот според ANSI-стандардот, додека на сл. 2-31 б) е претставен симболот според IEC/IEEE стандардот. Станува збор за коло кое или ја овозможува или ја прекинува врската помеѓу влезот и излезот во двете насоки, т.е. билатерално, заради што ова коло може да се прикаже и како надворешно контролиран механички прекинувач според сл. 2-32.



а) ANSI б) IEC/IEEE

Сл. 2-31. Симболички ознаки на билатерален прекинувач



а) Симбол б) Вклучен в) Исклучен

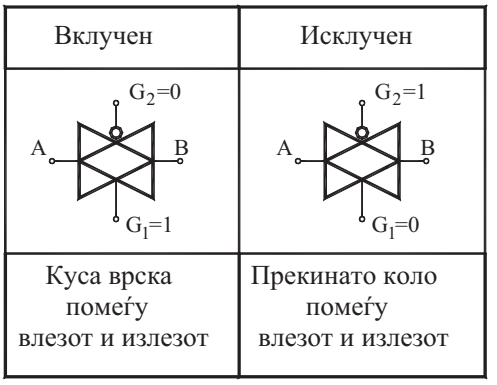
Сл. 2-32. Трансмисиона порта како надворешно контролиран механички прекинувач

Однесувањето на колото се управува преку логичките состојби на контролните влезови. Имено, трансмисионата порта се вклучува кога на контролниот влез G_1 се доведе 1, додека на вториот контролен влез G_2 *се носи* 0 со што помеѓу влезот и излезот се воспоставува куса врска и меѓу нив е овозможен пренос во двете насоки: од A кон B, т.е. од B кон A. Обратно, ако на управувачкиот влез G_1 *се донеси* 0 и едновремено на вториот контролен влез G_2 *се приклучи* 1, трансмисионата врата се исклучува бидејќи комуникацијата помеѓу влезот/излезот A не може да се оствари со влезот/излезот B заради што излезот е во прекин (откачен, исклучен) од влезот и се наоѓа во состојба на бесконечно голема (висока) отпорност, т.е. во трета состојба. Принципот на работа на билатералната порта е презентиран со таб. 2-25 и сл. 2-33.

Благодарејќи на начинот на работа, ова коло има голема примена во различни области на електрониката и затоа има уште неколку имиња. Тоа се нарекува и двонасочна порта, билатерален прекинувач, аналоген прекинувач или временски селектор.

G1	G2	Комуникација Влез – Излез
1	0	Куса врска ($R_{A-B} \rightarrow 0$)
0	1	Прекин (Hi-Z) ($R_{A-B} \rightarrow \infty$)

Таб. 2-25. Функционална таблица на трансмисионата порта



Сл. 2-33. Опис на работата на трансмисионата порта

Контролата на работата на билатералниот прекинувач може да се изведе и само преку единствена контролна влезна линија. Во овој случај логичкото ниво на едниот контролен влез треба да биде во директна форма, додека на другиот во комплементарен облик преку инвертор. Поврзувањето е во принцип исто со управувањето на тристатичките бафери прикажано на сл. 2-28, каде контролните влезови E1 и E2 беа поврзани во единствена контролна линија E.

2.7.3. АНАЛИЗА НА ПРЕКИНУВАЧКИ МРЕЖИ

Анализирањето на прекинувачката мрежа се однесува на функционален план, затоа што конструкцијата на мрежата е позната преку нејзината логичка шема. *Задачата на анализата е да ги објасни поодделните логички функции на зададената комбинациона мрежа, со крајна цел - одредување на логичките состојби во поедините точки на мрежата, и тоа за секоја комбинација на вредности на влезните променливи*. Според она што го забележавме, произлегува дека проблемот на анализа ќе биде посебно важен при експлоатацијата и одржувањето на дигиталните уреди.

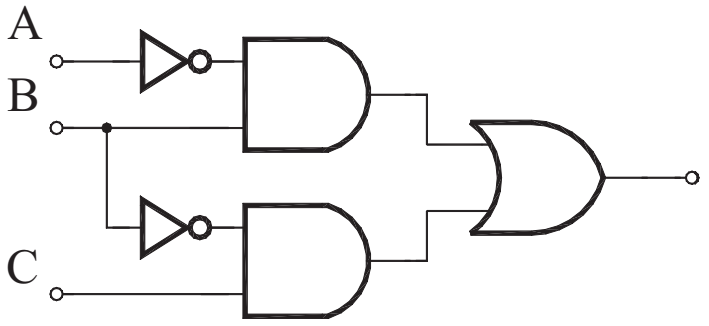
Логичката шема на мрежата треба да биде нацртана со примена на соодветните логички симболи на употребените логички кола, како и со означување на врските кои постојат помеѓу влезовите и излезите од секое логичко коло. За секоја прекинувачка мрежа познати се имињата на сите влезни променливи, т.е. независно променливите кои можат да се појават во директен (вистински, номинален) или комплементаран облик, како и сите имиња на излезните променливи, т.е. функциите.

По извршената анализа на зададената мрежа треба да се добие аналитичкиот облик на функцијата, односно функциите што таа мрежа ги реализира. Ако е возможно, тој облик треба што е можно повеќе да се упрости, или поточно минимизира. Конечно, ако треба, од добиената равенка може да се конструира и таблицата на вистинитост на прекинувачката мрежа.

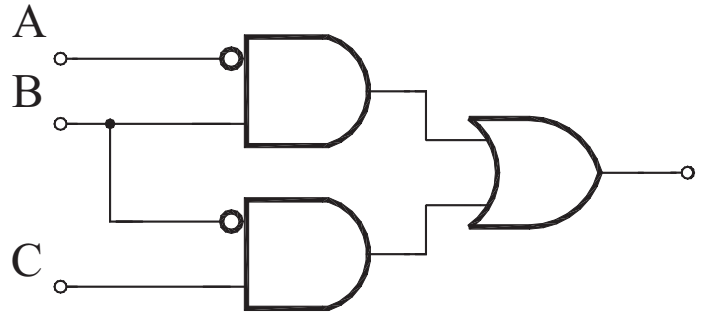
Анализата започнува од влезот кон излезот на мрежата и тоа така што се означуваат сите излези од логичките кола, и за секој излез се пишува соодветна равенка која ќе зависи од тоа за какво логичко коло станува збор. Значи, постапно се одредуваат аналитичките облици на прекинувачките функции на излезите од сите логички кола, и тоа движејќи се од влезот кон излезот од мрежата. Оваа постапка се извршува сè додека не се добие израз за секој излез од било кое логичко коло во мрежата (секоја функција).

Дури потоа се продолжува со упростувањето на секоја функција. Од последнава равенка, т.е. од равенката која повеќе не може да се поедноставува, ако се бара, се формира комбинационата таблица на мрежата.

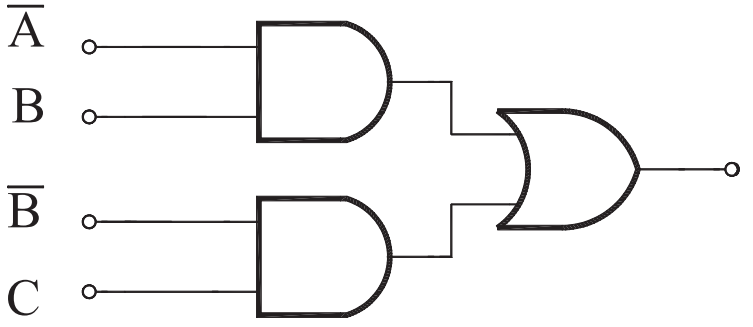
На сл. 2-34 а), б) и в) се прикажани три логички шеми кои на прв поглед меѓусебно се разликуваат. Сепак, ако истите ги анализираме, ќе заклучиме дека секоја од нив ја опишува прекинувачката мрежа со која се реализира функцијата $Y(A,B,C)=\overline{A}B+\overline{B}\overline{C}$. Разликата се јавува само во однос на прикажувањето на комплементирањето, така што секоја шема е валидна и во однос на реализацијата на добиената логичка функција, може подеднакво да се употребува како и другите две. Во врска со ова, на сл. 2-34 а) комплементирањето е означено со инвертор. Сл. 2-34 б) е малку поедноставна затоа што се користат кола со комплементарни влезови кои се означени со мал круг „○”, додека на сл. 2-34 в) означувањето е наједноставно бидејќи се претпоставува дека комплементите на променливите веќе се претходно добиени, што се индицира со цртички над нив „ ¯ ”.



а) логичка негација (комплементирање) со инвертори



б) логичка негација (комплементирање) со влезовите на логичките кола

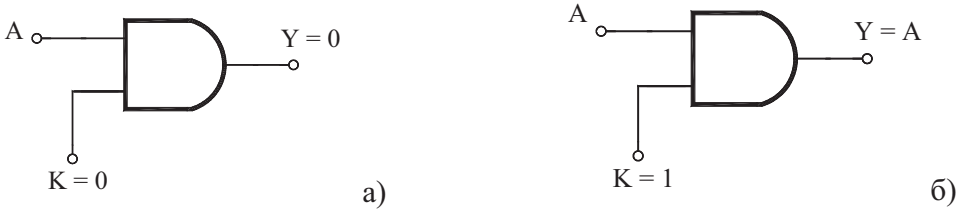


в) логичка негација (комплементирање) со комплементарни влезни променливи

Сл. 2-34. Логички шеми на функцијата $Y(A,B,C)=\overline{A}B+\overline{B}\overline{C}$

Овде сепак ќе истакнеме дека од гледна точка на практична реализација секој логички дијаграм ќе биде претставен со различно решение. Имено, во шемата од сл. 2-34 а) се применуваат два инвертори, две И кола и едно ИЛИ коло. Шемата од сл. 2-34 б) користи само три логички кола од кои две НИ кола кои можат да инвертираат по еден влез и едно ИЛИ коло. И шемата од сл. 2-34 в) користи три кола бидејќи комплементирањето на променливите е претходно веќе остварено со некоја друга логичка структура.

На следните две слики сл. 2-35 и сл. 2-36 последователно се претставени по два наједноставни примери за примена на И и ИЛИ коло кои често ќе ги сретнуваме понатаму.



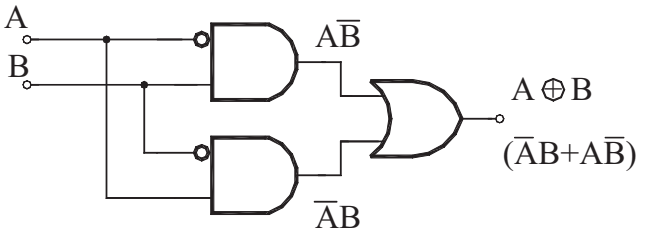
Сл. 2-35. Анализа на логичко И коло

Сл. 2-36. Анализа на логичко ИЛИ коло

Со анализа на логичките дијаграми презентирани на сл. 2-37 и сл. 2-38 се добиваат следниве две логички равенки:

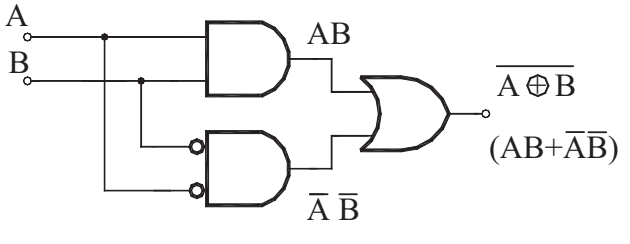
$$Y = A\overline{B} + \overline{A}B \qquad (2\text{-}28)$$

$$Y = AB + \overline{A}\,\overline{B} \qquad (2\text{-}29)$$



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

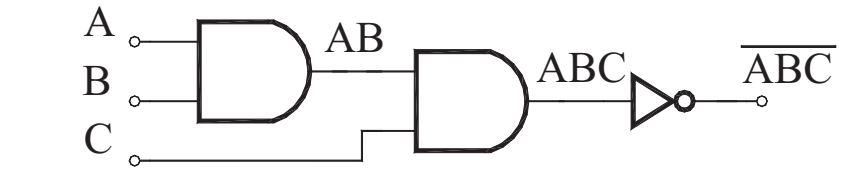
Сл. 2-37. Реализација на ЕКСИЛИ логичко коло



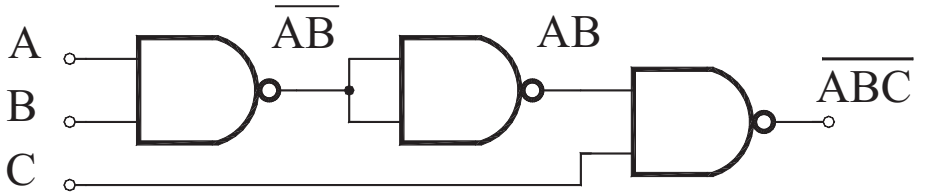
A	B	$AB + \overline{A}\,\overline{B}$
0	0	1
0	1	0
1	0	0
1	1	0

Сл. 2-38. Реализација на ЕКСНИЛИ логичко коло

Ако за секоја од нив се пополнат соодветни комбинациони таблици и ако тие се споредат со таблиците на вистинитост на логичките функции ЕКСИЛИ и ЕКСНИЛИ се докажува дека логичките шеми дадени на сл. 2-37 и сл. 2-38 практично ги реализираат функциите ЕКСИЛИ и ЕКСНИЛИ. На сл. 2-39 а), б) и сл.2-40 а) б) е прикажана анализа на уште четири едноставни, но карактеристични примери на логички дијаграми со кои се реализираат НИ и НИЛИ логичките функции зависни од три влезни променливи. Сликата сл. 2-41 претставува еден пример на анализа на логичка шема со бафер со три состојби.



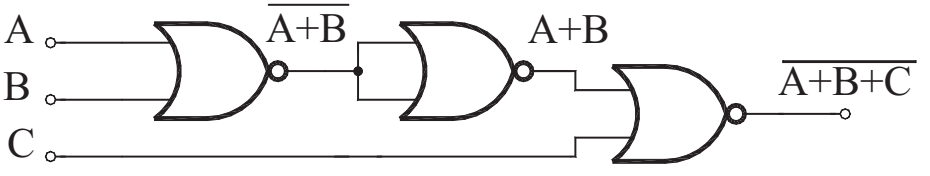
а)



б)

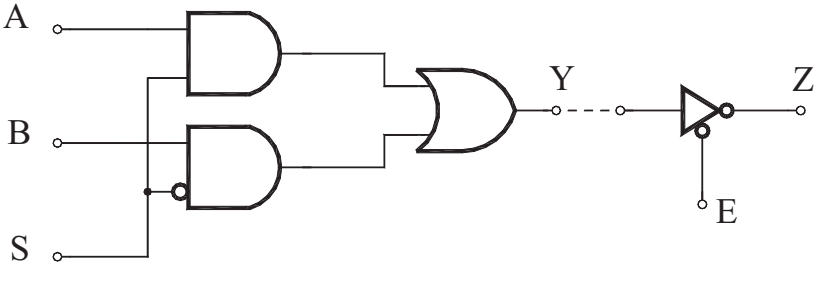
Сл. 2-39. Реализација на НИ логичко коло со три влеза.

а)



б)

Сл. 2-40. Реализација на НИЛИ логичко коло со три влеза



E	S	Y	Z
0	0	B	$\overline{B}$
	1	A	$\overline{A}$
1	0	B	HiZ
	1	A	HiZ

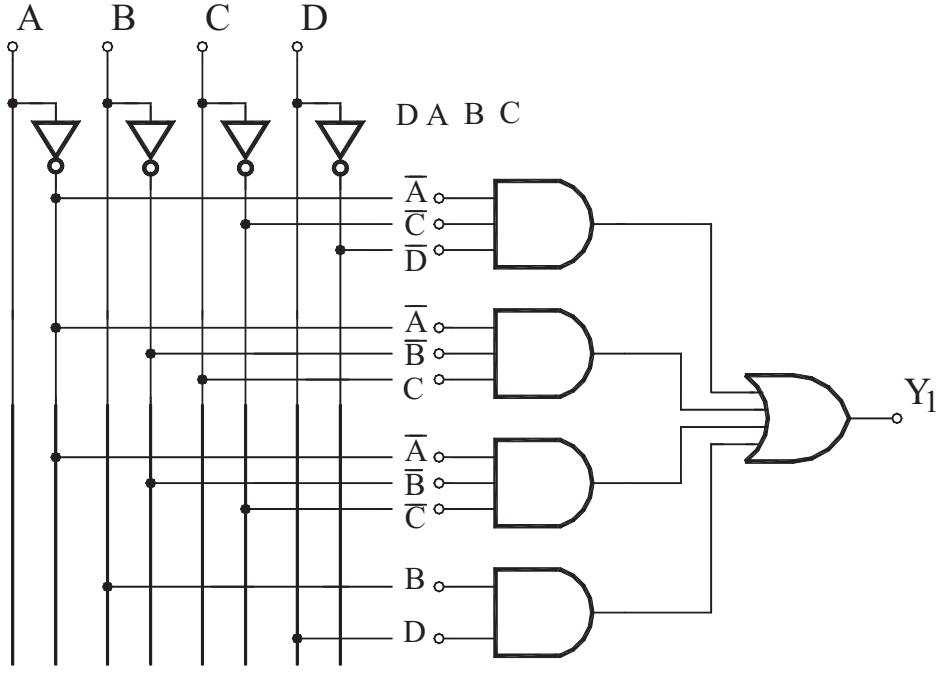
Сл. 2-41. Анализа на логички дијаграм со излезен тристатички бафер-инвертор

2.7.4. СИНТЕЗА НА ПРЕКИНУВАЧКИ МРЕЖИ

За да се изврши синтеза на некоја прекинувачка мрежа треба да е позната прекинувачката функција со која таа се опишува. Ова значи дека од некоја зададена нормална форма или таблица на вистинитост на логичката функција треба да се добие логичката шема на мрежата што таа функција физички ќе ја реализира. Многу важен критериум при создавањето на мрежата е секако вкупниот број на употребени логички кола. Јасно е дека ќе се стремиме кон тоа прекинувачката мрежа да ја реализираме со што е можно помал број логички кола и со што е можно помал број влезови по логичко коло, така што *прва работа што треба да се направи во процесот на синтезата е да се изврши минимизација на зададената прекинувачка функција* и таа да се сведе во облик на МДНФ или МКНФ.

Ако некоја логичка функција е претставена во МДНФ облик (минимална сума од производи), тогаш прекинувачката мрежа што неа ја реализира ќе се состои од одреден број на И кола, чии излези ќе се приклучат како влезови во едно ИЛИ коло, од чиј излез ќе се добие бараната функција. Секоја влезна променлива прво се проследува преку И коло, кое претставува прво ниво, а потоа се пренесува преку ИЛИ коло кое претставува второ ниво. Ваквата структура се нарекува *И-ИЛИ логички систем во две нивоа*.

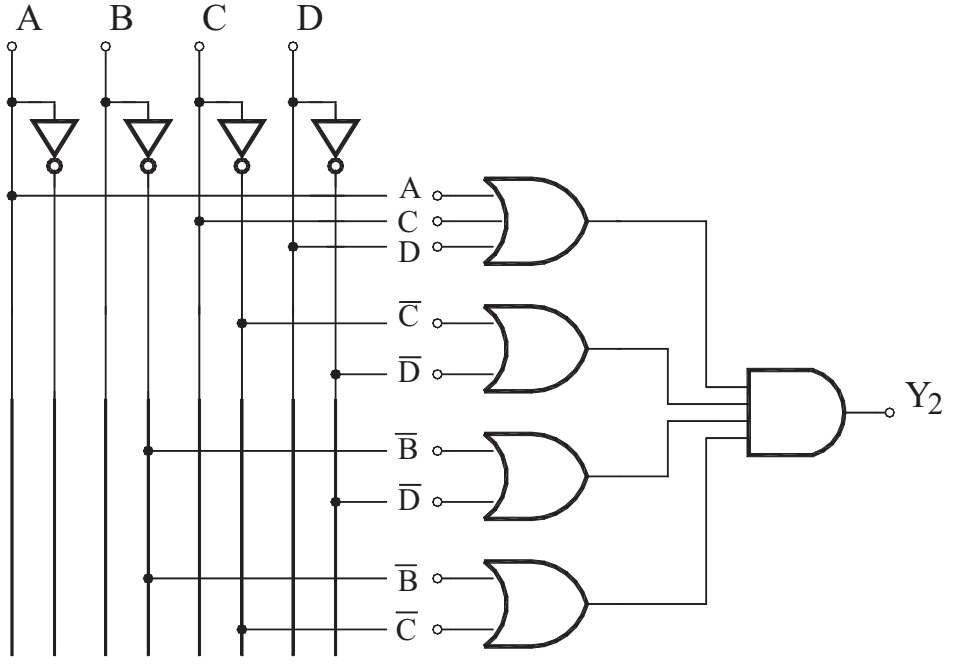
На сл. 2-42 е прикажан еден пример на ваква И-ИЛИ мрежа во две нивоа која ја реализира функцијата $Y_1 = \overline{A}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,C + A\overline{B}\,\overline{C} + BD$.



Сл. 2-42. Синтеза на функција од четири променливи со И-ИЛИ логичка структура во две нивоа

Ако некоја логичка функција е прикажена во МКНФ облик (минимален производ од суми), тогаш повторно ќе се добие логичка структура во две нивоа, но сега таа ќе биде од ИЛИ-И тип, што значи дека влезни кола ќе бидат ИЛИ колата и тие ќе го претставуваат првото ниво, додека второто ниво ќе биде излезното И коло од кое се добива функцијата.

Еден пример на ИЛИ-И мрежа во две нивоа со која се реализира функцијата $Y_2 = (A + C + D)(\overline{C} + \overline{D})(\overline{B} + \overline{D})(\overline{B} + \overline{C})$ е прикажан на сл. 2-43.



Сл. 2-43. Синтеза на функција од четири променливи со ИЛИ-И логичка структура во две нивоа

Во претходното излагање веќе кажавме дека секоја прекинувачка функција може да се претстави само со примена на НИ кола, или само со примена на НИЛИ кола. Покрај ова овие две функции се полесни за техничка реализација, така што и од практични причини подобро е да се користат НИ и НИЛИ логички кола. Со примерите што следат ќе објасниме како може да се добие физичка реализација на логичките функции само со примена на НИ кола, или само со примена на НИЛИ кола.

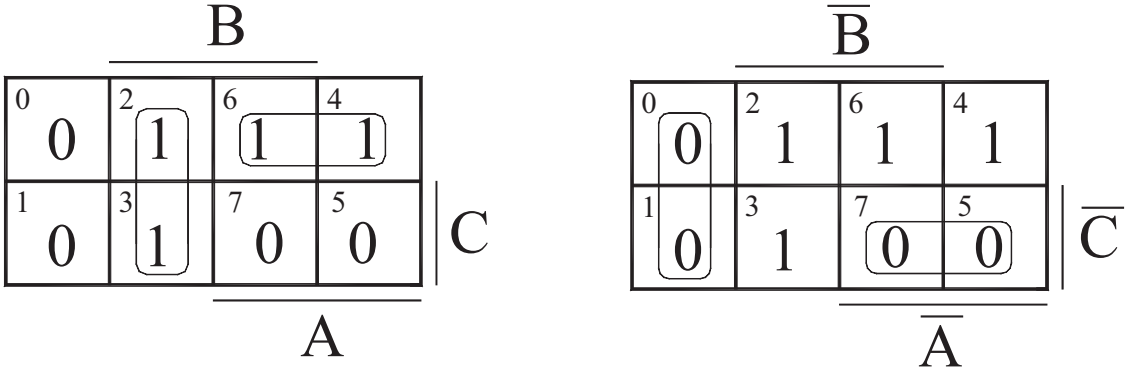
Да ја разгледаме прекинувачката функција Z од три променливи Y=*Y(A,B,C)*= $= \sum m(2,3,4,6) = \prod M(0,1,5,7)$ чија комбинациона таблица е претставена како таб. 2-26, додека нејзините КК во СДНФ и СКНФ облик се прикажани на сл. 2-44 а) и б). По извршената минимизација на функцијата, таа може да се запиши во МДНФ облик како $Y = \overline{A}B + A\overline{C}$, или во МКНФ облик како $Y = (A + B)(\overline{A} + \overline{C})$. Двонивовската И-ИЛИ структура која е прикажана на сл. 2-45 а) ја реализира функцијата во МДНФ облик, додека според МКНФ обликот се добива двонивовската ИЛИ-И мрежа прикажана на сл. 2-45 б).

i	*ABC*	*Y*
0	000	0
1	001	0
2	010	1
3	011	1
4	100	1
5	101	0
6	110	1
7	111	0

За да покажеме како се добива реализација само со НИ, и само со НИЛИ порти ќе извршиме двојно комплементирање посебно на МДНФ обликот на функцијата и посебно на нејзиниот МКНФ облик. Со ова практично ништо не сме промениле, но затоа да видиме што ќе се случи ако потоа ја примениме Де Морганова теорема.

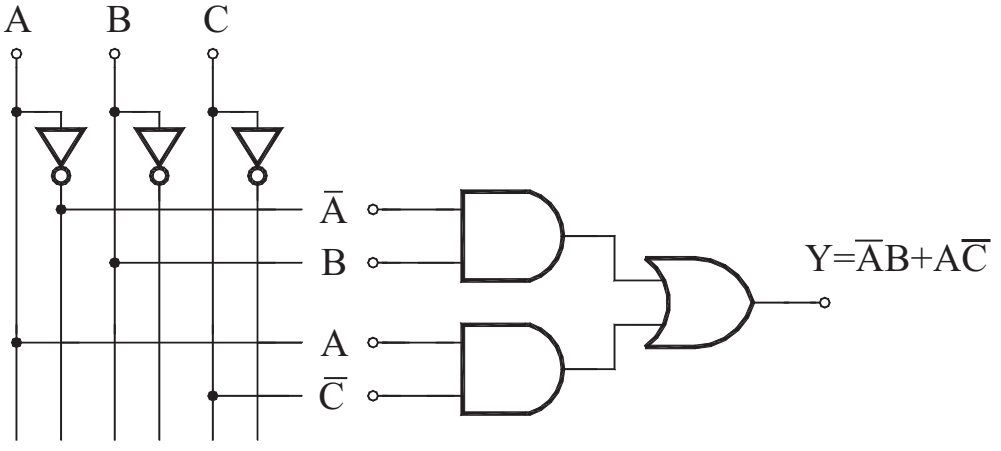
Таб. 2-26. Таблица на вистинитост на функцијата

$$Y(A,B,C) = \sum m(2,3,4,6) = \prod M(0,1,5,7)$$

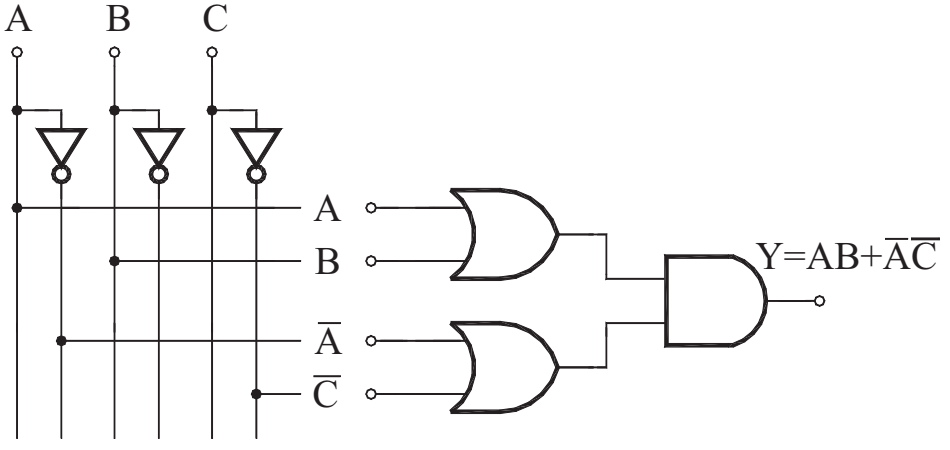


а) минимизација на СДНФ облик б) минимизација на СКНФ облик

Сл. 2-44. Карноова карта на функцијата $Y(A,B,C)=\sum m(2,3,4,6)=\prod M(0,1,5,7)$



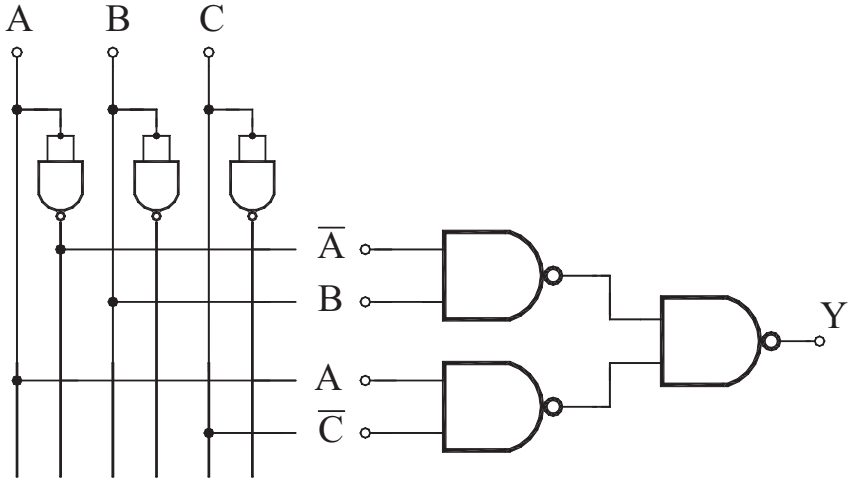
а) реализација со двонивовска И-ИЛИ комбинациона мрежа



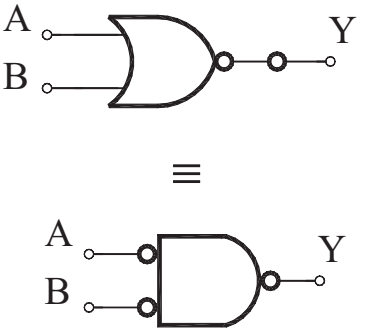
б) реализација со двонивовска ИЛИ-И комбинациона мрежа

Сл. 2-45. Синтеза на функцијата $Z(A,B,C)=\sum m(2,3,4,6)=\prod M(0,1,5,7)$ две нивоа

Поаѓајќи од МДНФ се добива $Y = \overline{\overline{Y}} = \overline{\overline{\overline{A}B + A\overline{C}}} = \overline{\overline{(\overline{A}B)} \cdot \overline{(A\overline{C})}}$. Применувајќи ја трансформацијата од сл. 2-46 добиениот МДНФ облик на функцијата може да се реализира само со користење на НИ кола како што е прикажано на сл. 2-47. Лесно се забележува дека конфигурацијата на оваа мрежа е иста со таа што е прикажана на сл. 2-45 а), само што секое логичко коло се заменува со НИ коло.

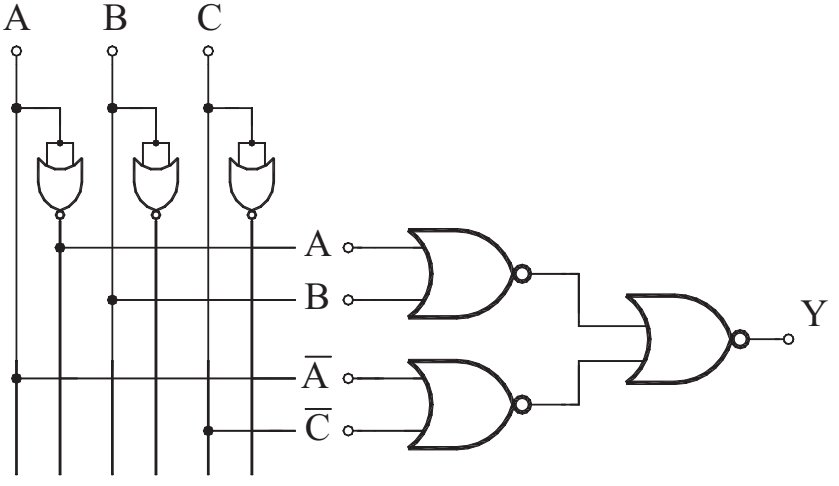


Сл. 2-47. Синтеза на функцијата Y = ∑ m(2,3,4,6) (=∏ M(0,1,5,7))

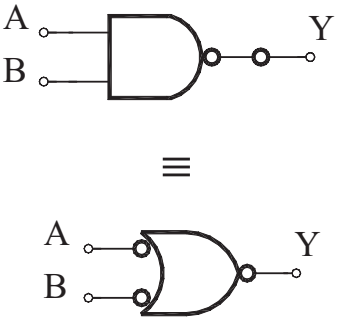


Сл. 2-46.

Слично, ако се тргне од МКНФ обликот на функцијата се добива: $Y = \overline{\overline{Y}} = \overline{\overline{(A+B)} \cdot \overline{(\overline{A}+\overline{C})}} = \overline{\overline{(A+B)} + \overline{(\overline{A}+\overline{C})}}$. Оваа форма на функцијата е реализирана само со примена на НИЛИ врати како што може да се види од сл. 2-49, при што е употребена трансформацијата од сл. 2-48. И во овој случај е очигледно дека последната добиена конфигурација и логичката шема од сл. 2-45 б) се разликуваат само по тоа што сите логички кола од првата шема се заменети со НИЛИ порти во втората.



Сл. 2-49. Синтеза на функцијата Z = ∏ M(0,1,5,7)(= $\sum m(2,3,4,6)$)



Сл. 2-48

Од сликите може да се забележи дека согласно равенката (2-22) $\overline{A} = \overline{A \cdot A}$ инверторите на сл. 2-47 се реализирани како двовлезни НИ кола чии влезови се поврзани во еден. Слично, имајќи ја во вид равенката (2-25) $\overline{A} = \overline{A + A}$ на сл. 2-49 инверторите се заменети со НИЛИ кола со два влеза, чии влезови се исто така поврзани во еден.

Од изложеното може да се изведе генерален заклучок дека *за да се добие мрежа, составена само од НИ кола,* треба да се почне така што дадената функција треба да се изрази во МДНФ. Потоа за оваа форма на функцијата треба да се нацрта соодветна И-ИЛИ конфигурација на логичка мрежа во две нивоа, и на крај сите логички кола во неа да се заменат со НИ кола.

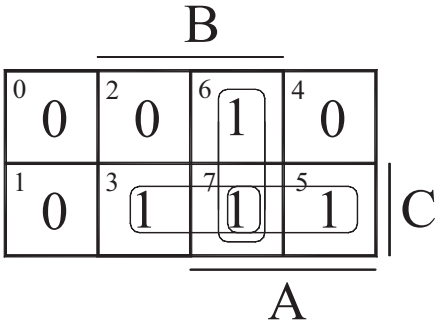
Слично на претходно наведеното, *за да се добие НИЛИ конфигурација на логичка мрежа* потребно е на почетокот функцијата да се претстави во облик на МКНФ. Понатаму се црта соодветната дво-нивовска ИЛИ-И мрежа, и на крај сите логички кола во неа се заменуваат со НИЛИ кола.

Проектирање на прекинувачки мрежи: На крај ќе презентираме постапка со која се проектира прекинувачка мрежа што го решава следниот едноставен проблем. Тројца членови на жири комисија гласаат со притискање на тастер пред секој од нив за тоа дали кандидатот кој испеал може да продолжи и во следната ТВ емисија за избор на најдобар талент или ќе мора да отпадне. Ако кандидатот добие најмалку два гласа од комисијата треба да светни зелена светилка што ќе покаже дека тој оди во следниот круг, додека ако се запали црвена светилка, кандидатот не покажал задоволителен квалитет и тој отпаѓа.

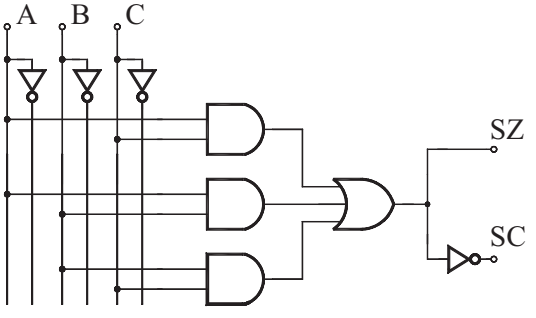
Најнапред ја формираме таблицата на вистинитост таб. 2-27 така што трите тастери ќе ги претставиме како независни (влезни) бинарни променливи A, B, C за кои вредноста 0 ќе значи дека тастерот не е притиснат, додека 1 дека е притиснат. Како функции, т.е. зависни (излезни) променливи, ќе ги земеме двете светла: зеленото SZ и црвеното SC. За нив вредноста 1 ќе индицира дека светлото свети, додека 0 ќе значи дека не свети. Од добиената таблица се заклучува дека светлата се меѓусебно комплементарни ($SC = \overline{SZ}$), што ни укажува дека е сосема доволно да ја решиме едната функција бидејќи другата ќе ја добиеме со комплементирање на првата. За остварување на оваа цел продолжуваме со минимизација на една од функциите, на пр. онаа за активирање на зеленото светло SZ применувајќи ја КК од сл. 2-50. Бидејќи нејзиното минимално решение го има следниот МДНФ облик SZ=AC+AB+BC, конечно можеме да ја нацртаме и комбинационата мрежа (сл. 2-51) со која го решаваме дадениот проблем.

i	*ABC*	*SZ*	*SC*
0	000	0	1
1	001	0	1
2	010	0	1
3	011	1	0
4	100	0	1
5	101	1	0
6	110	1	0
7	111	1	0

Таб. 2-27. Таблица на вистинитост



Сл. 2-50. Минимизација



Сл. 2-51. Логички шема

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

2-1. Што претставува Буловата алгебра?

2-2. Наведи ги аксиомите на Хантингтон?

2-3. Како се спроведува принципот на дуалност?

2-4. Набројги основните логички операции.

2-5. Според кој приоритет се извршуваат логичките операции при решавањето на логичките изрази?

2-6. Дефинирај ги наведените логички функции со нивните таблици на вистинитост и во аналитички облик со логички равенки: И, ИЛИ, НЕ (комплементирање), НИ, НИЛИ, ЕКСИЛИ и ЕКСНИЛИ.

2-7. Докажи ги теоремите (т. 2-12), (т. 2-14) и (т. 2-15) (а) по аналитички пат; (б) со методот на совршена индукција; (в) со примена на теоремата за експанзија.

2-8. Дадениве логички изрази упрости ги по аналитички пат: (а) $1+\overline{A}B+\overline{A}BC+B\overline{C}$; (б) $0+A\overline{B}D+\overline{B}D+C$; (в) $1(BC+\overline{B}C)$; (г) $0(\overline{A}BC+\overline{A}B+BC+AB\overline{C}+\overline{A}C)$.

2-9. Следниве логички изрази упрости ги по аналитички пат: (а) $\overline{(\overline{A}+B+\overline{C})(A+\overline{B})C}$; (б) $\overline{(\overline{A}+B+\overline{C})(\overline{A+\overline{B}})}$; (в) $\overline{(\overline{A\overline{B}})+(AB\overline{C})}$; (г) $\overline{\overline{(\overline{A}\overline{B})}+(AB\overline{C})}+C$.

2-10. Набарој ги облиците во кои може да се зададе било која прекинувачка функција.

2-11. Детално опиши го изгледот на комбинационата таблица за секоја логичка функција Y од *n* променливи. Колкав е бројот на редици и колони? Што се внесува во нив? Што е индекс и во кој опсег се движи? Според кој принцип редиците од табелата се нумерираат со индекси?

2-12. Наведи ги нормираните форми со кои може да се опише секоја логичка функција во аналитички облик.

2-13. Каков е ДНФ обликот на функцијата? Што претставува?

2-14. Што е импликанта? Што е минтерм? Што претставува СДНФ?

2-15. Каков е КНФ обликот на функцијата? Што претставува?

2-16. Што е имплицента? Што е макстерм? Што претставува СКНФ?

i	*A B C*	F_1	F_2	F_3
0	0 0 0	0	1	0
1	0 0 1	1	0	1
2	0 1 0	0	1	0
3	0 1 1	1	x	1
4	1 0 0	0	0	x
5	1 0 1	1	x	x
6	1 1 0	1	0	1
7	1 1 1	0	0	x

2-17. Со таблицата на вистинитост таб. 2-28 се претставени три функции $F_1(A,B,C)$, $F_2(A,B,C)$ и $F_3(A,B,C)$ од по три променливи. За секоја од нив напиши ги нивните СДНФ и СКНФ облици преку множествата на индекси.

Таб. 2-28. Комбинациони таблици на логичките функциии F_1, F_2, F_3 од задача 2-17

2-18. За секоја од дадените функции потцртај ги сите минтерми, односно макстерми, а потоа одговори кои функции се наведени во облик на ДНФ, СДНФ, КНФ односно СКНФ: (а) $F_1(A,B,C) = ABC + \overline{A}\,\overline{B}\,\overline{C}$; (б) $F_2(A,B,C) = A\overline{B}C + \overline{A}B\overline{C} + \overline{C}B$; (в) $Y(A,B,C) = (A+B+\overline{C})(\overline{A}+\overline{B}+C)(\overline{A}+\overline{B}+\overline{C})$;(г) $Z(A,B,C) = (A+C)(\overline{A}+\overline{B})(B+\overline{C})$.

2-19. За следниве функции зададени во аналитички ДНФ и КНФ облик (а) $F_1(X_1,X_2,X_3) = X_1X_2\overline{X}_3 + \overline{X}_1\overline{X}_2$, (б) $F_2(X_1,X_2,X_3) = (X_1+\overline{X}_2+X_3)(\overline{X}_2+\overline{X}_3)$, (в) $Y(A,B,C) = AB + \overline{A}C$, (г) $Z(A,B,C) = (A+C)(\overline{A}+B)$ (1) состави им ги нивните таблици на вистинитост; (2) прикажи ги преку множествата на индекси; (3) прикажи ги во СДНФ и СКНФ облици.

2-20. Зададените функции во СДНФ и СКНФ облик упрости ги по аналитички пат: (а) $Y(A,B,C) = A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C}$; (б) $Z(A,B,C) = (A+B+C)(A+\overline{B}+C)(\overline{A}+\overline{B}+C)$.

2-21. Изврши премин по аналитички пат од ДНФ во СКНФ облик на функциите:

(а) $Y(A,B,C,D) = B\overline{C} + \overline{A}\,\overline{B}D$; (б) $Y(A,B,C,D) = \overline{A}BD + CD$.

2-22. Изврши премин по аналитички пат од КНФ во СДНФ облик на функциите:

(а) $Z(A,B,C,D) = (\overline{B}+D)(A+B+\overline{D})$; (б) $Z(A,B,C,D) = (A+B+D)(\overline{C}+\overline{D})$.

2-23. Кои логички функции влегуваат во состав на множеството на функционално потполн систем на логички функции?

2-24. Набројги универзалните функции.

2-25. Изрази ги основните функции НЕ, И и ИЛИ со функцијата (а) НИ (б) НИЛИ.

2-26. Што е карактеристично за минималните форми МДНФ и МКНФ на функциите?

2-27. Според кои методи може да се врши минимизирање на логичките фунции?

2-28. Да се минимизираат по аналитички (алгебарски) пат функциите: (а) $Y(A,B,C) = (A+B)(B+C)(\overline{B}+C)(\overline{B}+\overline{C})$; (б) $Z(A,B,C) = AB + \overline{A}C + BC + \overline{B}\,\overline{C}$.

2-29. Да се минимизираат по (а) аналитички пат; (б) со примена на Карноовиот метод следниве функции: (а) $F_1(A,B,C,D) = \overline{A}BCD + A\overline{B}\,\overline{C}D + \overline{A}B\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$; (б) $F_2(A,B,C,D) = (A+B+C+D)(\overline{A}+\overline{B}+C+D)(A+B+\overline{C}+\overline{D})(\overline{A}+\overline{B}+\overline{C}+D)(A+B+C+\overline{D})$

2-30. Со примена на методот на Карноови карти да се минимизираат следниве функции зададени преку множествата на индекси:

(а) $Y(A,B,C) = \prod M(0,1,2,4,5)$

(б) $Y(A,B,C) = \prod M(0,1,4,7)$

(в) $Y(A,B,C,D) = \prod M(0,1,2,4,5,7,8,12,13,14,15)$

(г) $Y(A,B,C,D) = \prod M(0,2,4,5,6,8,10,11,14,15)$

(д) $Y(A,B,C,D) = \prod M(0,1,2,3,7,8,10,11,12,14)$

(ѓ) $Y(A,B,C) = \sum m(0,2,6,7)$

(е) $Y(A,B,C)\sum m(0,1,3,5,6)$

(ж) $Y(A,B,C,D)=\sum m(0,3,4,6,7,8,11,12,13,15)$

(з) $Y(A,B,C,D)=\sum m(4,5,6,7,8,9,10,12,13,15\ )$

(s) $Y(A,B,C,D)=\sum m(0,2,4,5,6,7,9,11,13,15\ \ )$

(и) $Y(A,B,C,D)=\sum m(0,2,3,4,6,8,9,10,11,15\ )$

2-31. Со примена на методот на Карноови карти да се минимизираат следниве функции зададени во ДНФ и КНФ облик:

(а) $F(A,B,C)=A\overline{B}+\overline{A}\,\overline{C}$

(б) $F(A,B,C,D)=ABC\overline{D}+\overline{A}\,\overline{B}\,\overline{C}$

(в) $F(A,B,C,D)=\overline{A}BD+\overline{B}\,\overline{C}$

(г) $F(A,B,C,D)=BC\overline{D}+\overline{A}CD$

(д) $F(A,B,C,D)=\left(A+\overline{B}+C+\overline{D}\right)\left(\overline{A}+\overline{C}+\overline{D}\right)$

(ѓ) $F(A,B,C,D)=\left(B+C+D\right)\left(\overline{A}+D\right)$

(е) $F(A,B,C,D)=\left(\overline{B}+\overline{C}+\overline{D}\right)\left(\overline{A}+B+\overline{D}\right)$

2-32. Со примена на методот на Карноови карти да се минимизираат следниве некомплетно дефинирани функции зададени преку множествата на индекси:

(а) $Y(A,B,C)=\prod M(0,5,7)\prod_{xM} M(1,4,6)$

(б) $Y(A,B,C,D)=\prod M(1,3,4,5,6,8,12,14)\prod_{xM} M(7,10,15)$

(в) $Y(A,B,C,D)=\prod M(1,2,8,10,11)\prod_{xM} M(0,3,4,5,15)$

(г) $Y(A,B,C,D)=\prod M(0,2,4,5,6,11,15)\prod_{xM} M(8,10,14)$

(д) $Y(A,B,C)=\sum m(2,3,7)+\sum_{xm} m(5,6)$

(ѓ) $Y(A,B,C,D)=\sum m(0,4,5,6,8,12,14,15)+\sum_{xm} m(1,2,10)$

(е) $Y(A,B,C,D)=\sum m(1,2,3,4,5,9,11,12)+\sum_{xm} m(10,13,15)$

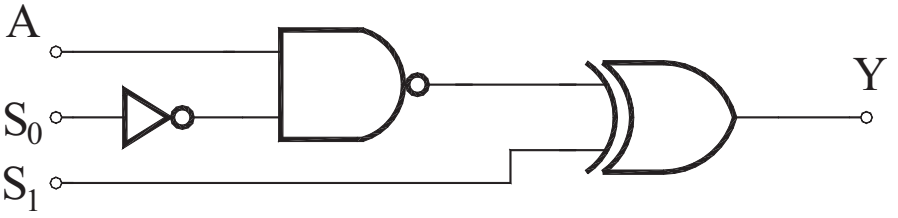
2-33. Нацртај ги логичките симболи на следниве логички кола: (а) И коло со три влеза; (б) Инвертор; (в) НИ коло со два влеза; (г) НИЛИ коло со три влеза; (д) ЕКСИЛИ коло со два влеза; (ѓ) ЕКСНИЛИ коло со два влеза; (е) Баферско коло; (з) Бафер со три состојби; (ж) Бафер-инвертор со три состојби; (з) билатерална (трансмисиона) порта.

2-34. Која е разликата помеѓу Бафер со три состојби и билатерална (трансмисиона) порта?

2-35. Како се формира прекинувачка мрежа?

2-36. Која е разликата помеѓу комбинациона и секвенцијална прекинувачка мрежа?

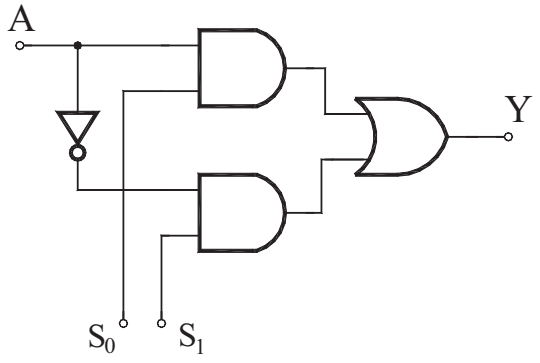
2-37. На сл. 2-52 е прикажана една комбинациона мрежа. За сите комбинации на влезовите S_1 и S_0 нацртај ја таблицата на вистинитост и одреди ја излезната функција *Y*. Од добиениот резултат коментирај го однесувањето на мрежата.



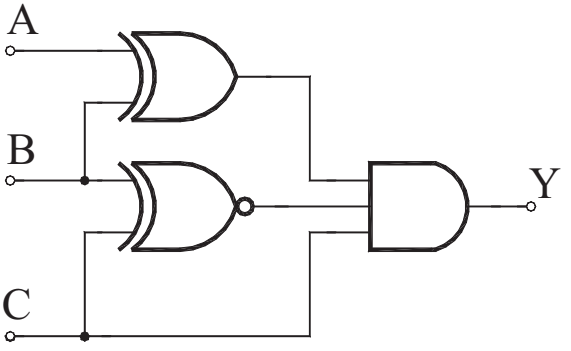
Слика за задача 2-37

2-38. За комбинационата мрежа прикажана на сл. 2-53 треба да се формира и пополни табела на вистинитост со S_1 и S_2 како влезни променливи, а Y како излезна. Врз основа на пополнетата табела објасни ја работата на мрежата.

2-39. За која комбинација на влезните променливи A, B и C излезот Y на логичката мрежа прикажана на сл. 2-54 ќе биде **1**?

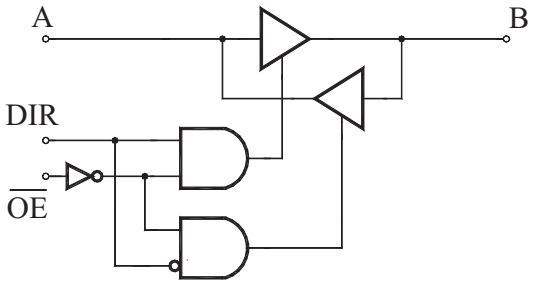


Слика за задача 2-38



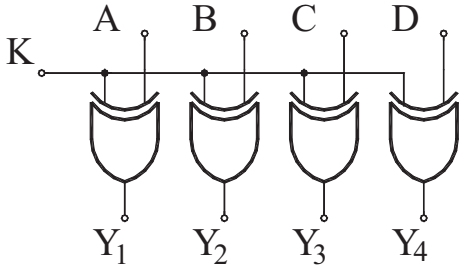
Слика за задача. 2-39

2-40. За комбинационата мрежа прикажана на сл. 2-56 треба да се пополни придружната комбинациона таблица и потоа да се опише и објасни нејзината функција и практична примена.



Контролни сигнали		Излези
		A B
0	0	
	1	
1	x	

Слика за задача 2-40



2-41. За логичката мрежа прикажана на сл. 2-55 претпостави дека променливата К еднаш има вредност 0, а еднаш 1 и за секој случај одреди ги логичките нивоа на излезните сигнали. Каква е улогата на променливата К? Образложи!

Слика за задача 2-41

2-42. Реализирај ги инверторот, И и ИЛИ логичките кола со по два влеза само со примена на (а) НИ; (б) НИЛИ кола со по два влеза.

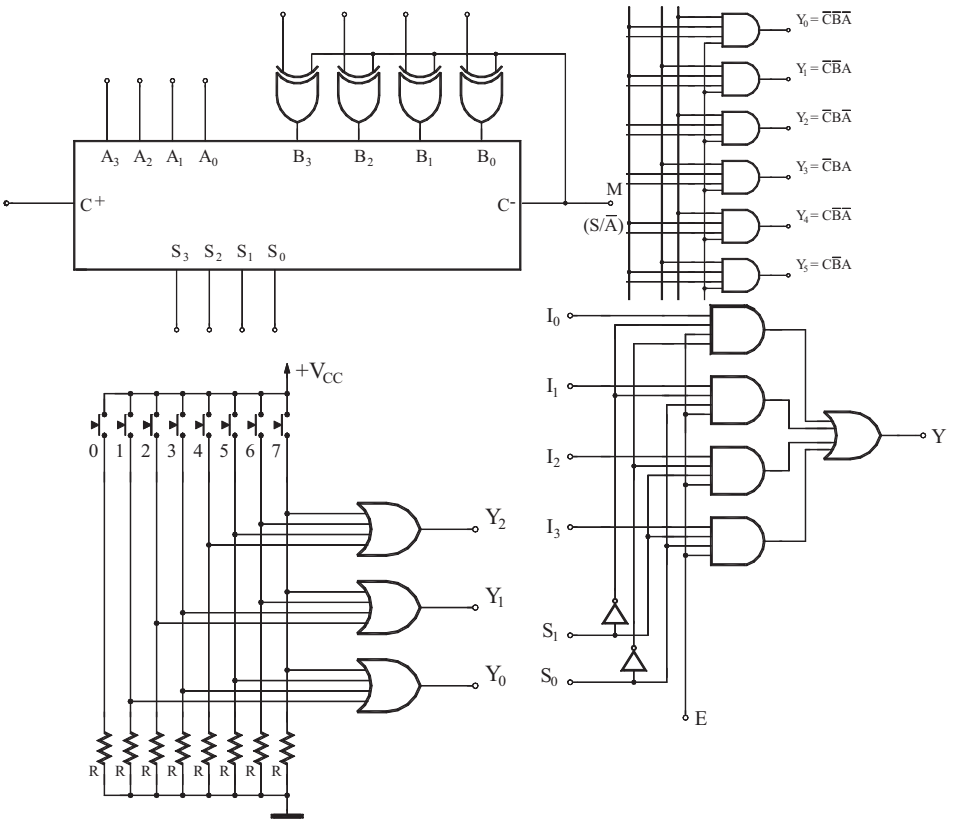
2-43. Со примена на И-ИЛИ мрежа во две нивоа, а потоа со НИ кола, да се реализираат функциите (а) $Y(A,B,C,D)=\overline{A}B+\overline{B}\overline{C}D+\overline{D}$; (б) $F(A,B,C,D)=A\overline{B}+BC\overline{D}+C$.

2-44. Со примена на ИЛИ-И мрежа во две нивоа, а потоа со НИЛИ кола, да се реализираат функциите (а) $Z(A,B,C,D)=(A+\overline{C})(\overline{A}+B+\overline{D})D$;
(б) $F(A,B,C,D)=(\overline{A}+B)(\overline{B}+C+D)\overline{D}$.

2-45. Функцијата $Y(X_1,X_2,X_3,X_4)=X_1X_2+X_1X_3X_4+X_1X_2X_3X_4$ треба да се реализира само со примена на НИ логички кола, (*) при што секое коло да има само по два влеза.

2-46. Функцијата $Y(X_1,X_2,X_3,X_4)=(X_1+X_2)(X_1+X_2+X_3)(X_1+X_2+X_3+X_4)$ да се реализира со користење на НИЛИ порти, (*) при што секоја да има само по 2 влеза.

2-47. (*) Функцијата $Y(X_3,X_2,X_1,X_0)=\sum m(0,1,2,5,7,12,13,14)$ да се претстави во СКНФ, а потоа да се минимизира и реализира со (а) НИЛИ (б) НИ логички кола.



3. КОМБИНАЦИСКИ МРЕЖИ

По изучувањето на оваа тематска целина

- ќе ја разберете логичката структура на следниве комбинациски мрежи:
  - коло за собирање и коло за одземање;
  - кодер, декодер;
  - мултиплексер, демултиплексер;
- ќе можете да анализирате поедноставни комбинациски мрежи;
- ќе умеете да го опишувате начинот на функционирање на поедноставни комбинациски мрежи;
- ќе можете да формирате посложени комбинациски мрежи;
- ќе можете да решавате задачи со комбинациски мрежи;

I) КОЛА ЗА РЕАЛИЗАЦИЈА НА АРИТМЕТИЧКО - ЛОГИЧКИ ФУНКЦИИ

3.1. ВОВЕД

Голем број од сложените операции во дигиталната техника можат да се извршат со погодно врзување на повеќе соодветни логички кола. *Добиените дигитални мрежи, во ситуација кога излезните сигнали од мрежата зависат само од моменталните комбинации на вредности на влезните сигнали, претставуваат* **комбинациски мрежи**. Претходните состојби (вредности) на излезите од мрежата немаат влијание врз нивните следни состојби, од каде произлегува дека комбинациските мрежи немаат особина на помнење на информациите.

Комбинациските мрежи имаат многу широка примена заради што се среќаваат скоро во секој дигитален уред. Во компјутерските системи овие мрежи се користат за извршување на пресметувачките операции, аритметички или логички, за генерирање на одредени нумерички вредности, за кодирање и декодирање на информациите, за пронаоѓање на адресираните локации во мемориите, за селектирање на одредени поврзувања итн. Разновидноста на нивната примена е една од причините заради која комбинациските мрежи обично се нарекуваат според функцијата што ја извршуваат, како на пример: собирач, одземач, комплементер, компаратор, кодер, декодер, мултиплексер (селектор), демултиплексер (дистрибутер) итн. Некои од комбинациските мрежи се така организирани и реализирани за да можат да се користат во праксата како мемориски компоненти чија содржина може само да се чита, иако немаат мемориски ќелии.

3.2. КОЛА ЗА СОБИРАЊЕ И ОДЗЕМАЊЕ

Во дигиталните уреди, како што се на пример калкулаторите и компјутерите, процесот на обработка на информациите се реализира со примена на посебни кола коишто извршуваат аритметички и/или логички операции.

Во продолжение ќе ги опишеме основните комбинациски кола за извршување на двете основни пресметувачки операции: собирање и одземање. Бидејќи при реализација на одземањето се јавува потреба за комплементарните вредности на броевите, ќе го анализираме и колото кое ја извршува логичката операција комплементирање.

3.2.1. БИНАРНИ СОБИРАЧИ

Колото за собирање има базична улога при извршувањето на основните аритметички операции. Да напоменеме и тоа дека дигитална сметачка машина може да се конструира само со примена на едно коло за собирање кое ќе игра улога на аритметичка единица. Другите три операции можат да се изведат со програмирање, односно со задавање на инструкции со кои собирачот ќе се искористи за реализирање на останатите три операции. Согледувајќи ја неговата важност, најнапред ќе се задржиме на реализацијата на собирачот.

Полусобирач: Основен модул којшто се користи во логичките аритметички компоненти е *полусобирачот* (анг. *half adder*, *HA*). Улогата на полусобирачот е да собере два бита, и како резултат да ја даде нивната сума, но воедно да генерира и бит за пренос, според правилата за собирање дадени во табелата таб. 3-1. Табелата на вистинитост на полусобирачот е прикажана како таб. 3-2, а неговиот логички симбол е даден на сл. 3-1.

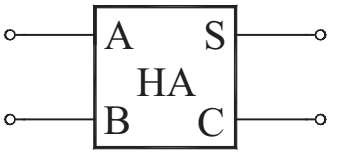
Од сл. 3-1 се гледа дека полусобирачот има два влеза за по еден бит A и B, и два излеза: еден за битот на збирот (сумата) S и еден за битот на преносот (анг. *carry*) C.

0+0=0
0+1=1
1+0=1
1+1=0 и пренос 1.

Таб. 3-1. Правила за аритметичко собирање

A B	S C
0 0	0 0
0 1	1 0
1 0	1 0
1 1	0 1

Таб. 3-2. Комбинациона табела на полусобирач

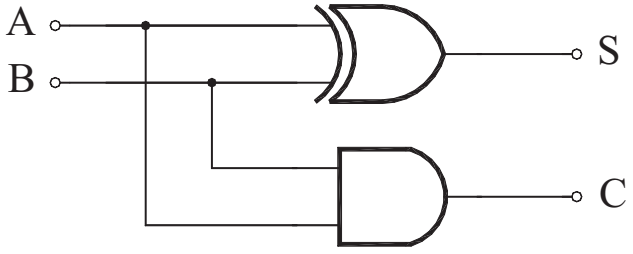


Сл. 3-1. Логички симбол на полусобирач

Битот на збирот ќе биде 0 (S = 0) ако битовите A и B се меѓусебно еднакви: или двата се истовремено 0 или се 1. Во првиот случај битот за пренос ќе биде 0 (C = 0), а во вториот ќе се појави пренос (C = 1). Збирот ќе има вредност 1 (S = 1) само ако A и B имаат меѓусебно комплементарни вредности при што и за двата случаи нема да има пренос (C = 0). Овие односи можат да се напишат во аналитички облик, со следниве логички равенки:

$$S = A\overline{B} + \overline{A}B \quad \text{и} \quad C = AB \tag{3-1}$$

Од нив произлегува и самата реализација на полусобирачот, која е прикажана на сл. 3-2. Битот на сумата S се добива од ЕКС-ИЛИ коло чии два влеза се A и B, додека битот на преносот C се формира од И колото со истите влезни битови A и B.



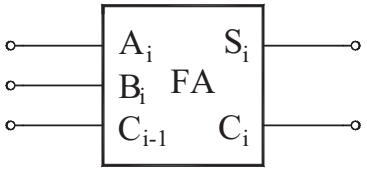
Сл. 3-2. Логичка шема на полусобирач

Целосен (потполн) собирач: Со полусобирачот можат да се собираат само еднобитни бинарни броеви од причини што кај него како влез не се јавува преносот од собирањето на битовите од претходното ниво, т.е. од местото со помала тежина. Колото што го решава овој проблем се вика *целосен (потполн)* собирач (анг. *full adder, FA*), а неговиот логички симбол е прикажан на сл. 3-3. Од сликата се гледа дека кај потполниот собирач постои додатен влез на кој се носи преносот од претходното тежинско ниво. Ознаките доаѓаат оттаму што претпоставуваме дека потполниот собирач ги собира битовите кои се наоѓаат на одредено i-то место кај секој бинарен број. Табелата на вистинитост на целосниот собирач е претставена како таб. 3-3.

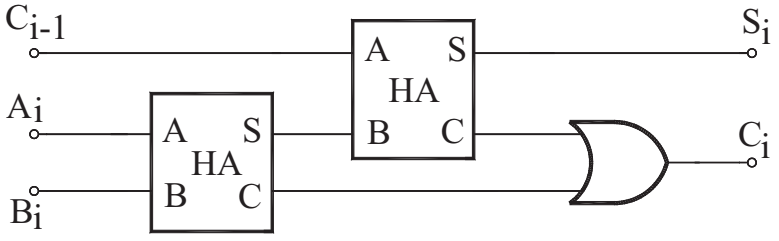
Целосниот собирач може да се конструира со примена на два полу-собирачи и едно ИЛИ коло како што е прикажано на сл. 3-4. Оваа реализација не е најекономична и затоа целосниот собирач обично се изведува директно со анализа на табелата на вистинитост која е дадена во таб. 3-3, од која произлегуваат различни практични решенија.

A_i B_i C_{i-1}	S_i C_i
0 0 0	0 0
0 0 1	1 0
0 1 0	1 0
0 1 1	0 1
1 0 0	1 0
1 0 1	0 1
1 1 0	0 1
1 1 1	1 1

Таб. 3-3. Табела на вистинитост



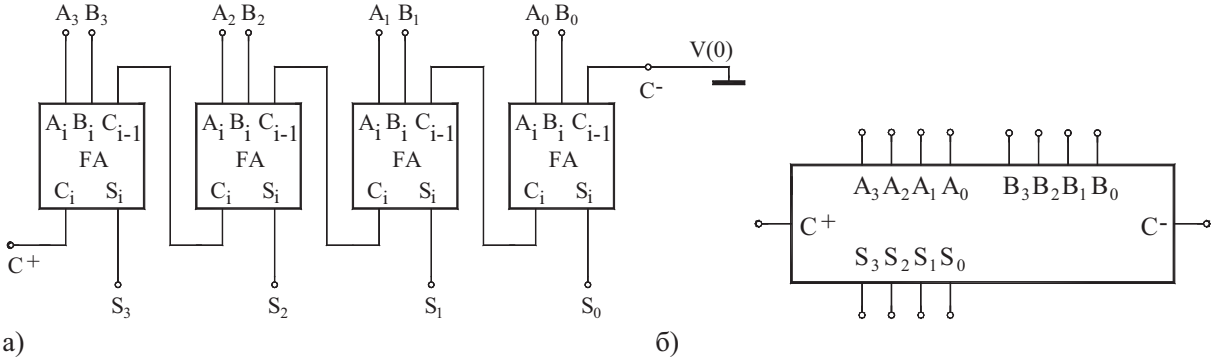
Сл. 3-3. Логички симбол на целосен собирач



Сл. 3-4. Логичка шема на целосен бинарен собирач

Бинарен паралелен собирач: Со примена на повеќе потполни собирачи може да се проектира собирач за повеќебитни броеви. Знаејќи дека со еден потполн собирач може да се соберат само два бита, станува јасно дека бројот на употребените целосни собирачи зависи од тоа колкава е должината на броевите што се собираат.

На сл. 3-5 е прикажан еден *паралелен собирач* за позитивни 4 битни броеви (броеви со должина од еден *nibble*). Оваа реализација користи 4 потполни собирачи, при што потполниот собирач со кој се собираат двата најмалку значајни битови (LSB-битовите) од броевите на влезот C_{i-1} има логичка 0. Ова е од причини што тие битови се од најниското позиционо ниво, па кај нив не може да се јави пренос од претходното ниво, бидејќи такво ниво нема. Собирањето на овие два бита не се остварува со употреба на полусобирач затоа што тогаш нема да може да се изврши поврзување на повеќе вакви собирачи со цел реализација на собирање на броеви со должина од два или повеќе нибли.



а) б)

Сл. 3-5. Логичка структура и симболичка ознака на паралелен 4 битен собирач

При собирањето на два броја карактеристична е појавата на *префрлување* (анг. *overflow*). Тоа фактички претставува пренос кој се појавува по собирањето на двата најмногу значајни битови (MSB – битовите). Ако се појави 1 на излезот C_i од потполниот собирач кој ги собира третите битови (последниот потполн собирач), тоа значи дека добиениот збир е број кој не може да се претстави само со четири бита, т.е. број што е поголем од $1111_{(2)}=15_{(10)}$ како најголем број што може да се прикаже со 4 бита. Заради тоа оваа линија се означува како линија за *детекција на префрлувањето* (C+).

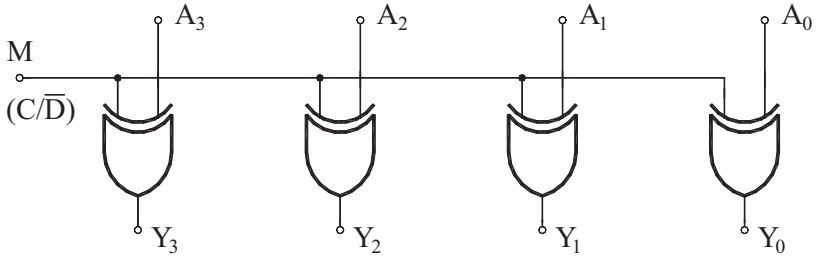
3.2.2. КОЛО ЗА КОМПЛЕМЕНТИРАЊЕ

Пресметувачките операции во дигиталните уреди најчесто се изведуваат во природниот бинарен броен систем или во некој од бинарните системи за прикажување на броеви со предзнак. Според ова и операцијата комплементирање ќе се извршува со различни логички кола, со конфигурација која ќе зависи од тоа за кој систем станува збор. Во принцип, комплементарните вредности и во двата система се добиваат на ист начин со примена на познатата равенка $\overline{A} = K\text{-}A$ за првиот и за вториот комплемент на некој даден број А. Да се потсетиме дека првиот комплемент се однесува на дополнување до најголемиот број во бројниот систем, а вториот до опсегот на броеви на бројниот систем. Така, во бинарниот броен систем постојат комплемент на единица (единечен комплемент, 1's) и комплемент на двојка (двоен комплемент, 2's). Покрај ова, уште еднаш ќе потенцираме дека комплементарните вредности на броевите се од посебен интерес во бинарниот броен систем, бидејќи со нив се претставуваат негативните броеви.

Единечниот комплемент се добива така што секој бит од бројот се заменува со неговата комплементарна вредност. Да претпоставиме дека бинарниот број е зададен во обликот $A=A_3A_2A_1A_0$, и како еден илустративен пример, да го земеме бројот $A=1011$. Неговиот прв комплемент $A_{(1s)}$ ќе биде $A_{(1s)}=0100$. Јасно е дека добивањето на првиот комплемент може едноставно да се изведе со употреба на инверторски (НЕ) кола, бидејќи $A_{(1s)} = \overline{A_3}\ \overline{A_2}\ \overline{A_1}\ \overline{A_0}$.

Кога се применува сериски начин за извршување на оваа операција, тогаш комплементот на зададениот број може да се добие само со еден инвертор. Имено, на влезот од инверторот последователно се носат поодделните битови на бројот A_0, A_1, A_2, A_3, а на излезот од инверторот ќе се добијат нивните комплементарни вредности $\overline{A_0}, \overline{A_1}, \overline{A_2}, \overline{A_3}$. Серискиот начин на работа е прилично бавен, заради што обично се применува паралелна обработка на податокот.

При паралелната постапка бројот на инверторски кола мора да биде еднаков со должината на зборот, т.е. со вкупниот број битови со кои се претставува секој поединечен податок. Добивањето на единечниот комплемент наједноставно се врши со употреба на ЕКСИЛИ логички кола, според шемата на логичкиот комплементер прикажана на сл. 3-6.



M	*Y*
0	*A*
1	$\overline{A}$

Сл. 3-6. Логичка шема на коло за комплементирање и функционална табела

За секој излез од ЕКСИЛИ колата важи равенката:

$$Y_i = M\overline{A_i} + \overline{M}A_i \qquad (3\text{-}2)$$

Од оваа логичка функција произлегува дека за M=1 на излезите од комплементорот Y_i се добиваат поединечните комплементни вредности на секој бит од бројот што се доведува на неговиот влез ($Y_i = \overline{A_i}$), а со тоа се добива комплементот $Y = \overline{A} = A_{(1s)}$ на зададениот број А.

Од друга страна, ако на контролната линија M доведеме 0 (M=0) на излезите *Y* од комплементерот се добиваат битовите на влезниот број A ($Y_i=A_i$), а со тоа и самиот број во директен (вистински) облик ($Y=A$).

Вториот комплемент на даден бинарен број се добива кога на единечниот комплемент му се додаде 1. Така на пример, првиот комплемент на бројот $A=0101$ е $\overline{A} = A_{(1s)} = 1010$, а вториот комплемент $A_{(2s)}=1011$. Според ова, мрежата за реализација на вториот комплемент, покрај ЕКСИЛИ колата со кои се генерира единечен комплемент ќе треба да користи и собирач за собирање на првиот комплемент со логичка константа 1.

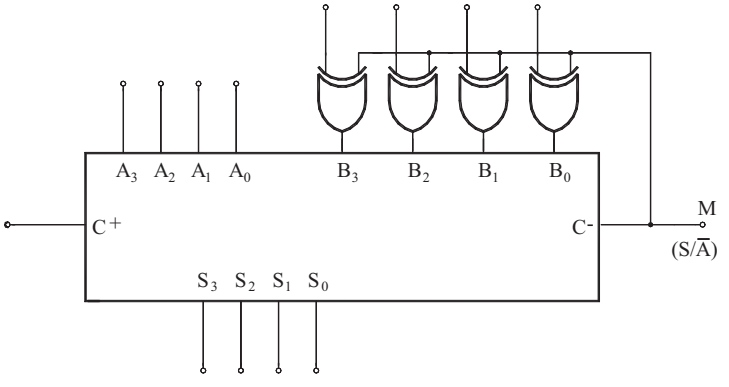
3.2.3. КОЛО ЗА ОДЗЕМАЊЕ

Аритметичките операции што треба да се извршат кај бинарните броеви со знак, вообичаено се изведуваат со примена на *комплементарната аритметика*, со примена на директните (вистинските) вредности на броевите и нивните комплементи. Ваквиот начин на работа има одредени предности, посебно кога се работи за операции со негативни броеви. Веќе знаеме дека негативните броеви можат да се прикажат на три начини: со предзнак во SM системот, со прв комплемент во DC системот или со вториот комплемент во RC системот. Во сите овие случаи најзначајниот бит - MSB битот има вредност 1, со што и се означува негативната вредност на било кој бинарен број во машинскиот јазик. *Изразувањето на негативните броеви во комплементарен облик овозможува операцијата одземање да се сведе на собирање, при што со битот за знак се манипулира на ист начин како и со вредносните битови.*

Во случај броевите да се претставени со својата директна вредност и бит за предзнак во SM системот, тие како целина можат да се собираат само кога имаат исти предзнаци. Оваа констатација очигледно посочува на оправданоста за претставување на негативните броеви со помош на нивните комплементи.

Операциите на собирање и одземање со првиот комплемент во DC системот се едноставни, бидејќи лесно се добива првиот комплемент на бројот, меѓутоа при појавата на пренос во резултатот треба да се изврши собирање на пренесената 1 со резултатот, така што доаѓа до отежнување при реализацијата на споменатите операции.

Од друга страна, операциите собирање и одземање со примена на вториот комплемент во RC системот се поедноставни отколку со првиот, бидејќи отпаѓа постапката за додавање на 1 од преносот, така што работата со двојниот комплемент е далеку повеќе застапена. Имено, одземањето на позитивните броеви може да се оствари посредно така што ќе се соберат намалениот и вториот комплемент на намалителот. Постапката за посредно одземање е прикажана на сл. 3-7 со примена на паралелниот четирибитен собирач од сл. 3-5 и на дигиталниот комплементер од сл. 3-6.



Сл. 3-7. Логичка шема на мрежа за собирање и одземање

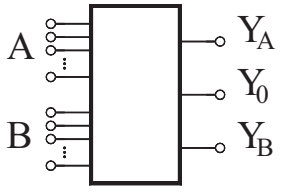
Оваа конфигурација овозможува собирање, ако на контролниот влез M ($S/\overline{A}$) се донесе ниско ниво (M=0), додека одземање ако M=1. Првиот четирибитен број се носи директно на влезовите од собирачот A_i, додека вториот број се носи на влезовите B_i преку ЕКСИЛИ колата на колото за комплементирање. Прикажаната логичка структура ќе работи како собирач ако на контролната линија M се донесе ниско логичко ниво (M=0), така што ЕКСИЛИ колата едноставно го пренесуваат четирибитниот собирок B до влезот од собирачот во неизменет – директен облик.

Меѓутоа, ако на контролната линија M се донесе високо логичко ниво (M=1), мрежата се трансформира во коло за одземање. Имено, во овој случај, на излезите од ЕКСИЛИ колата се формира првиот комплемент на намалителот B, но затоа што M=1, собирачот практично ја додава таа 1 со што вредноста на бројот B се изразува во втор комплемент. Така на влезовите од интегрираната компонента се присутни намаленикот A и вториот комплемент $B_{(2s)}$ на намалителот B, т.е. неговата негативна вредност, па резултатот од собирањето означен со S кој се добива на излезите S_i каде i=1,2,3,4 е всушност разликата на броевите A и B.

3.3. ДИГИТАЛЕН КОМПАРАТОР

Од дигиталните уреди многу често се бара да извршат споредување на два бинарни броја, така што како резултат се добива информација за тоа дали едниот број е поголем, еднаков или помал од другиот. Логичката мрежа која ја реализира оваа функција се вика **дигитален** или **бинарен компаратор,** а нејзината блок шема е претставена на сл. 3-8.

Бинарните броевите што се споредуваат имаат еднаков број битови (n) кои ги претставуваат влезовите во логичката компонента. Информацијата за тоа кој од броевите е поголем, односно дали се тие еднакви меѓу себе, се добива преку трите излезни линии, соодветно на функционалната таб. 3-4. Од табелата се гледа дека соодветно на односот помеѓу двата броја, само на една од излезните линии ќе се појави логичка 1.



Сл. 3-8. Логичка шема на дигитален компаратор

A B	Y_0 Y_A Y_B
$A = B$	1 0 0
$A > B$	0 1 0
$A < B$	0 0 1

Таб. 3-4. Функционална табела

Заради разбирање на суштината на проблемот, ќе претпоставиме наједноставен случај: споредување на два еднобитни бинарни броја A и B. Врз основа на функционалната табела (таб. 3-4), може да се изведе табелата на вистинитост на оваа логичка компонента која е дадена како таб. 3-5.

Логичката шема на еднобитниот компаратор кој врши споредување на битот A со битот B, е прикажана на сл. 3-9. На сликата покрај единственото ЕКСНИЛИ логичко коло, се применети и два инвертори и две И логички кола.

Централното место во средината на логичката шема го зазема ЕКСНИЛИ логичкото коло бидејќи ова коло препознава еднакви влезни комбинации. Тоа е очигледно од неговата табела на вистинитост (таб. 3-5) од која лесно може да се заклучи дека ЕКСНИЛИ колото фактички игра улога на детектор на еднаквост, затоа што неговиот излез е 1 само кога и двата влезни битови се меѓусебно еднакви. Од друга страна, излезот ќе биде 0 само ако битовите се различни меѓу себе.

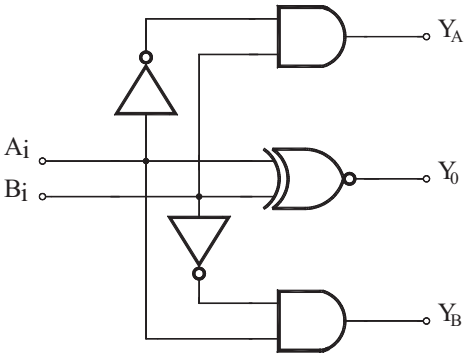
Искажаниот услов може да се запише со следнава логичка равенка:

$$Y_0 = (\overline{A \oplus B}) = AB + \overline{A}\overline{B} = \begin{cases} 1 \text{ ако A=B}, \\ 0 \text{ ако A}\neq\text{B}. \end{cases} \quad (3\text{-}3)$$

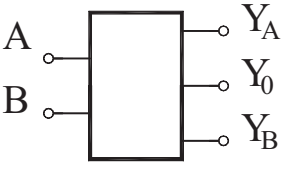
Условот $A>B$ ќе биде исполнет со равенката $Y_A = A\overline{B}$, бидејќи само кога A=1 и B=0 се добива Y_A=1. Слично, условот $A<B$ ќе биде исполнет согласно со равенката $Y_B = \overline{A}B$. Во овој случај Y_B =1, ако и само ако A=0 и B=1.

A	B	Y_0	Y_A	Y_B
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

Таб. 3-5. Табела на вистинитост



а)



б)

Сл. 3-9. Логичка шема на еднобитен компаратор и негов симбол

Презентираната анализа и принцип на резонирање може понатаму да се развива и надоградува заради реализација на кола за споредување на два повеќебитни броја.

Како компаратор на два четирибитни бинарни броеви A и B, може да се употреби и дигиталната мрежа за собирање и одземање од сл. 3-7. Имено, ако $A=B$, сите излези S_i каде i=1,2,3,4 ќе се наоѓаат на ниско логичко ниво, т.е. ќе бидат 0-и. ($S_i=0$). Меѓутоа, ако $A>B$ на преносот од највисокото ниво, т.е. на излезот (C+) ќе се појави логичка 1, додека при $A<B$ на оваа излезна линија ќе се појави логичка 0.

II) ПРЕКИНУВАЧКИ МАТРИЦИ

3.4. ВОВЕД

Прекинувачките матрици се посложени комбинациски логичко-прекинувачки мрежи со поголем број влезови и излези. За нив е најважно тоа што логичките состојби на било кој излез од мрежата зависат само од моменталните логички состојби на влезовите. **Прекинувачките матрици** *се составени од прекинувачки елементи кои се наредени во низи, по редици и колони, така што формираат матрични структури*. Освен ова, и самите прекинувачки функции што се реализираат со овие мрежи се дадени во облик на матрици. Меѓутоа, овие логички мрежи многу почесто се сретнуваат со нивните конкретни функционални имиња, од кои како најпознати ќе ги наведеме: кодерите, декодерите, мултиплексерите и демултиплексерите.

Прекинувачките матрици можат да се реализираат со примена на логички кола, но многу почесто, можат да се најдат и комплетни решенија што се изведени во техника на интегрирани кола.

3.5. КОДЕРИ И ДЕКОДЕРИ

Кодерот (анг. encoder) *е логичка мрежа која врши кодирање (претворување) на некоја нумеричка информација од еден броен систем кој е примарен во друг, секундарен, броен систем или код*. Како примарен броен систем се подразбира декадниот систем, додека секундарен систем најчесто е природниот бинарен броен систем, или некој бинарен код како на пр. природниот *BCD* код (*NBCD)*, т.е. *8421*-кодот, или некој друг.

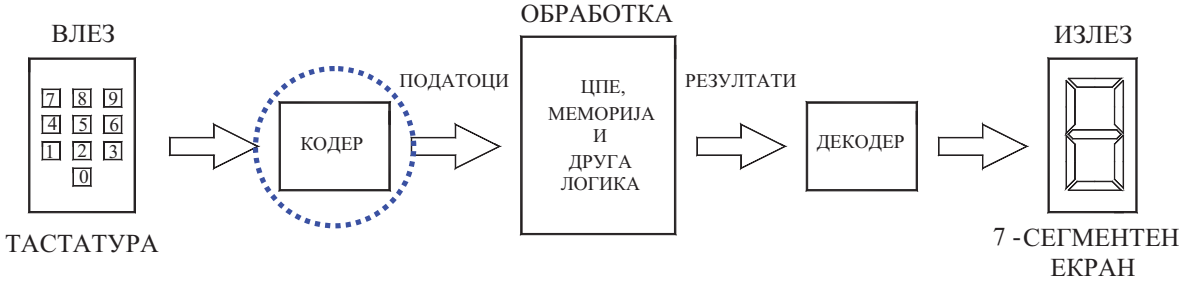
И **декодерот** врши претвоурање, меѓутоа во обратна насока. *Тој ја декодира дигиталната информација од секундарниот броен систем во примарниот,* т.е. податокот запишан во бинарна форма го трансформира во декаден облик.

Покрај овие два типа комбинациски мрежи постојат и логички мрежи што се викаат **конвертори на код**. Тие ја преведуваат информацијата од еден секундарен систем во друг секундарен систем или код.

Со текот на времето и развојот на дигиталните системи се дојде до тоа декадниот систем да не е секогаш примарен систем, така што една иста логичка мрежа за еден систем претставува кодер, за друг декодер, а за трет конвертор на код. Затоа попрактично е за сите три типа кола да се користи еден заеднички назив: **преведувач** или **транслатор на код**. Меѓутоа, во практиката и понатаму најчесто се применуваат класичните функционални имиња кои најпрво ги дефиниравме: кодер или декодер. Од излагањето што следи ќе видиме дека кодерот е всушност ИЛИ матрична структура, додека декодерот е матрична структура со И кола.

3.5.1. КОДЕР

Пред да ја започнеме анализата на кодерот, за кратко ќе се фокусираме на сл. 3-10 која претставува една многу едноставна блок-шема на калкулатор. Во овој дигитален систем декадниот влез од тастатурата мора да биде претворен во бинарен облик за да може понатаму како влезен податок да се обработува во процесорот.



Сл. 3-10. Местото на кодерот во наједноставна блок-шема на калкулатор

Најчеста постапка за изразување на декадните цифри е кодирањето во познатиот *NBCD* код (8421 или природен бинарен код), чија кодна табела е означена со таб. 3-6. Дигиталната компонента за оваа намена е позната како *декаден-во-NBCD кодер (DEC/NBCD)*. Тој треба да има десет влезови, по еден за секоја декадна цифра и четири излези на кои се добиваат четирите битови D, C, B и A на *NBCD* кодниот збор.

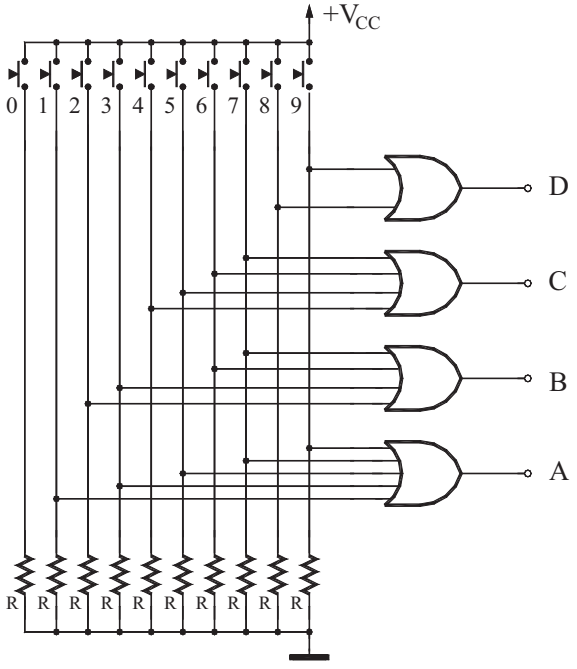
Изборот и поврзувањето на прекинувачките елементи во формирањето на кодерската матрица ќе го објасниме со примерот за добивање на третиот бит (*B*) во кодниот *NBCD* збор. Од таб. 3-6 лесно се забележува дека битот *B* ќе има вредност 1 (*B*=1) ако се активира влезот на цифрата 2, или цифрата 3, или ако се појави цифрата 6, или ако се

активира линијата на цифрата 7. Од анализата е јасно дека излезот *B* може да се реализира со едно ИЛИ логичко коло што има четири влеза и тоа линиите 2, 3, 6 и 7. Слично се размислува и се донесува заклучок за изведбата на другите три бита: *D*, *C*, A.

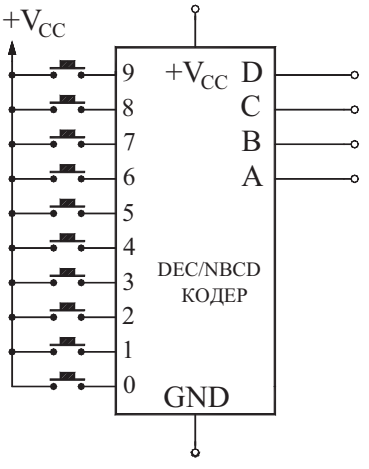
Декад. цифра	8421 (NBCD) D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Таб . 3-6. Кодна табела на 8421 (NBCD) кодот

На сл. 3-11 е прикажана една реализација на DEC/NBCD кодер, изведена со употреба на еднонивовска матрична структура формирана од ИЛИ логички кола. Од сликата се гледа дека кај овој кодер влезовите се активни на високо ниво, бидејќи со притиснување на еден од тастерите на соодветната влезна линија доведуваме логичка 1, т.е. напон на напојување $+V_{CC}$. Така со притиснување на одреден тастер на некоја од влезните линии се побудуваат соодветните ИЛИ кола и на нивните излези се добива високо логичко ниво (*1*), додека на останатите излези од ИЛИ колата кои не се побудени се јавува ниско логичко ниво (*0*). На овој начин се добива *NBCD* кодниот збор кој одговара на декадната цифра која се наоѓа на тастерот што беше активиран. На сл. 3-12 е прикажан вака реализираниот DEC/NBCD кодер како посебна логичка компонента.



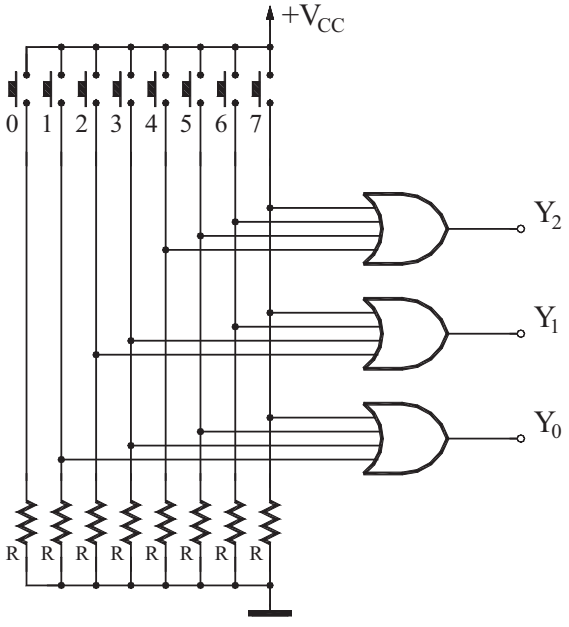
Сл. 3-11. Логичка шема на DEC/NBCD кодер



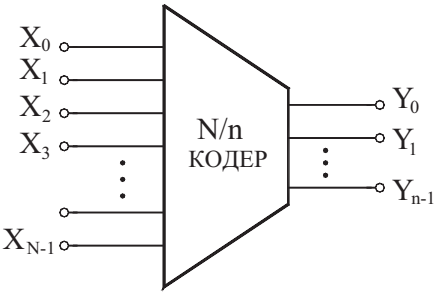
Сл. 3-12. Блок шема на DEC/NBCD кодер

Од претходните слики се гледа дека DEC/NBCD кодерот има 10 влезови и 4 излези. Знаејќи дека со 4 бита може да се кодираат 2^4=16 различни комбинации, може да се заклучи дека кај DEC/NBCD кодерот остануваат неискористени 6 од можните 16 излезни комбинации (состојби) и дека тие никогаш нема да се појават на излезот од мрежата.

Имајќи го предвид кажаното, може да се заклучи дека во *општ случај кодер со* $N=2^n$ *влеза, максимално може да има n излези*. Во врска со ова, на претходната логичка шема може да ѝ се додадат уште шест влезови според табелата на хексадецималниот броен систем со што би се добил хексадецимален-во-бинарен кодер (HEX/BIN). Слично, ако станува збор за октален-во-бинарен кодер може (OCT/BIN) да се примени табелата за конверзија од октален во бинарен броен систем и да се добие логичката структура која е прикажана на сл. 3-13 чија симболичка ознака е дадена на сл. 3-14. Комбинационата табела на вистинитост таб. 3-7 го дообјаснува принципот на работа на ваквиот кодер.



Сл. 3-13 Логичка структура на OCT/BIN кодер



Сл 3-14. Симболичка ознака на OCT/BIN кодер

Влезови								Излези			Индекс
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	Y_2	Y_1	Y_0	i
1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1	1
0	0	1	0	0	0	0	0	0	1	0	2
0	0	0	1	0	0	0	0	0	1	1	3
0	0	0	0	1	0	0	0	1	0	0	4
0	0	0	0	0	1	0	0	1	0	1	5
0	0	0	0	0	0	1	0	1	1	0	6
0	0	0	0	0	0	0	1	1	1	1	7

Таб. 3-7. Комбинациона табела на OCT/BIN кодер

Окталниот-во-бинарен кодер ќе има осум влезни линии I_0 до I_7, при што секоја линија презентира по една октална цифра и три излезни линии Y_2, Y_1 и Y_0 на кои ќе се добиваат тробитните бинарни еквивалентни вредности. Логичките равенки со кои се опишува принципот на работа на кодерот се:

$$Y_0 = I_1 + I_3 + I_5 + I_7; \quad Y_1 = I_2 + I_3 + I_6 + I_7; \quad Y_2 = I_4 + I_5 + I_6 + I_7; \qquad (3\text{-}4)$$

Општо земено, најважно за кодерот е да се забележи дека за секоја од $N=2^n$–те влезни линии на n-те излезни линии треба да се генерира единствен бинарен коден збор според конкретната кодна табела.

Кај кодерите реализирани според сл. 3-10 и сл. 3-12 како проблем се јавува фактот што на излезот се добиваат сите нули како да е активирана влезната линија за цифрата 0, а всушност не е активирана ниту една влезна линија. Излезниот резултат е ист и ако навистина се активира тастерот на 0-та линија. Заради елиминирање на овој проблем на кодерот може да му се додаде уште една влезна линија со која ќе може да се контролира неговото овозможување да работи или не. Покрај неа, и на излезот може да се додаде уште една линија која ќе има вредност 1 ако е притиснат било кој од влезните тастери, односно вредност 0 ако ниту еден од влезовите не е активен.

Имајќи ја во предвид досегашната анализа, може да се заклучи и тоа дека при реализацијата на презентираните кодери беше претпоставено дека во даден момент само една од влезните линии можеше да се наоѓа на високо ниво, т.е. на логичка 1. Заради ова, евентуалното истовремено притиснување на два или повеќе тастери кај кодерите од сл. 3-10 и сл. 3-12 ќе предизвикаше грешка во работата на кодерот и појава на неважечки (нелегални, нелегитимни) излезни комбинации.

3.5.2. ПРИОРИТЕТЕН КОДЕР

Недостатокот на претходно анализираните кодери да генерираат грешка при активирање на два или повеќе влезови се надминува со т.н. *кодери со приоритет*. Во практиката, ваквите логички компоненти најчесто се изведуваат во интегрирана техника. Кај приоритетните кодери ако истовремено се притиснат два или повеќе тастери, на излезот ќе се добие кодната комбинација која одговара на оној тастер од притиснатите што има поголема нумеричка вредност. Според ова, кодерите со приоритет ќе имаат модификувана табела на вистинитост како што е на пр. табелата на OCT/BIN (окталниот-во-бинарен) кодер со приоритет презентирана како таб. 3-8.

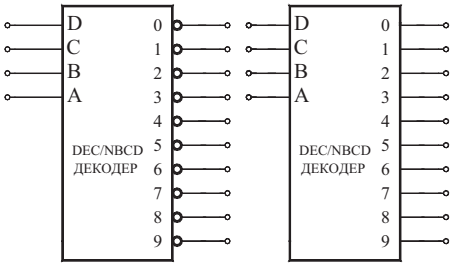
Влезови								Излези			
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	Y_2	Y_1	Y_0	V
0	0	0	0	0	0	0	0	x	x	x	0
1	0	0	0	0	0	0	0	0	0	0	1
x	1	0	0	0	0	0	0	0	0	1	1
x	x	1	0	0	0	0	0	0	1	0	1
x	x	x	1	0	0	0	0	0	1	1	1
x	x	x	x	1	0	0	0	1	0	0	1
x	x	x	x	x	1	0	0	1	0	1	1
x	x	x	x	x	x	1	0	1	1	0	1
x	x	x	x	x	x	x	1	1	1	1	1

Таб. 3-8. Комбинациона табела на OCT/BIN кодер со приоритет

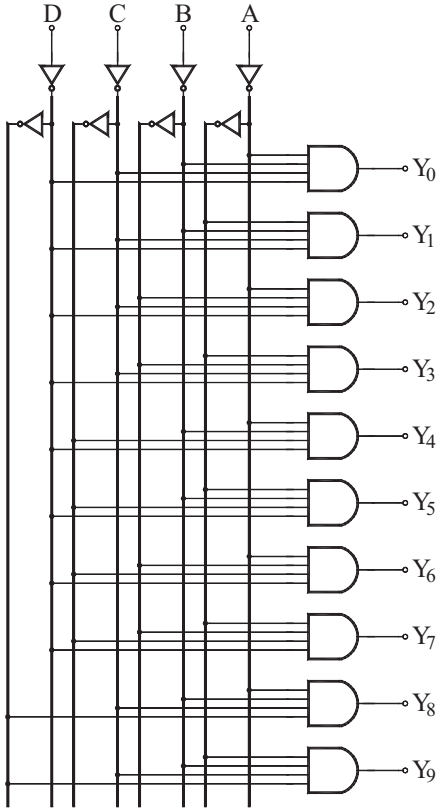
Ако добро ја погледнеме оваа табела, ќе забележиме дека без оглед на логичките состојби на другите влезови, на излезот секогаш ќе се добива кодниот збор кој одговара на онаа линија од активираните која има најголема нумеричка вредност. Во презентираната табела на вистинитост се појавува уште една излезна линија означена со V која укажува на тоа дали некоја од влезните линии е активна или не. Логичката вредност присутна на овој излез овозможува да се прави разлика помеѓу случаите кога е активирана нултата влезна линија (кога е притиснат 0-иот тастер) и кога не е активиран ниту еден влез.

3.5.3. ДЕКОДЕР

Бидејќи сите информации во дигиталната обработка се претставуваат во бинарен облик, јасно е дека тие податоци треба да бидат преведени во декадни броеви секогаш кога нив ќе треба да ги употреби човекот. *Постапката со која се преведуваат бинарните кодови во друг облик што е попогоден за понатамошна употреба се вика декодирање, а самата дигитална компонента која тоа го реализира се вика* **декодер**.



а) симболичка ознака



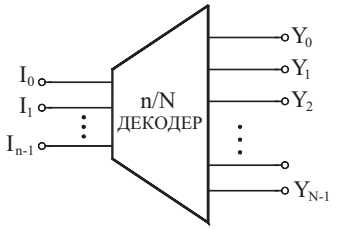
б) логичка шема (структура)

Сл.3-15. NBCD/DEC декодер

Декодирањето на бинарните податоци во декадни броеви е процедура што е инверзна на кодирањето. Заради ова, блок-шемата на *NBCD*-во-декадниот декодер (NBCD/DEC) која е дадена на сл. 3-15 а) ќе има десет излези, а четири влеза. Излезите се означени од Y_0 до Y_9, а влезовите се *D, C, B, A*. Излезите се активни на високо ниво, што значи дека логичка 1 ќе се појави само на оној излез чиј влезен *NBCD* коден збор ја претставува нумеричката вредност на тој излез во бинарен облик.

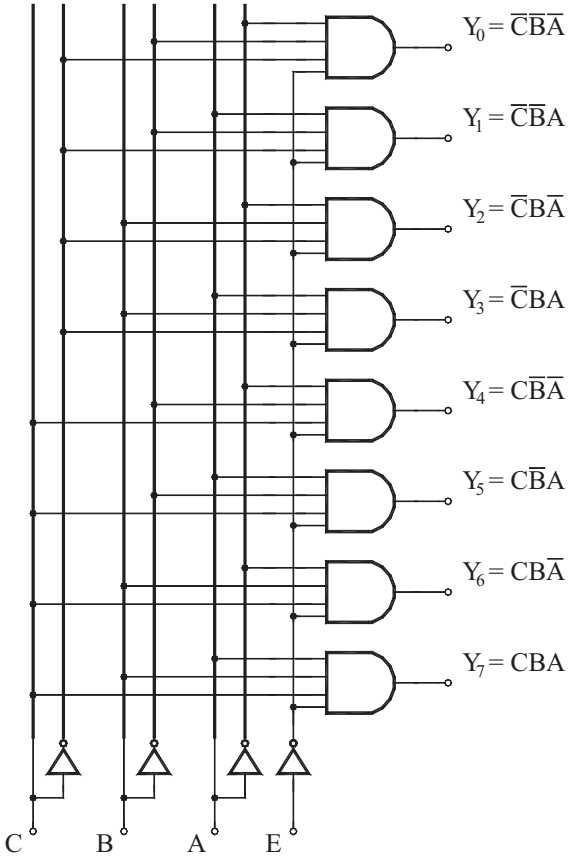
Една реализација на *NBCD/DEC* декодер е прикажана на сл. 3-15 б). Декодерската матрица е добиена со следнава анализа. Активен, т.е. да оди на високо ниво на логичка 1, треба да биде оној излез што одговара на конкретната *NBCD* кодирана декадна цифра која е донесена на влезот од декодерот. Така на пр. на излезот Y_6 ќе треба да се појави 1 само ако на влезот се јави комбинацијата DCBA = 0110, т.е. ако $\overline{D}$=1 и C=1 и B=1 и $\overline{A}$=1 ($Y_6 = \overline{D}CB\overline{A}$). Ова значи дека излезното логичко коло ќе биде И коло со четири влеза при што C и B се носат директно, а D и A комплементарно. Поврзувањето за останатите девет цифри се добива со размислување на аналоген начин со што конечно се добива *еднонивовска матрична И структура*.

Декодерот од сл. 3-15 б) ги отфрла погрешните информации затоа што ако на влезот се појави неважечка комбинација, т.е. коден збор кој не претставува *NBCD* цифра, како што се: 1010, 1011, 1100, 1101, 1110 или 1111, тогаш нема да биде активна ниту една излезна линија. Во праксата се среќаваат и такви изведби на декодери кои не ги отфрлуваат погрешните влезни комбинации.

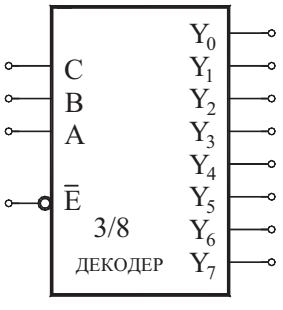


Сл. 3-16. Логички симбол на декодер

Знаејќи дека со 4 бита може да се кодираат 2^4=16 различни комбинации, станува јасно дека кај NBCD/DEC декодерот остануваат неискористени 6 од можните 16 излезни комбинации (состојби). Во врска со ова, може да се заклучи дека в*о општ случај декодер со n влеза може максимално да има N=2^n излези*.



Сл. 3-17. Логичка шема на BIN/OCT декодер со излези активни на високо ниво



Сл. 3-18. Блок шема на BIN/OCT декодер со излези активни на високо ниво

Така на пример, на претходната логичка структура може да и се додадат уште шест влезови според табелата на хекса-децималниот броен систем со што би се добил бинарен-во-хексадецимален (BIN/HEX) декодер. Слично, ако станува збор за бинарен-во-октален (BIN/OCT) декодер може да се примени табелата за конверзија од октален во бинарен броен систем и да се добие логичката структура која е прикажана на сл. 3-17, чија блок шема е даден на сл. 3-18. Комбинационата табела 3-9 дополнително го објаснува принципот на работа на ваквиот кодер.

Бинарниот-во-октален (BIN/OCT) декодер има три влезни линии C, B, A и осум излезни линии: од Y_0 до Y_7. Логичките равенки со кои се опишува принципот на работа на декодерот се дадени покрај секоја излезна линија. За секоја влезна комбинација постои еднозначно определена излезна линија, така што кога на влезот ќе се појави одреден коден збор на излезот се активира онаа излезна линија чија бројна вредност одговара на влезниот бинарен еквивалент. Така секоја од можните 2^n влезни бинарни комбинации продуцира активирање на различна излезна линија според табелата на окталниот броен систем.

Практично реализираните декодери најчесто содржат уште една влезна линија со која се овозможува работата на декодерот. Тоа се остварува така што на секое излезно И коло му се додава уште по еден влез кој се приклучува на ваквата влезна линија за дозвола на работа.

Влезови				Излези							
$\overline{E}$	C	B	A	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
1	x	x	x	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	0	0	0	1	0
0	1	1	1	0	0	0	0	0	0	0	1

Таб. 3-9. Комбинациона табела на BIN/OCT декодер

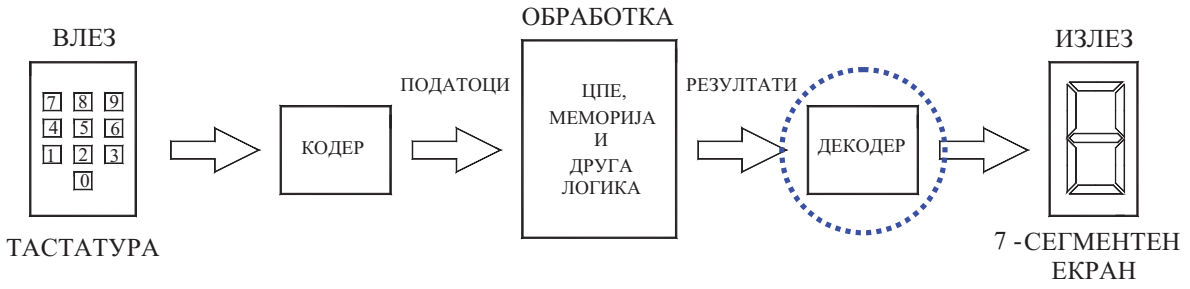
Влезот за дозвола $\overline{E}$ (анг. enable) е активен ако на него се доведе логичка 0 и во тој случај декодерот најнормално работи. Меѓутоа, ако на овој влез ($\overline{E}$) се донесе високо логичко ниво, т.е. 1, ниту една излезна линија, а со тоа ниту еден минтерм нема да биде селектиран бидејќи сите излези ќе бидат 0-и. Декодерите кои имаат влез $\overline{E}$ за овозможување на работата можат да се поврзуваат заради добивање на декодери со поголем број на влезови и излези.

Доколку има потреба излезите да бидат активни на ниско ниво, тогаш И колата ќе треба да се заменат со НИ кола, или на секој излез ќе се додаде по еден инвертор со што секој од излезните минтерми ќе биде даден во комплементарен облик. Во овој случај на ниско ниво (логичка 0) ќе се наоѓа само оној излез чија бројна вредност соодветствува на влезната бинарна комбинација, додека сите останати излези ќе бидат 1-и.

За декодерите е многу важно да се каже и тоа дека секоја логичка функција од n – променливи која е зададена во ДНФ облик како сума од минтерми може да се реализира со примена на еден n-во-2^n декодер кој ќе ги генерира минтермите и едно ИЛИ логичко коло кое ќе го формира нивниот збир. Излезните линии од декодерот одговараат на минтермите на функцијата кои ќе се искористат како влезови во ИЛИ колото на чиј излез ќе се добие бараната логичка функција. Дополнително, секоја комбинациона мрежа која има n-влезови и m-излези може да се реализира со еден n-во-2^n декодер и m ИЛИ логички кола. Како што ќе видиме подоцна, декодерите со n-во-2^n логичка структура имаат огромна примена и се составен дел при реализацијата на ROM и RAM мемориските компоненти.

3.5.4. NBCD-ВО-7 СЕГМЕНТЕН ДЕКОДЕР

Во практиката многу често се сретнува *NBCD-во-*7 *сегментен декодер* (*NBCD/7S*). Ова е затоа што најголем дел од дигиталните уреди резултатите ги покажуваат на 7-сегментен екран со светлечки диоди или со течни кристали (*LED* или *LCD*). Кажаното многу едноставно може да се илустрира со начинот на којшто калкулаторот ги прикажува резултатите добиени од извршената обработка. На неговата блок-шема од сл. 3-19 се гледа дека *NBCD* кодираните излезни податоци од процесорот мора да се декодираат во таков облик што секоја декадна цифра ќе може да се прикаже на излезниот 7-сегментен дисплеј.



Сл. 3-19. Местото на декодерот во наједноставна блок-шема на калкулатор

Ако претпоставиме дека дисплејот има заедничка анода, тогаш излезите од *NBCD/7S* декодерот треба да се активни ако се наоѓаат на ниско ниво. Функционалната табела 3-10 претставува основа за синтеза на логичката мрежа на декодерот која може да се добие со минимизација на секоја од седумте сегменти како функции: *a, b, c, d, e f, g,* кои зависат од четирите битови на NBCD кодниот збор како влезни променливи: D, C, B и A.

BCD 8421	Дек. циф.	LED екран	Екран со заедничка анода						
			a	b	c	d	e	f	g
0000	0		0	0	0	0	0	0	1
0001	1		1	0	0	1	1	1	1
0010	2		0	0	1	0	0	1	0
0011	3		0	0	0	0	1	1	0
0100	4		1	0	0	1	1	0	0
0101	5		0	1	0	0	1	0	0
0110	6		0	1	0	0	0	0	0
0111	7		0	0	0	1	1	1	1
1000	8		0	0	0	0	0	0	0
1001	9		0	0	0	0	1	0	0

Таб. 3-10. Табела на кодовите за седумсегментен екран со LED диоди и заедничка анода

Ако станува збор за поврзување со 7-сегментен дисплеј со заедничка катода тогаш излезите од *NBCD/7S* декодерот ќе треба да се активни на високо ниво на логичка 1.

3.6. МУЛТИПЛЕКСЕР И ДЕМУЛТИПЛЕКСЕР

Мултиплексерот и демултиплексерот се користат во оние уреди со кои се врши претворување на податоците од сериска во паралелна форма, и обратно.

Мултиплексерот *во дигиталната електроника има улога на преклопник со повеќе положби*. Имено, тој има поголем број на влезови за бинарните податоци, а само еден податочен излез. *Податоците можат да дојдат на било кој влез, а на излез се избира еден од нив. Кој сигнал ќе се појави на излезот зависи од тоа каква е состојбата на адресните (селекционите) влезови на мултиплексерот*. Заради начинот на работа мултиплексерите се викаат и **селектори**.

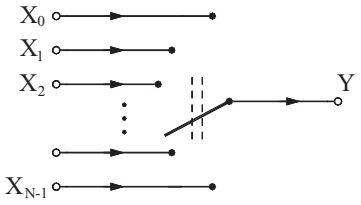
Демултиплексерот *има инверзна задача: бинарниот податок кој доаѓа на единствениот влез, треба да го проследи до еден од повеќето излези.* И во овој случај постојат адресни линии со кои се врши селекција за тоа на која излезна линија ќе се појави податокот. Заради начинот на работа демултиплексерот се вика и **дистрибутер**.

3.6.1. МУЛТИПЛЕКСЕР

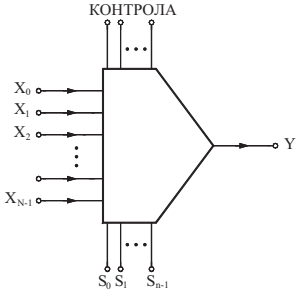
Воопштено може да се каже дека мултиплексерот е таква прекинувачка мрежа, која врши избор на една од повеќе влезни линии (N) и битот која е присутен на таа линија го пренесува на единствениот излез (Y). Покрај влезните линии за податоци, кај мултиплексерот постојат и т.н. влезни селекциони линии (n) со кои се врши избор *на конкретната влезна линија чија логичка состојба треба да се пренесе на излезот.* Односот помеѓу бројот на влезните линии за податоци (N) и бројот на линиите за селектирање (n) е даден со равенката

$$N=2^n . \qquad (3\text{-}5)$$

На овој начин, ако мултиплексерот има *n адресни линии*, тогаш со нив може да се врши избор на еден од $N=2^n$ *влезови*. На сл. 3-20 а) е прикажана еквивалентната шема на мултиплексерот како контролиран прекинувач, а на сл.3-20 б) неговиот симбол.



а) Еквивалентна шема



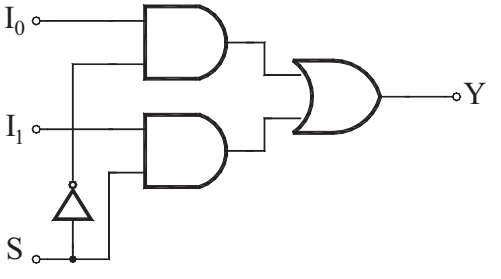
б) Логички симбол

Сл. 3-20. Мултиплексер

Наједноставната реализација на мултиплексер со примена на елементарни логички кола се однесува на 2-на-1 (двоположбен) мултиплексер. Нејзината логичка шема е дадена на сл. 3-21. Излезното ИЛИ коло проследува еден од двата сигнали, а кој сигнал ќе биде пропуштен се селектира со состојбата на влезот S кој ги контролира влезните И кола. Ако *S=0* на излезот *Y* ќе се појави сигналот од влезот I_0, додека ако *S=1,* сигналот од влезот I_1.

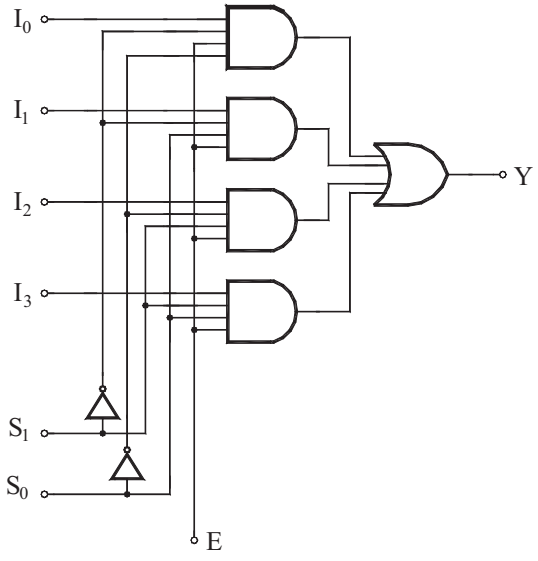
S	Y
0	I_0
1	I_1

Таб. 3.11. Функционална таблица на 2-во-1 мултиплексер



Сл. 3-21. Логичка шема на 2-во-1 мултиплексер

На сл. 3-22 е дадена логичката структура на 4-на-1 (четириположбен) мултиплексер. Тој има четири влеза за податоци, еден излез и две адресни линии. Принципот на работа може да се илустрира со табелата на вистинитост таб.3-12.



Сл. 3-22. Логичка шема на 4-во-1 мултиплексер

E	*S1*	*S0*	*Y*
0	x	x	*0*
1	0	0	*I0*
	0	1	*I1*
	1	0	*I2*
	1	1	*I3*

Таб. 3-12. Табела на вистинитост на 4-во-1 мултиплексер

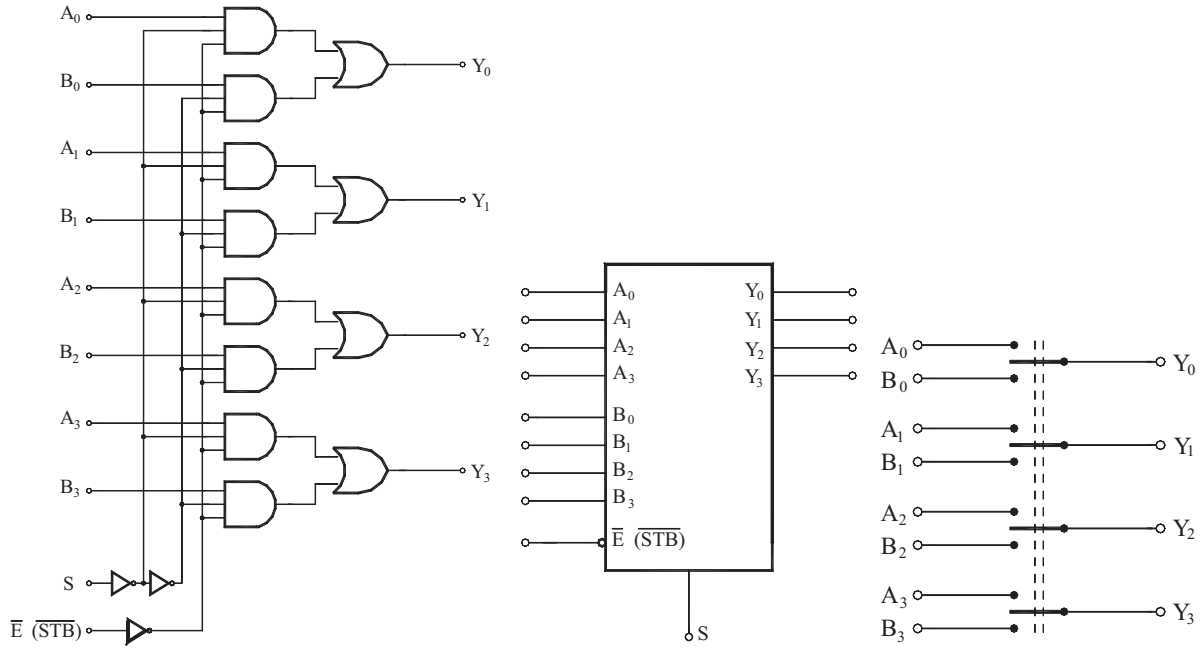
На прикажаната логичка шема може да се види присуството на уште еден влезен сигнал. Тоа е контролниот сигнал за овозможување (дозвола) на работа означен со E. Овој управувачки сигнал се јавува кај мултиплексерите кои се изведуваат како монолитни интегрирани кола. Со контролната линија практично се вклучува или исклучува интегрираното коло. Имено, мултиплексерот активно ќе работи само ако E=1 со што тој добива дозвола за работа и функционира најнормално според претходно дадената таб. 3-12. Така, неговото однесување ќе зависи од влезни комбинации доведени на селекционите влезови:

- Ако S_1S_0=00 (0_{DEC}) на излезот Y ќе се појави логичкото ниво присутно на влезот I_0;
- Ако S_1S_0=01 (1_{DEC}) на излезот Y ќе се појави логичкото ниво присутно на влезот I_1;
- Ако S_1S_0=10 (2_{DEC}) на излезот Y ќе се појави логичкото ниво присутно на влезот I_2;
- Ако S_1S_0=11 (3_{DEC}) на излезот Y ќе се појави логичкото ниво присутно на влезот I_3.

Меѓутоа, ако на управувачкиот влез E се донесе ниско ниво (E=0), излезот е секогаш нула (Y=0). Контролната линија E има важна улога и заради тоа што со неа може да се оствари поврзување на повеќе исти чипови со цел формирање на мултиплексери со поголем број на влезови.

Сите досега наведени мултиплексери вршеа избор на еден податочен влез од повеќето расположиви. Меѓутоа, покрај ваквите мултиплексери во интегрирана техника се изведуваат и мултиплексери кои вршат селекција на една влезна група податочни линии од повеќето такви групи што се јавуваат како можни влезови. Овие мултиплексери имаат онолку излези, колку што линии содржи секоја од влезните групи. Логичкиот блок-дијаграм на еден ваков мултиплексер е даден на сл. 3-23 а), симболот на сл. 3-23 б), а неговиот еквивалент како контролиран механички ррекинувач со повеќе положби на сл. 3-23 в). Оваа селекторска матрица претставува четирикратен двоположбен мултиплексер, бидејќи на излез дава една од двете влезни групи што содржат по четири информациони линии.

Колото има осум влеза, четири излеза, еден селекторски влез и еден влез за контрола. Ако влезот *S=0*, тогаш на излезите Y_0, Y_1, Y_2, Y_3 ќе се појават податоците присутни на влезовите A_0, A_1, A_2, A_3, додека кога S=1 на излезите ќе се појават B_0, B_1, B_2, B_3. Се разбира дека при ова контролниот влез означен со $\overline{E}$ или $\overline{STB}$ треба да се наоѓа на ниско ниво и мора да важи $\overline{E}=0$ или $\overline{STB}=0$.

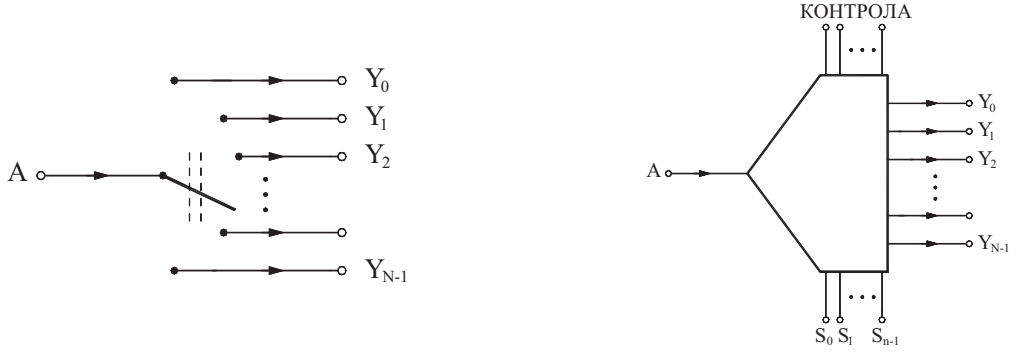


а) логичка шема б) блок шема в) механички еквивалент

Сл. 3-23. Четирикратен двоположбен мултиплексер

3.6.2. ДЕМУЛТИПЛЕКСЕР

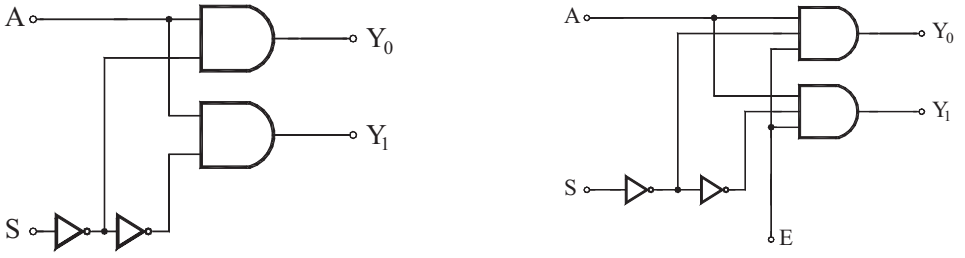
Демултиплексерот го прима податокот преку единствениот влез, и истиот го пренесува до еден од повеќето постоечки излези. Конкретниот излез што треба да ја добие влезната информација се избира со помош на линии за селекција, слично како кај мултиплексерот. И кај демултиплексерот важи равенката (3-5) со единствена разлика што сега со *n*-те адресни линии се врши избор на една од N-те *($N=2^n$) излезни, а не влезни линии.* Сликата сл. 3-24 а) го прикажува механичкиот еквивалент на демултиплексерот, а сл. 3-24 б) неговата симболичка ознака.



а) Еквивалентна шема б) Логички симбол

Сл. 3-24. Демултиплексер

Наједноставната реализација на демултиплексер е прикажана на сл. 3-25 а). Тоа е 1-на-2 (двоположбен) демултиплексер. Влезниот сигнал А ќе биде проследен на излезот Y_0, ако на селекторскиот влез постои ниско ниво, т.е. ако $S=0$. Меѓутоа, ако на линијата за селекција доведеме високо ниво ($S=1$), тогаш влезниот податок ќе се појави на излезот Y_1. Од сликата се забележува дека бројот на излезните И кола е еднаков со бројот на излези од демултиплексерот и дека влезниот податок што треба да се проследи до еден од излезите се доведува истовремено на по еден влез од секое И коло. Отворањето на конкретното И коло се контролира преку логичката состојба на селекционата линија која на влезот од едното И коло се носи во директен облик, а кај другото во комплементарен облик преку инвертор. На овој начин е остварена целта: во даден момент кога ќе се побуди влезот за селекција (S) да биде отворено само едно од двете И кола и тоа она коло за кое е наменет битот присутен на влезот А.

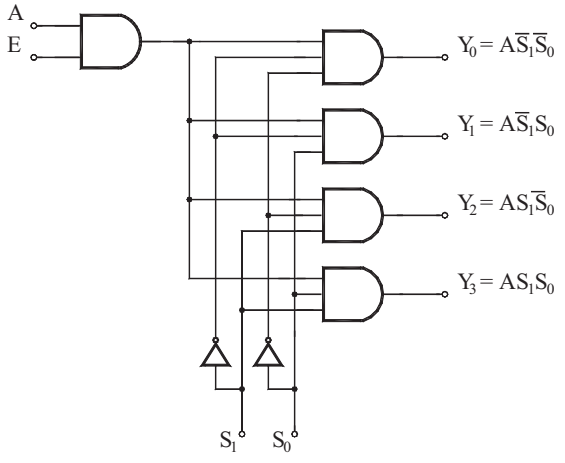


а) логичка шема б) со влез за овозможување на работата

Сл. 3-25. 1-на-2 Демултиплексер

На сл. 3-25 б) е прикажана реализација на истиот демултиплексер со влез за дозвола за (овозможување на) работа Е, активен на високо: ако Е=1 демултиплексерот функционира најнормално, но ако Е=0 сите излези се на логичка 0.

Следејќи го овој принцип, може да се изведе и логичката структура на 1-на-4 (четириположбен) демултиплексер која е прикажана на сл. 3-26. Начинот на работа на компонентата најлесно се објаснува преку нејзината табела на вистинитост 3-13.



Сл. 3-26. Логичка структура на 1-во-4 демултиплексер

E	S_0	S_1	Y_0	Y_1	Y_2	Y_3
0	x	x	0	0	0	0
1	0	0	A	0	0	0
	0	1	0	A	0	0
	1	0	0	0	A	0
	1	1	0	0	0	A

Таб. 3-13. Комбинациона табела на 1-во-4 демултиплексер

И овој демултиплексер располага со влез за овозможување на работата (Е) со таа разлика што сега контролата на излезните И кола се изведува посредно. Кај ова решение е воведено влезно И коло со два влеза: едниот за податокот А, а другиот за влезот за дозвола Е. Ако Е=1 податокот ја минува влезната И порта и се носи до излезните И кола со што е овозможена нормална работа. Но, ако влезот Е=0, тогаш излезот од влезното И коло е 0 која едновремено ги побудува сите излезни И кола со што сите излези се 0-и.

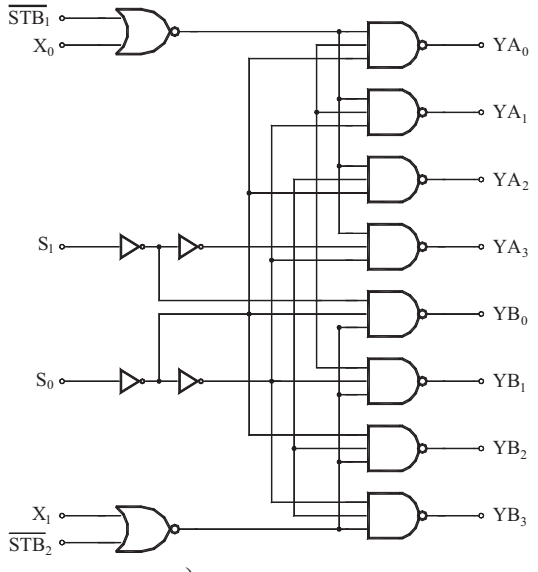
Споредувајќи го однесувањето на декодерот и демултиплексерот, може да се забележи дека демултиплексерот може да се користи како декодер ако на влезната линија на демултиплексерот се донесе фиксно ниво на логичка 1. Поконкретно, секој 1-во-2^n демултиплексер може да се користи како n-во-2^n декодер ако на влезната линија на демултиплексерот се приклучи константна вредност 1, додека неговите n линии за селекција се користат како n влезни линии на декодерот, а излезните 2^n линии се излезни линии и за декодерот.

Слична транформација на декодер во демултиплексер може да се реализира само ако декодерот има влез за овозможување на работа. Така на пр. 2-во-4 декодер може да се употреби како 1-во-4 демултиплексер. Влезот за дозвола на декодерот е влез за податокот кај демултиплексерот, двете влезни линии на декодерот се земаат како линии за селекција на демултиплексерот, додека четирите излези се исти и за двете компоненти.

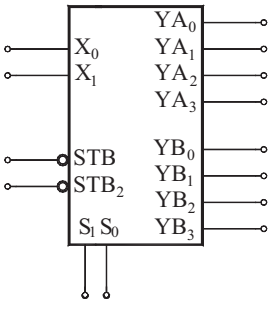
Демултиплексерите кои се изведуваат во интегрирана техника, покрај податочните и адресните линии вообичаено содржат и контролни линии со кои компонентата може да се вклучи во работа или да се поврзи со исти такви демултиплексери за да се формираат компоненти со поголем број излези.

Сите наведени демултиплексери вршеа пренос на единствениот влез до еден од повеќето расположливи излези. Меѓутоа, покрај ваквите демултиплексери во интегрирана техника се изведуваат и такви што вршат дистрибуција на една влезна група податочни линии до една од повеќето такви групи што се јавуваат како можни излезни групи. Овие демултиплексери имаат онолку влезови, колку што линии содржи секоја од поединечните излезни групи.

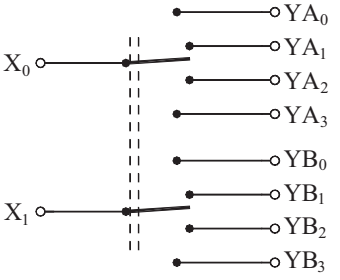
Логичкиот блок-дијаграм на еден ваков демултиплексер е даден на сл. 3-27 а), симболот на сл. 3-27 б), а неговиот механички еквивалент како прекинувач на сл. 3-27 в). Дадениот пример е двократен четириположбен демултиплексер: секоја од четирите групи излези има по две линии. Интегрираното коло има два податочни влеза, осум излези, две адресни линии и две контролни линии со што се добива поголема флексибилност. Имено, со едната контролна линија се овозможува пренос на првиот бит од податокот, а со втората на вториот бит. Примената на две контролни линии овозможува од една страна, чипот да се користи како два поединечни 1-на-4 демултиплексери, а од друга страна, ако двата контролни влеза се врзат во единствен контролен влез се добива двократен четириположбен демултиплексер. Со селекционите линии се врши избор за тоа на кои две излезни линии од четирите пара ќе се појават сигналите присутни на двете влезни линии.



а)логичка шема



б) логички симбол



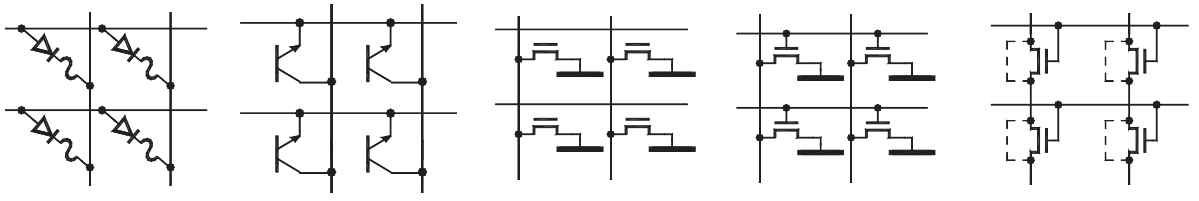
в) механички еквивалент

Сл. 3-27. Двократен четириположбен демултиплексер

III) ПРОГРАМАБИЛНИ ЛОГИЧКИ СТРУКТУРИ

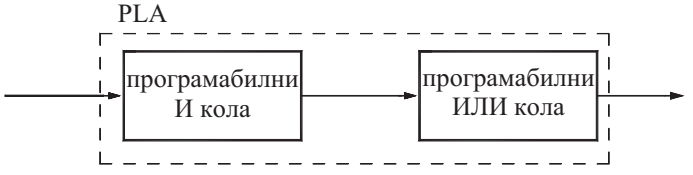
3.7. ВОВЕД И ПОДЕЛБА

Програмабилните логички структури (ПЛС, анг. Programabile Logic Devices. PLD) *се интегрирани компоненти, кои се користат за реализација на различни комбинациски мрежи по пат на програмирање.* Во процесот на ***програмирањето*** *всушност* станува збор за програмирање на врските на мрежата, поточно на воспоставување или прекинување на поврзувањето во внатрешноста на интегрираната компонента помеѓу логичките кола, кои влегуваат во нејзин состав. Во принцип овие мрежи имаат матрична структура слична на сите што до сега ги анализиравме, но во две нивоа. Разликата е и во тоа што во непрограмиран случај ПЛС-и имаат вградени прекинувачки елементи (диоди или биполарни или униполарни транзистори) како спојни елементи со осигурувач на пресеците помеѓу вертикалните и хоризонталните водови на мрежата. На следната слика (сл. 3-28) се прикажани неколку различни типови на прекинувачки нискоомски (куси врски) или високоомски (прекинати кола) елементи со кои се остваруваат поврзувањата.

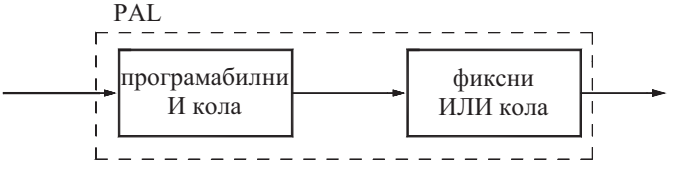


Сл. 3-28. Матрични структури со спојни елементи со диоди и биполарни транзистори

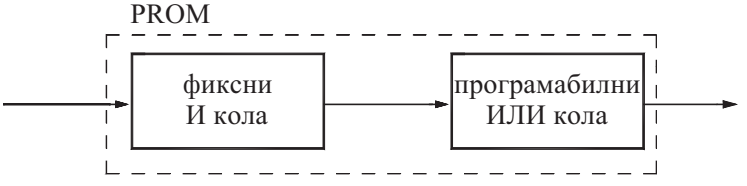
На сл. 3-29, 30 и 31 се означени основните структури на програмабилните компоненти. Имено, *секоја структура содржи две сериски врзани мрежи: една е со И логички кола, а втората со ИЛИ порти*. Според ова, за разлика од претходно споменатите едностепени матрични структури, сега станува збор за *двостепена матрична структура за изведување на логичките функции со И-ИЛИ мрежа во две нивоа.*



Сл. 3-29. Блок-шема на PLA програмабилна компонента



Сл. 3-30. Блок-шема на PAL програмабилна компонента



Сл. 3-31. Блок-шема на PROM програмабилна компонента

Првите две програмабилни структури се однесуваат на интегрираните компоненти чија намена е реализација на сложени логички мрежи со повеќе влезови и излези. Кај компонентите изведени според сл. 3-29 е овозможено *програмирање како на И, така и на ИЛИ колата*. Оваа структура на логички мрежи позната е под името *PLA (анг. Programabile Logic Array)*, што значи **програмабилна логичка матрица**. Од друга страна, концепцијата на интегрираните компоненети изведени според сл. 3-30 *овозможува програмирање само на И колата, додека ИЛИ колата се изведени како фиксни*. Оваа структура е позната според скратениот назив PAL (Programabile Array Logic), што значи **програмабилна матрична логика**.

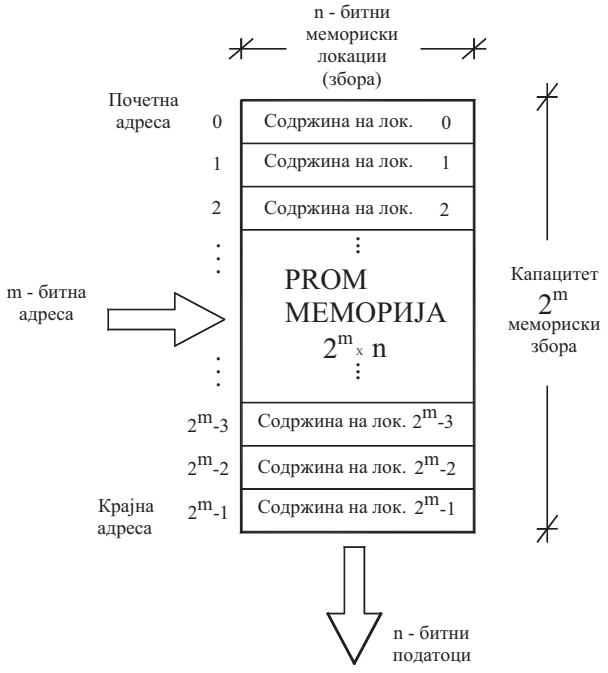
На крај, сл. 3-31 се однесува на интегрираните компоненти кај кои *програмирањето се врши само на ИЛИ колата, додека И колата се изведени фиксно*. Овие структури се познати како **програмабилни мемории што можат само да се читаат или накратко** како *PROM (Programabile Read Only Memory)*.

Програмабилните логички структури во принцип имаат универзален карактер, со одредени предности и недостатоци во поедини специфични области на примена. За разлика од *PLA* и *PAL* матриците со кои се реализираат сложени логички функции, *PROM* имаат посебна намена, а тоа е запамтување на информации кои не треба да се менуваат. Во продолжение фокусот ќе го ставиме токму врз работата на *PROM* мемориите.

3.8. PROM МЕМОРИЈА

PROM -от претставува *дигитална компонента изведена во интегрирана техника која* во практиката најчесто се применува како посебен тип на полупроводничка меморија. Бидејќи нејзината содржина во процесот на работа може само да се чита, таа спаѓа во групата на т.н. недеструктивни мемории. Запамтената содржина во PROM-от не се губи дури и по престанок на напојувањето на уредот каде е вградена, заради што истата се користи при иницијализација на компјутерските системи. Така на пр. во PROM-от се сместува системскиот софтвер (BIOS-от) на компјутерот кој ја тестира исправноста на компонентите што влегуваат во негов состав, како на пр. графичката картичка, дискот, и сл. и потоа го стартува неговиот оперативен систем, на пр. Windows-от.

Имајќи ја во предвид блок шемата на PROM-от од сл. 3-31, забележуваме дека на влезот постои И матрица со веќе формирани фиксни врски. Тоа е декодер со m-влезови и 2^m интерни излези. Овие излези може да се користат како влезови за излезната матрица која содржи m ИЛИ логички кола. Секој излез од ИЛИ колото може да претставува произволна функција од m-те влезни логички променливи. Според ова, од организациски аспект, на PROM-от може да се гледа како на мемориска компонента која содржи $N=2^m$ мемориски зборови секој со непроменлива должина од n-бита која обично се изразува во бајти *1[B]=8[b]*. Секој од запомнатите $N=2^m$ зборови претставува одредена комбинација од n-битови согласно потребите на корисникот. Во меморијата постои соодветна локација за сместување на секој збор, што значи дека *меморијата може да се смета како едно конечно и уредено множество со голем број мемориски локации* според сл. 3-32. Секоја локација е формирана од одреден број прекинувачки елементи. Бројот на прекинувачките елементи е еднаков со должината на мемориските збор. Зборот, т.е. информацијата, која е сместена во било која локација ја претставува нејзината *содржина*. Пристапот до секоја од локациите оди со задавање (специфицирање) на број кој еднозначно ја определува соодветната локација. Овој број се вика *адреса на локацијата*. Ако е позната адресата на мемориската локација, практично е дефинирана нејзината местоположба во меморијата. Така, пристапот до било која мемориска локација на PROM -от е директен со задавање на адресата заради што времето потребно за пристап до секој запомнет податок е исто.



Сл. 3-32. Организациска структура на PROM мемориска програмабилна компонента

Во врска со претходното, како *капацитет на* PROM се дефинира вкупниот број адреси, т.е. вкупниот број зборови којшто може да се запомни во PROM –от помножен со должината на меморискиот збор. Бидејќи PROM-от има голем број локации, *капацитетот* вообичаено се изразува во килобајти или во мегабајти. $1[MB] = 2^{20}[B] = 2^{10}[KB] = 1024[KB]$.

Кај типичната PROM мемориска компонента сите врски се воспоставени заради што неговата содржина е пополнета со сите 1-и. Во процесот на програмирање се користи посебен уред т.н. *PROM програматор*. Со него се прегоруваат само прекинувачките елементи на оние места каде треба да се запишат 0-и и тоа со пуштање на напонски импулси со поголема вредност од напонот на кој ќе се приклучи PROM-от при неговата вообичаена работа во уредот за кој е наменет.

i	A_2	A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	1	0	0	0
1	0	0	1	0	0	0	1
2	0	1	0	0	0	1	1
3	0	1	1	0	0	1	0
4	1	0	0	0	1	1	0
5	1	0	1	0	1	1	1
6	1	1	0	0	1	0	1
7	1	1	1	1	1	0	0

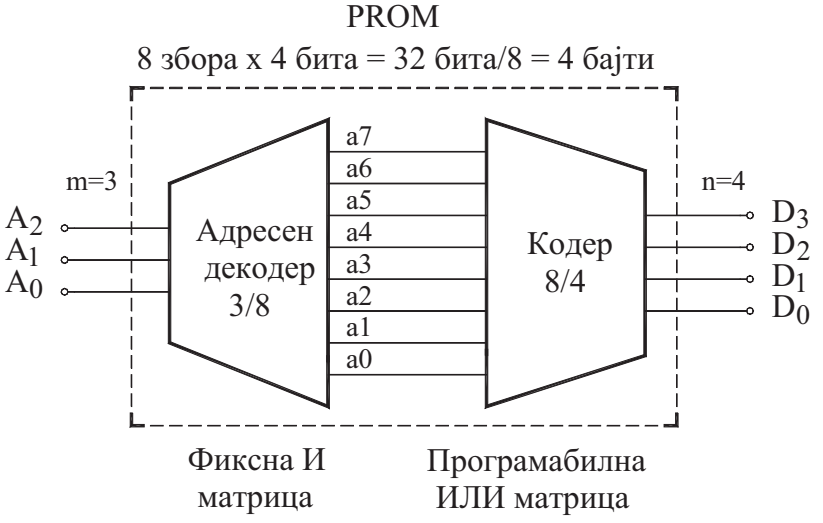
Таб. 3-14. Осум четирибитни податоци кои треба да се запомнат во PROM

За полесно разбирање на однесувањето на PROM-меморијата ќе го разгледаме следниот пример. Станува збор за трајно зачувување на осум четирибитни податоци во PROM-мемориска компонента според таб. 3-14.

Од неа се гледа дека содржина на првата локација со адреса 000 треба да биде податокот 1000, во втората локација чија адреса е 001 е сместен податокот 0001, итн. сé до последната, осмата мемориска локација со адреса 111 чија содржина е 1100.

Бидејќи треба да се меморираат N=8 податоци, ќе избереме PROM-меморија со соодветен број на адресни линии m за да биде исполнет условот $8=2^m$, од каде добиваме дека m=3. Од друга страна, секој податок што ќе биде внесен во меморискиот чип има должина од 4 бита, заради што е јасно дека бројот на излезни податочни линии n ќе биде 4 (n=4). Всушност, секој податок ја претставува содржината на секоја од осумте адресирани мемориски локации. На сл. 3-33 е прикажана блок-шемата на PROM компонента која ќе се примени за решавање на дадениот пример.

Адресниот декодер практично се реализира со фиксна матрична И структура, додека внесувањето на информациите оди по пат на програмирање на ИЛИ матрица која фактички претставува кодер со онолку влезови колку што има излези од адресниот декодер, а излези колку што е должината на меморискиот збор.



Сл. 3-33. Блок-шема на *PROM* компонента

Начинот на имплементација на табелата е прикажан со логичката шема на оваа PROM компонента претставена на сл. 3-34. Имено, секој бит од податокот, т.е. меморискиот збор D_i ќе биде функција од трите влезни променливи A_2, A_1 и A_0, а може да се добие со програмирање за тоа кои од осумте излези од адресниот декодер: $a_0, a_1, \ldots, a_7$ кои ќе бидат влезови во секое ИЛИ коло соодветно на дадената кодна табела. Во врска со ова, излезните битови D_3, D_2, D_1 и D_0 ќе се добијат од следниве СДНФ форми:

$$D_3 = a_0 + a_7 = A_2 A_1 A_0 + \overline{A_2}\,\overline{A_1}\,\overline{A_0}$$

$$D_2 = a_4 + a_5 + a_6 + a_7 = \ldots$$

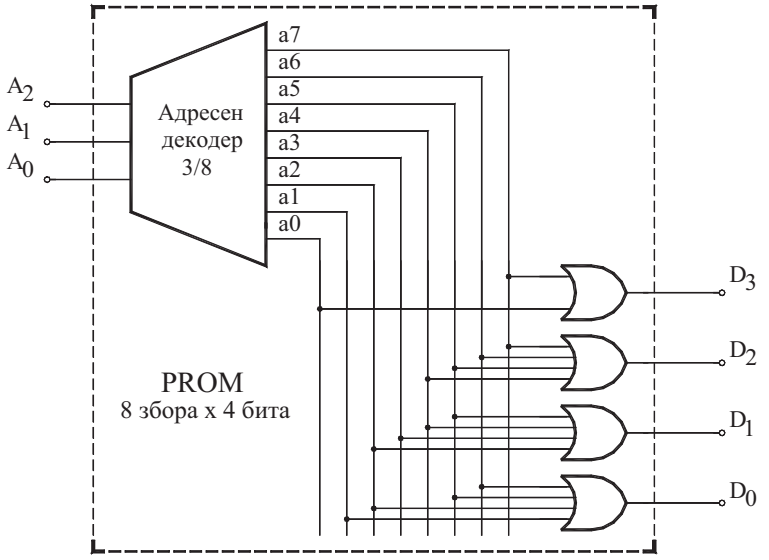
$$D_1 = a_2 + a_3 + a_4 + a_5 = \ldots$$

$$D_0 = a_1 + a_2 + a_5 + a_6 = \ldots$$

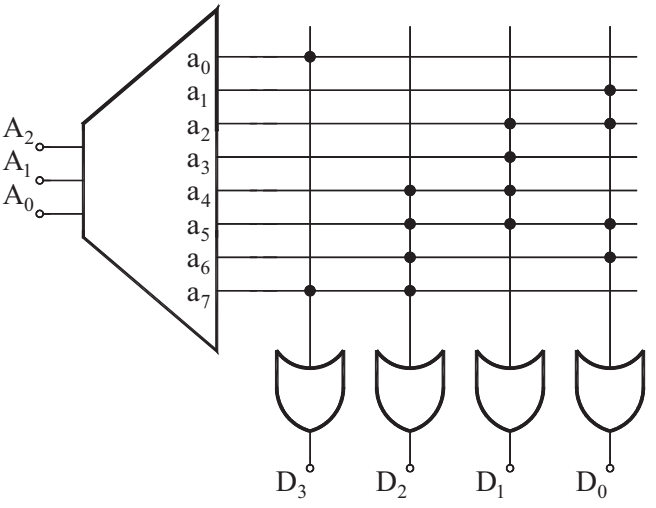
Логичките равенки произлегуваат од самата табела. Од неа се гледа дека за да се добие излезот D_3, треба да биде активна или нултата или седмата адресна линија (a_0 или a_7) заради што само со нив се формираат врски како влезови на првото ИЛИ коло. Останатите шест влезови не треба да бидат поврзани. Слично, за да се добие D_2 се програмираат врските само со a_4, a_5, a_6 и a_7, за D_1 се остварува поврзување само со a_2, a_3, a_4 и a_5, додека за да се добие D_0 на соодветното ИЛИ коло се поврзуваат само влезовите a_1, a_2, a_5 и a_6.

Бидејќи производувачот на компонентата го има направено адресниот декодер во склоп на самиот чип, корисникот сам ги внесува содржините во локациите со помош на програматорот. Преку адресните влезови се задава адресата на мемориската локација, додека преку податочните линии во специфицираната мемориска локација се внесува соодветниот мемориски збор како нејзина содржина по пат на воспоставување или елиминирање на споевите помеѓу излезите од адресниот декодер и влезовите на секое ИЛИ коло од излезната матрица.

Заради поголема прегледност, поврзувањето од сл. 3-34 се прикажува како на сл. 3-35. На оваа слика остварените врски се означени со ставање на спојна точка во пресекот на соодветните излезни линии од адресниот декодер и влезовите во ИЛИ колата.



Сл. 3-34. Логички дијаграм на *PROM* меморија со капацитет 8 збора по 4 бита



Сл. 3-35. Логички дијаграм на *PROM* меморија со капацитет 8 збора по 4 бита

По завршувањето на програмирањето, т.е. по внесувањето на потребната содржина во PROM-от, тој може да се употребува во нормален режим на работа така што се внесува во уредот за кој е наменет.

Бидејќи содржината на PROM-от може само да се чита тој спаѓа во групата на т.н. програмабилни *ROM мемории (анг. Programmable Read Only Memory)*. Слаба страна на PROM мемориските чипови е таа што тие може да се програмираат само еднаш и потоа внесената содржина повеќе не може да се менува. Во праксата се сретнуваат и EPROM мемориски компоненти кои можат повеќе пати да се програмираат, а со тоа и повеќе пати да им се менува нивната содржина (анг. *Erasable Programmable ROM*). Бришењето на старата содржина на EPROM-ите се остварува со експозиција на ултра-виолетови (UV) зраци заради што EPROM-овите се сретнуваат и под името UVEPROM. За EEPROM или E^2PROM мемориските чипови (анг. *Electrically Erasable Programmable ROM*) е исто така карактеристична можноста за повеќекратно запишување, но во овој случај бришењето се врши со електрични импулси. Последниве неколку години во праксата нашироко се применуваат и т.н. Flash мемориски чипови како посебен вид на EEPROM мемории кои се користат кај USB Flash компонентите и мемориските картички.

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

3-1. Кои се главни карактеристики на комбинациските мрежи?

3-2. Кои операции се извршуваат со комбинациските мрежи?

3-3. Која е улогата на бинарниот собирач?

3-4. Која е улогата на полусобирачот?

3-5. Нацртај ја табелата на вистинитост, логичката шема и логичкиот симбол на полусобирачот.

3-6. Која е улогата на целосниот собирач?

3-7. Нацртај ја табелата на вистинитост, логичката шема и логичкиот симбол на целосниот собирач.

3-8. Нацртај логичка шема на паралелен собирач за два двобитни броеви користејќи целосни собирачи. За кои влезни комбинации, т.е. вредности на влезните броеви, излезната линија за префрлување ќе стане активна (C+=1) и зошто?

3-9. Која е улогата на компараторот?

3-10. Нацртај ја табелата на вистинитост и логичката шема на еднобитен компаратор.

3-11. (*) Проектирај дигитален (бинарен) компаратор за два двобитни броеви: $A=A_1A_0$ и $B=B_1B_0$ кои се појавуваат на неговите влезни приклучоци. Компонентата треба да има

а) три излези Y_A, Y_B и Y_0 согласно таб. 3-4;

б) два излези G и L чии логички состојби зависат од односот помеѓу дава броеви што се споредуваат при што:

- Ако A > B тогаш G = 1 и L = 0;
- Ако A < B тогаш G = 0 и L = 1;
- Ако A = B тогаш G = 0 и L = 0.

3-12. Која е улогата на колото за (единечно) комплементирање?

3-13. Нацртај ја логичката шема на (единечен) комплементер за двобитни броеви.

3-14. (*) Применувајќи паралелно коло за собирање на двобитни броја и коло за единечно комплементирање на двобитни броеви, нацртај логичка шема на коло што ќе извршува двоен комплемент на двобитни броеви.

3-15. Нацртај ја логичката шема на коло за собирање/одземање на два двобитни броеви.

3-16. Какви комбинациски мрежи се прекинувачките матрици?

3-17. Објасни каква функција врши кодерот!

3-18. Нацртај табела на вистинитост на *DEC/NBCD* кодер. За која влезна комбинација битот (а) *D;* (б) *C*; (в) *B*; (г) *A* од *NBCD* кодниот збор има вредност 1.

3-19. Нацртај реализација на *DEC/NBCD* кодер како матрична еднонивовска ИЛИ структура чиишто влезови се активни на високо ниво, како и нејзиниот логички симбол.

3-20. (*) Нацртај реализација на *DEC/NBCD* кодер со влезови што се активни на ниско ниво применувајќи НИ логички кола, како и неговиот логички симбол.

3-21. Нацртај реализација на OCT/*BIN* кодер како матрична еднонивовска ИЛИ структура чиишто влезови се активни на високо ниво, како и нејзиниот логички симбол. (*) Додај влез за дозвола на работа!

3-22. Која е разликата помеѓу стандардниот кодер и кодерот со приоритет?

3-23. Ако претпоставиме дека на располагање имаш два кодери: еден OCT/BIN без приоритет и еден OCT/BIN со приоритет и дека и на двата едновремено ги притиснете тастерите [2], [3] и [4] одговори која е состојбата на излезите од двата кодери. Дали меѓусебно се разликуваат? Образложете!

3-24. (*) Нацртај реализација на *HEX/BIN* кодер.

3-25. Ако на влезот од кодерот доаѓаат *N* различни симболи, кој услов треба да го исполнува бројот на излезите n, т.е. должината на кодниот збор, со кој може да се кодираат сите влезни симболи?

3-26. Која е улогата на декодерот?

3-27. Нацртај табела на вистинитост за *NBCD* /*DEC* декодер чии излези се активни на (а) ниско (б) високо ниво.

3-28. Нацртај реализација на *NBCD*/*DEC* декодер чиишто излези се активни на високо ниво, како матрична еднонивовска И мрежа со примена на И логички кола. Каква модификација треба да се изврши на мрежата, за излезите да бидат активни на ниско ниво.

3-29. За која влезна комбинација излезот (а) 0 (б) 1 (в) 2 (г) 3 (д) 4 (ѓ) 5 (е) 6 (ж) 7 (з) 8 (ц) 9 од *NBCD* /*DEC* декодерот ќе биде активен.

3-30. Нацртај табела на вистинитост за *NBCD* /*7S* декодер. (*) Проектирај ја оваа логичка мрежа со претпоставка дека на располагање стојат сите потребни логички кола со произволен број на влезови.

3-31. Кој услов треба да го задоволува максималниот број на излези *N,* ако на влез во декодерот доаѓаат кодни зборови со должина од n бита?

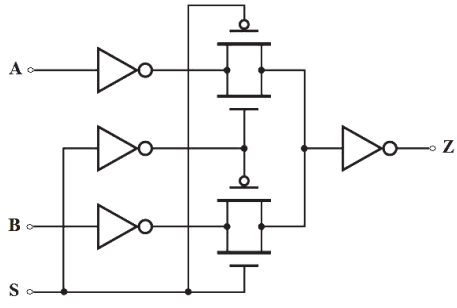
3-32. (*) Реализирај комбинациона мрежа на 4/16 (BIN/HEX) декодер со примена на а) И б) НИ логички кола и влез за дозвола на работа E кој треба да биде активен на 1) ниско 2) високо ниво.

3-33. (*) Аки имаш на располагање два 2-во-4 декодери со влез за дозвола E изврши нивно поврзување за да добиеш еден 3-во-8 декодер.

3-34. Која е функцијата на мултиплексерот?

3-35. Да се нацрта комбинационата табела, логичката шема и логичкиот симбол на (а) 2-во-1 (б) 4-во-1 (в) 8-во-1 мултиплексер. (*) со влез за овозможување на работата активен на 1) ниско 2) високо ниво.

3-36. (*) На следната слика е прикажана логичка мрежа која покрај инвертори содржи и трансмисиони порти. Формирај ја и пополни ја таблицата на вистинитост за мрежата и објасни го нејзиното однесување, симболичка ознака и евентуалната примена.



Слика за задача 3-36.

3-37. Нацртај ја логичката шема на четирикратен-двоположбен (4-во-2) мултиплексер.

3-38. (*) Нацртај ја логичката шема на (а) двократен - двоположбен (2-во-2) (б) двократен-четириположбен (2-во-4) мултиплексер.

3-39. Каква функција врши демултиплексерот?

3-40. Да се нацрта комбинациската табела, логичката шема и логичкиот симбол на (а) 1-во-2 (б) 1-во-4 (в) 1-во-8 демултиплексер. (*) со влез за овозможување на работата активен на 1) ниско 2) високо ниво.

3-41. Ако имаш на располагање 1-во-4 демултиплексерот изврши негова трансформација како во 2-во-4 декодер.

3-42. Ако имаш на располагање 2-во-4 декодер со посебен влез за дозвола на работа, изврши негова трансформација во 1-во-4 демултиплексер.

3-43. Нацртај логичка шема на двократен-четириположбен демултиплексер (2-во-4) со посебен влез за селекција на компонентата активен на ниско ниво.

3-44. Нацртај логичка шема на (а) двократен-двоположбен (2- во-2) (б) четирикратен-двоположбен (4-во-2) демултиплексер со посебен влез за селекција на компонентата активен на ниско ниво.

3-45. Што претставуваат програмабилните логички структури?

3-46. Како се остварува програмирањето на програмабилните логички структури?

3-47. Прикажи по два примера на (а) нискоомски (б) високоомски електронски елементи што се користат за остварување на споевите кај програмабилните логички структури.

3-48. Каква структура е *PLA* и која е нејзината главна примена?

3-49. Нацртај блок-шема на PAL компонента.

3-50. Каква структура е *PAL* и која е нејзината главна примена?

3-51. Нацртај блок-шема на *PROM* компонента.

3-52. Каква структура е *PPOM* и која е нејзината главна примена?

3-53. Како е организирана меморијата?

3-54. Во каков облик се запамтува и каде се сместува секоја информација во меморијата?

3-55. Што претставува адреса на мемориска локација?

3-56. Што е капацитет на меморијата и како се изразува?

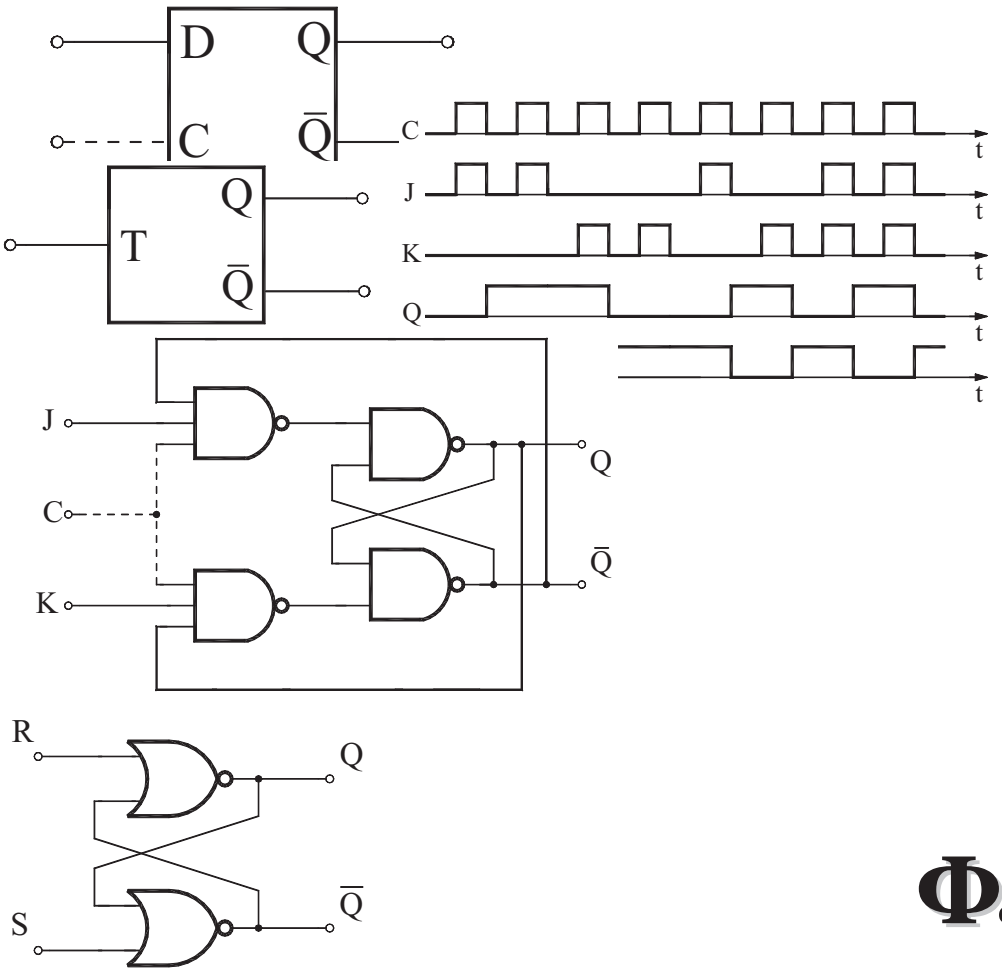
3-57. Како се врши внесување на нова содржина во *PROM* ?

3-58. Кои типови *PROM* мемории постојат и по што се разликуваат меѓу себе?

3-59. Нацртај блок-шема на *PROM* (а) 16x4 (16 збора по 4 b) (б) 128x8 (128 збора по 8 b).

3-60. Даден е *PROM* со организација 64 K x 16. (а) колку адресни влезни линии (б) колку податочни излезни линии, има оваа мемориска компонента (в) колкав е капацитетот на *PROM* -от изразен во (а) зборови (б) бајти.

3-61.(*) Нацртај комбинациона таблица на *PROM* со соодветен капацитет и должина на мемориските зборови, со кој во локацијата со најмала адреса 0000 се сместува најголемиот четирибитен број 1111, итн. со зголемување на адресите се намалуваат броевите што во нив се сместуваат, па така во последната мемориска локација што има адреса 1111, се сместува податокот 0000. (**) За секој бит од излезниот податок напиши ја соодветната совршена нормална форма во функција од (а) излезите (б) влезовите од адресниот декодер.



4.
ФЛИП - - ФЛОПОВИ

По изучувањето на оваа тематска целина

- ќе го разберете принципот на работа на флип-флоповите како елементарни секвенцијални кола со стандардна и master-slave конфигурација и тоа:
  - SR флип-флоп;
  - JK флип-флоп,
  - T флип-флоп:
  - D флип-флоп;
- ќе умеете да демонстрирате различни трансформации на флип-флоповите;
- ќе се запознаете со примената на флип-флоповите во реализирањето на посложени секвенцијални компоненти:
  - коло за заклучување;
  - RAM мемориска ќелија.
- ќе ја сфатите примената на флип-флоповите во градбата на посложените секвенцијални мрежи;

4.1. ВОВЕД И ОСНОВНИ ПОИМИ

Бистабилниот мултивибратор кој популарно се нарекува **флип-флоп** претставува основен елемент во дигиталните системи кој овозможува меморирање на податоците. Тој претставува елементарна мемориска ќелија, бидејќи може да запомни еднобитен податок, односно најмало количество на информација. Способноста за помнење е резултат на фактот што флип-флопот може да се најде во една од две стабилни состојби.

Флип-флопот спаѓа во групата на регенеративни кола, затоа што при неговата реализација мора да постои позитивна повратна врска. Постојат различни типови флип-флопови, зависно од тоа каква им е изведбата и начинот на функционирање. Во дискретната техника, флип-флопот се реализира со спрегнување на два засилувачки степени, наједноставно со два транзистори, потоа со погодно врзување на две логички кола, при што најчесто се користат две меѓусебно поврзани НИ или НИЛИ кола, или во интегрирана техника кога се изведува како посебна дигитална компонента.

Секој флип-флоп има два излеза кои се комплементарни еден на друг, и еден или повеќе влезови. Со меѓународна конвенција е усвоено состојбата на флип-флопот да се изразува преку вредноста на излезот означен со Q (номиналниот, директниот излез), а со тоа практично се дефинира и вредноста на излезот означен со $\overline{Q}$ (комплементарниот излез). За флип-флопот се вели дека е **поставен** или **сетиран** ако Q=1 ($\overline{Q}$=0), односно **избришан** или **ресетиран**, тогаш кога Q=1 ($\overline{Q}$=0), што може да се види и од таб. 4-1.

Состојба на флип-флопот	Излези	
	Q	$\overline{Q}$
Сетиран (Поставен)	1	0
Ресетиран (Избришан)	0	1

Таб. 4-1. Состојби на флип-флопот

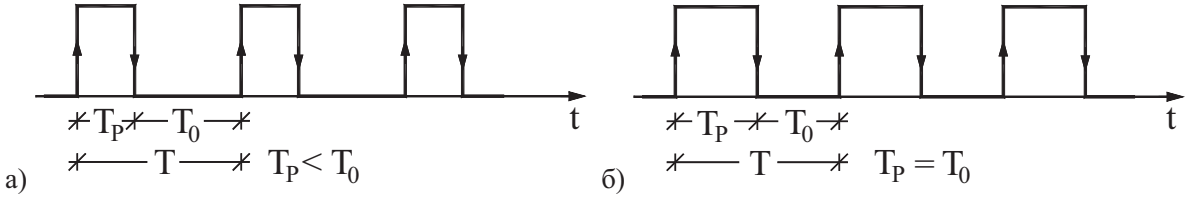
Преку влезовите се управува неговата работа и/или се доведуваат податоците што ќе бидат запомнети во него. Состојбата на излезите $Q, \overline{Q}$ теоретски може да се задржи бесконечно долго, а практично сè додека на некој од влезовите не се донесе побуда која ќе има ефективно дејство врз излезите, т.е. побуда што ќе ја промени состојбата на излезите. Во зависност од тоа дали влезовите имаат директно (непосредно) или индиректно (посредно) влијание врз излезите, постојат асинхрони флип-флопови и синхрони флип-флопови.

Кај **асинхроните** флип-флопови состојбата на излезите зависи само од нивото на побудните сигнали што се јавуваат на информационите (податочните) влезови. Состојбите на овие влезови детерминираат дали и како ќе се променат излезните логички нивоа.

Кај **синхроните** флип-флопови покрај податочните влезови постои и еден посебен влез за **такт** (*clock*) сигнал кој може да биде означен со: CLK, CK, CL, C, C_P, или T_1. Тој претставува напонски правоаголен импулсен облик, или многу почесто заради практични потреби, квадратен бранов облик.

Тактот има точно определени временски интервали на високото ниво, т.е. 1-та, импулсот T_p, и на ниското ниво, т.е. 0-та, паузата T_0, а со тоа е дефиниран и неговиот период T, $T=T_p+T_0$. Станува збор за сигнал за временско водење и усогласување (синхронизирање) на работата на флип-флопот, кој определува во кој момент (кога) нешто ќе се случи, па затоа овие флип-флопови се викаат и тактирани. Имено, состојбата на флип-флопот може да се промени во зависност од комбинацијата на влезните сигнали, но само во моментот кога на влезот за такт-сигналот ќе се појави активно (прекинувачко) ниво. Активно ниво на тактот е она ниво за чие времетраење другите влезови можат да имаат ефективно дејство врз излезите од флип-флопот (да можат да предизвикаат „*нешто да се случи на излезите*"). Во општ случај, активно ниво може да биде или нивото на 0, или нивото на 1, но вообичаено е за активно ниво да се зема високото ниво. Кај оние изведби на флип-флопови кај кои активно ниво е ниското ниво, тоа посебно ќе биде нагласено.

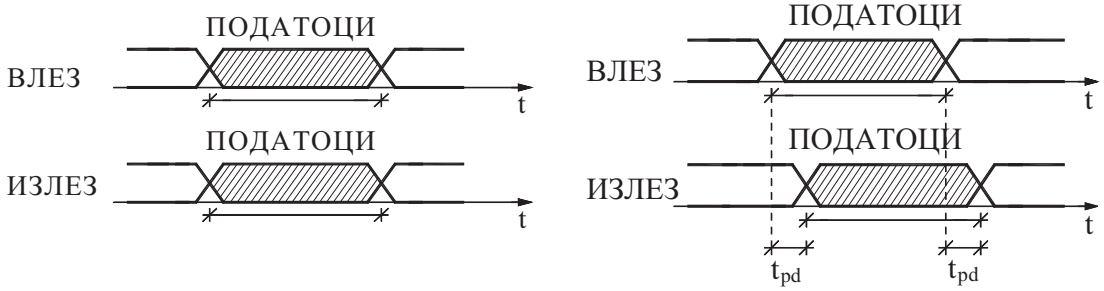
На сл. 4-1 а), б) се прикажани временски дијаграми на правоаголен и квадратен такт-сигнал. На сликата се означени двата најважни делови на тактот: предниот (позитивниот, растечкиот) раб, и задниот (негативниот, опаѓачкиот) раб. Ова е важно да се забележи, бидејќи најголем број на флип-флопови што се во употреба се активираат со појавата на растечкиот или со опаѓачкиот раб на такт-сигналот. **Активниот раб** *се вика и прекинувачки или ефективен раб (срп. окинувачки, анг. triggering), затоа што само тогаш може да настане промена на излезите од флип-флопот***. Циклусот** (**интервалот**) на такт-сигналот го претставува времето од едниот активен раб до другиот активен раб, а тоа време практично е еднакво со периодот на такт-сигналот T.



Сл. 4-1. Временски дијаграми на правоаголен и квадратен такт-сигнал

Без оглед на тоа со кој раб се активира флип-флопот, ќе напоменеме дека излезот од флип-флопот реагира на побудата по определен временски период, кој е многу краток, но сепак е присутен како последица од доцнењето на сигналот кога тој минува низ флип-флопот. Ова време се вика време на доцнење и се означува со t_{pd}, t_d, или Δt. Тоа варира во зависност од технологијата во која е направен флип-флопот, а обично изнесува од неколку *[ns]* до неколку десетици *[ns]*. Заради поедноставно разбирање на принципот на работа на флип-флоповите, при цртањето на временските дијаграми, ќе разгледуваме идеални случаи, односно ќе го занемаруваме времето на доцнење ($t_{pd}=0$), така што излезот ќе се јавува без задоцнување, истовремено, во однос на влезот. Реални случаи, односно временска разлика меѓу излезот и влезот ($t_{pd}>0$), ќе опишуваме само тогаш кога ќе сакаме да ја потенцираме појавата на одредени проблеми кои можат да настанат при самата практична примена на флип-флопот во работата.

На сл. 4-2 а) е прикажана појавата на влезните сигнали, како и одзивот на нив, односно појавата на излезните сигнали, за еден пример на флип-флоп кај кого е занемарено доцнењето. На сл. 4-2 б) е прикажан еден реален случај каде постои одредено доцнење од побудата до одзивот.



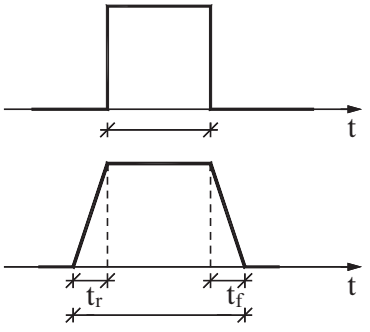
а) иделен случај без доцнење влез - излез б) реален случај со доцнење влез - излез

Сл. 4-2. Временски дијаграми на влезни и и излезни сигнали кај флип-флопот

Излезот од флип-флопот, а со тоа и неговиот начин на работа, може да се прикаже аналитички, преку одредена логичка равенка која е специфична за секој флип-флоп, и затоа се вика *карактеристична равенка* или *фукција на премин*. Освен овој алгебарски начин, постои и табеларен приказ, со примена на два типа табели: едната е т.н. *карактеристична табела* или *табела на премин и излез*, а другата е *табела на побуда* или *екситација*. Табелата на премин се употребува при анализата на секвенцијалните мрежи, додека табелата на побуда при реализацијата на секвенцијалните мрежи. Веќе е истакнато дека за илустрација на работата исто така може да се користи и графичката претстава со временските дијаграми на напоните на влезовите и излезите од флип-флопот.

Излезот на флип-флопот е функција од сигналите доведени на неговите влезови, од нивото на тактот ако флип-флопот е тактиран, но и од претходната состојба на излезите од флип-флопот. Ова значи дека се јавува зависност од временскиот редослед, т.е. од секвенцата на логичките состојби низ кои поминал флип-флопот. Според ова, флип-флопот претставува и елементарна секвенцијална компонента. Бидејќи при прикажувањето на излезот од флип-флопот ќе фигурираат логички состојби во различни временски интервали кои следат еден по друг, се воведуваат ознаки за тоа дали состојбата на колото се однесува на претходниот (сегашниот) временски период: t или T, или за следниот временски период t+T, t+1, t_n+1, или t^+. Во овој контекст за состојбата на излезот од флип-флопот се користат следниве ознаки $Q(t)$ и $Q(t+T)$, или $Q(t)$ и $Q(t+1)$, или $Q(t_n)$ и $Q(t_n+1)$, или Q_n и Q_{n+1}, или $Q(t)$ и $Q(t^+)$, или наједноставно Q и Q^+.

Во понатамошното излагање ќе претпоставиме дека напонските сигнали имаат идеален правоаголен облик според сл. 4-3 а) занемарувајќи ги времињата што се потребни сигналот да го достигне високото или ниското ниво, кои се јавуваат во случај на реален импулс кој е даден на сл. 4-3 б).



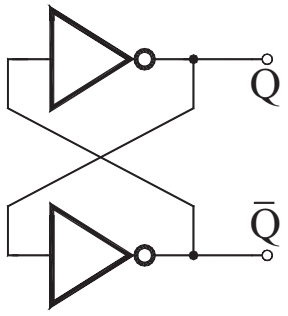
а) идеален импулс со $t_r = t_f = 0$

б) реален импулс со $t_r \neq 0$ и $t_f \neq 0$.

Сл. 4-3. Напонски импулс на влезот или излезот од флип-флопот

4.2. SR ФЛИП-ФЛОП

Во принцип, флип-флоп може да се реализира со вкрстена врска на две инверторски кола така што излезот од едното коло се врзува на влезот од второто коло, а излезот од второто оди на првото, како што се гледа на сл. 4-4.



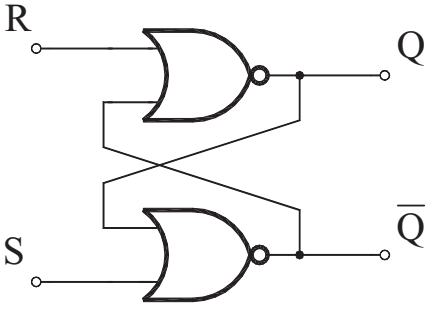
Сл. 4-4. Вкрстена врска на две инверторски кола

Ваквото поврзување овозможува излезите од флип-флопот да се стабилни и меѓусебно комплементарни. Со вкрстувањето се остварува позитивна повратна врска која е потребна за создавање на регенеративен процес со кој времето потребно за премин од едната во другата стабилна состојба е занемарливо мало, практично еднакво на нула. Меѓутоа вака реализираниот флип-флоп ќе има случајна состојба која не може да се одреди и не е дефинирана: таа ќе биде или Q=1 ($\overline{Q}$= 0), или Q=0 ($\overline{Q}$ = 1). Покрај ова, не постои можност за управување со состојбата на излезите бидејќи не постојат влезови.

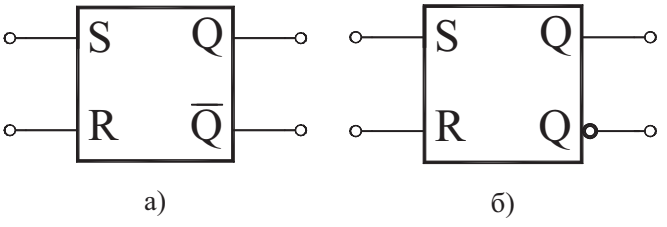
Заради овие причини се применува вкрстување на логички кола со повеќе влезови и тоа на различни начини за што ќе зборуваме во продолжение.

4.2.1. SR ФЛИП-ФЛОП ОД НИЛИ ТИП

Флип-флопот кој се добива со вкрстување на две НИЛИ кола со по два влеза популарно се вика **SR** или **RS флип-флоп.** Во англиската терминологија ваквото коло е познато и како SR Latch, коло за заклучување или задржување. Во понатамошното излагање терминот **леч** ќе го превземеме и ќе го користиме без преведување. Логичката структура на **SR флип-флопот** е прикажана на сл. 4-5, додека неговите симболички ознаки на сл. 4-6 а) и б).



Сл. 4-5. Логичка шема на SR флип-флоп



Сл. 4-6. Симболички ознаки на SR флип-флоп

Влезот S (SET) служи за поставување (сетирање) на флип-флопот (Q=1). Тоа се изведува така што на него се носи 1 (S=1), а влезот R се држи на 0. Штом S ќе го достигне

нивото 1, на излезот се јавува состојбата Q=1 ($\overline{Q}$= 0): ако претходно излезот бил на 1, тој ќе остане на 1, но ако претходно излезот бил на 0, тој ќе го промени нивото и ќе оди на 1.

Преку вториот влез означен со R (RESET), флип-флопот се брише, се доведува во ресетирана состојба (Q=0) на начин спротивен од претходниот: сега на влезот R се носи 1, а влезот S се воспоставува 0. Тогаш кога нивото на R ќе ја достигне вредноста 1, на излезот се јавува 0, т.е. Q=0 ($\overline{Q}$=1) ако претходно излезот бил на 0, тој ќе остане на 0, но ако бил на 1, тогаш ќе се промени на 0. Доколку влезовите S и R истовремено се 0 (S=R=0), флип-флопот ја задржува својата претходна состојба.

Ако на влезовите S и R истовремено се донесе 1, (S=R=1), тогаш излезната состојба на флип-флопот ќе биде таква што и Q и $\overline{Q}$ ќе бидат на 0. Значи, нема да важи почетната претпоставка дека излезите од флип-флопот меѓусебно се комплементарни. Освен ова, може да се случи следната состојба на флип-флопот да не биде дефинирана. Имено, ако претпоставиме дека по побудата S=1 и R=1, и двата влеза одат на ниско ниво S=0 и R=0, тогаш излезот ќе зависи од оној сигнал што подолго останал на 1. Излезното ниво ќе биде одредено од состојбата на оној влезен сигнал којшто подоцна (втор) се спуштил на ниско ниво, а бидејќи зедовме дека и двата сигнала истовремено се менуваат, излезното ниво на Q и $\overline{Q}$ не може прецизно да се определи. Заради ова влезната комбинација S=1 и R=1 кај SR флип-флопот не е дозволена.

Од кажаното произлегува дека ефективно дејство врз флип-флопот, т.е. промена на излезот, може да се случи само ако постои појава на 1 на некој од влезовите S или R. Заради ова велиме дека влезовите S и R се активни на 1, т.е. на високо ниво.

Работата на SR флип-флопот целосно може да се определи од табелата на премин и излез означена со таб. 4-2, додека неговата табела на побуда е прикажана како таб. 4-3. Поаѓајќи од табелата на премин 4-1, принципот на работа на SR флип-флопот може да се опише и по аналитички пат, со следнава карактеристичната равенка:

$$Q^+ = S + \overline{R}\,Q, \text{ или } Q^+ = (S + Q)\,\overline{R}, \text{ при што SR=0.} \qquad (4\text{-}1)$$

S R	Q^+
0 0	Q
0 1	0
1 0	1
1 1	?

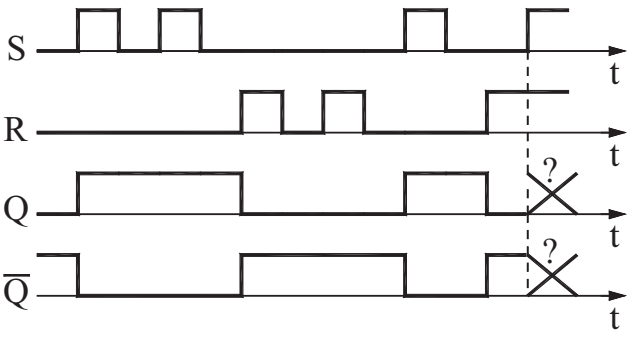
Таб. 4-2. Табела на премин и излез на SR флип-флоп

Q Q^+	S R
0 0	0 x
0 1	1 0
1 0	0 1
1 1	x 0

Таб. 4-3. Табелата на побуда на SR флип-флоп

Во равенката 4-1 мора да важи SR=0. Ограничувањето SR=0 ја кажува претходно споменатата забрана дека во ист момент влезот S и влезот R не смеат да бидат активни, т.е. тие не смеат истовремено да бидат на логичка 1.

Функционирањето на SR флип-флопот е илустрирано со временските дијаграми дадени на сл. 4-7, при што претпоставено е дека неговата почетна положба била Q=0, ($\overline{Q}$=1) , односно дека тој на почетокот бил ресетиран.

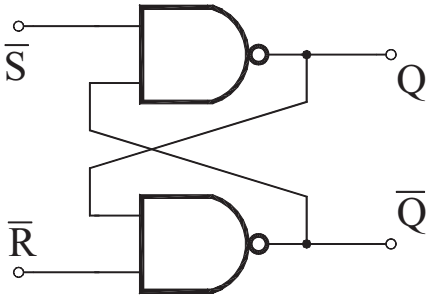


Сл. 4-7. Временски дијаграми во каратеристичните точки на влезот и излезот од SR флип-флопот

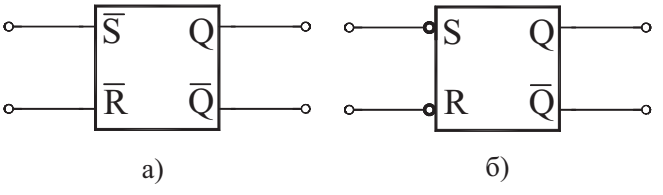
Веќе наведовме дека прикажаната конфигурација на SR флип-флопот од сл. 4-5 се нарекува леч. За него е карактеристична појавата на транспарентност бидејќи кај ова коло секоја промена на логичките нивоа на влезовите предизвикува промена (се рефлектира) на излезните нивоа. Поедноставно кажано, излезот "може да се гледа" од страната на влезот и влезот "може да се гледа" од страната на излезот.

4.2.2. $\overline{S}\ \overline{R}$ ФЛИП-ФЛОП ОД НИ ТИП

Покрај SR флип-флоп од НИЛИ тип постои и SR флип-флоп од НИ тип реализиран со вкрстена врска на две НИ кола со по два влеза според сл. 4-8. За разлика од SR НИЛИ флип-флопот кој беше активен на 1, SR НИ флип-флопот е активен на ниво на 0, бидејќи неговиот излез реагира на појавата на 0 на еден од двата влеза. Заради ова во литературата ваквиот флип-флоп обично се означува со $\overline{S}\ \overline{R}$, или на влезовите S и R се става мал круг (○). Логичките симболи на $\overline{S}\ \overline{R}$ флип-флоп се прикажани на сл. 4-9 а) и б).



Сл. 4-8. Логичка шема на $\overline{S}\ \overline{R}$ флип-флоп



Сл. 4-9. Симболи на $\overline{S}\ \overline{R}$ флип-флоп

Табелата за премин и излез за $\overline{S}\ \overline{R}$ флип-флопот е означена со таб. 4-4, додека неговата табела на побуда (екситација) е дадена како на таб. 4-5.

$\overline{S}\ \overline{R}$	Q^+
0 0	?
0 1	1
1 0	0
1 1	Q

Таб. 4-4. Табела на премин и излез на $\overline{S}\ \overline{R}$ флип-флоп

Q Q^+	$\overline{S}\ \overline{R}$
0 0	0 x
0 1	0 1
1 0	1 0
1 1	x 1

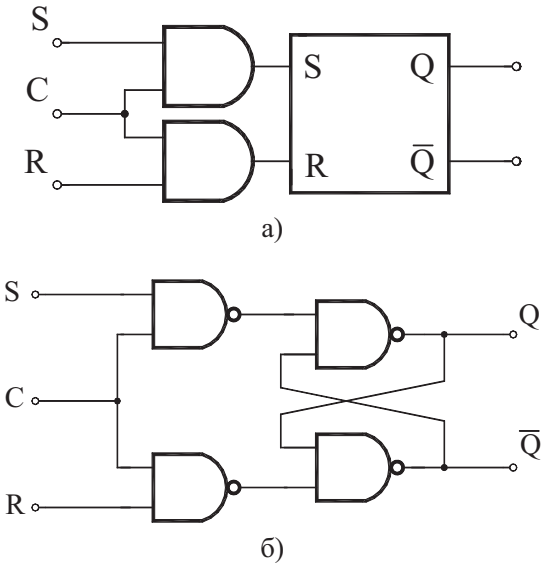
Таб. 4-5. Табелата на побуда на $\overline{S}\ \overline{R}$ флип-флоп

Поаѓајќи од дадената комбинациона табела, лесно може да се објасни работата на $\overline{S}\,\overline{R}$ флип-флопот. Имено, ако на влезот се донесе комбинација $\overline{S}=0$ и $\overline{R}=1$, тогаш флип-флопот ќе биде сетиран, т.е. $Q=1$, ($\overline{Q}=0$). Од друга страна, ако влезовите се побудат со $\overline{R}=0$ и $\overline{S}=1$, тогаш флип-флопот ќе биде ресетиран, т.е. $Q=0$, ($\overline{Q}=0$). Кога и двата влеза се наоѓаат на логичка 1, следната состојба на излезот од флип-флопот ќе остане иста со претходната. Кај овој флип-флоп е забрането на двата влеза истовремено да се појават 0-и. Во овој случај, ако и двата влеза истовремено се 0-и ($\overline{S}=\overline{R}=0$), излезите од флип-флопот Q и $\overline{Q}$ нема да бидат меѓусебно комплементарни, туку ќе се наоѓаат на ниво на 1, што лесно може да предизвика појава на недефиниран излез.

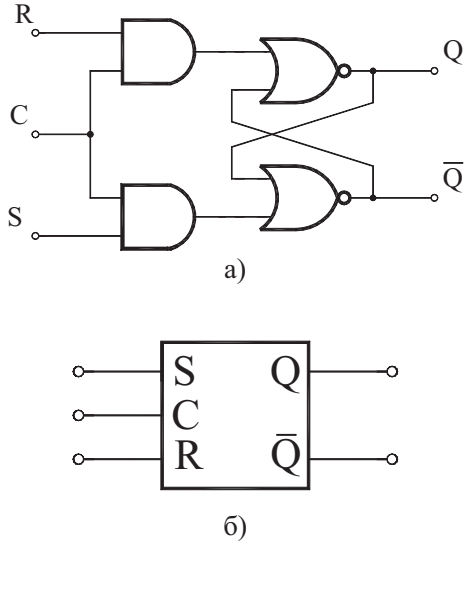
Логичката шема на $\overline{S}\,\overline{R}$ флип-флопот од сл. 4-8 со НИ логички кола може да се трансформира и да се однесува како SR флип-флоп од НИЛИ тип доколку секој од влезните сигнали на сл. 4-8 се доведе до НИ вратите преку инверторски кола.

4.2.3. SR ФЛИП-ФЛОП ТАКТИРАН СО НИВОТО НА ТАКТ-СИГНАЛОТ

Ваквиот флип-флоп треба да биде управуван со логичката состојба на влезовите S и R, но во моментот на појавата на високо логичко ниво на такт-сигналот C (C=1) што во принцип може да се оствари со поврзувањето прикажано на сл. 4-10 а). Логичката структура на **тактираниот** SR **флип-флоп** реализиран со НИЛИ кола е прикажана на сл. 4-10 б), со НИ кола на сл. 4-11 а), додека неговиот симбол е даден на сл. 4-11 б. На влезовите S и R се доведуваат соодветните логички напонски нивоа, а заради присуството на двете И кола чиј излез се контролира преку логичкото ниво на тактот C кој истовремено се носи како влез и до двете И кола, ефективно дејство врз состојбата на флип-флопот ќе се изврши само кога ќе се појави предниот (растечкиот, позитивниот) раб на влезот за такт-сигналот C. И ова коло претставува леч бидејќи и кај него е карактеристична појавата на транспарентност, но само ако сигналот за такт C е висок (C=1), заради што овој сигнал може да се гледа и како сигнал за дозвола (овозможување) на работа E.



Сл. 4-10. Логичка структура на тактиран SR флип-флоп



Сл. 4-11. Симбол на тактиран SR флип-флоп

Бидејќи влезовите S и R можат да дејствуваат врз состојбата на флип-флопот само ако се синхронизирани со такт-сигналот, тие се викаат синхрони влезови, а самиот флип-флоп **синхронизиран** или **тактиран** SR флип-флоп.

Работата на тактираниот SR флип-флоп може да се илустрира со табелата на премин и излез прикажана на сл. 4-6 и табелата на екситација дадена на сл. 4-7. Тие практично се исти со табелите на премин и излез за асинхрониот SR флип-флоп, со една забелешка дека важат само кога тактот преминува од ниско логичко ниво (0) на високо (1) и има стабилно високо ниво што се случува со појавата на предниот раб на тактот кој ја овозможува работата на логичките кола.

Ова подобро може да се види од неговата карактеристична равенка со такт влез C:

$$Q+ = (S + \overline{R}\, Q) C + Q\, \overline{C} \text{ , или } Q+ = [(S + Q) \overline{R}]C + Q\, \overline{C} \qquad (4\text{-}2)$$

при што треба да биде исполнет условот SR = 0.

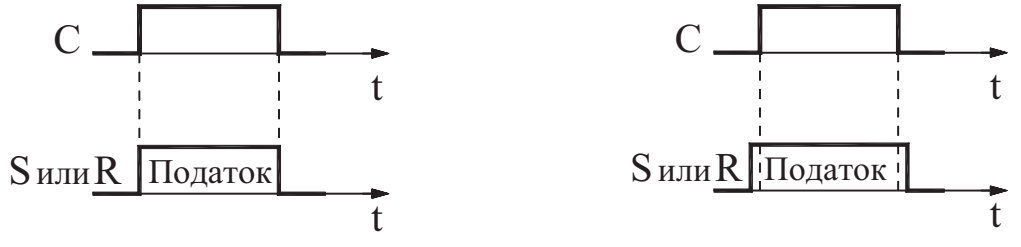
S R	Q^+
0 0	Q
0 1	0
1 0	1
1 1	?

Таб. 4-6. Табела на премин и излез на SR флип-флоп

Q Q^+	S R
0 0	0 x
0 1	1 0
1 0	0 1
1 1	x 0

Таб. 4-7. Табела на побуда на SR флип-флоп

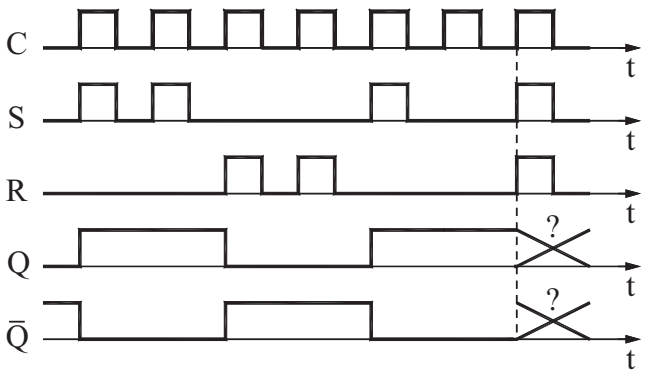
За сигурна работа на тактираниот флип-флоп, нивото на влезовите S и R треба да биде стабилно додека трае такт-импулсот C, или поточно податоците на влезовите S и R треба да се појават непосредно пред предниот раб на тактот, а да завршат непосредно по неговиот заден раб, како што е прикажано на сл. 4-12 а). За ваквиот тип на флип-флоп се вели дека е управуван, т. е. дека се префрлува од една состојба во друга со нивото на такт-сигналот (анг. level-triggered или pulse-triggered). Ние обично ќе разгледуваме идеални случаи, опишани со сл. 4-12 б). Ако овој услов не биде исполнет, односно ако некој од влезовите S или R ја промени својата логичка состојба, таа промена ќе помине низ влезните кола кои се отворени, бидејќи тактот е на високо ниво, со што ќе дојде до промена на излезот за време на такт-импулсот.



а) б)

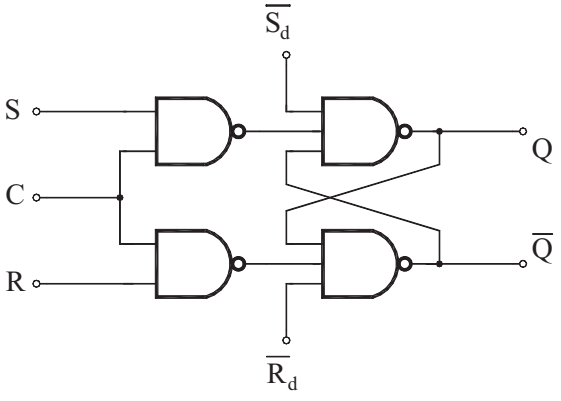
Сл. 4-12. Усогласеност на влезните сигнали кај SR флип-флоп управуван со нивото на тактот

Со примерот од сл. 4-13 е илустриран принципот на работа кај вака тактираниот SR флип-флоп, при што е претпоставено дека неговата почетна состојба била ресетирана: $Q = 0$, ($\overline{Q} = 1$). Од презентираните временски дијаграми се гледа дека сега транспарентноста се јавува само ако C=1 бидејќи само во тој временски интервал промената на влезовите ги афектира излезите (само тогаш тие меѓусебно „се гледаат”).

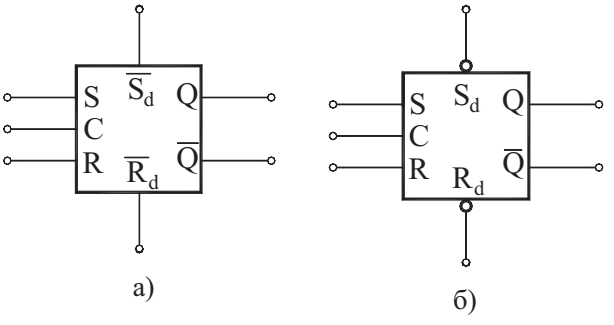


Сл. 4-13. Временски дијаграми за начинот на работа на тактиран SR флип-флоп

Кај тактираниот флип-флоп е вообичаено да се додаваат уште два влеза, како што е прикажано на сл. 4-14. Логичкиот симбол на ваквиот флип-флоп е даден на сл. 4-15 а) и б). Новите влезови се активни на ниско ниво и се означуваат со $\overline{Sd}$ или $\overline{PRS}$ *(анг. PRE-SET, почетно сетирање*) и $\overline{Rd}$ или $\overline{CLR}$ (анг. CLEAR, бришење, почетно ресетирање).

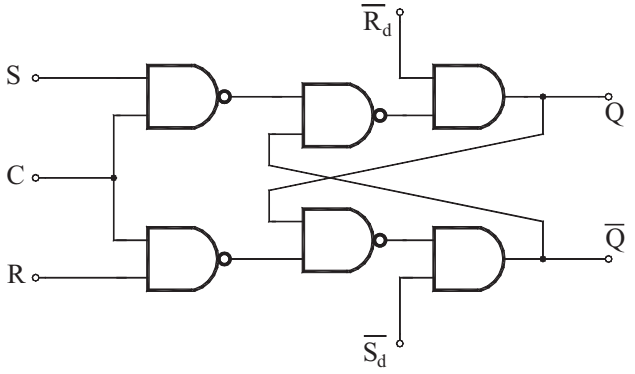


Сл. 4-14. Логичката шема на тактиран SR флип-флоп со директни (асинхрони) влезови



Сл. 4-15. Симболи на тактиран SR флип-флоп со директни (асинхрони) влезови

Овие два влеза директно влијаат на излезот од флип-флопот независно од такт-сигналот, па затоа и се викаат *директни* или *асинхрони* влезови. Нивна задача е да овозможат дефинирање на почетната состојба на флип-флопот, и евентуално негово управување независно од тактот. Кога $\overline{Sd}$=0, флип-флопот почетно се поставува на високо ниво (Q=1), а ако $\overline{Rd}$=0 флип-флопот почетно се поставува на ниско ниво (Q=0). И за овие влезови важи условот дека истовремено не смеат да бидат активни. Нивна забранета комбинација ќе биде $\overline{Sd}=\overline{Rd}$=0. Така, во отсуство на такт-импулсот состојбата на флип-флопот целосно ќе биде одредена од директните влезови. Кај вака изведените директни влезови активни нивоа можат да се донесат само ако нивото на тактот е ниско. Меѓутоа, ако нивото на тактот е високо, а тие се активни заедно со податочните влезови S и R, но со спротивни барања, како на пр. S=1 и $\overline{Rd}$=0, или R= 1 и $\overline{Sd}$ = 0, тогаш излезот од флип-флопот Q ќе биде недефиниран. Овој проблем се избегнува со додавање на уште две И кола на излезот од флип-флопот, како што е прикажано на сл. 4-14, така што се добива конфигурацијата како на сл. 4-16. Сега директните влезови го надвладуваат (доминираат над) такт-сигналот. Имено, излезот од флип-флопот зависи само од состојбата на влезовите $\overline{Sd}$ и $\overline{Rd}$, независно од нивото на тактот C и влезовите за податоците S и R.



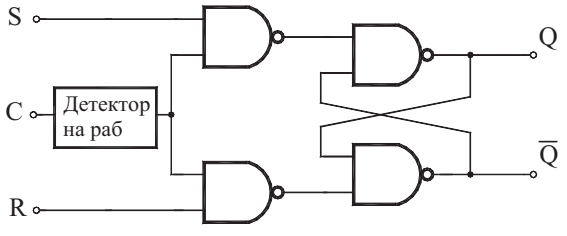
Сл. 4-16. Логичка шема на тактиран SR флип-флоп со доминантни директни (асинхрони) влезови

Обично секогаш кога ќе употребуваме синхрони флип-флопови со директни влезови ќе претпоставуваме дека директните влезови го надвладуваат тактот. Тактираните флип-флопови можат да работат под дејство на такт-сигналот и нивното однесување да зависи само од синхроните влезови S и R, единствено кога директните (асинхроните) влезови се пасивни, за што треба да се држат на високо ниво.

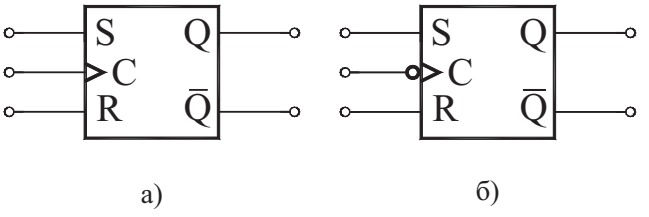
Треба да спомнеме и тоа дека кај некои изведби на флип-флопови директните влезови можат да бидат активни на високо ниво. Во овие случаи тие се означуваат со S_d R_d, односно PRS и CLR. За ваквите флип-флопови забранета комбинација ќе биде доведувањето на две 1-и на директните влезови, т.е. $S_d=R_d=1$. Однесувањето на флип-флопот ќе биде контролирано од синхроните влезови S и R само кога директните влезови истовремено се на ниско ниво ($S_d=R_d=0$).

4.2.4. SR ФЛИП-ФЛОП ТАКТИРАН СО РАБОТ НА ТАКТ-СИГНАЛОТ

Покрај флип-флоповите кои се управувани со нивото на тактот, во практиката голема примена наоѓаат и т.н. прекинувачки флип-флопови, кои својата состојба ја менуваат само при појавата на работ на такт сигналот (анг. *edge-triggered)* и тоа при неговиот премин од ниско на високо ниво, или од високо на ниско. Во првиот случај станува збор за менување на состојбата при појава на позитивниот (растечкиот) раб на тактот, додека во вториот случај на негативниот (опаѓачкиот). Кај овие флип-флопови такт сигналот може да биде висок и да се случи било каква промена на логичката состојба на влезовите, но тоа нема да има никакво влијание врз состојбата на излезите, што не беше случај кај претходно анализираните флип-флопови. Ваквите флип-флопови во принцип може да се добијат ако такт сигналот од сл. 4-10 прво се донесе на коло кое ќе врши детекција на работ (анг. edge-triggered detector) на такт сигналот според сл. 4-17. Симболичката ознака на ваквите флип-флопови е прикажана на сл. 4-18 а) и б).

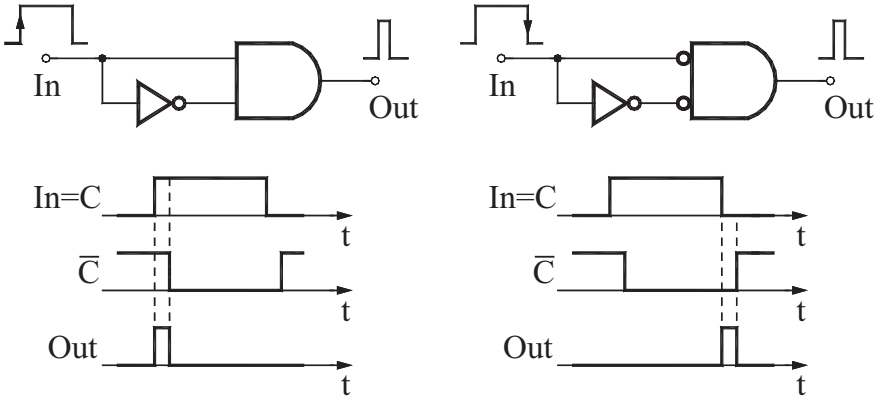


Сл. 4-17. Логичка шема на прекинувачки флип-флоп.



Сл. 4-18. Симболи на прекинувачки флип-флоп

Детекторот на растечкиот или опаѓачкиот раб, такт-импулсот го трансформира (претвора) во многу тесен импулс широк само неколку наносекунди. На сл. 4-19 а) и б) последователно се прикажани типични и наједноставни кола за детекција на растечкиот и опаѓачкиот раб на тактот. Такт сигналот се носи истовремено на двата влеза од И колото. При ова, едниот влез се побудува откога тактот ќе помине низ влезното инверторско коло. Улогата на инверторот е да внесе минимално доцнење од неколку наносекунди заради минувањето на такт-сигналот низ него и со тоа на влезот од И колото да донесе два минимално временски поместени инвертирани сигнали. Ваквото поврзување овозможува на излезот од детекторот да се формира позитивен импулс точно во моментот на појава на работ на тактот (на предниот раб според сл. 4-19 а), а на задниот според сл. 4-19 б)), со времетраење кое е еднакво со времето на доцнење на инверторот. Како замена на логичкото И коло од сл. 4-19 б) може да се употреби НИЛИ коло.



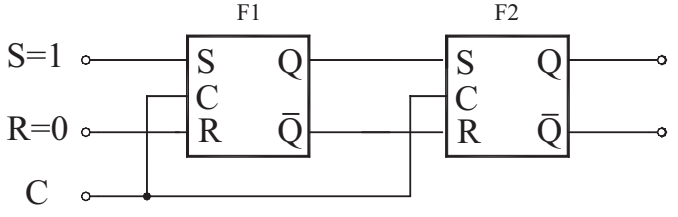
а) детекција на преден раб б) детекција на заден раб

Сл. 4-19. Реализација на детектор на раб и негови временски дијаграми

Кај ваквите флип-флопови што реагираат на појавата на работ на такт сигналот, а не на неговото ниво, тактот во нивната симболичка ознака се означува со мал триаголник (Δ) како што е прикажано на сл. 4-18 а) и б). Симболот на флип-флопот од сл. 4-18 а) се однесува на флип-флоп управуван со позитивниот раб на тактот, додека сл. 4-18 б) означува флип-флоп управуван со негативниот раб на тактот.

4.2.5. SR ФЛИП-ФЛОП СО MASTER-SLAVE СТРУКТУРА

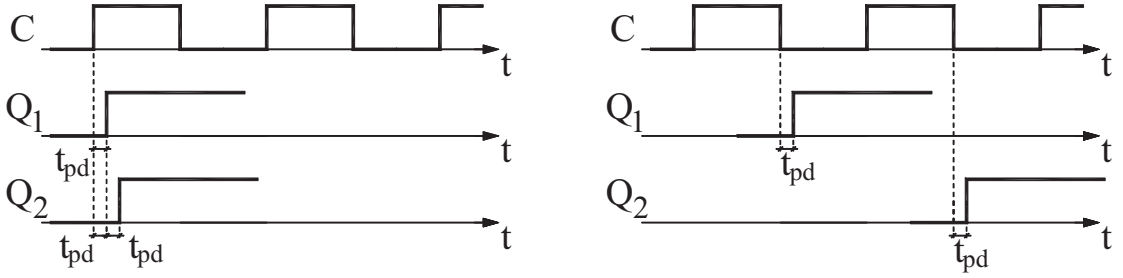
Тактираниот SR флип-флоп ги менуваше своите излези (Q и $\overline{Q}$) со појавата на растечкиот раб на влезот за такт-сигнал. Меѓутоа, ваквото однесување, во голем број на дигитални склопови би предизвикал неправилно функционирање. Проблемот произлегува од следниов факт: дигиталните уреди користат поголем број меѓусебно поврзани флип-флопови, така што секој флип-флоп реагира на податокот што е присутен на неговиот влез пред да се случи активниот премин, но и на новиот податок, кој е резултат на промената на излезите на другите флип-флопови, кои исто така, го промениле излезот. За да го разјасниме овој проблем, ќе го разгледаме примерот на два каскадно врзани SR флип-флопови прикажани на сл. 4-20.



Сл. 4-20. Два каскадно поврзани тактирани SR флип-флопови

Ќе претпоставиме дека и двата флип-флопови F_1 и F_2, реагираат со појавата на предниот раб на тактот и дека почетната состојба на флип-флоповите била ресетирана (Q_1=0 и Q_2=0). На влезот во каскадата се доведува влезната комбинација S_1=1 и R_1=0, која во првиот флип-флоп за време на првиот такт-циклус треба да внесе 1, а во вториот флип-флоп за време на вториот такт-циклус. Попрецизно, кога ќе се појави првиот такт-импулс, излезот Q_1 треба да оди на 1 (Q_1=1), додека излезот Q_2 треба да остане на 0 (Q_2=0). Со појавата на вториот такт импулс, првиот флип-флоп треба да остане на 1 (Q_1=1), а излезот од вториот флип-флоп да оди на 1 (Q_2=1). Меѓутоа, склопот нема да работи правилно. Имено, кога ќе се појави предниот раб на тактот првиот флип-флоп оди на 1 (Q_1=1), бидејќи S_1 = 1 и R_1= 0, но со доцнење t_{PD} кое време е потребно за сигналот да помине низ првиот флип-флоп. За ова кусо време и излезот од вториот флип-флоп не се менува бидејќи $S_2=Q_1$=0 и $R_2=\overline{Q}_1$=1. Но, веднаш по завршувањето на t_{PD}, S_2 оди на 1 (S_2=1), а R_2 (R_2=0). Сега, бидејќи тактот C сèуште е активен, излезот од вториот флип-флоп оди на 1 (Q_2=1) уште за време на првиот такт-импулс, а не онака како што требаше, за време на вториот. Ова е илустрирано со временските дијаграми од сл. 4-21 а).

Еден начин за ова да се спречи е да се генерира такт-сигнал кој има времетраење на импулсите многу помало во однос на траењето на паузите. Ваквиот такт-сигнал, кој има многу кусо времетраење на активното ниво, покусо од времето на доцнење t_{PD}, може да предизвика несигурна работа при префрлување на флип-флоповите, затоа што тоа време може да биде недоволно за тие да реагираат на влезните податоци. Накратко, времето за кое се овозможени логичките кола на флип-флоповите е покусо отколку што треба. Заради ова, се практикува едно поинакво решение со кое се избегнуваат овие проблеми и кое обезбедува врската помеѓу влезовите и излезите да се прекине пред излезот да се смени. Станува збор за употребата на т.н. **главен-извршен** или **MS** (анг. master-slave, главен-помошен, двомемориски) **флип-флоп**. MS флип-флопот се управува со оној раб на тактот кога тој преминува од овозможување на логичките кола во работа на нивно оневозможување, а во случајов тоа е опаѓачкиот раб на импулсот за такт синхронизација.

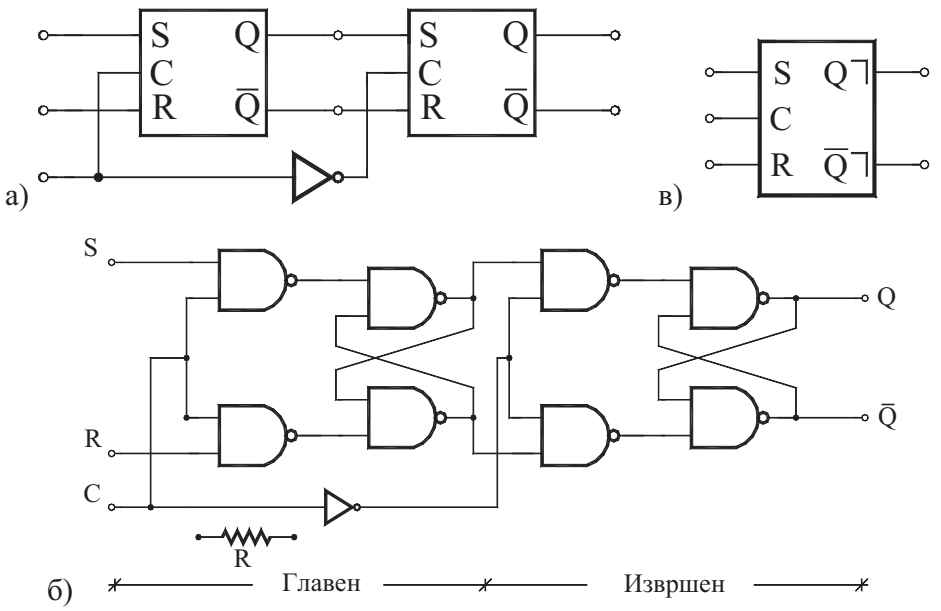


а) управувани со нивото на тактот б) управувани со MS структура

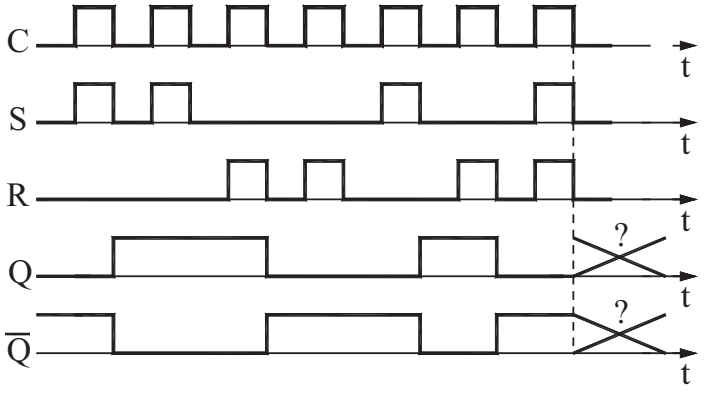
Сл. 4-21. Временски дијаграми кај два каскадно поврзани тактирани SR флип-флопови

Одзивот на истата побуда, но со MS флип-флопови е прикажан на сл. 4-21 б). Излезите Q_1 и $\overline{Q}_1$ од првиот флип-флоп се менуваат на опаѓачкиот раб од тактот и не можат да дејствуваат на вториот флип-флоп, бидејќи нивото на тактот е ниско. Промената на вториот излез Q_2 ќе се случи со појавата на опаѓачкиот раб на такт-сигналот, така што сега склопот работи правилно, а освен ова за време на вториот такт-циклус податоците можат да се стабилизираат на потребните нивоа $S_2=Q_1$=1 и $R_2=\overline{Q}_1$= 0.

MS структура на флип-флоп се добива со каскадно поврзување на два тактирани флип-флопови од сл. 4-10 со што се добива сл. 4-22 а) и б) која ја претставува логичката структура на SR MS флип-флопот. Неговиот логички симбол е прикажан на сл. 4-22 в).



Сл. 4-22. Логичката структура (а, б) и логички симбол (в) на SR MS флип-флоп



Сл. 4-23. Временски дијаграми за однесувањето на тактиран SR MS флип-флоп

Функционирањето на овој флип-флоп е илустрирано со временските дијаграми прикажани на сл. 4-23. Принципот на работа е идентичен со работата на обичниот SR флип-флоп, со единствена разлика што сега логичките состојби на излезот се јавуваат по задниот раб на такт-сигналот. Ова е овозможено со употребата на двата SR флип-флопови кои се тактирани со меѓусебно комплементарни тактови.

Имено, предниот раб на тактот дејствува на состојбата на главниот (анг. *master,* господарот, меморискиот) флип-флоп, а состојбата на извршниот (slave, робот, излезниот) флип-флоп останува непроменета затоа што на неговиот влез се јавува опаѓачкиот раб на тактот. Кога ќе се појави задниот раб на тактот, престанува дејството врз главниот флип-флоп, но инвертираната вредност на овој напонски облик т.е. преден раб, сега се јавува на извршниот флип-флоп и тој ја менува состојбата под дејство на излезите Qm и $\overline{Qm}$ бидејќи тие се сега негови влезови: $S_S=Qm$ и $RS=\overline{Qm}$. Ова значи дека сè додека тактот е константен (на ниво 1 или 0), флип-флоповите меѓусебно се одвоени, а со тоа и влезот од излезот кај MS флип-флопот. Со појавата на предниот раб на тактот работи главниот флип-флоп, додека извршниот флип-флоп е исклучен, па влезните податоци присутни на S и R влезовите се запишуваат во главниот флип-флоп. Извршниот флип-флоп податоците ги прима со појавата на задниот раб на тактот, бидејќи тогаш се вклучува. Во тој момент тие се појавуваат и на излезот, а воедно е оневозможен и главниот флип-флоп.

И кај MS флип-флопот, за неговата правилна работа, податоците не смеат да се менуваат за времето кога тактот е активен, т.е. сигналите присутни на влезовите S и R мора да се стабилни сè додека C=1. Овој недостаток може да се избегне ако логичката структура на главниот флип-флоп се модифицира така да тој реагира само на појавата на предниот раб на такт сигналот.

Кај MS структурите на флип-флопови често се додаваат приклучоци за асинхрони влезови со кои директно се поставува или се брише флип-флопот. Тие може да се активни на високо или ниско ниво, и зависно од изведбата, смеат да се донесат само во паузата на тактот, или целосно да го надвладеат дејството на тактот (да доминираат над него).

Ако направиме осврт на сè она што досега беше изнесено, може да заклучиме дека основен недостаток на SR флип-флоповите од било кој тип, е тоа што постои забранета влезна комбинација кога влезовите S и R не смеат истовремено да бидат активни со што се ограничува неговата примена. Дури и системски да се обезбеди да не дојде до појавување на ваква побуда, заради дејството на шумот или пречките, некоја дозволена влезна комбинација може да стане нелегитимна. Поради ова се изработуваат и такви SR флип-флопови кај кои еден од влезовите е доминантен тогаш кога ќе се појави нелегална влезна комбинација, па излезот ќе биде одреден од тој доминантен влез. Сепак, проблемот со забранетата влезна комбинација целосно е надминат со проектирањето на други типови флип-флопови кои имаат дефинирани состојби за секој влез. Нивната анализа ќе биде презентирана во понатамошното излагање.

4.3. ЈК ФЛИП-ФЛОП

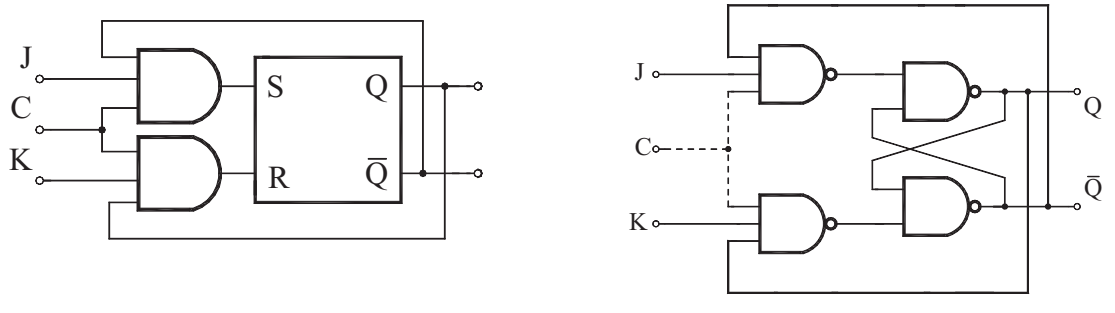
ЈК флип-флопот по својата логичка структура претставува модифициран SR флип-флоп за кој нема забранета влезна комбинација. Неговата логичка конфигурација е прикажана на сл. 4-24 а) б) од каде се гледа дека JK флип-флопот се добива со примена на SR флип-флоп со изведени две вкрстени повратни врски. При ова, JK флип-флопот ќе биде тактиран ако на влезните НИ кола се додаде уште по еден влез на кој ќе се носи тактот C како што е прикажано на сл. 4-24 со испрекинати линии. Симболичката ознака на JK флип-флопот е дадена на сл. 4-25, табелата на премин и излез е означена со таб. 4-8, додека табелата на екситација како таб. 4-9.

Врз основа на табелата на премин и излез може да се напише карактеристичната равенка за JK флип-флопот. За асинхрониот JK флип-флоп, таа ќе биде дадена со следнава логичка равенка:

$$Q^+ = J\overline{Q} + \overline{K}Q \qquad (4\text{-}3)$$

Логичката равенка на синхрониот ЈК флип-флоп со влез за такт C е:

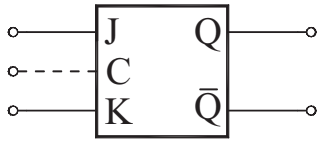
$$Q^+ = (J\overline{Q} + \overline{K}Q)C + \overline{Q}C \qquad (4\text{-}4)$$



а) б)

Сл. 4-24. Логичка структура на JK флип-флоп

За JK флип-флопот нема забранета влезна комбинација. Имено, од комбинационата табела, а и од функцијата на премин станува јасен неговиот принцип на работа: JK флип-флопот работи како SR флип-флоп за сите влезни комбинации, при што аналоген влез на S е J, додека на влезот R соодветствува влезот K, освен за комбинацијата кога и двата влеза J и K се на ниво на 1. Кога JK флип-флопот ќе се побуди со влезна комбинација J = 1 и K = 1, состојбата на излезот ќе биде дефинирана и ќе претставува комплемент на неговата претходна состојба, т.е. ќе важи $Q^+ = \overline{Q}$.



Сл. 4-25. Логички симбол на JK флип-флоп

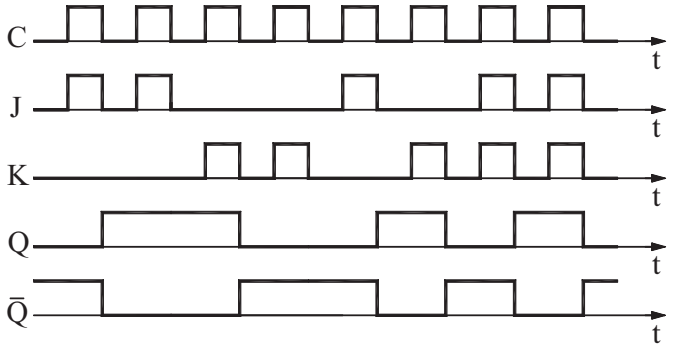
J K	Q^+
0 0	Q
0 1	0
1 0	1
1 1	$\overline{Q}$

Таб. 4-8. Табела на премин и излез кај JK флип-флоп

Q Q^+	J K
0 0	0 x
0 1	1 x
1 0	x 1
1 1	x 0

Таб. 4-9. Табела на побуда на JK флип-флоп

Еден пример за однесувањето на JK MS флип-флопот е прикажан со временските дијаграми дадени на сл. 4-26 со ресетирана почетна положба.



Сл. 4-26. Временски дијаграми за однесувањето на тактиран JK флип-флоп

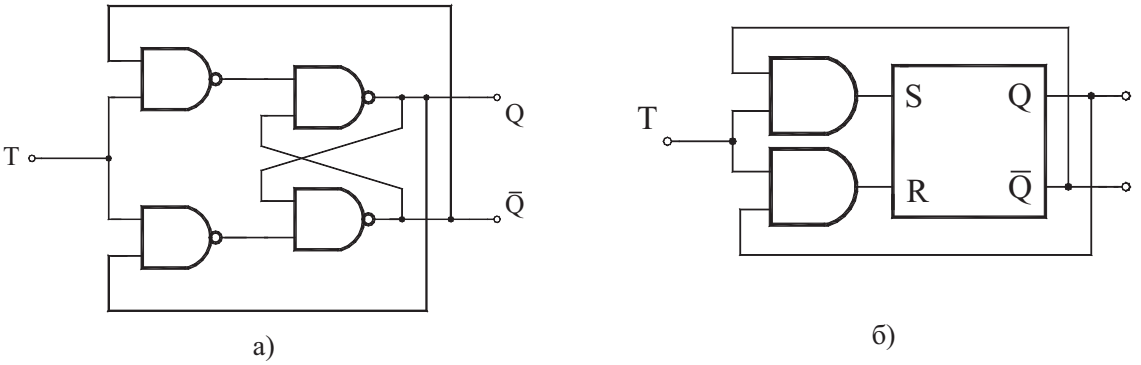
Кај тактираните JK флип-флопови обично се додаваат асинхрони влезови кои можат да бидат активни на ниско ниво и се означуваат со $\overline{S_d}$ ($\overline{PRS}$) и $\overline{R_d}$ ($\overline{CLR}$), додека ако се активни на високо ниво се означуваат со S_d (PRS) и R_d (CLR).

Кај JK флип-флопот можат да настанат одредени проблеми во неговото функционирање. Така на пр. да претпоставиме дека J=1 и K=1 и дека почетната состојба на флип-флопот била $Q = 0$ ($\overline{Q} = 1$). Кога ќе се појави растечкиот раб на тактот, по одредено задоцнување t_{pd} кое е помало од траењето на импулсот од такт-сигналот, на излезот ќе се појави $Q = 1$ ($\overline{Q} = 0$). Оваа промена од излезот се враќа на влезот, а бидејќи импулсот од тактот сèуште трае, повторно ќе се предизвика менување на излезот $Q = 0$ ($\overline{Q} = 1$) по време t_{pd}, и оваа промена пак се враќа на влезот итн. сè додека трае такт-импулсот. Ова значи дека излезот осцилира меѓу 0 и 1 сè додека тактот е активен (C = 1), па така кога тактот ќе се спушти на пасивно (ниско) ниво, излезот нема да биде дефиниран. Едно решение би било повратните сигнали да се задоцнат со што доцнењето t_{pd} ќе биде поголемо од времетраењето на такт-импулсот, но воедно и помало од траењето на паузата, а второто импулсот да биде пократок од времето на доцнење t_{pd}. Бидејќи и двете решенија создаваат други проблеми, наведениот недостаток се надминува ако

излезот се менува со појавата на задниот раб на тактот, или ако JK флип-флопот се реализира со MS структура слично на сл. 4-22. Во овој случај излезот се враќа на влезот, но сега влезот на такт-сигналот за вториот (извршниот) флип-флоп се наоѓа на ниско ниво заради што вратениот сигнал не може да предизвика дополнителна промена.

4.4. Т ФЛИП-ФЛОП

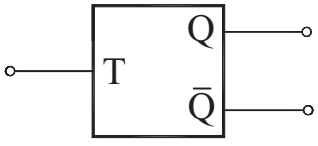
Овој тип флип-флоп има само еден влез означен со Т. Логичката структура на Т флип-флопот е прикажана на сл. 4-27 а) и б), а симболот на сл. 4-28. Од сл. 4-27 се гледа дека и овој флип-флоп се добива од SR (или JK) флип-флоп на кој се изведени две вкрстени повратни врски, додека два влеза се врзуваат во еден.



а) б)

Сл. 4-27. Логичка структура на Т флип-флоп

Табелата на премин и излез е прикажана како таб. 4-10, додека табелата на побуда е презентирана како таб. 4-11 и од нив лесно може да се сфати неговиот начин на работа.



Сл. 4-28. Логички симбол на Т флип-флоп

T	Q^+
0	Q
1	$\overline{Q}$

Таб. 4-10. Табела на премин и излез кај Т флип-флоп

Q	Q^+	T
0	0	0
0	1	1
1	0	1
1	1	0

Таб. 4-11. Табела на побуда на Т флип-флоп

Ако на влезот Т се донесе 0 (T=0), тогаш следната состојба на излезот ќе биде непроменета во однос на претходната: $Q^+ = Q$. Ако, пак, на влезот Т се донесе 1 (T=1), тогаш следната состојба на излезот ќе биде комплемент на претходната $Q^+ = \overline{Q}$. Поради овој начин на работа, Т флип флопот се вика и префрлувачки флип-флоп (од анг. **t**oggle или **t**rigger, срп. окидачки).

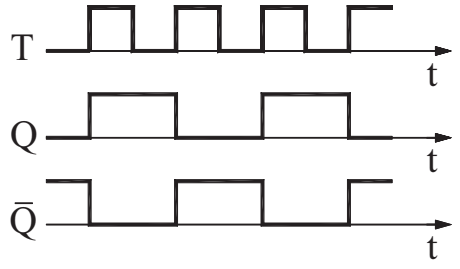
Врз основа на табелата на премин може да се напише и карактеристичната равенка на Т флип-флопот. Таа е дадена со следнава логичка равенка:

$$Q^+ = (Q \oplus T) \quad \text{или} \quad Q^+ = Q\overline{T} + Q\overline{T} \ . \tag{4-5}$$

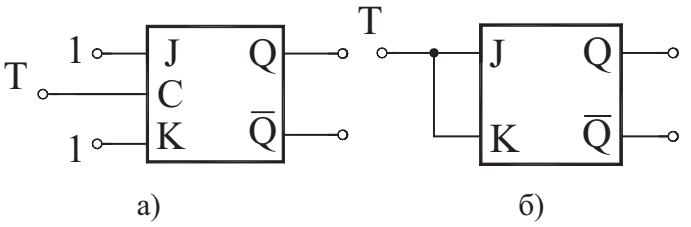
За тактиран флип-флоп со такт С карактеристичната равенка ќе биде

$$Q^+ = (Q \oplus T)C + Q\overline{C} \ \text{ или } \ Q^+ = (Q\overline{T} + Q\overline{T})C + Q\overline{C} \ . \tag{4-6}$$

На сл. 4-28 е прикажан еден пример за однесувањето на Т флип-флоп, со почетна ресетирана состојба $Q = 0$ ($\overline{Q} = 1$).



Сл. 4-28. Временски дијаграми за однесувањето на Т флип-флоп



Сл. 4-29. Реализација на Т флип-флоп со ЈК флип-флоп

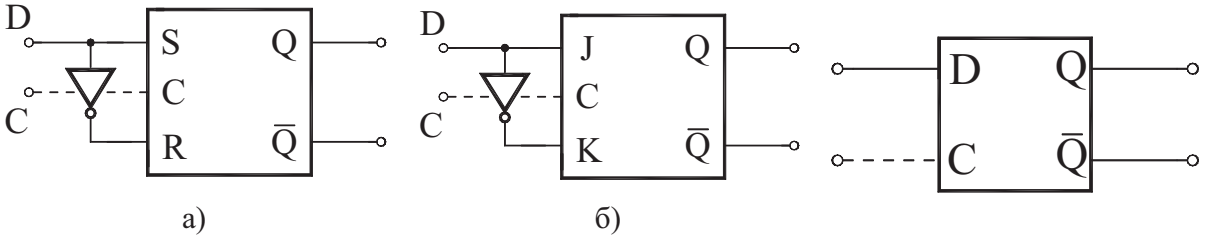
Ако се погледне карактеристичната табела 4-8 на ЈК флип-флопот, се забележува дека со поврзување на двата влеза во еден заеднички сме направиле транформација во Т флип-флоп. Слично, доведувањето на две 1-и на J и K влезовите (J=K=1) кога тактот е активен, предизвикува неговата следна состојба да биде комплементарна на претходната. Од тука произлегуваат изведбите на Т флип-флоп прикажани на сл. 4-29 со примена на нетактиран и тактиран ЈК флип-флоп чии влезови константно се држат на логичка 1.

Бидејќи Т флип-флопот ја менува состојбата при појава на влезен импулс, произлегува дека тој врши детекција на секој импулс, а тоа може да се примени за броење на влезните импулси, заради што за него се вели дека е и бројачки флип-флоп. Во врска со ова, Т флип-флопот се користи како основен градбен елемент при реализацијата на бројачки мрежи за кои ќе стане збор во една од темите што следуваат.

4.5. D ФЛИП-ФЛОП

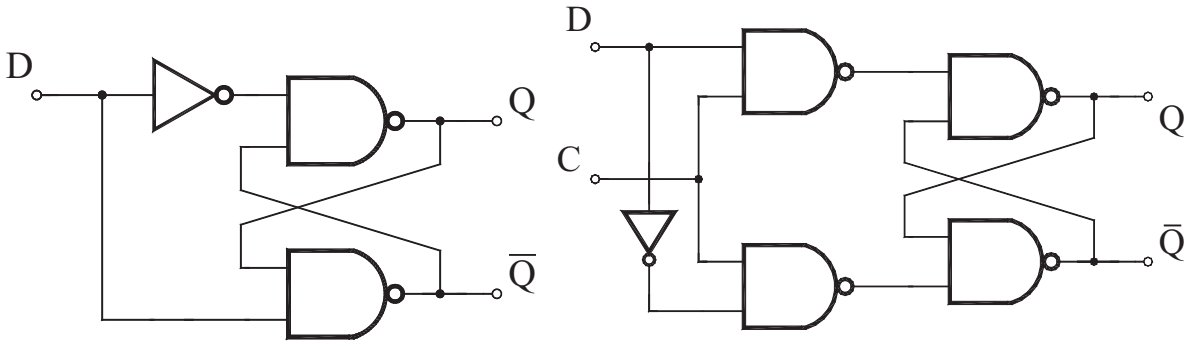
Ако се погледнат и анализираат карактеристичните табели 4-6 и 4-8 на SR и ЈК флип-флоповите лесно се забележува дека доведувањето на влезна комбинација со меѓусебно комплементарни влезови предизвикува следната состојба на флип-флопот да биде или 1 или 0. Оттука произлегува изведбата на D флип-флоп прикажана на сл. 4-30 која користи SR (или JK) флип-флоп и еден инвертор. Овој флип-флоп има само еден влез означен со D на кој се доведува податокот. Целта е да се добие дигитална компонента која ќе памти еден бит доведен на овој влез, така што следната состојба на флип-флопот секогаш ќе биде иста со логичката состојба на влезот D.

Симболичката ознака на D флип-флопот е дадена на сл. 4-31, додека неговата логичка структура на сл. 4-32 а) и б). Од сликите се гледа дека кога има потреба неговата работа да биде синхронизирана со други дигитални кола во единствен систем, се додава уште еден влез за такт (C) како што е претставено на сликите со испрекинати линии.



Сл. 4-30. Реализација на D флип-флоп со SR и JK флип-флоп

Сл. 4-31. Логички симбол на D флип-флоп



Сл. 4-32. Логичка структура на D флип-флоп

Комплетната работа на овој флип-флоп најдобро може да се илустрира со помош на неговата табела на премин и излез 4-12 и табелата на побуда (екситација) 4-13.

T	Q^+
0	0
1	1

Таб. 4-12. Табела на премин и излез кај T флип-флоп

Q	Q^+	T
0	0	0
0	1	1
1	0	0
1	1	1

Таб. 4-13. Табела на побуда на T флип-флоп

Врз база на овие табели може да се сфати принципот на работа на D флип-флопот. Без оглед на тоа каква е претходната состојба на излезот Q, следната состојба на флип-флопот ќе биде иста со доведената состојба на влезот D: $Q^+ = D$ така што ако D = 0 тогаш $Q^+ = 0$, а ако D =1 тогаш $Q^+ = 1$. Оттука фактички произлегува и скратениот назив на D флип-флопот. Имено, сигналот од влезот се пренесува на излезот, т.е. податокот *(анг.* ***D****ata)* присутен на влезот се проследува до излезот. При ова секогаш постои одредено доцнење *(анг.* ***D****elay)* заради неговото поминување низ флип-флопот, колку и да е тоа занемарливо мало.

Кај тактираните D флип-флопови влезот D зависи од појавата на такт-сигналот C и затоа треба да биде синхронизиран со него. Кај ваквиот флип-флоп излезот го следи влезот, почнувајќи од појавата на предниот раб на тактот, па сè додека тактот е висок, т.е. додека има активно ниво. Обично на тактираниот флип-флоп му се додаваат уште еден или два директни асинхрони влезови кои се независни од тактот и доминираат со него.

Начинот на функционирањето на D флип-флопот се опишува и во аналитички облик преку неговата карактеристична равенка:

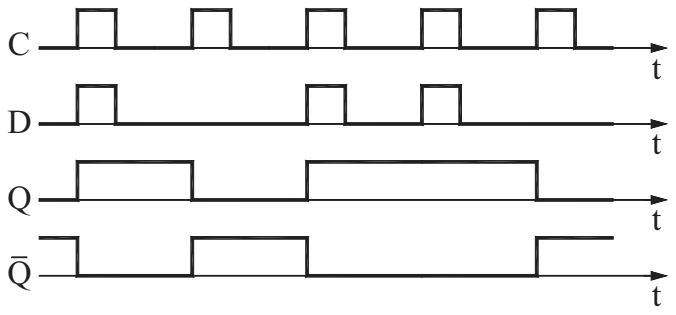
$$Q^+ = D \tag{4-7}$$

за асихрон флип-флоп, или

$$Q^+ = DC + Q\overline{C} \tag{4-8}$$

за синхрон D флип-флоп.

Принципот на работа на D флип-флопот е илустриран со следниот пример на временски дијаграми на влезот и излезот од D флип-флопот кои се прикажани на сл. 4-33.



Сл. 4-33. Временски дијаграми за однесување на D флип-флоп

Овој флип-флоп најмногу се употребува во изведбата на т.н. стационарни и поместувачки регистри како мемориски компоненти со мал капацитет кои ќе бидат предмет на анализа во следната тема каде ќе бидат обработени во поголеми детали.

Практичната реализација на D флип-флопот овозможува поголема густина на пакување во однос на RS или JK флип-флоповите, заради што обично D флип-флопот се користи како основна компонента за градба на покомплексни секвенцијални компоненти.

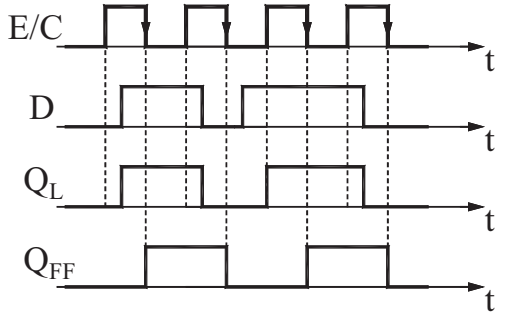
4.5.1 КОЛО ЗА ЗАКЛУЧУВАЊЕ

Од претходното излагање може да се констатира дека колото за заклучување (за задржување), или лечот, е всушност основната градбена единица на сите флип-флопови. Негова главна карактеристика е транспарентноста во работата: влезот и излезот се во комуникација и „меѓусебно се гледаат“.

Една од главните практични примени на D флип-флопот како леч е реализацијата на кола кои вршат прифаќање на податоците, нивно заклучување (задржување) и во одреден момент, нивно понатамошно проследување. Колото за задржување има два влеза: еден на кој се носи податокот D и вториот E кој се нарекува влез за дозвола бидејќи врши контрола на пропуштањето на сигналот на податокот од влезот D кон излезот Q.

	Влезови		Излез
Операција	E	D	Q^+
Пропушта (Q го следи D)	1	0	0
		1	1
Заклучува (Q е заклучен)	0	x	Q

Таб. 4-14. Кобинационата табела на колото за заклучување



Сл. 4-34. Временски дијаграми за однесување на колото за заклучување

Во врска со претходното,

- излезот Q го следи влезот D сè додека E е активен, т.е. со појавата на неговиот преден раб, па сè додека тој се наоѓа на високо ниво (E = 1);
- излезот Q ја заклучува (задржува) состојбата на влезот D затекната при задниот раб на влезот за дозвола E, т.е. во моментот кога тој преминува од високо на ниско ниво (од E = 1 паѓа на E = 0). Затекнатата состојба на излезот Q при преминот на влезот за дозвола E од високо на ниско ниво останува непроменета за цело време додека E е пасивен, т.е. сè додека E се наоѓа на ниско ниво (E = 0).

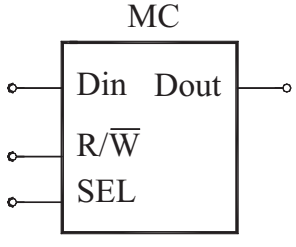
$$Q^+ = DE + D\overline{E} \qquad (4\text{-}9)$$

Принципот на работа се објаснува со функционалната табела 4-14 и карактеристична равенка (4-9). Дополнително, на сл. 4-34 се дадени временски дијаграми кои прикажуваат едноставен пример на однесување на лечот за дадена влезна побуда. Со цел да се забележи разликата помеѓу однесувањето на D лечот и D флип-флопот кој е управуван со опаѓачкиот раб на тактот, на сл. 4-34 се прикажани и временските дијаграми на излезите и кај двете компоненти кои се побудени со идентични влезни сигнали.

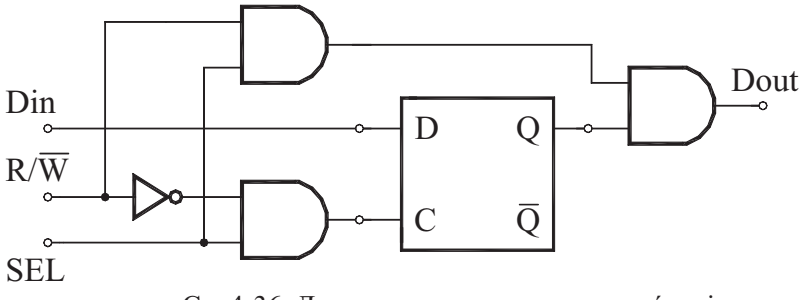
4.5.2. ОСНОВНА МЕМОРИСКА ЌЕЛИЈА

Од досега изложеното може да се претпостави на кој начин флип-флопот може да се употреби како елементарна мемориска ќелија (МС) која памти податок од еден бит. Постојат најразлични изведби на мемориски ќелии, но за сите нив е карактеристична појавата на повеќе влезови и најмалку еден излез.

За мемориската ќелија што овде ќе биде анализирана ќе претпоставиме дека таа има еден влез (Din) за запишување на содржина во неа (еднобитен податок), со другиот влез (SEL) се врши селекција на ќелијата, додека преку третиот влез ($R/\overline{W}$) се избира операцијата што таа ќе ја извршува: читање на memorираниот податок или запишување на нов. Од излезот на ќелијата (Dout) се врши читање на содржината (запамтениот податок).



Сл. 4-35. Логички симбол



Сл. 4-36. Логичка шема на мемориска ќелија

Симболичката ознака на ваквата мемориска ќелија е прикажана на сл. 4-35, додека логичката шема што неа ја реализира е дадена на сл. 4-36. Од нив се гледа дека централното место го зазема D лечот со неговиот влез за контрола C. Принципот според кој треба да работи мемориската ќелија ќе го опишеме во продолжение.

За да може да се работи со мемориската ќелија, поточно за да може да се чита битот (податокот) што е запамтен во неа или да се запише нов, таа треба да стане активна. За оставрување на оваа цел, мемориската ќелија треба да биде адресирана, а тоа се врши со доведување на сигнал со високо логичко ниво на адресната (селекционата) линија (SEL=1). Во овој случај постојат две можности:

- ЗАПИШУВАЊЕ на нов податок: Ако сигналот $R/\overline{W}$ се наоѓа на ниско ниво ($R/\overline{W}$ =0), тогаш преку инверторот се добива 1 која минува низ И колото и го активира сигналот за дозвола (C=1) со што податокот присутен на влезот Din се запишува во флип-флопот, а на излезот се добива нула (Dout=0);
- ЧИТАЊЕ на запамтен податок: Ако сигналот $R/\overline{W}$ се наоѓа на високо ниво ($R/\overline{W}$ =1), тогаш од една страна влезот за дозвола се наоѓа на ниско ниво (C=0) кое не дозволува нивото на линијата за податок Din да влијае врз состојбата на флип-флопот заради што тој ја задржува старата состојба, а од друга страна се отвора излезното И коло со што податокот (постоечката состојба на флип-флопот) се јавува на излезната линија (Dout=Q).

Бидејќи цело време не се работи само со една иста ќелија, туку и со другите, треба да постои можност мемориската ќелија да не биде селектирана. Во овој случај, кога ќелијата не е адресирана нивото на линијата за селекција е ниско (SEL=0), таа е пасивна и го задржува податокот што е внесен во неа. Кога ќелијата не е селектирана, влезот за дозвола оди исто така на нула (C=0) со што D флип-флопот ја задржува својата постоечка состојба и не ја менува, а истовремено на излезот се добива ниско ниво (Dout=0).

Презентираниот принцип на работа на момориската ќелија од сл. 4-35 покомпактно е претставен со функционалната табела 4-15.

Операција	Влезови			Излези	
	SEL	$R/\overline{W}$	Din	Q+	Dout
Пасивност	0	x	x	Q	0
Запишување	1	0	0 1	0 1	0
Читање	1	1	x	Q	Q

Таб. 4-15. Функционална табела на мемориска ќелија

Покрај ваквите асинхрони мемориски ќелии, постојат и синхрони чија работа е детерминирана со дополнителен такт сигнал кој ги контролира сите влезови и излези на ќелијата. Мемориската ќелија во овој случај се однесува идентично како и претходно, кога е асинхрона, само што сега читањето и запишувањето се вршат во точно определено време, со појавата на предниот или задниот раб на тактот.

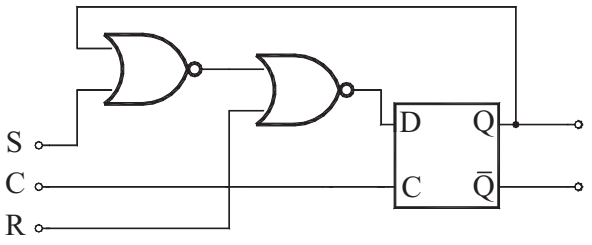
Реалните мемориски компоненти може да имаат мемориски ќелии со единствен влез за податок: во процесот на читање таа линија да е излезна, додека при запишување влезна. Покрај ова, кога мемориската ќелија не е адресирана, најчесто линиите за податок за наоѓаат во трета состојба. За да се добие вакво однесување треба да се изврши модификација на основната логичка структура на ќелијата и да се воведе баферско коло за што ќе зборуваме во темата за мемории која следува понатаму во учебникот.

4.5.3. ТРАНСФОРМАЦИЈА НА ЛОГИКАТА НА D ФЛИП-ФЛОПОТ

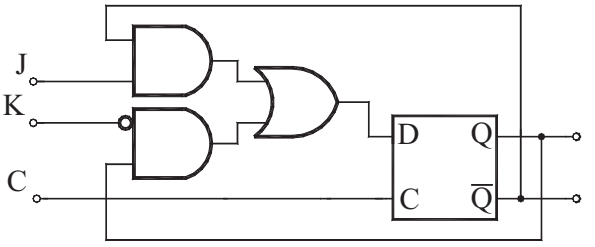
Веќе напоменавме дека D флип-флопот сè повеќе се користи за реализација на флип-флоповите со друга логика. Ова значи дека D флип-флопот, во некоја рака станува стандардна компонента за изградба на секвенцијалните мрежи. Затоа е корисно да се прикажат некои решенија на претходно анализираните типови флип-флопови изведени со примена на D флип-флопот. Еден од битните недостатоци на SR флип-флоповите секако е нивната недефинирана состојба за влезна комбинација R = 1 и S = 1. Меѓутоа, веќе рековме дека можат да се изградат SR флип-флопови со еден доминантен влез. Така на пример, за да се оствари SR флип-флоп со доминантен R влез со примена на D флип-флоп, треба да се употреби начинот на поврзување прикажан на сл. 4-36 а). Кај оваа конфигурација појавата на влезна комбинација R=1 и S=1 предизвикува ресетирање на излезот. Со ова е отстранета недефинираноста при побуда со R=1 и S=1 бидејќи флип-флопот се ресетира ($Q^+=0$)

На сл. 4-37 б) е прикажан начинот на поврзување на D флип- флопот, така што со него се реализира JK флип-флоп.

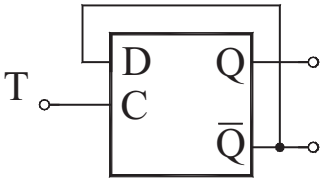
Т флип-флопот се применува и како тактиран, но често пати и како нетактиран. На сл. 4-37 в) е прикажана една изведба на нетактиран Т флип-флоп, додека на сл. 4-37 г) на тактиран Т флип-флоп. И за двете реализации е применет D флип-флоп.



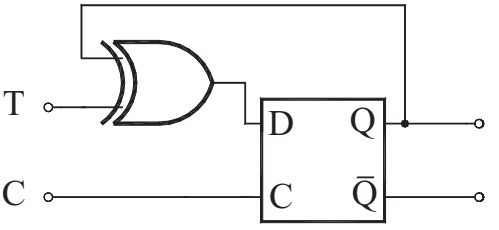
а) SR флип-флоп со доминантен R влез



б) Тактиран JK флип-флоп



в) Т флип-флоп



г) Тактиран Т флип-флоп

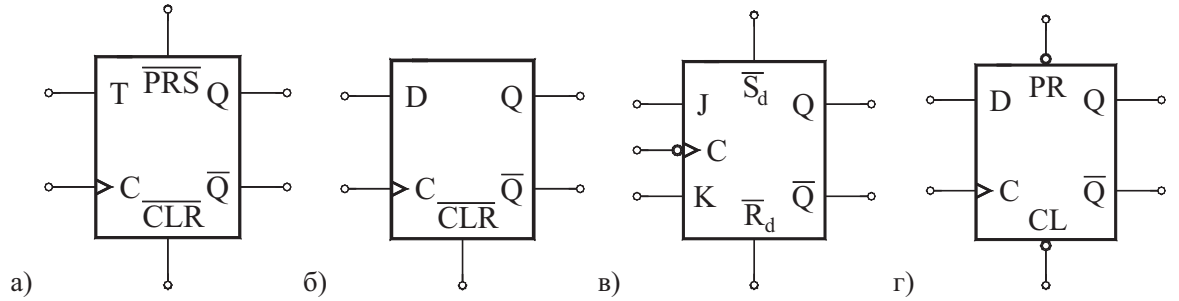
Сл. 4--37. Изведби на различни типови флип-флопови со примена на D флип-флоп

4.6. ИНТЕГРИРАНИ ФЛИП-ФЛОПОВИ

Како што може да се забележи од претходните примери, пооделни флип-флопови имаат доста комплексна конфигурација, иако се конструирани со помош на интегрирани логички кола. Напредокот на технологијата овозможи и многу комплексните структури на флип-флопови да се реализираат во облик на единствено монолитно интегрирано коло. Овие флип-флопови имаат подобри особини, а нивната примена во сложените системи е многу поедноставна земајќи го предвид фактот што сите врски помеѓу поединечните функционални елементи на флип-флопот веќе се изведени во внатрешноста на интегрираното коло. Дизајнерот нив ги третира како единствена функционална целина, водејќи сметка само за зависноста на промените на излезните состојби во функција од комбинацијата на влезните нивоа. Овие флип-флопови во шемите се прикажуваат на ист начин како и досега со квадрат на чија десна страна се означуваат излезите, а на левата, горната и долната страна влезовите на флип-флопот. Вообичаено е лево да се означуваат синхроните влезови и тактот, додека горе и долу директните (асинхроните) влезови.

На сл. 4-38 а), б), в) и г) се прикажани симболи на неколку често применувани интегрирани флип-флопови. Во праксата се сретнуваат и означувања на асинхроните влезови со мал круг (“○”). Ова значи дека ефективното дејство врз состојбата на флип-флопот, а со тоа и на неговиот излез, ќе се реализира ако на него се донесе ниско ниво (0).

Кругот на влезот за такт-сигнал С означува дека промената на состојбата на флип-флопот ќе се врши со појавата на задниот раб на такт-сигналот, но ако тој е испуштен промената ќе настапи со појавата на неговиот преден раб. Кога на влезот за такт постои триаголник (Δ) тоа покажува дека се работи за флип-флоп кој е тактиран (управуван) со предниот раб на тактот ако нема мал круг, односно со задниот раб ако кругчето постои.



Сл. 4-38. Симболи на интегрирани флип-флопови

Кај флип-флоповите изведени во интегрирана техника не мора секогаш да постојат двата асинхрони влезови, ниту пак двата меѓусебно комплементарни излеза, туку само еден од нив.

Технологијата на интегрирани кола овозможи производство на такви интегрирани кола кои во себе содржат повеќе независни флип-флопови. Така на пример, во TTL и CMOS технологиите, чии ознаки се 74xx, односно 40xx, се сретнуваат најразлични изведби на двократни или четирикратни флип-флопови интегрирани во една компонента.

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

4-1. Што претставува флип-флопот?

4-2. Колку излези има флип-флопот, како се викаат и како се означуваат?

4-3. Кога флип-флопот е сетиран, а кога ресетиран?

4-4. Флип-флопот, во општ случај, има ... влезови такви што некои од нив се ..., а некои се ...

4-5. Што се носи на податочните влезови?

4-6. Зошто служат контролните влезови?

4-7. Од што зависи состојбата на излезите кај асинхроните флип-флопови?

4-8. Од што зависи состојбата на излезите кај синхроните флип-флопови?

4-9. Што претставува такт сигналот?

4-10. Што се случува кога ќе се појави активно ниво, односно активен раб на тактот?

4-11. Што е време на доцнење?

4-12. Опиши го начинот на работа на флип-флопот!

4-13. Од што зависи состојбата на излезите од флип-флопот? Зошто се нарекува елементарна мемориска компонента?

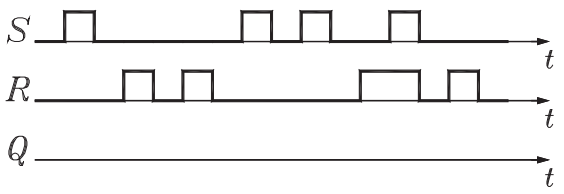
4-14. Нацртај ја логичката структура и логичкиот симбол на SR флип-флоп (флип-флоп од НИЛИ тип).

4-15. Нацртај ја табелата на премин и излез, табелата на екситација и напиши ја карактеристичната равенка на SR флип-флопот.

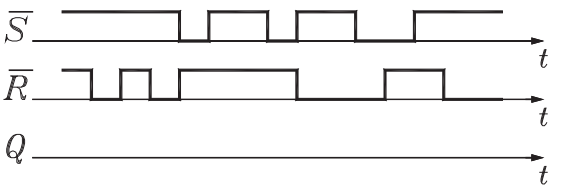
4-16. Кога SR флип-флопот не ја менува својата состојба? Што треба да се направи за тој да се постави на 1, а што за да се ресетира?

4-17. Кој е основниот недостаток на SR флип-флопот и како тоа аналитички се искажува?

4-18. На следната слика се прикажани временските дијаграми на влезните сигнали S и R кај SR флип-флопот. Нацртај го временскиот дијаграм на излезот Q, ако почетната состојба била (а) Q = 0; (б) Q = 1.



Слика за прашање 4-18



Слика за прашање 4-2

4-29. Нацртај ја логичката структура, логичкиот симбол и табелата на премин за $\overline{S}\overline{R}$ флип-флоп (флип-флоп од НИ тип).

4-20. Дали постои нелегитимна (забранета) влезна комбинација за $\overline{S}\overline{R}$ флип-флопот и, ако постои, која е таа?

4-21. На следната слика се прикажани временските дијаграми на влезните сигнали $\overline{S}$ и $\overline{R}$ кај $\overline{S}\ \overline{R}$ флип-флопот. Нацртај го временскиот дијаграм на излезот Q, ако почетната состојба на излезот била (а) Q = 0 (б) Q = 1

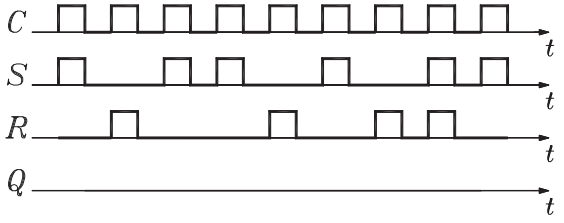
4-22. Нацртај ја логичката структура и симболичката ознака на тактиран SR флип-флоп!

4-23. Нацртај ја табелата на премин и излез, табелата на екситација и напиши ја карактеристичната равенка за тактиран SR флип-флоп.

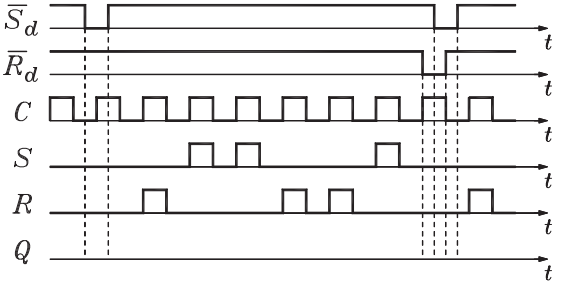
4-24. Кога сигналите доведени на S и R влезовите кај тактираниот SR флип-флоп имаат ефективно дејство врз неговата состојба?

4-25. Какви треба да бидат нивоата на сигналите доведени на влезовите S и R за правилна работа на тактираниот SR флип-флоп? Што ќе биде во спротивниот случај?

4-26. На сликата се прикажани временските дијаграми на тактот C и на влезните сигнали S и R кај тактираниот SR флип-флоп. Нацртај го временскиот дијаграм на излезот Q ако тој почетно се наоѓал на ниско ниво.



Слика за прашање 4-26



Слика за прашање 4-29

4-27. Нацртај ја логичката структура и симболичката ознака на тактиран SR флип-флоп со асинхрони (директни) влезови активни на ниско ниво $\overline{S_d}$ и $\overline{R}_d$ кои доминираат над тактот C.

4-28. Која е улогата на директните влезови кај тактираниот флип- флоп? Кој е условот што треба да биде исполнет помеѓу директните влезови за флип-флопот да работи исправно?

4-29. На сликата се прикажани временските дијаграми на сигналите доведени на асинхроните влезови $\overline{S_d}$ и $\overline{R}_d$, на тактот C, и на податочните влезови S и R кај тактираниот SR флип-флоп. Нацртај го временскиот дијаграм на излезот Q ако почетно тој се наоѓал на ниско ниво.

4-30. Објасни ја разликата помеѓу флип-флоповите што реагираат на нивото од тактот и оние што реагираат на појавата на неговиот раб?

4-31. Нацртај ја логичката структура и логичкиот симбол на MS SR флип-флоп.

4-32. Која е разликата помеѓу тактираниот SR флип-флоп (лечот) и MS SR флип-флопот?

4-33. Која е разликата помеѓу MS SR флип-флопот и тактираниот SR флип-флоп управуван со работ на такт сигналот?

4-34. Какви треба да бидат податоците присутни на податочните влезови S и R за правилна работа на MS SR флип- флопот?

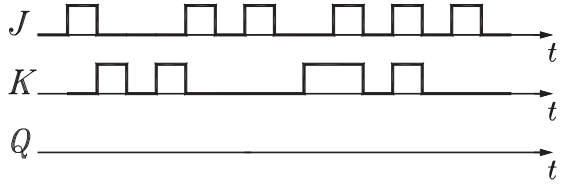
4-35. Нацртај го логичкиот симбол на MS SR флип-флоп со асинхрони влезови активни на (а) ниско (б) високо ниво.

4-36. Нацртај ја логичката шема и логичкиот симбол на (а) асинхрон (б) синхрон JK флип-флоп.

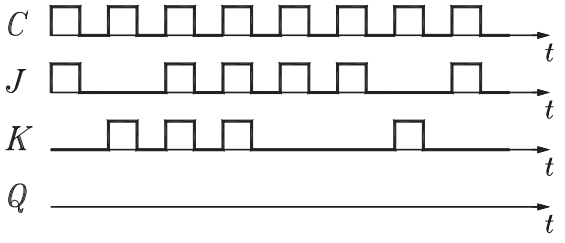
4-37. Нацртај ја табелата на премин, табелата на побуда и карактеристичната равенка на (а) асинхрон (б) синхрон JK флип-флоп.

4-38. Која е разликата во однесувањето помеѓу SR и JK флип-флопот?

4-39. На сликата се прикажани временските дијаграми на влезните сигнали J и K кај JK флип-флопот. Нацртај го временскиот дијаграм на излезот Q ако тој почетно се наоѓал на ниско ниво.



Слика за прашање 4-39



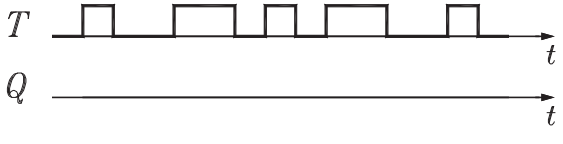
Слика за прашање 4-40

4-40. На следната слика се прикажани временските дијаграми на тактот C и на влезните сигнали J и K кај тактираниот JK флип-флоп. Нацртај го временскиот дијаграм на излезот Q ако тој почетно се наоѓал на ниско ниво.

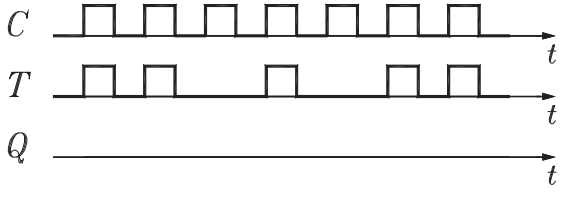
4-41. Нацртај ја логичката шема и логичкиот симбол на (а) асинхрон (б) синхрон T флип-флоп.

4-42. Нацртај ја табелата на премин и излез, табелата на побуда и карактеристичната равенка на (а) асинхрон (б) синхрон T флип-флоп!

4-43. На сликата е прикажан временскиот дијаграм на сигналот доведен на T влезот кај T флип-флоп. Нацртај го временскиот дијаграм на излезот Q ако неговата почетна состојба била Q=0.



Слика за прашање 4-43



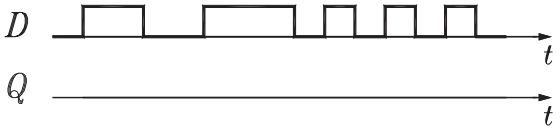
Слика за прашање 4-44

4-44. На сликата се прикажани временските дијаграми на сигналите доведени на тактот C и на податочниот влез T кај T флип-флоп. Нацртај го временскиот дијаграм на излезот Q ако неговата почетна состојба била Q = 0.

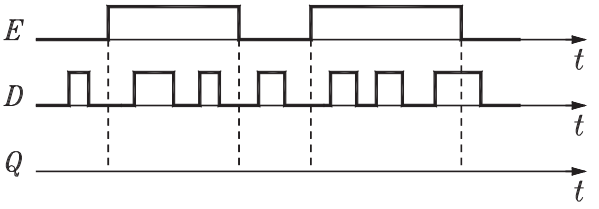
4-45. Нацртај ја логичката шема и логичкиот симбол на (а) асинхрон (б) синхрон D флип-флоп.

4-46. Нацртај ја табелата на премин и излез, табелата на побуда и карактеристичната равенка на (а) асинхрон (б) синхрон D флип-флоп.

4-47. На следната слика е прикажан временскиот дијаграм на сигналот доведен на влезот за податок D кај D флип-флопот. Нацртај го временскиот дијаграм на излезот Q ако неговата почетна состојба била Q = 0.



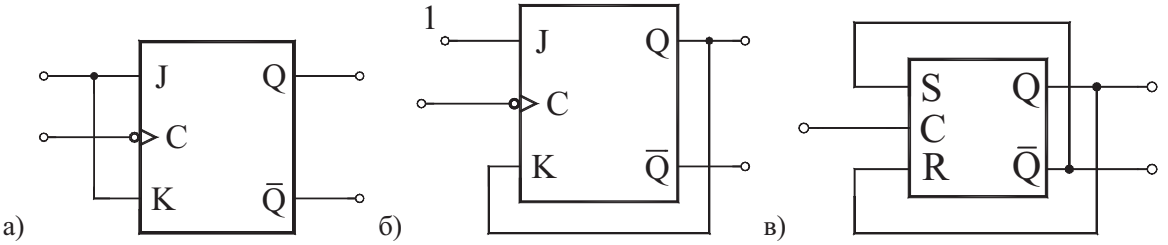
Слика за прашање 4-47



Слика за прашање 4-49

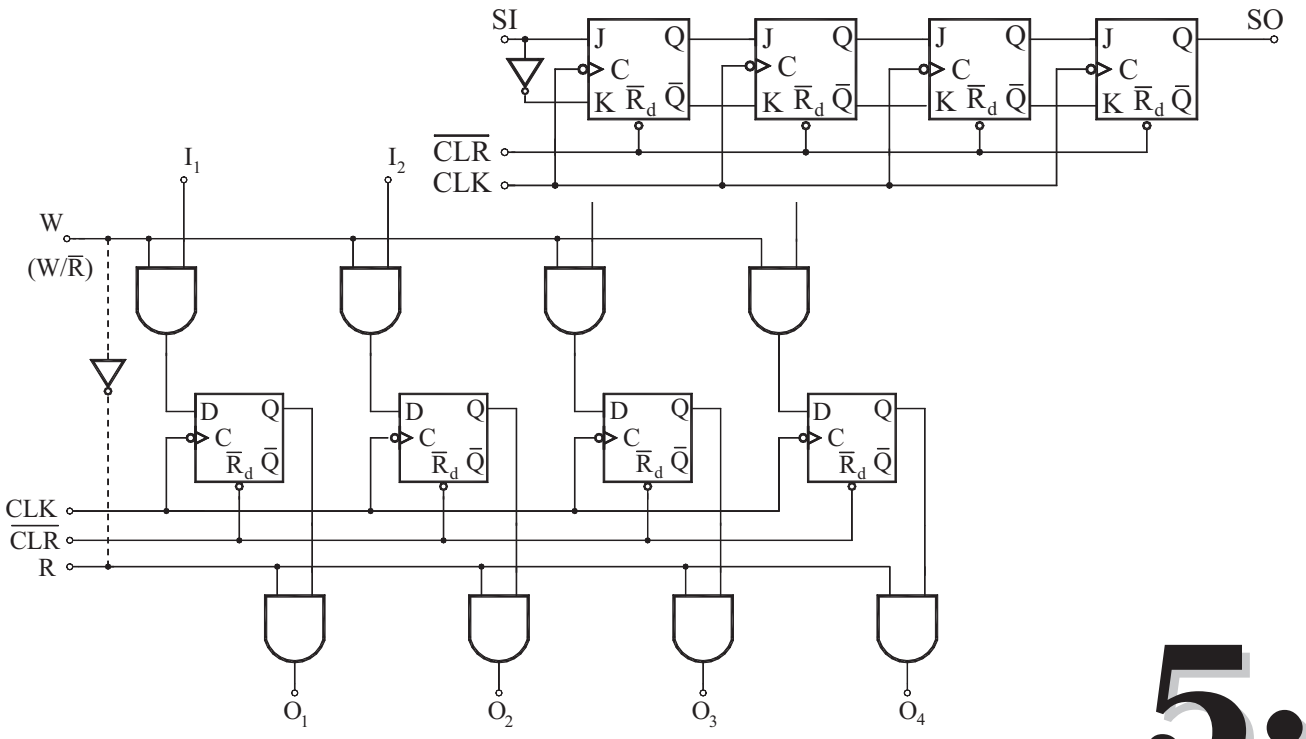
4-48. Објасни ја разликата помеѓу D флип-флопот и D лечот (колото за заклучување).

4-49. На следната слика се прикажани временскиот облик на сигналите доведени на податочниот влез D и влезот за контрола E кај D лечот (колото за заклучување). Нацртај го брановиот облик на излезот Q, ако тој почетно се наоѓал на ниско ниво. Нацртај го брановиот облик на излезот Q со претпоставка дека истите побудни сигнали се носат на D флип-флоп кој е управуван а) со предниот б) задниот раб на тактот.



Слики за прашање 4-50

4-50. Одреди какви типови на флип-флопови се реализираат со логичките шеми прикажани на следните слики.



5. РЕГИСТРИ

По изучувањето на оваа тематска целина

- Ќе ја сфатите логичката структура на регистрите;
- Ќе го разберете и ќе можете да го опишете принципот на работа и примената на стандардните регистри:
  - стационарен регистер,
  - поместувачки регистер,
  - регистер со комбиниран влез,
  - регистер со комбиниран излез,
  - универзален регистер;
- Ќе ги разликувате типовите на регистри според функцијата и примената;

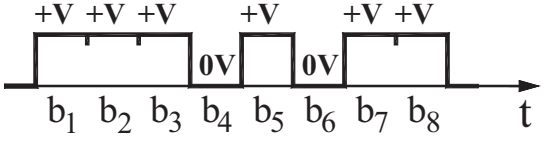
5.1. ВОВЕД И ОСНОВНИ ПОИМИ И КОНЦЕПТИ

Регистрите се сложени секвенцијални мрежи и истите многу често се користат при градбата на дигиталните уреди. **Регистерот** претставува склоп кој служи за меморирање на произволен бинарен податок со ограничена должина што се вика збор. Бидејќи за помнење на секој бит од зборот треба по еден бистабилен елемент, т.е. флип-флоп, следува дека за збор со должина од n-бита ќе бидат потребни n-флип-флопови.

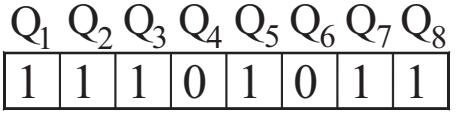
Регистрите се користат како акумулатори за привремено запомнување, односно прифаќање на влезните податоци, меѓурезултатите или конечните резултати во процесот на обработката на податоци, потоа неопходни се како бафери на местата каде е потребна врска на делови од дигиталниот уред кои работат со различни брзини. Исто така, тие се користат при реализирањето на аритметичките и логичките операции: комплементирање, специјални случаи на множење и делење, потоа во преносот на информации како конвертори за претворување на податоците од паралелен во сериски облик и обратно, итн.

На сл. 5-1 е прикажан брановиот облик на напонот кај некој дигитален склоп. Станува збор за сигнал кој претставува одреден 8-битен податок, во примеров 11101011. Напонското ниво на логичката 1 е високо V(1)=+Vcc, додека на логичката нула е ниско V(0)=0V. На сл. 5-2 дадена е една многу едноставна блок-шема за регистер со 8 флип-флопови во кој може да се смести дадениот податок. Бидејќи податокот ќе се запомни во регистерот, истиот ја претставува и неговата **содржина**. За нашиот пример, содржината на регистерот ќе биде 11101011.

За влез на податокот во регистерот се користи терминот **полнење, внесување** или **сместување** (анг. *load*), или поретко **запишување** (анг. *write*) во регистер, додека за излез на податокот од регистерот се користи терминот **читање** (анг. *read*) на содржината.



Сл. 5-1. Осум битен податок



Сл. 5-2. Блок шема на 8-битен регистер

При запишувањето во регистерот претходната (старата) содржина се губи, а се запомнува новиот податок доведен на влезните податочни линии кои се и влез во флип-флоповите од кои е составен регистерот. За разлика од ова, најчесто, содржината на регистерот може да се чита повеќе пати затоа што при читањето не доаѓа до промена на податокот што е запишан во него. Ваквото читање не е деструктивно. Меѓутоа, постои и деструктивно читање кога запишаната содржина се губи, заради што прочитаниот податок мора да се обновува, т.е. повторно да се полни во регистерот. При читањето, содржината на регистерот се појавува на излезните податочни линии што всушност се излези од неговите флип-флопови. Состојбата на регистерот се претставува со вредноста на излезите од секој поединечен флип-флоп, во дефиниран временски период. Затоа, пред да почне со работа, во регистерот се одредува почетната состојба, така што сите мемориски елементи се доведуваат во нулта состојба, т.е. во регистерот се запишуваат сите нули. Оваа операција се вика **бришење** или **чистење** (анг. *clear*) на регистерот. Понекогаш регистерот се дизајнира така што почетната состојба е различна од нула, односно во почетниот момент во некои флип-флопови се запишуваат единици. Оваа операција се вика почетно **поставување** (анг. *preset*) на регистерот.

Бришењето и поставувањето, како и некои други операции што можат да се извршуваат на содржината внесена во регистерот, како на пример полнење или читање, се управуваат преку посебни контролни линии. Кога регистерот работи синхроно, како посебен влез ќе биде присутен и влезот за тактирање, при што сите флип-флопови што влегуваат во составот на регистерот се тактирани од истиот такт сигнал. Јасно е дека покрај контролните, регистерот има и влезни, односно излезни податочни (информациони) линии преку кои тој се полни, односно се чита неговата содржина (податокот што е внесен во него).

Имајќи го предвид начинот на читање на содржината на регистерот, односно начинот на кој се запишува новата информација во него, постојат два начина на читање или запишување на содржината:

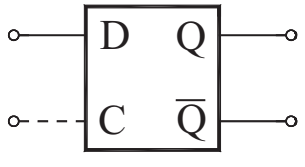
1. **Паралелно (просторно)**. сите битови се запишуваат или читаат истовремено, во еден временски интервал, и

2. **Сериски (временски).** битовите се запишуваат или читаат редоследно еден по еден (бит по бит), за секој бит по еден временски интервал, и тоа почнувајќи од првиот или последниот.

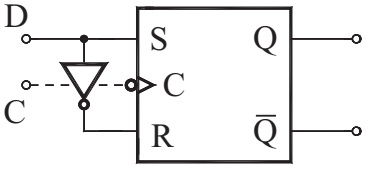
Со комбинација на опишаните начини за влез и излез на податоците се добиваат повеќе различни типа на регистри:

1. Регистри со паралелен влез и паралелен излез;
2. Регистри со сериски влез и сериски излез;
3. Регистри со сериски влез и паралелен излез;
4. Регистри со паралелен влез и сериски излез и
5. Регистри со комбинирани можности за запис и/или читање.

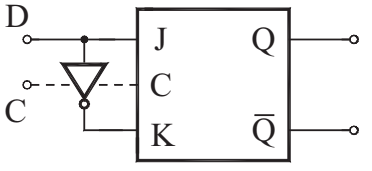
Првиот тип регистер е **стационарен,** бидејќи податокот што се запишува во регистерот останува запомнет во појдовниот облик како содржина на регистерот која не се менува. Кај другите три типа регистри содржината на регистерот т.е. запишаниот податок постојано се поместува по еден бит на лево или на десно, па затоа овие регистри се викаат **поместувачки** или **динамички** (анг. *shift*) регистри. Во практиката, често пати влезот или излезот на регистрите се изведува комбинирано, така што е можен сериски или паралелен влез, и/или сериски или паралелен излез.



Сл. 5-3. D флип-флоп



а) реализиран со SR



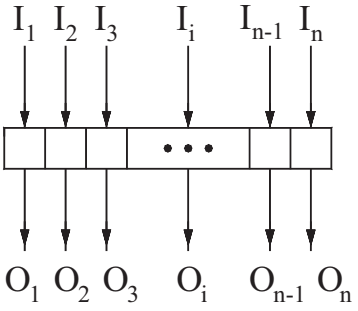
б) реализиран со ЈК флип-флоп

Сл. 5-4. D флип-флоп

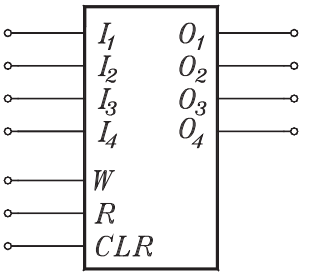
Бидејќи D флип-флопот претставува елементарна секвенцијална компонента која според начинот на работа памти податок со должина од еден бит, за градба на регистрите се користат поголем број на тактирани D флип-флопови според сл. 5-3 соодветно на должината на податокот за кој станува збор, изразена во битови. Употребените флип-флопови најчесто имаат и асинхрони влезови со кои се поставува почетна положба на регистрите или се користат заради проширување на можностите и за добивање на комбинирани типови регистри. Во практичните реализации на регистри ќе сретнуваме и логички структури кои како основни елементи користат SR или ЈК флип-флопови, но тоа не треба да нè зачудува бидејќи секогаш таквите флип-флопови ќе бидат поврзани во конфигурации што реализираат D флип-флоп според сл. 5-4 а) или б).

5.1. СТАЦИОНАРЕН РЕГИСТЕР

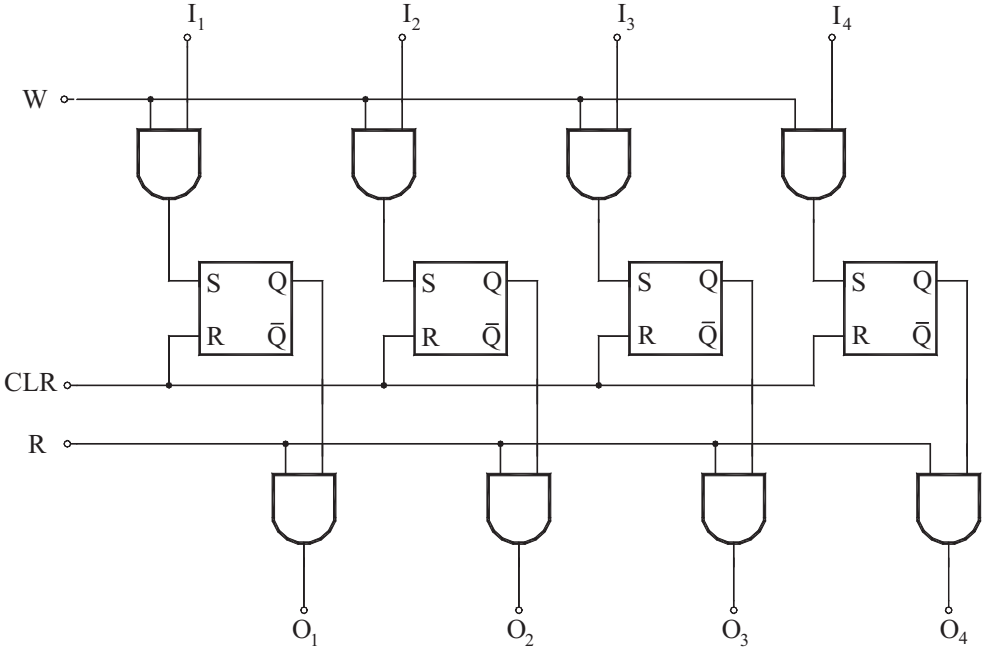
Стационарниот регистер има паралелен влез и паралелен излез на податоците како што е и прикажано на едноставната блок-шема од сл. 5-5 а). На сл. 5-5 б) е прикажана една реализација на четирибитен стационарен регистер со употреба на четири SR флип-флопови означени како FF_i, каде i=1,2,3,4. Прикажаниот регистер има осум податочни (информациони) линии, од кои четири се влезни: I_1, I_2, I_3, и I_4, додека четири се излезни: O_1, O_2, O_3 и O_4. Покрај нив постојат и три контролни (управувачки) линии: за бришење на регистерот CRL, потоа за запишување на нова содржина W, како и линија за читање R. Логичкиот симбол на ваквиот тип регистер е прикажана на сл. 5-5 в).



а) Едноставна блок-шема



б) Симболичка ознака



в) Логичка шема

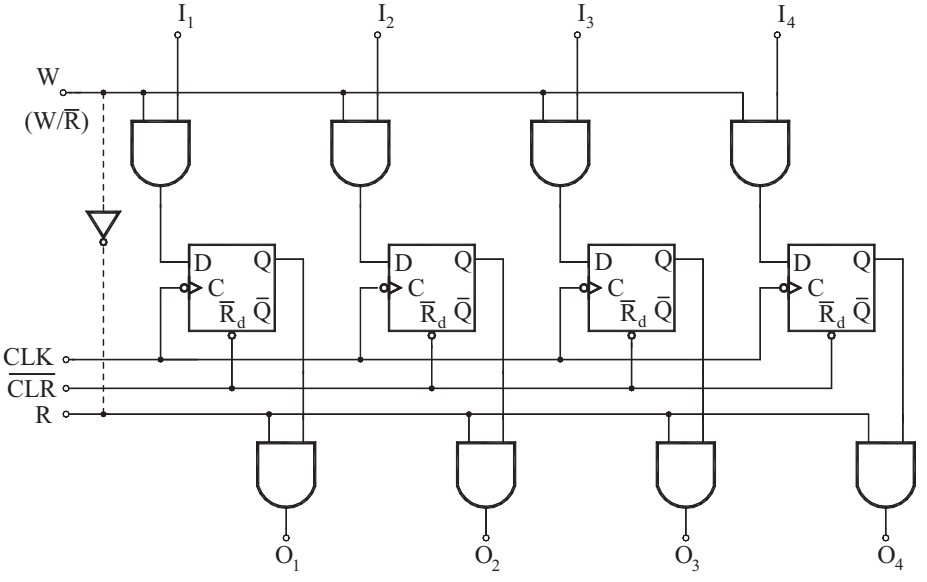
Сл. 5-5. Четирибитен стационарен регистер со SR флип-флопови

Регистерот се доведува во почетна состојба на тој начин што преку влезот за бришење CRL се доведува 1, т.е. високо напонско ниво: (CRL=1). Со ова флип-флоповите ќе се избришат бидејќи сите нивни влезови за ресетирање се активни R_i=1 (R_1=R_2=R_3=R_4=1), така што неговата почетна состојба ќе биде: Q_1=0, Q_2=0, Q_3=0 и Q_4=0.

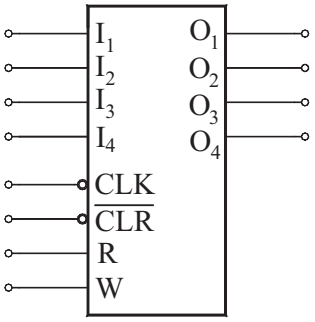
Запишувањето во регистерот може да се изведе кога сите битови од податокот ќе се доведат на влезовите I_1, I_2, I_3 и I_4, а потоа на линијата W се носи високо ниво (W=1) со што истовремено се „отвораат" сите И кола и секој бит се запишува во соодветниот флип-флоп: $S_i=I_i$. Практично, состојбата ја менуваат само оние флип-флопови на чии влезови се јавува 1, додека таму каде што влезот е на 0, флип-флопот ја задржува својата претходна состојба. Заради ова, пред секое полнење на регистерот тој мора претходно да се избрише со доведување на високо ниво на влезот за бришење CRL. Запишаниот податок останува мемориран во регистерот како негова содржина и тоа сè додека не се појави друг податок кој ќе треба да се внесе на влезните информациони линии I_i.

За да се прочита содржината на регистерот, потребно е на влезот R да се донесе 1, (R=1) со што истовремено ќе се отворат сите излезни И кола: $O_i=Q_i$. Со ова, состојбата на излезот од секој флип-флоп ќе се појави на излезот од регистерот Q_0, Q_1, Q_2, Q_3, што значи се добива податокот што е запомнет во регистерот без тој да се промени. Вака содржината на регистерот може да се отчитува повеќе пати, а притоа таа да не се менува. Бидејќи кај ваквите регистри внесената содржина не се поместува односно не се движи, тие се викаат стационарни. Да не заборавиме дека за правилна работа на регистерот даден на сл. 5-5 а) не е дозволено истовремено W=1, R=1, или W=1 CRL=1,или R=1, CRL=0 кога се запишува W=1, R=0, CRl=0, и обратно, ако се чита R=1, W=0, CRL=0, додека ако содржината на регистерот се брише треба CRL=1, W=0, R=0.

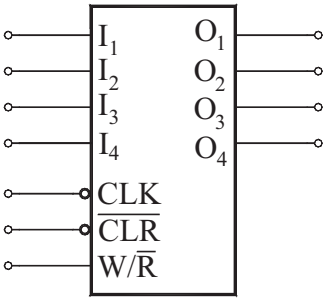
Времето потребно за внесување нова информација во регистерот може да се скрати ако тој се изведе на таков начин, што ќе овозможи да биде избегната потребата од претходно бришење на неговата моментална содржина. Логичката шема на еден ваков регистер е прикажана на сл. 5-6 а). Тој е изведен со D флип-флопови, така што било кое логичко ниво на податокот доведено на секој од D влезовите директно се внесува. Покрај ова, работата на регистерот е синхрона со задниот раб на тактните импулси CLK бидејќи се применети тактирани D флип-флопови. За регистерот е карактеристично и појавувањето на директни влезови за ресетирање R_d кои се користат за бришење на неговата содржина. Тоа е овозможено со нивно заедничко врзување на една линија означена со $\overline{CRL}$ која ќе биде активна на ниско ниво, т.е. регистерот се брише ако на неа се донесе 0 ($\overline{CRL}$ =0).



Сл. 5-6 а) логичка шема на четирибитен стационарен регистер со тактирани D флип-флопови



б) симбол со два контролни сигнали



в) симбол со еден контролен сигнал

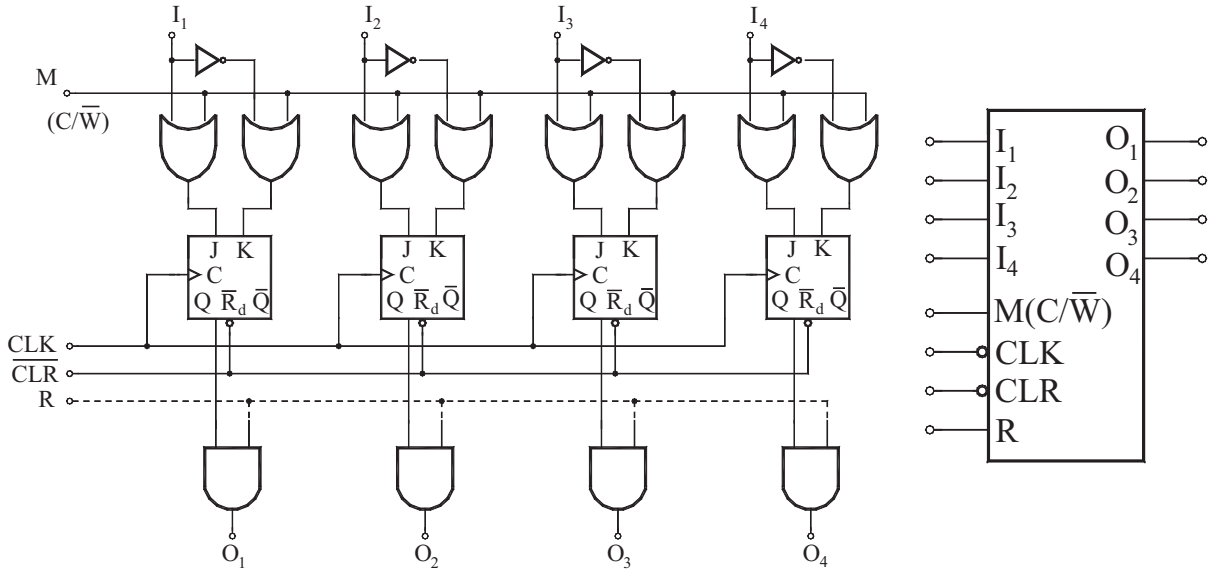
Сл. 5-6. Четирибитен стационарен регистер со тактирани D флип-флопови

За правилна работа на регистерот, мора да биде исполнет условот за меѓусебна комплементарност на состојбите на контролните линии за внесување (полнење, запишување) W и за читање R: W·R=0, т.е. високо ниво не смее да се наоѓа истовремено на двете линии.

Процесот на запишување или читање, може да се поедностави ако се користи само една контролна линија и за читање и за запишување, како што е прикажано на сл. 5-5 а) со испрекинати линии. Сега линијата за читање R се зема преку инвертор од линијата за запис W, така што секогаш ќе биде исполнет условот за комплементарност помеѓу состојбите на линиите W, R. Оваа единствена влезна линија ќе може да се означи со ($W / \overline{R}$), бидејќи ако на неа се донесе високо ниво овозможено е запишување, а кога ќе се донесе ниско ниво овозможено е читање. Имено, ако $W / \overline{R}$=1, тогаш е овозможено внесување на нова содржина во регистерот, бидејќи влезните И кола ги пропуштаат битовите на податокот $D_i=I_i$, со што $Q_i^+=D_i$, при што на излезите од регистерот се добиваат само нули $O_i=0$. Тоа е така бидејќи на секое излезно И коло има по една нула која произлегува од комплементирањето на 1 што е присутна на контролната линија за читање/запис ($W / \overline{R}$). Од друга страна, ако $W / \overline{R}$=0 практично се врши читање на податокот што се наоѓа во регистерот. Меѓутоа, сега читањето е деструктивно, т.е. со читањето автоматски се брише содржината на регистерот затоа што на сите влезови од флип-флоповите се јавуваат нули $D_i=0$, па јасно е дека $Q_i^+=0$.

Симболичката ознака на регистерот изведен со две контролни линии за читање и запишување прикажана е на сл. 5-6 б), додека симболот на реализацијата со само една контролна линија за читање или внесување е претставен на сл. 5-6 в).

На сл. 5-7 а) е прикажана логичката структура на уште еден стационарен регистер реализиран со JK флип-флопови, кој работи синхроно, а чија симболичка ознака е дадена на сл. 5-7 б). И кај оваа изведба нема потреба од бришење на содржината на регистерот пред внесувањето на нов податок бидејќи се користат JK флип-флопови поврзани како D флип-флопови. Покрај ова, преку линијата за контрола M на начинот (режимот) на работа (анг. *mode*), може да се внесува новиот податок ако M=0, односно да се комплементира содржината што се наоѓа во регистерот ако M=1. Во првиот случај, кога M=0, $J_i=I_i$, $K_i=T_i$, така што за наредниот интервал ќе важи $Q_i^+ =I_i$. Ако, пак, на линијата M донесеме високо ниво M=1, тогаш $J_i=K_i=1$, така што наредната состојба на излезите од сите флип-флопови ќе биде комплемент на сегашната: $Q_i^+ =Q_i$. Кај вака реализираниот регистер, неговата содржина е присутна на излезните линии веднаш по внесувањето на новиот податок или по комплементирањето. Меѓутоа, со додавање на уште една линија за читање R, како што е прикажано на сл. 5-7 а) со испрекинати линии, може да се врши контрола и на моментот на читање со донесување на високо ниво на влезот за читање R, (R=1) бидејќи тогаш $O_i=Q_i$. Од друга страна, ако R=0, тогаш сите излези од регистерот се 0-и: $O_i=0$.



а) логичка структура б) симбол

Сл. 5-7. Четирибитен стационарен регистер со тактирани со JK флип-флопови

И кај овој регистер се користат флип-флопови со асинхрони влезови за ресетирање R_d со кои се брише неговата содржина. Тоа се изведува со нивно поврзување во единствена линија, означена со CRL која ќе биде активна на ниско ниво (регистерот се брише ако е исполнет условот CRL=0).

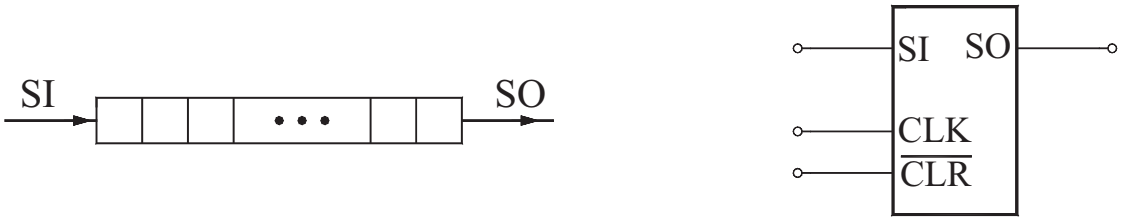
За стационарните регистри реализирани со тактирани флип-флопови може да се употребат флип-флопови кои реагираат на предниот раб од тактот *CLK*, како и флип-флопови од *MS* структура кои реагираат на појавата на задниот раб од такт-сигналот.

5.2. ПОМЕСТУВАЧКИ РЕГИСТЕР

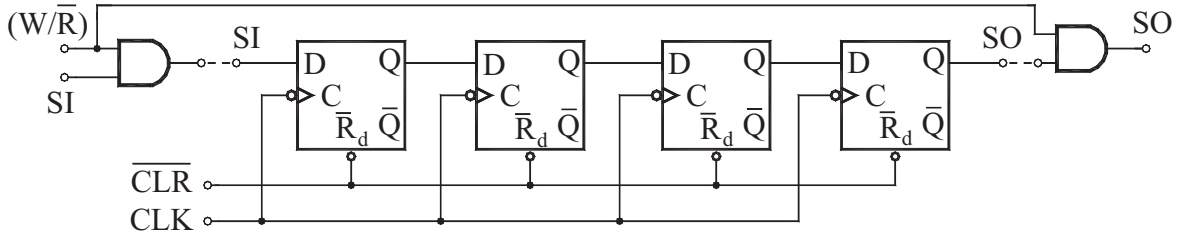
За D флип-флопот знаеме дека веднаш по појавата на активниот (прекинувачкиот) раб (премин) од такт-сигналот на неговиот влез, излезот Q од флип-флопот ќе ја добие онаа состојба, која беше присутна на влезот D непосредно пред да се појави активниот раб: $Q^+=D$. Токму заради ова, со примена на D флип-флопови може да се оствари поместувачки (анг. *shift*) регистер со сериски влез и сериски излез на неговата содржина, како што е претставено на едноставната блок-шема дадена на сл. 5-8 а). Имено, **поместувачкиот регистер** во принцип претставува последователна врска (каскада) од неколку D флип-флопови.

Влезниот податок се појавува на влезот од првиот флип-флоп, а на неговиот излез ќе биде задоцнет за еден временски интервал кој е еднаков со периодот на такт-сигналот T_P. На излезот од вториот флип-флоп тој ќе се појави подоцна за уште еден такт-интервал, итн. така што доаѓа до сукцесивно поместување кон десно на информацијата низ регистерот: $D_i=Q_{i-1}$ и $Q_i^+ = Q_{i-1}$, или $D_{i+1}= Q_i$ и $Q_{i+1}^+ =Q_i$. Така без оглед на тоа каква низа од битови ќе се појави на влезот од регистерот, со секој такт импулс таа група ќе биде поместувана за еден флип-флоп надесно. Оттука овие регистри и го добиле своето име.

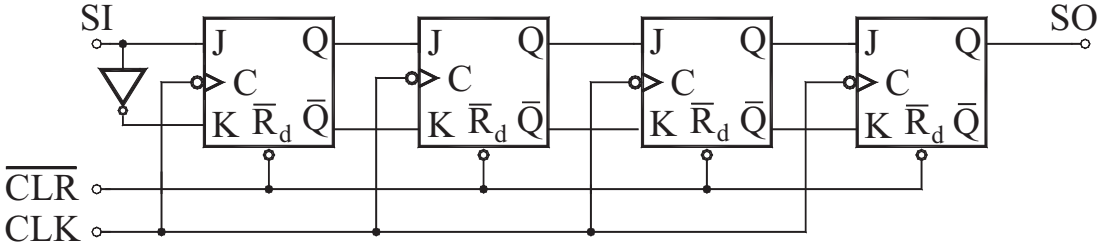
Во современите конструкции на регистри, при нивната изведба се користат тактирани флип-флопови активни на предниот или задниот раб од тактот или се реализираат со примена на *MS* флип-флопови. Еден таков четирибитен поместувачки регистер кој функционира со задниот раб на тактните импулси е прикажан во две изведби на сл. 5-8 в) и г). Покрај стандардната изведба со *D* флип-флопови (в), употребени се и JK флип-флопови поврзани како D на познатиот начин со меѓусебно комплементирање на J и K влезовите (г). Симболот на поместувачкиот регистер е прикажан на сл. 5-8 (б).



а) едноставна блок-шема б) симболичка ознака

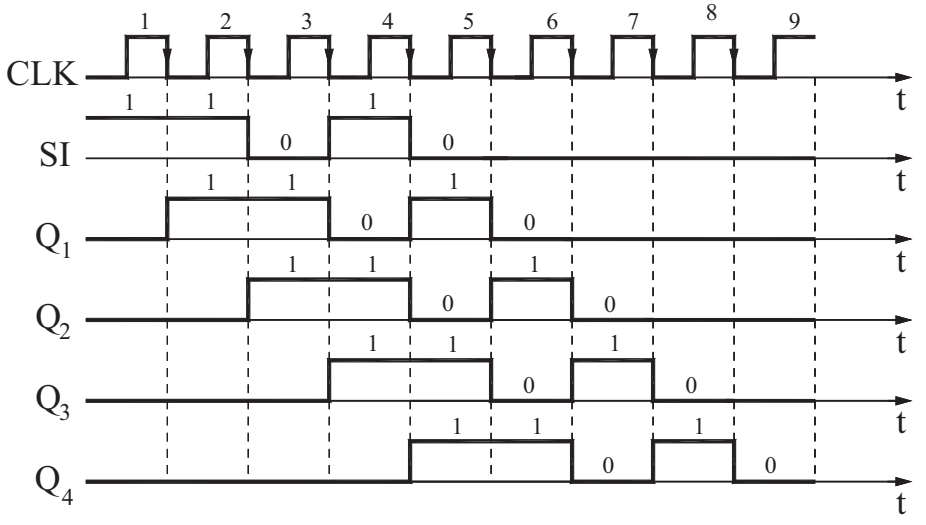


в) логичка шема со D флип-флопови



г) логичка шема со JK флип-флопови

Сл. 5-8. Четирибитен поместувачки регистер



Сл. 5-9. Временски дијаграми на поместувачки регистер

Пред почетокот со работа на асинхрониот влез CRL се носи ниско ниво (логичка 0), со што се брише претходната содржина на регистерот и тој се иницијализира, се доведува на почетната, нултата состојба. Работата на овој регистер наједноставно може да се опише со помош на временски дијаграми, како и со табелата на премин. Во неа се запишува состојбата на секој флип-флоп од компонентата за секој поединечен такт-интервал.

За дополнително објаснување и подобро разбирање на начинот на работа на овој регистер ќе претпоставиме едноставен пример, а тоа е запишување на податокот 11010 во регистерот, којшто се јавува во t=0. Временските дијаграми во карактеристичните точки се прикажани на сл. 5-9. За почетната состојба на регистерот, пред доаѓањето на првиот такт-интервал, поточно првиот опаѓачки (заден) раб на такт сигналот, ќе претпоставиме дека сите флип-флопови се ресетирани, односно содржината во нив е 0 (Q_1=0, …, Q_4=0).

Бидејќи од почетниот момент на разгледување t=0, на влезот од првиот флип-флоп се појавува влезен податок со високо ниво (логичка 1), јасно е дека првиот флип-флоп ќе се сетира со доаѓањето на првиот опаѓачки раб. Сите други флип-флопови на своите влезови имаат ниско ниво, т.е. 0, па тие ќе останат и натаму ресетирани. Веднаш по овој активен премин на влезот од вториот флип-флоп се јавува преден раб на импулс, премин од 0 на 1, бидејќи тоа е излезот од првиот флип-флоп, но со тоа не се сетира вториот флип-флоп бидејќи тој не реагира на преден раб, туку "го чека" задниот. Влезовите на сите останати флип-флопови се наоѓаат на ниско ниво.

CLK	SI (D_i)	$Q_1Q_2Q_3Q_4$
1	**1**	0 0 0 0
2	**1**	**1** 0 0 0
3	**0**	**1 1** 0 0
4	**1**	**0 1 1** 0
1	0	**1 0 1 1**
6	0	0 **1 0 1**
7	0	0 0 **1 0**
8	0	0 0 0 **1**
9	0	0 0 0 0
10	0	0 0 0 0

Таб. 5-1. Табела на премин на состојби кај поместувачки регистер

При појавувањето на вториот прекинувачки раб доаѓа до сетирање и на првиот и на вториот флип-флоп. Кај првиот флип-флоп тоа е бидејќи и вториот бит од податокот е 1, додека кај вториот флип-флоп тоа е затоа што неговиот влез е фактички излезот од првиот флип-флоп, кој во претходниот интервал отиде на високо ниво. Излезите Q_3, Q_4 од третиот и четвртиот флип-флоп и натаму остануваат на 0. Со секој наредeн активен премин на тактот, податокот „влегува" во регистерот. Очигледно е дека по четвртиот такт-импулс на излезот од првиот флип-флоп од регистерот ќе се појави четвртиот бит од податокот (низ овој флип-флоп „поминале" првите три битови од податокот и се јавува четвртиот), додека на излезот од четвртиот флип-флоп, т.е. на излезот од регистерот се појавува првиот бит од податокот кој може да се чита.

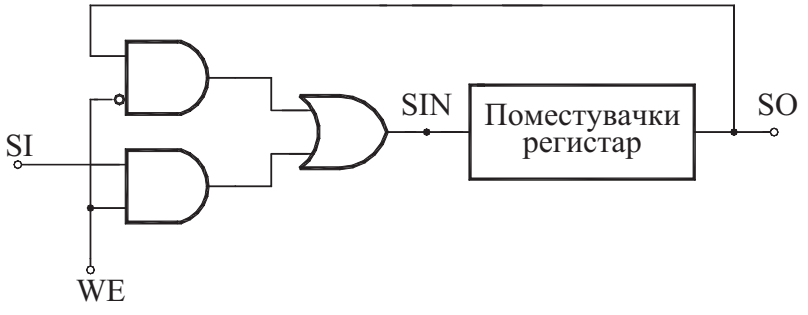
Ваквиот начин на отчитување на битовите од содржината на регистерот претставува начин на работа по принципот *FIFO (анг. First-In-First-Out)*: првиот внесен, прв излегува. Исчитувањето на содржината на регистерот се врши во сериска форма, бит по бит и трае онолку такт интервали, колку што има битови влезниот податок. Табелата на премин на состојбата на регистерот таб. 5-1 се однесува на влезната комбинација 1101.

Од она што е досега изложено, станува јасно дека содржината на овој регистер нема да се изгуби само ако читањето се изврши точно навреме. Имено, треба да се почне со читање веднаш по *n*-тиот импулс, каде *n* е бројот на употребени флип-флопови (во

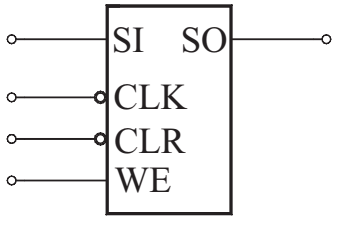
примеров n=4), т.е. по n-тиот опаѓачки раб. При ова не треба да се внесува нов податок, бидејќи битот што е поместен надвор од последниот флип-флоп неповратно ќе се изгуби. За остварување на оваа цел, на регистерот од сл. 5-8 може да му се додадат две И кола: влезно и излезно (сликата прикажано со испрекината линија). Дополнителната контролна линија ($W / \overline{R}$) овозможува запишување ако се наоѓа на високо ниво ($W / \overline{R}$=1), односно читање ако на неа се доведе ниско ниво ($W / \overline{R}$=0).

5.3. КРУЖЕН РЕГИСТЕР

Битовите запомнети во регистерот може да се сочуваат доколку излезот од последниот флип-флоп се поврзи назад со влезот на првиот флип-флоп. Кај овој регистер, битовите на податокот ќе циркулираат низ него, поместувајќи се за по еден флип-флоп со секој такт импулс. Вака поврзаниот регистер се вика **кружен** или **циркуларен регистер**, или **поместувачки регистер со враќање на преносот од крајот** (анг. *end-around-carry shift register)*. Овде треба да напоменеме дека кружењето на битовите во регистерот е можно само ако се откачи серискиот влез кога регистерот се „наполни" и веднаш се формира повратната врска преку која ќе може да се враќаат излезните битови, на влезот еден по еден. За таа цел, во овој случај ќе мора да се формира додатна логичка мрежа со соодветни контролни сигнали, такви што на влезот од првиот флип-флоп ќе може да го поврзува или директниот сериски влез во регистерот, или излезот од регистерот. Еден ваков пример на кружен регистер е прикажан на сл. 5-10, додека неговата симболичка ознака е презентирана на сл. 5-11.



Сл. 5-10. Логички дијаграм на кружен поместувачки регистер



Сл. 5-11. Логички симбол на кружен поместувачки регистер

Сигналот за овозможување на запишување *WE* (анг. *write enable*), ја контролира линијата за влез на податоците во регистерот *SIN* преку два-во-еден мултиплексер добиен со две И и едно ИЛИ коло кои ја реализираат логичката равенка

$$\text{SIN} = SI \cdot WE + SO \cdot \overline{WE} \qquad (11\text{-}1)$$

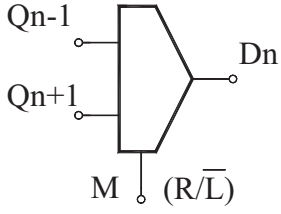
Кога WE=1, се отвора И вратата на која е приклучен влезот за податокот SI (Y_{SI}=SI), а се затвора И портата на која е приклучен излезот SO од регистерот (Y_{SO}=0). Така, преку ИЛИ колото во регистерот се запишуваат битовите на новиот податок Y_{ILI}=SIN=SI.

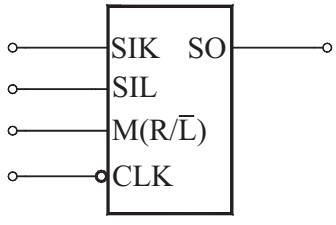
Кога WE=0 ситуацијата е обратна: се отвора И вратата која го пропушта внесениот податок (Y_{SO}=SO), а преку другото И коло кое е затворено не се дозволува влез на нов податок (Y_{SI}=0). Така преку ИЛИ колото во регистерот повторно се полни запомнетата информација Y_{ILI}=SIN=SO со што се добива нејзино кружење низ флип-флоповите на регистерот.

5.4. ДВОНАСОЧЕН ПОМЕСТУВАЧКИ РЕГИСТЕР

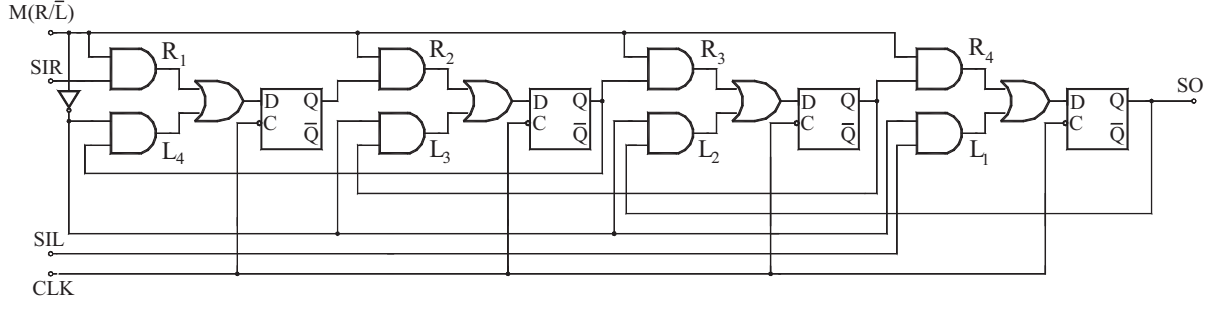
Претходно разгледаниот поместувачки-регистер го поместува податокот бит по бит, но само во десно. Во практиката често пати се јавува потреба за регистер кој треба да ги поместува податоците кон лево. Познавајќи го принципот за реализација на поместувачкиот регистер во десно, може да го претпоставиме начинот на поврзување на флип-флоповите во регистерот којшто треба да врши поместување во лево. Кај ваквиот регистер, влез во претходниот флип-флоп ќе биде излезот од следниот флип-флоп: $D_i=Q_{i+1}$.

Како наредени пример ќе разгледаме регистер кој има можност да врши поместување на содржината и во лево и во десно, или т.н. двонасочен поместувачки регистер. Влезот во секој регистер треба да биде или излезот од претходниот степен, ако сакаме поместување кон десно, или излезот од следниот степен, ако треба да се изведе поместување во лево: $D_i = Q_{i-1} + Q_{i+1}$. И во овој случај, пред секој флип-флоп ќе треба да се употреби по еден два–во–еден мултиплексер (сл. 5-12), секој контролиран со единствена линија за насока на движење. Од анализата произлегува логичката шема на двонасочниот поместувачки регистер прикажана на сл. 5-13 б, чиј симбол е претставен на сл. 5-13 а.





Сл. 5-12. Два–во–еден мултиплексер Сл. 5-13 а). Симбол на двонасочен поместувачки регистер



б). Логичка структура

Сл. 5-13. Двонасочен поместувачки регистер

Насоката на поместувањето зависи од контролниот влез M (анг. *mode*), со кој се определува начинот (режимот) на работа, бидејќи за влезот D_i на секој флип-флоп i = 1, 2, 3, 4 важи веќе познатата логичка равенка $D_i=(M \cdot Q_{i-1} + \overline{M} \cdot Q_{i+1})$. Кога M=1, содржината на регистерот се поместува во десно, додека ако M=0 неговата содржина се поместува во лево. При ова внесувањето на податоците се изведува преку два влеза: едниот е означен со SIR и на него се донесуваат битовите на оној податок што треба да се поместува во десно, додека за податокот чиишто битови треба да се поместуваат во лево постои влезот SIL.

Имено, кога *M=1* е овозможен пренос на сигналот преку R вратите, а оневозможен преку L вратите чиишто излези се на ниво на 0 заради присуството на 0 на по еден влез од секоја L порта. Во овој случај, битовите на податокот се носат на влезот од првиот флип-флоп D_0, чијшто излез Q_0 е врзан за влезот од вториот флип-флоп D_1 итн., така што регистерот ја поместува содржината во десно, на начин кој веќе го опишавме.

Кога *M=0*, ситуацијата е обратна затоа што сите R врати се затворени, излезите им се на 0, додека сите L врати се отворени, и врските помеѓу флип-флоповите се во инверзна насока: прво сигналот се носи на влезот од третиот флип-флоп D_3, потоа од неговиот излез Q_3 оди на влезот од вториот флип-флоп D_2, па од Q_2 на D_1 итн., што значи дека сега битовите ќе се поместуваат еден по еден во лево.

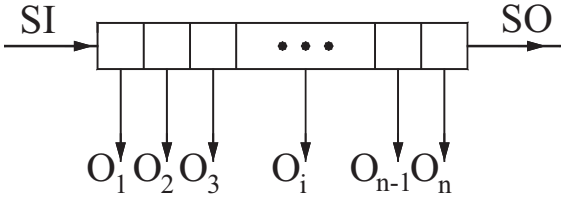
Многу е важно да напоменеме дека сигналот M може да се менува само кога нивото на такт сигналот е 0 (CLK=0), инаку може да дојде до несакана промена на содржината на регистерот.

Овој тип регистер остава можност за читање на битови од податокот според принципот *LIFO*, (анг. Last-In-First-Out): последен внесен, прв изнесен. Тоа се реализира така што битовите на податокот се внесуваат во регистерот преку поместување во десно, т.е. воспоставување M=1 и употреба на влезот SIR, а потоа доведување на M=0 и полнење на регистерот преку влезот SIL.

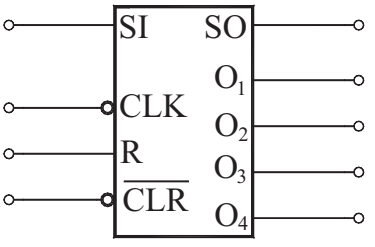
5.5. ПОМЕСТУВАЧКИ РЕГИСТЕР СО СЕРИСКИ ВЛЕЗ И КОМБИНИРАН ИЗЛЕЗ

Едноставната блок-шема на регистерот, кој има сериски влез и можност за сериски или паралелен излез е прикажана на сл. 5-14, а една негова реализација на сл. 5-15 б), која повторно претставува шема на некој четирибитен регистер. Симболичката ознака дадена е на сл. 5-15 а).

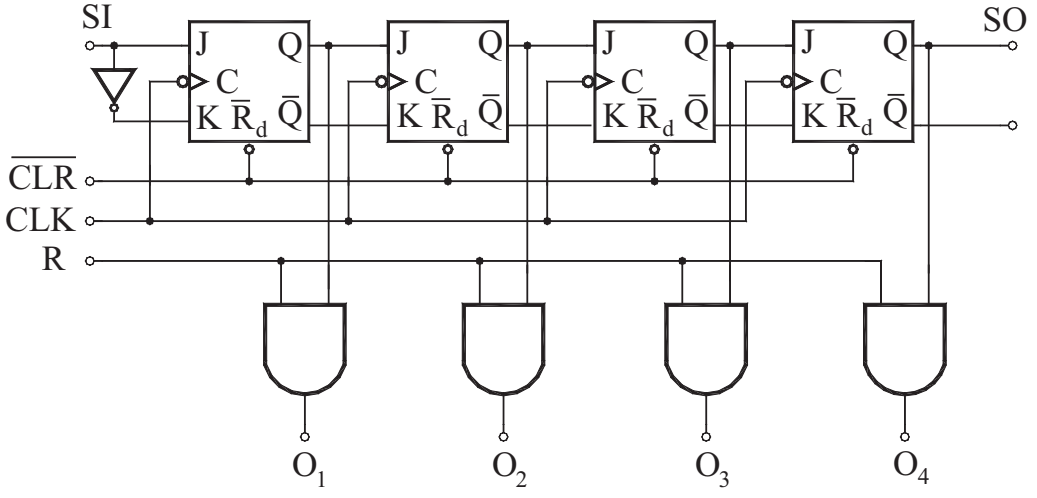
Работата на регистерот, што се однесува до серискиот влез и излез, веќе ни е позната од претходното објаснување: податокот доаѓа во сериска форма на влезот од првиот флип-флоп, а сериски се чита од излезот на последниот флип-флоп.



Сл. 5-14. Едноставна блок-шема



Сл. 5-15. а) симболичка ознака



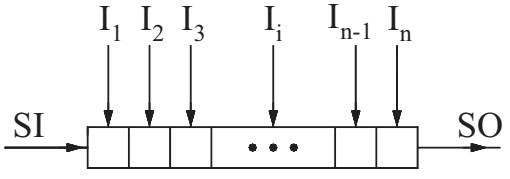
б) логичка шема

Сл.5-15. Четирибитен регистер со сериски влез и комбиниран излез

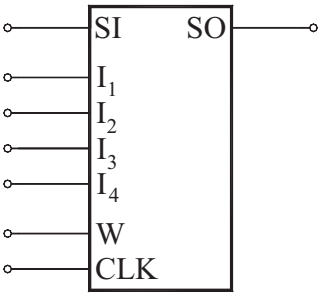
Паралелното читање на содржината на регистерот се врши со доведување на позитивен импулс на линијата за читање R (R =1) кој ги отвора логичките И кола и со тоа овозможува состојбата на излезот од секој флип-флоп да се појави на соодветната излезна линија: $O_1=Y_{11}=Q_{1,...,}O_4=Y_{I4}=Q_4$ со што содржината на регистерот (податокот) се добива во паралелна форма. Од изнесеното станува јасно дека овој регистер претставува сериско-паралелен конвертор (претворувач) на код, бидејќи податоците се запишуваат сериски, а се читаат паралелно. Сериски кодираните бинарни информации ги претвора во паралелен бинарен код, односно врши конверзија од временски во просторен облик.

5.6. ПОМЕСТУВАЧКИ РЕГИСТЕР СО КОМБИНИРАН ВЛЕЗ И СЕРИСКИ ИЗЛЕЗ

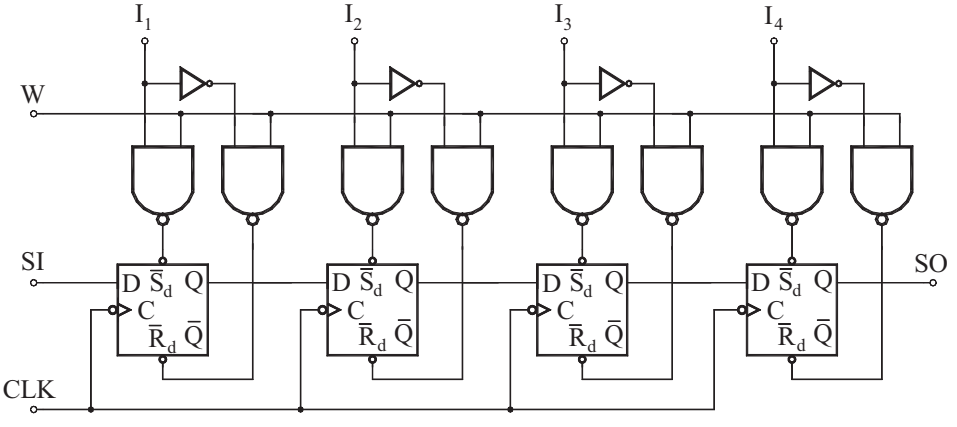
Едноставниот блок-дијаграм на овој регистер кој има опционен сериски или паралелен влез и сериски излез, е прикажан на сл. 5-16, а една негова изведба е дадена на сл. 5-17 б). Тоа повторно е четирибитен регистер, чија симболичка ознака ја гледаме на сл. 5-17 а). Паралелниот влез на податоците се остварува преку асинхроните влезови $\overline{S_d}, \overline{R_d}$ на секој флип-флоп. Директните вредности на влезниот податок го побудуваат влезот $\overline{S_d}$ преку логички НИ врати, чиј втор влезен сигнал е линијата за запишување W. Овој контролен сигнал (W) исто така ги контролира и НИ портите чии втори влезови се коплементите од секој влезен бит, а излезите го побудуваат директниот влез за ресетирање $\overline{R_d}$. На овој начин, асинхроните влезови $\overline{S_d}, \overline{R_d}$ на секој флип-флоп секогаш меѓусебно се комплементарни со што е овозможено во секој флип-флоп да се внеси логичката состојба присутна на секој влез што обезбедува правилно функционирање на оваа дигитална компонента. Паралелното запишување на податокот се врши кога на линијата за запишување W ќе се донесе високо ниво кое ги отвора НИ портите.



Сл. 5-16. Едноставна блок-шема



Сл.5-17 а) симболичка ознака



б) логичка шема

Сл. 5-17 Четирибитен регистер со комбиниран влез и сериски излез

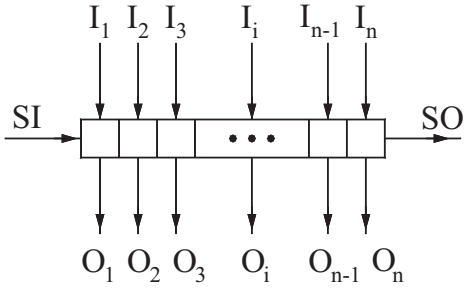
На овој начин, конечната состојба на секој флип-флоп, а со тоа и на регистерот, зависи само од логичките нивоа на влезните сигнали, а не и од неговата тековна содржина. Со ова се добива забрзување на работата при полнење со нова содржина, затоа што нема потреба од бришење на податокот кој се наоѓа во регистерот.

Информацијата може да биде запишана и во сериска форма преку влезот на првиот флип-флоп, додека отчитувањето на содржината на регистерот се врши во сериски облик на излезот од последниот флип-флоп.8

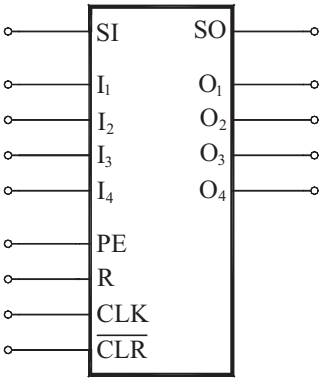
Ако се врши паралелен запис и сериско читање, паралелно кодираната бинарна информација фактички се претворува во сериски кодирана, т.е. се врши конверзија од просторен во временски код. Ова е многу корисна особина затоа што на овој начин информациите можат да се запишат на непериодичен начин, секој бит во друго време, а потоа да се отчитуваат периодично, со секој такт-импулс, бит по бит на сериски начин.

5.7. УНИВЕРЗАЛЕН РЕГИСТЕР

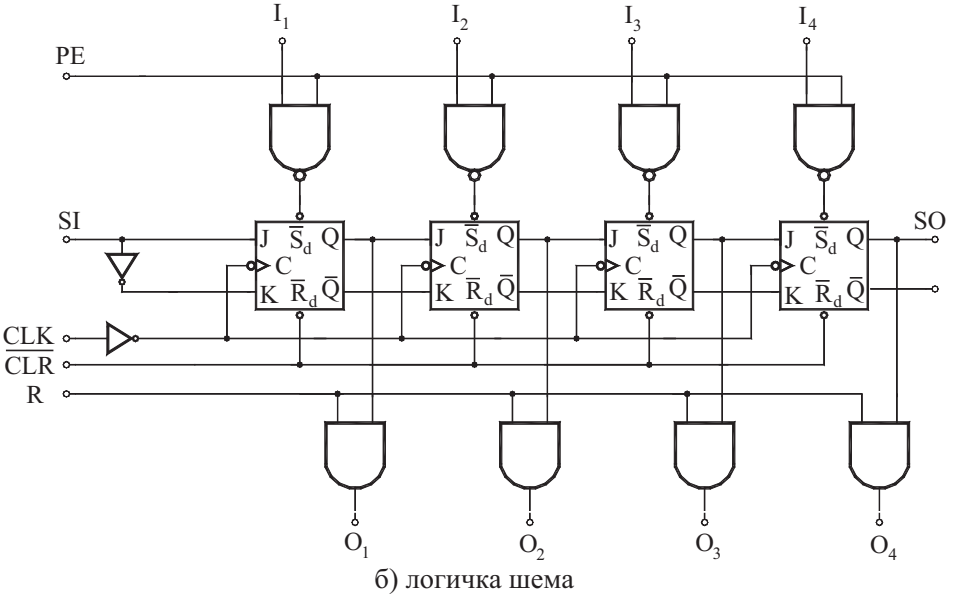
На сл. 5-18 е прикажана едноставната блок-шема на овој регистер, од која веднаш се гледа дека кај овој регистер постои можност и за двата начина на запишување или читање на податоците. На сл. 5-19 б) е дадена една реализација на ваков четирибитен регистер, чија симболичка ознака е претставена на сл. 5-19 а).



Сл. 5-18. Едноставна блок-шема



Сл. 5-19. а) симболичка ознака



б) логичка шема

Сл. 5-19. Четирибитен универзален регистер

Бришењето се изведува со носење на ниско ниво на влезот $\overline{CLR}$. Преку влезот SI *(serial input)* податоците се внесуваат во сериска форма бит по бит, а преку паралелните влезови I_1,I_2,I_3,I_4, регистерот може да се постави во било која почетна состојба. Влезот за поставување во почетна положба PE (*preset enable*-овозможено поставување) ја овомножува токму оваа операција кога на него ќе се доведе високо ниво. Претходно, регистерот мора да биде избришан, што важи генерално: операцијата на бришење мора да претходи на операцијата за внесување на податок во паралелен облик. Излезната информација може да се добие во сериски облик на излезот од последниот флип-флоп, или во паралелен облик ако се доведе високо ниво на линијата за читање R *(read)* со што состојбата на секој флип-флоп се јавува на излезните линии.

Кај овој пример на регистер, такт-импулсите прво се инвертираат, а потоа се доведуваат на влезовите за такт-сигнал C кај сите флип-флопови. Со ова практично се овозможува работа на регистерот на предниот раб од тактот (со доцнење од еден период), иако употребените флип-флопови се со MS структура.

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

11-1. Што претставува регистерот? Од што е составен?

11-2. Каква е примената на регистрите во дигиталните сметачки машини?

11-3. Што претставува содржината на регистерот?

11-4. Што претставува операцијата полнење (внесување, запишување)?

11-5. Што се врши со операцијата читање?

11-6. Што се случува со старата содржина при полнење на регистерот со нова содржина?

11-7. Какви начини на читање постојат во зависност од тоа дали при читањето содржината што се наоѓа во регистерот се губи или не?

11-8. Каде се јавува секој бит при читањето на содржината на регистерот?

11-9. Со што се дефинира состојбата на регистерот?

11-10. Што се прави со операцијата бришење?

11-11. Што се прави со операцијата поставување?

11-12. Кои додатни линии постојат кај секој регистер и за што служат?

11-13. Набројги различните начини за читање, односно запишување на содржината на регистерот!

11-14. Што се случува со битовите на содржината при паралелниот начин?

11-15. Што се случува со битовите на содржината при серискиот начин?

11-16. Што е карактеристично за стационарните регистри?

11-17. Што е карактеристично за поместувачките регистри?

11-18. Каков регистер е прикажан на сл. 5-5 в)? Нацртај ја неговата блок-шема и логички симбол.

11-19. Објасни како се доведува во почетната состојба регистерот од сл. 5-5 в)!

11-20. Објасни како се изведува полнењето на регистерот од сл. 5-5 в)!

11-21. Објасни како се извршува читањето на содржината на регистерот од сл. 5-5 в)!

11-22. Каков регистер е прикажан на сл. 5-6 в)? Нацртај ја неговата блок шема и логичкиот симбол.

11-23. Како се доведува во почетна состојба регистерот од сл. 5-6 в)?

11-24. Како се изведува полнењето на регистерот од сл. 5-6 в) со а) две линии б) една линија за читање и запишување?

11-25. Како се врши читањето на содржината на регистерот од сл. 5-6 в) со а) две линии б) една линија за читање и запис?

11-26. Каква треба да биде логичката состојба на сите линии од регистерот даден на сл. 5-6 в) за да а) во него се запише б) од него се прочита податокот 1) 1101; 2) 1011.

11-27. Каков регистер е прикажан на сл. 5-7 а)? Нацртај ја неговата блок шема и логичкиот симбол.

11-28. Како се доведува во почетна состојба регистерот од сл. 5-7 а)?

11-29. Што прави линијата M за регистерот од сл. 5-7 а) кога M=1 б) кога M=0?

11-30. Како се врши читањето на содржината на регистерот од сл. 5-7 а)?

11-31. Каква треба да биде логичката состојба на сите линии од регистерот даден на сл. 5-7 а) за да а) во него се запиши б) од него се прочита податокот 1) 1001; 2) 1110.

11-32. Каков регистер е прикажан на сл. 5-8 в) и г)? Нацртај ја неговата блок шема и логичкиот симбол.

11-33. Како се доведува во почетна состојба регистерот од сл. 5-8 в) и г)?

11-34. Преку кои линии се внесува и чита содржината на регистерот од сл.5-8 в) и г)?

11-35. Нацртај ги временските дијаграми на излезите од флип-флоповите, т.е. на состојбата на регистерот Q_1, Q_2, ..., Q_4 од сл. 5-8 в) и г) ако на влезот доаѓа податокот 1011. Како ќе изгледаат дијаграмите ако наместо флип-флопови кои се управувани со работ на такт-сигналот, се применат а) MS D флип-флопови б) D лечови? Дали ваквиот регистер работи исправно или не? Образложи!

11-36. Нацртај ги временските дијаграми за состојбата на регистерот од сл. 5-8 в) и г) како и неговата табела на премин ако на влез доаѓа податокот: 1) 10110; 2) 01010.

11-37. Според кој принцип се врши отчитување на содржината од регистерот прикажан на сл. 5- 8 в) и г)? Зошто?

11-38. Дали постои можност при читањето на содржината од регистерот прикажан на сл.5-8а,б да дојде до нејзино губење, и под кои услови?

11-39. Објасни ја логичката шема на четирибитен кружен регистер со линија за сериски влез, линија за сериски излез на податокот, линија за бришење и контролна линија за запишување на нова содржина, односно за кружење на старата прикажана на сл. 5-10.

11-40. *) Нацртај логичка шема и логички симбол на четирибитен поместувачки регистер во лево со D флип-флопови водени со работ на такт сигналот, и посебен влез за бришење.

11-41. Опиши го начинот на работа на двонасочниот регистер од сл. 5-12 ако (а) M=0 (б) M=1.

11-42. На кој начин може да се изврши отчитување на содржината на регистерот од сл. 5-12 така што првиот внесен бит, да се прочита последен, т.е. по принципот LIFO.

11-43. Каков регистер е прикажан на сл. 5-15 б)? Нацртај ја неговата блок-шема и симболичка ознака.

11-44. На кој начин се полни регистерот од сл. 5-15 б)? Како може да се чита неговата содржина?

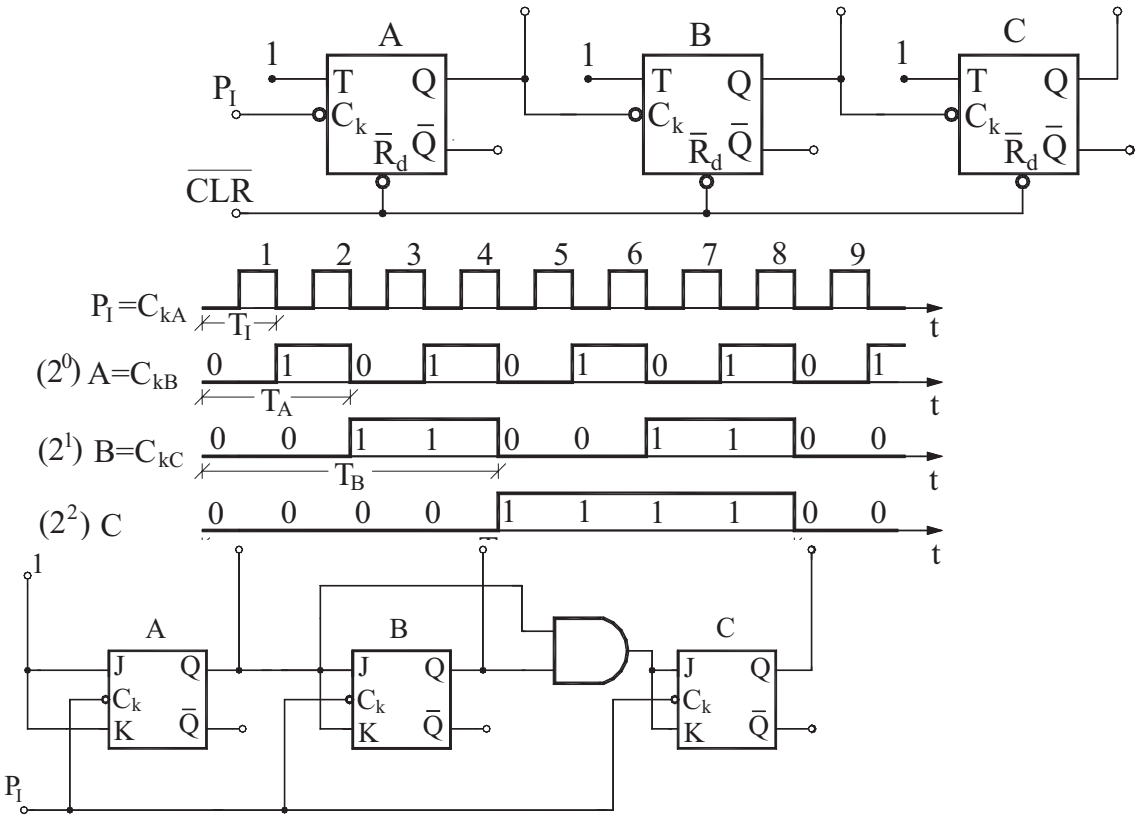
11-45. Каква конверзија овозможува регистерот од сл.5-15 б).

11-46. Каков регистер е прикажан на сл. 5-17 б)? Нацртај ја неговата блок шема и симболичка ознака.

11-47. На кој начин се врши полнењето на регистерот сл. 5-17 б)? Како може да се чита неговата содржина?

11-48. Каков регистер е прикажан на сл. 5-19 б)? Нацртај ја неговата блок шема и логички симбол.

11-49. Објасни го принципот на работа на регистерот од сл. 5-19 б).



6.

БРОЈАЧИ

По изучувањето на оваа тематска целина

- ќе ја разберете логичката структура на бројачите;
- ќе знаете да го опишете принципот на работа и примената на асинхроните бројачи:
  - бинарен бројач,
  - бројач со произволен модул,
  - бројач наназад,
  - регистер со комбиниран излез,
  - двонасочен бројач.
- ќе знаете да го опишете принципот на работа и примената на синхроните бројачи:
  - бинарен бројач,
  - бројач со произволен модул,
  - декаден бројач,
  - кружен бројач.
- ќе умеете да проектирате бројачи со различен модул на броење.

6.1. ВОВЕД И ОСНОВНИ ПОИМИ И КОНЦЕПТИ

Бројачите (анг. *counters*) се секвенцијални логички мрежи кои ги бројат импулсите што се доведуваат на нивниот влез, а на својот излез го даваат редниот број на секој влезен импулс во одреден бинарен код. Најчесто тоа е природниот бинарен систем, или природниот бинарен код т.е. NBCD или 8421 кодот. Овие дигитални склопови како основни градбени компоненти користат конечен број мемориски елементи, поточно флип-флопови, најчесто во комбинација со различни логички кола.

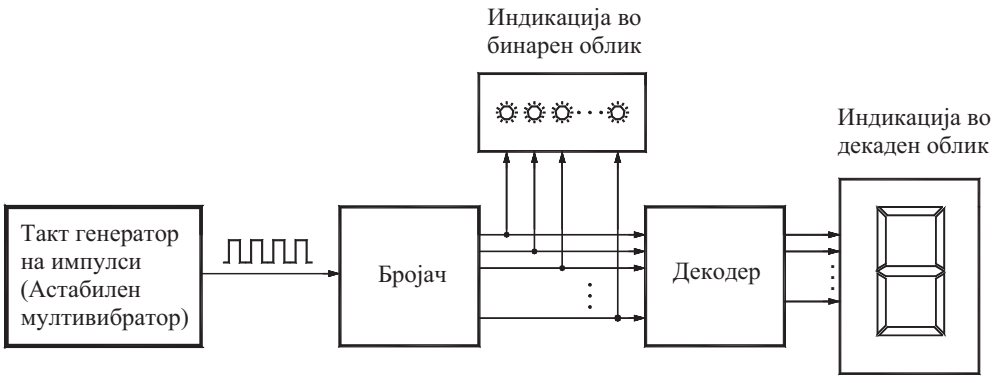
Бидејќи секој флип-флоп има две состојби, тоа значи дека множество од *n* флип-флопови може да се најде најмногу во 2^n состојби, каде што една состојба на множеството се дефинира како конкретна комбинација од состојбите на поединечните флип-флопови. Кај бројачите, флип-флоповите се поврзани во таква логичка структура што секој импулс на влезниот сигнал го предизвикува бројачот да ја менува состојбата и од една состојба да влегува во друга. Ако бројачот почне да брои од некоја почетна состојба и во неа се врати по *M* влезни импулси, таквиот бројач се вика бројач со *основа M* или *по модул M*. Ако некој бројач е составен од *n* флип-флопови и поминува низ секоја од можните 2^n состојби, таквиот бројач се вика *бинарен бројач*.

Бројачите се базираат на тактираните T флип-флопови во префрлувачки (прекинувачки) режим на работа според таб. 4-10 кога влезот T=1 предизвикува флип-флопот да ја менува состојбата при секој импулс на такт сигналот. Станува збор за T флип-флопови со MS структура (анг. master-slave, т.е. главен-извршен) кои се активни на задниот раб од такт-сигналот. Флип-флоповите каскадно се поврзуваат така што првиот флип-флоп ја менува својата состојба на секој бројачки импулс, вториот на секој втор импулс, третиот на секој четврт, четвртиот на секој осми, итн. што соодветствува на промената на битовите со различни тежини кај природниот бинарен систем. Токму заради ова, излезната информација на бројачот се добива кога истовремено (паралелно) се зема состојбата на излезот од секој флип-флоп кој влегува во состав на бројачката мрежа. Затоа излезот на бројачот претставува бинарен вектор со должина од *n*-бита, каде што *n* е бројот на употребените флип-флопови, т.е. степени.

Покрај броењето, од изнесеното станува јасно дека каскадното поврзување на T флип-флоповите и нивниот начин на работа овозможува после секој флип-флоп во состав на бројачката мрежа, фреквенцијата на влезните бројачки импулси редоследно да се дели со два, заради што бројачите можат да се користат и како делители на фреквенција.

Инаку, во практичните реализации на бројачките мрежи многу често, наместо T се среќаваат каскадно поврзани JK MS флип-флопови, кои според таб. 4-8 работат исто како T флип-флопови во префрлувачки режим**,** бидејќи на J и K влезовите им се приклучува високо ниво (J=1 и K=1). Имено, заради своите два влеза, JK флип-флоповите нудат пошироки можности за изведби на различни типови бројачи, како по логичката структура, така и по основата на броење.

Имајќи го предвид претходно наведениот принцип на работа на T или JK каскадата, како и формирањето на излезната информација на бројачот, станува јасно дека секој изброен импулс на излезот од бројачот ќе биде презентиран со соодветен бинарно кодиран број кој лесно може да биде прикажан со помош на LED (светлечки) диоди, како што може и да се види од сл. 6-1,која претставува блок шема на бројач во општ случај.



Сл.6-1. Блок шема на бројач со бинарно и декадно покажување

За визуелно претставување на резултатите од броењето, кога бројот на импулсите треба да биде лесно разбирлив од страна на човекот, на излезот од бројачот може да се додаде декодер и седум-сегментен екран како индикатор, според истата сл. 6-1.

6.2. ОСНОВА И КАПАЦИТЕТ НА БРОЈАЧИТЕ

Веќе наведовме дека состојбата на бројачот е дефинирана преку состојбата на секој од *n*-те флип-флопови (степени) што влегуваат во составот на бројачката мрежа. Во врска со ова, секој бројач има своја *почетна состојба*, а тоа е состојбата на секој од флип-флоповите пред да дојде првиот влезен импулс. Со секој нов импулс бројачот ја менува состојбата за тој, по завршувањето на еден полн опсег на броење, да се врати во почетната состојба и отпочне нов циклус. Вкупниот број на различни состојби се вика *должина на бројачкиот циклус* и ја дефинира *основата* на бројниот систем во кој тој брои, т.е. *модулот* на бројачот. Според основата на броење бројачите се делат во две групи:

- бинарни бројачи, и
- небинарни бројачи, т.е. бројачи со произволна основа.

Основата на бинарните бројачи се означува со M_0 и тоа може да биде било кој степен на бројот 2, како на пр. 2, 4, 8, 16 и сл. даден со равенката:

$$M_0=2^n, \qquad (6\text{-}1)$$

каде $n = 1, 2, 3, \ldots$ е бројот на употребени флип-флопови кои го формираат бројачот.

Бројачите со произволна основа M се делат на декадни бројачи, чија основа е 10 ($M=10$) и на други небинарни бројачи.

Капацитетот на даден бројач се означува со N_K и се дефинира како најголемата децимална вредност што тој бројач ја дава на својот излез. Капацитетот на бројачот може да се претстави преку неговата основа:

$$N_K = M_0 - 1 \qquad (6\text{-}2)$$

ако е бинарен бројач (во природен бинарен код), односно

$$N_K = M - 1 \qquad (6\text{-}3)$$

во случај на бројач со произволна основа.

Кога се работи за бинарен бројач, тогаш

$$N_K = 2^n - 1 . \qquad (6\text{-}4)$$

Ова значи дека кога ќе се достигне капацитетот на бројачот N_K, сите негови флип-флопови ќе бидат сетирани (излезите ќе им се наоѓаат во состојба на логичка единица).

Пример: Нека се дадени два бинарни бројачи реализирани со а) n=3 б) n=4 флип-флопови. За секој од нив да се определи основата, т.е. модулот (M_0) и капацитетот (N_K).

а) За n=3 се добива $M_0=2^3=8$ па $N_K=8-1=7$ ($=111_{BIN}$), што значи дека станува збор за бројач со основа 8, т.е. октален бројач, додека

б) за n=4 се добива $M_0=2^4=16$ па $N_K=16-1=15$ ($=1111_{BIN}$), што значи дека се работи за бројач по модул 16, т.е. хексадецимален бројач.

6.3. ПОДЕЛБА НА БРОЈАЧИТЕ

Бројачите се делат по различни основи односно критериуми. Така на пр., претходно веќе наведовме дека според модулот на броење тие се делат на бинарни бројачи и бројачи кои бројат според произволна основа. Од друга страна, според начинот на доведување на импулсите постојат *асинхрони бројачи,* кои уште се викаат и бројачи со *сериски* или *реден влез,* и *синхрони бројачи* или *бројачи со паралелен влез.* Кај асинхроните бројачи импулсите се приклучуваат на влезот за такт сигнал во првиот флип-флоп, а секој излез од претходниот флип-флоп се поврзува на тактот кај следниот. За разлика од нив, кај синхроните бројачи импулсите се доведуваат паралелно (истовремено) до влезовите за такт на сите флип-флопови.

Почетната и крајната состојба на бројачот зависат од насоката на броењето, па според тоа тие се делат на *бројачи нанапред* и *бројачи наназад.* Бројачите нанапред почнуваат од состојбата што одговара на најмалата вредност на бројачот и со секој нов импулс неа ја зголемуваат за еден движејќи се кон најголемата вредност, за потоа, по нејзиното достигнување, повторно да почнат од почеток: од најмалата кон најголемата менувајќи ја секоја претходна состојба за еден, со што циклусот на броење континуирано се повторува. Спротивно на бројачите нанапред, бројачите наназад почнуваат од најголемата вредност и со секој нов импулс неа ја намалуваат за еден, броејќи кон најмалата вредност, за по нејзиното достигнување, да почнат повторно од најголемата кон најмалата, итн. И во овој случај циклусот на броење непрекинато се повторува.

Од кажаното следува дека за бинарен бројач нанапред, кој брои според природниот броен систем, неговата почетна состојба ќе биде ресетирана, т.е. излезите на сите флип-флопови се *0*-и, додека за случај на бинарен бројач наназад неговата почетна состојба ќе биде сетирана, т.е. излезите на флип-флоповите ќе им се *1*-и. Значи, ваквиот бројач нанапред, во децимална нотација, почнува да брои: *0, 1, 2, 3, ..., (N_K-1), N_K,* повторно *0, 1, 2, 3, ...* итн., додека бројачот наназад започнува од *N_K, (N_K-1), ... 2, 1, 0,* пак *N_K, (N_K-1), ...,* итн. каде што N_K е капацитетот на бројачот, т.е. неговата најголема декадна вредност.

Пример: Нека е даден октален бројач (бинарен бројач со основа $M_0=8$) кој брои а) нанапред б) наназад и дека за него треба да се определи бројачката секвенца.

Окталниот бројач е составен од n=3 флип-флопови заради што $M_0=2^3=8$ па $N_K=8-1=7$. а) Окталниот бројач нанапред ќе почне да брои од 000 (0_{DEC}) до 111 (7_{DEC}) и ќе ја генерира низата ... *000, 001, 010, 011, 100, 101, 110, 111 ...* т.е. во декадно означување ... 0, 1, 2, 3, 4, 5, 6, 7 ... б) Окталниот бројач наназад брои обратно: ќе почне од 111 (7_{DEC}) до 000 (0_{DEC}) и ќе ја генерира низата ... *111, 110, 101, 100, 011, 010, 001, 000 ...* т.е. декадно ... 7, 6, 5, 4, 3, 2, 1, 0 ....

Во практиката многу често се среќаваат и т.н. *кружни бројачи* кои по својата логичка структура значајно се разликуваат и од асинхроните и од синхроните бројачи. Имено, и кружните бројачи се реализираат како секвенцијални мрежи формирани од повеќе флип-флопови, но тие се затворени со повратна врска од излезот на последниот флип-флоп до влезот на првиот и тоа без употреба на логички кола. Покрај ова, кружните

бројачи најчесто се изградени од каскадно поврзани D, а не T или ЈК флип-флопови, заради што, од конструктивен аспект, потсетуваат на поместувачки регистри. Кај кружните бројачи, токму заради специфичниот начин на поврзување во затворена логичка мрежа, на влезната низа бројачки импулси не и соодветствуваат наредни броеви во растечки или опаѓачки редослед како кај бројачите нанапред/наназад, туку се добиваат броеви во различен бинарен код, зависно од конкретната структура на кружниот бројач.

Изучувањето на бројачите опфаќа нивна анализа и синтеза. Анализата се однесува на откривање на начинот на работа на веќе зададен бројач со позната логичка структура и формирање на неговата комбинациона табела и временски дијаграми. Од друга страна, при процесот на синтеза треба да се проектира (дизајнира) бројач со однапред зададени карактеристики: тип на бројач (синхрон или асинхрон), основа (модул) на броење (со бинарна или произволна основа), насока на броење (напред или назад) и типови на флип-флопови (T, ЈК или други). За да се стекне целосна слика за бројачите во натамошниот текст ќе бидат обработени и двата проблема.

6.4. АСИНХРОНИ БРОЈАЧИ

Асинхроните бројачи кои уште се викаат и *бројачи со сериски* или *реден влез* (анг. *ripple counters*) имаат наједноставна конфигурација и се употребуваат во случаи кога нема потреба од поголеми брзини во работата.

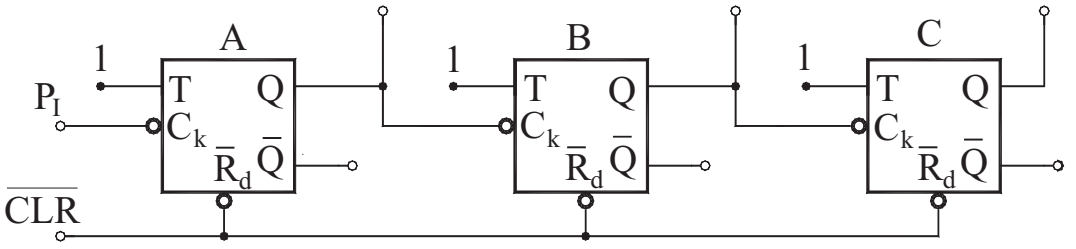
Најнапред, фокусот ќе го ставиме на бројачите што бројат нанапред за потоа, во куси црти да се задржиме и на принципот на работа на бројачите наназад, а ќе ги анализираме и т.н. двонасочни или билатерални бројачи кои можат да бројат и нанапред и наназад зависно од логичката состојба на посебниот контролен влез со кој се избира насоката на броење. На крај, ќе го објасниме и процесот на проектирање бројачи со небинарна основа.

6.4.1. БИНАРЕН АСИНХРОН БРОЈАЧ

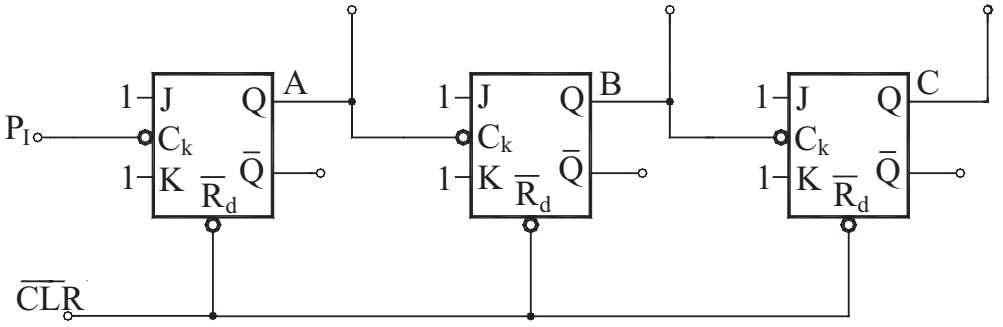
Броењето во дигиталните уреди се темели на бинарните бројачи кои воедно се и наједноставни за реализација. Покрај ова, поаѓајќи од структурата на бинарните бројачи, со одредени модификации релативно едноставно можат да се добијат небинарни бројачи.

Имајќи ја предвид равенката (6-1) за пресметување на основата M_0 на било кој бинарен бројач $(M_0=2^n)$, како и фактот дека бројачите ги прикажуваат изброените импулси во бинарен облик, станува јасно дека за реализација на бинарен бројач со основа M_0 која е цел степен на бројот 2 ќе треба да се употребат n флип-флопови.

На сл. 6-2 а), односно б), се прикажани две основни изведби на еден бинарен бројач со примена на три T, односно ЈК, MS флип-флопови ($n=3$) што значи дека станува збор за тростепена бројачка мрежа. Флип-флоповите имаат и директен влез за ресетирање (бришење), а меѓусебно се каскадно поврзани преку своите излези при што T, односно J и K влезовите на сите флип-флопови се држат на ниво на логичка **1** заради што, според последните редици од таб. 4-8 и таб. 4-10, тие работат во префрлувачки режим: со секој такт импулс состојбата на флип-флопот се менува во состојба, комплементарна на претходната ($Q=\overline{Q^+}$). Бидејќи флип-флоповите се со MS структура, тие ја менуваат состојбата при појава на задниот раб на такт сигналот, па секој следен степен ќе ја промени својата состојба во моментот кога излезот на претходниот степен преминува од високо на ниско ниво (од *1* на *0*). Значи кај ваквата структура, појавувањето на импулс на влезот за такт (C_K) кај било кој од флип-флоповите, предизвикува промена на неговата моментална состојба во следна која е комплементарна на неа.



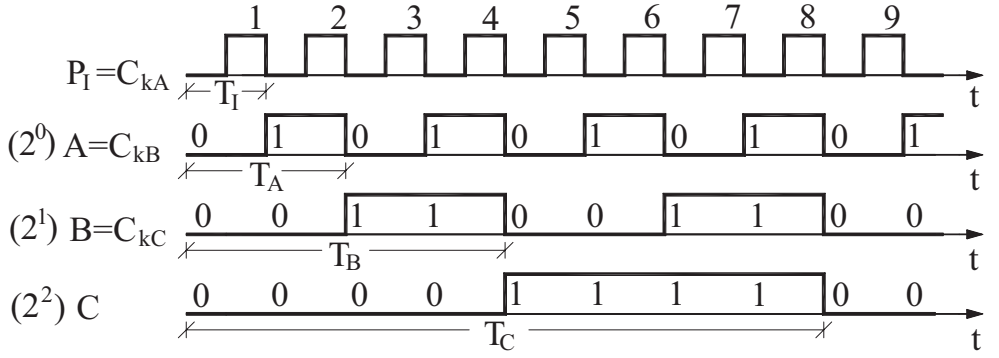
а) на октален асинхрон бројач со Т флип-флопови



б) на октален асинхрон бројач со ЈК флип-флопови

Сл. 6-2. Логичка структура

Побудните импулси (P_I) што треба да се избројат се доведуваат на влезот за такт сигнал на првиот флип-флоп А ($C_{KA}=P_I$), додека излезот Q од секој претходен флип-флоп претставува такт за следниот. Бидејќи се работи за тактирани Т, односно ЈК, MS флип-флопови кај кои $T=1$, односно $J=1$ и $K=1$, со секој бројачки импулс се менува состојбата на првиот флип-флоп А и тоа со појавата на неговиот заден раб. И флип-флоповите В и С ќе ја менуваат состојбата само кога на излезите од флип-флоповите што нив им претходат се јавува промена од високо на ниско ниво, т.е. во моментот кога се воспоставува логичка нула. Состојбата на флип-флопот В, т.е. неговиот излез Q_B зависи од излезот на А (Q_A) бидејќи тактот на В е токму излезот на флип-флопот А ($C_{KB}=Q_A$); слично, излезот на С (Q_C) зависи од излезот на В (Q_B), бидејќи такт за флип-флопот С е излезот на В ($C_{KC}=Q_B$). Начинот на работа на бројачот најдобро се гледа од временските дијаграми на напоните на излезите од овие мемориски елементи (сл. 6-3). Бидејќи се работи за секвенцијална мрежа, најнапред ќе треба да се претпостави почетната состојба која во овој случај, кога се работи за бројач нанапред, ќе треба да биде 0_{DEC} што бинарно се кодира како $ABC=000$, т.е. $Q_A=Q_B=Q_C=0$. Почетната состојба се дефинира со доведување на краткотрајно ниско ниво (логичка 0) на влезот за бришење $\overline{CLR}$ кој е поврзан на влезот за директно ресетирање на секој флип-флоп, со што истовремено се ресетираат сите флип-флопови.



Сл. 6-3. Временски дијаграми на октален асинхрон бројач

Со доведување на првиот импулс на влезот на бројачот се сетира флип-флопот А и на неговиот излез се воспоставува високо напонско ниво ($Q_A=1$). Ова не влијае врз флип-флопот В бидејќи станува збор за преден раб на неговиот такт, па тој останува во ресетирана состојба ($Q_B=0$). Излезот од флип-флопот В е такт за флип-флопот С но, бидејќи тој е низок, не ја менува состојбата на флип-флопот С, па така и флип-флопот С останува ресетиран ($Q_C=0$). Вториот влезен импулс повторно ја менува состојбата на флип-флопот А, сега тој се ресетира, па на неговиот излез се јавува опаѓачки раб и ниско напонско ниво ($Q_A=0$). Ова се пренесува до влезот за такт на следниот степен, флип-флопот В, со што флип-флопот В се сетира и на неговиот излез се појавува логичка единица ($Q_B=1$), преден растечки раб што не влијае врз следниот степен, флип-флопот С, кој и понатаму останува ресетиран ($Q_C=0$).

Третиот импулс повторно го сетира флип-флопот А, но не делува на флип-флоповите В и С ($Q_A=1$, $Q_B=1$, $Q_C=0$). По четвртиот импулс се завршени два циклуса на промени на состојбите на А, еден циклус на В и започнува промената на излезот од флип-флопот С ($Q_C=1$). Еден циклус на С, т.е. повторна промена на неговата состојба, но сега од високо на ниско ниво (од *1* на *0*) ќе се случи дури по осмиот импулс, кога истовремено сите флип-флопови од сетирана состојба *ABC=111,* т.е. $Q_AQ_BQ_C=111$, што претставува број 7 во декаден систем, преминуваат во ресетирана состојба *ABC=000,* т.е. $Q_AQ_BQ_C=000$ (0_{DEC}) со што повторно се воспоставува почетната состојба, од која броењето повторно продолжува преку *1, 2, ...* кон *7*. Јасно е дека по избројните 8 импулси се завршува циклусот на броење што произлегува и од равенката 6-1 ($M_0=2^n$), според која со замена за $n=3$ се добива

$$M_0 = 2^3 = 8 \qquad (6\text{-}5)$$

што значи дека модулот (основата) на броењето на овој бинарен бројач е 8 ($M_0=8$), заради што тој и се вика *октален бројач*.

За објаснување на работата на октални от бројач може да се користи и неговата комбинациона табела таб. 6-1 во која се дадени сите состојби на бројачот S_i, конкретните вредности во бинарен облик (комбинациите) на излезите од флип-флоповите за секоја од состојбите, како и соодветните индекси (декадните вредности) K_i на секоја комбинација. Од таб. 6-1 се гледа дека состојбите на бројачот S_0, S_1, S_2, ... S_7 одговараат на комбинационите вредности на бинарните триади на октални от броен систем: *000, 001, 010, 011, 101, 110* и *111* т.е. *0, 1, 2, 3, 4, 5, 6* и *7* декадно.

Состојби		Флип-флопови
S_i	K_i	C B A
0	0	0 0 0
1	1	0 0 1
2	2	0 1 0
3	3	0 1 1
4	4	1 0 0
5	5	1 0 1
6	6	1 1 0
7	7	1 1 1
0	0	0 0 0

Таб. 6-1. Комбинациона табела на октален асинхрон бројач

Имено, комбинационите вредности на бројачот можат да се определат со помош на излезите од флип-флоповите *A*, *B* и *C*. Во секоја редица се запишува состојбата (S_i) во која може да се најде бројачот, како и нејзиниот индекс (K_i) кој ја претставува во зависност од поединечната состојба на секој флип-флоп, каде што $i = 0, 1, 2, 3, 4, 5, 6, 7$.

Редоследот на состојбите и вредностите на индексите во табелата целосно одговараат на броењето во природниот бинарен броен систем.

Од табелата се констатира дека првиот степен т.е. флип-флопот *A* има позиција со најмала тежинска вредност 2^0, следниот флип-флоп *B* има тежина 2^1, додека крајниот степен *C* има најголема позициона вредност 2^2. Така, во декадно означување, секој изброен импулс може да се претстави со соодветен број *N*:

$$N=2^0Q_A+2^1Q_B+2^2Q_C \qquad (6\text{-}6)$$

Вкупниот број на состојби за овој бројач е осум, при што броењето започнува од почетната - првата состојба кога $N=0$ ($S_0=Q_AQ_BQ_C=000$), па сè до последната - осмата состојба кога се достигнува капацитетот на бројачот $N=N_K=7$ ($S_7=Q_AQ_BQ_C=111$).

Во општ случај, за *n*-степен бинарен бројач вкупниот број на состојби ќе изнесува 2^n, при што се почнува од почетната состојба кога $N=0$, па сè до крајната состојба кога $N=N_K=2^n-1$.

$$N=2^0Q_A+2^1Q_B+2^2Q_C+2^3Q_D\ldots+2^{n-1}Q \quad .. \qquad (6\text{-}7)$$

Делител на фреквенција: Бинарниот бројач всушност работи и како делител на фреквенција што лесно се забележува од неговите временски дијаграми прикажани на сл. 6-3. Бидејќи однесувањето на секој флип-флоп зависи од неговиот такт сигнал што е всушност излез од неговиот претходник, на излезот од секој флип-флоп се генерираат два пати помалку импулси во однос на тој пред него, т.е. сигнал со два пати поголема периода, така што односот на делење на влезните бројачки импулси зависи од тоа после кој флип-флоп се зема излезот. Факторот на делење изнесува 2^i, каде *i* е редниот број на флип-флопот ($i=1, 2, 3, .., n$). Така на пример, влезната бројачка низа е такт за првиот флип-флоп, па односот на делење на неговиот излез *A* е 2 (2^1), по вториот флип-флоп *B*, односот е 4 (2^2), итн. заради што ако фреквенцијата на влезните бројачки импулси е $f_I=1/T_I$, каде што T_I е нивната периода, фреквенцијата на импулсите по првиот флип-флоп А ќе биде $f_A=f_I/2^1=f_I/2$, по вториот $f_B=f_A/2=f_I/2^2=f_I/4$, по третиот $f_C=f_B/2=f_I/2^3=f_I/8$, итн. според равенката

$$f_i=\frac{f_I}{2^i}\text{, каде што } i = 1(A), 2(B), 3(C), \ldots n. \qquad (6\text{-}8)$$

Селекција на секој од сигналите со фреквенција дадена со равенката (6-8) може да се добие ако на бројачот се поврзе соодветен мултиплексер.

6.4.2. БИНАРЕН АСИНХРОН БРОЈАЧ НАНАЗАД

Принципот на работа на бројачите наназад е спротивен во однос на бројачите нанапред. За разлика од бројачите нанапред кои почнуваат да бројат од најмалата кон најголемата вредност, *бројачите наназад* поаѓаат од најголемата вредност и со секој нов изброен импулс нивната вредност се намалува за еден сé додека не стигнат до најмалата вредност – нула, за веднаш по неа повторно да влезат во својата почетна состојба - најголемата вредност на бројачот, која пак ќе се намалува за еден сé до најмалата, итн. циклусот на броење континуирано се повторува.

Идентично како и кај бинарните бројачи нанапред, бројот на потребни флип-флопови *n* за реализација на бинарен бројач наназад со основа M_0 која е цел степен на бројот 2, произлегува од познатата равенка (6-1) според која $M_0=2^n$.

Како пример за анализа повторно ќе земеме октален бинарен бројач со основа 8 ($M_0=8$) реализиран со 3 флип-флопови ($n=3$), но сега таков што ќе брои наназад. Комбинационите вредности, т.е. состојбите низ кои минува бројачот се внесени во табелата таб. 6-2.

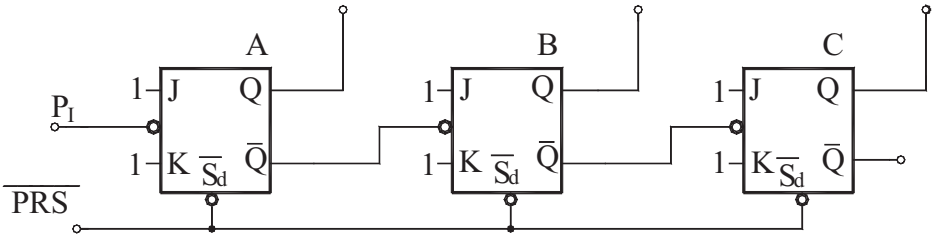
Состојби		Флип-флопови
S_i	K_i	C B A
0	7	1 1 1
1	6	0 1 1
2	5	1 0 1
3	4	1 0 0
4	3	0 1 1
5	2	0 1 0
6	1	0 0 1
7	0	0 0 0
0	7	1 1 1

Таб. 6-2. Табела на вистинитост на октален бројач наназад

Бидејќи се поаѓа од највисока вредност, во почетниот момент сите флип-флопови се сетирани: $Q_AQ_BQ_C=111$ (почетната состојба $S_0=111$ е декадно *7*), за нивната состојба да се промени во *110* ($S_1=110$, т.е. *6-ка*), па 101 ($S_2=101$, т.е. *5-ка*), итн. Сè додека бројачот не влезе во последната – осмата состојба кога сите флип-флопови се ресетирани: $Q_AQ_BQ_C = 000$ ($S_7=000$ или декадно *0*), по која повторно влегува во почетната состојба кога покажува 7-ка, па 6-ка, па пак 5-ка, 4-ка, итн.

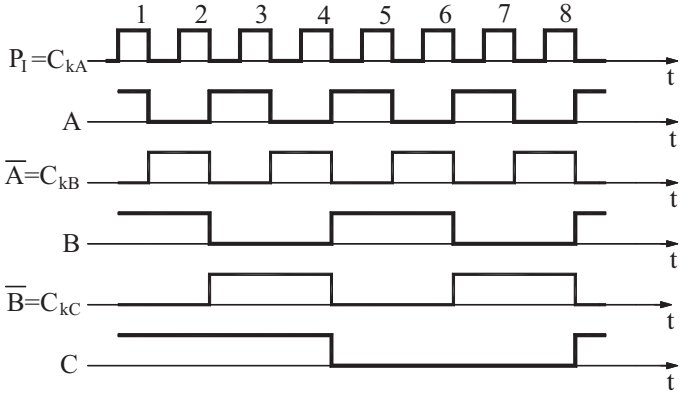
Ваков начин на работа може наједноставно да се добие ако се употребат каскадно врзани Т или ЈК флип-флопови слично како кај бинарниот бројач нанапред но во овој случај поврзувањето од еден до друг флип-флоп оди преку нивните комплементарни излези, а не преку директните.

Логичката структура на бројачот е прикажана на сл. 6-4. Тој се реализира со ЈК MS флип-флопови во префрлувачки режим на работа (*J=1* и *K=1*), а неговите временски дијаграми се претставени на сл. 6-5.



Сл. 6 -4. Логичка структура на октален бројач наназад

Бидејќи се работи за бројач наназад, неговата почетна состојба S_0 треба да биде *7*-ка декадно која бинарно се кодира како *ABC=111*, т.е. $Q_A=1$, $Q_B=1$ и $Q_C=1$, заради што употребените флип-флопови имаат директен влез за сетирање $\overline{S_d}$. Токму на сите овие влезови за директно сетирање се поврзува влезот за поставување $\overline{PRS}$, на кој со краткотрајно доведување на логичка нула се поставува почетната состојба на бројачот, т.е. сетирањето на сите флип-флопови ($Q_AQ_BQ_C=111$).



Сл. 6-5. Временски дијаграми на октален бројач наназад

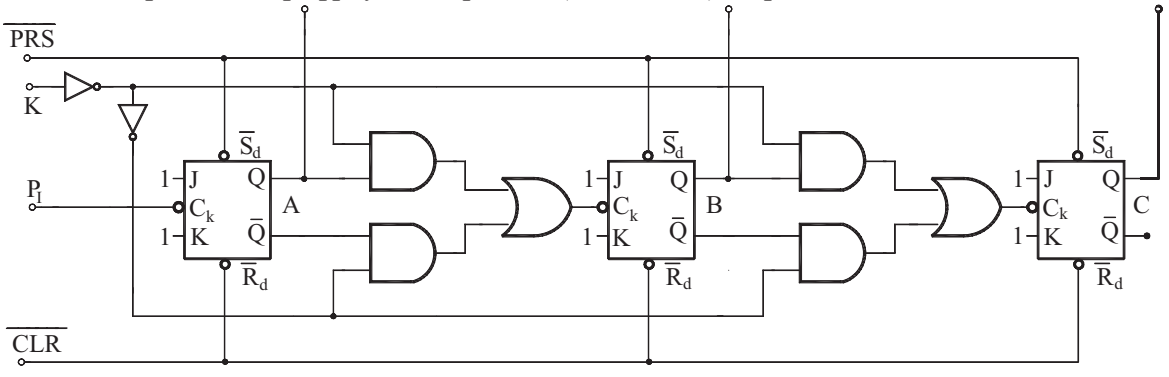
Принципот на работа на овој бројач е таков што првиот флип-флоп ја менува состојбата со секој нов влезен бројачки импулс, додека сите останати флип-флопови ја менуваат состојбата само кога на комплементарниот излез од претходниот флип-флоп се воспоставува состојба на логичка нула. Според ова, и со претпоставка дека бројачот се наоѓа во почетната состојба $S_0=111$ кога сите флип-флопови се сетирани $Q_A=1$, $Q_B=1$ и $Q_C=1$, доведувањето на првиот бројачки импулс ќе го ресетира флип-флопот А ($Q_A=0$) и на неговиот излез ќе се појави опаѓачки раб, додека комплементарниот излез ќе оди на високо ниво. Бидејќи комплетарниот излез на флип-флопот А е такт за следниот флип-флоп В, а на него се појавува растечки раб на напонот ($\overline{Q_A}=1$), тоа не влијае на состојбата на В која останува непроменета ($Q_B=1$). Следниот импулс го враќа флип-флопот А во состојба на логичка единица ($Q_A=1$), на неговиот комплементарен излез се појавува опаѓачки раб што го ресетира флип-флопот В ($Q_B=0$), но не влијае врз состојбата на флип-флопот С чиј такт е комплементарниот излез на В, а на него се појавил растечки раб, заради што С останува сетиран ($Q_C=1$). Состојбата на флип-флопот С ќе се промени од сетирана во ресетирана со појавата на задниот раб на четвртиот импулс бидејќи тогаш на комплементарниот излез на В ќе се појави опаѓачки раб ($\overline{Q_A}=0$). Циклусот на броење ќе заврши по осмиот импулс кога бројачот се враќа на почетната состојба на сетирани флип-флопови ($S_0=Q_AQ_BQ_C=111$).

6.4.3. БИНАРЕН АСИНХРОН ДВОНАСОЧЕН БРОЈАЧ

Од досегашното излагање може да се заклучи дека и бројачите нанапред и наназад имаат иста конструкција, но различно последователно поврзување. Тоа иницира примена на логичка мрежа со која ќе може да се реализираат и двата начини на броење.

Бројачот со ваква карактеристика се вика *билатерален бројач* (анг. up-down counter) бидејќи има можност да брои и во двете насоки: нанапред или наназад, во зависност од состојбата на посебната линија со која ќе се контролира насоката на броење.

За да се добие асинхрон двонасочен бројач, неговата вообичаена структура треба да се дополни со логички кола кои ќе овозможат правилно поврзување на флип-флоповите и при едниот, но и при другиот начин на броење. Ваков бројач, реализиран со ЈК MS флип-флопови поврзани во префрлувачки режим ($J=1$ и $K=1$) е прикажан на сл. 6-6.



Сл. 6-6. Логичка структура на октален двосмерен бројач

Бидејќи бројачот треба да има можност за броење и напред и назад, поврзувањето на флип-флоповите се изведува преку 2-во-1 мултиплексери (секој реализиран со по две И и едно ИЛИ коло) кои обезбедуваат избор на насоката на броење со меѓусебна исклучивост: ако бројачот брои нанапред, не може наназад, и обратно. Од сликата се гледа дека кај мултиплексерите едната влезна линија обезбедува поврзување на директниот излез на секој претходен флип-флоп до влезот за такт на секој следен, додека преку втората линија на влезовите за такт се поврзуваат комплементарните излези.

Изборот на насоката на броење се реализира со контролната линија K која се приклучува како линија за селекција на сите мултиплексери. Кога бројачот треба да брои нанапред, на оваа линија (*K*) со која се дефинира насоката на броење се поставува логичка нула (*K=0*) која преку колото за комплементирање како *1*-ца се проследува до горните И-кола и воедно истите ги отвора. Влезовите на овие И-порти се поврзани со директните излези од флип-флоповите, а нивните излези преку ИЛИ кола се дистрибуираат до влезот за такт сигнал на секој следен флип-флоп. Со ова практично секој од мултиплексерите овозможува пренос на сигналот од директниот излез на претходниот флип-флоп, до влезот за такт на следниот флип-флоп, а со тоа и броење нанапред.

Ако е потребно да се брои наназад, тогаш на контролната линија *K* се поставува високо ниво (*K=1*), со што се отвораат И-вратите на комплементарните излези, па заради преносот на сигналот од комплементарните излези на секој претходен до такт-влезовите на секој нареден флип-флоп, се добива инверзен процес на броење наназад.

Преку влезовите $\overline{CLR}$ и $\overline{PRS}$ се поставува почетната состојба на бројачот: кога треба да се брои напред сите флип-флопови се ресетираат па поради тоа на $\overline{CLR}$ треба да се донеси ниско ниво и на $\overline{PRS}$ високо ($\overline{CLR}=0$ и $\overline{PRS}=1$), додека за броење наназад побудата на овие два влеза треба да е комплементарна на претходната ($\overline{CLR}=1$ и $\overline{PRS}=0$) со што на стартот сите флип-флопови ќе се сетираат. Од изнесеното произлегува дека почетната состојба може да се поставува и со една влезна линија, на пример *P,* која истовремено ќе се носи директно до влезовите $\overline{S_d}$ на сите флип-флопови, но и комплементарно, преку инвертор, до сите нивни $\overline{R_d}$ влезови.

6.4.4. ПРОЕКТИРАЊЕ НА АСИНХРОН БРОЈАЧ СО ПРОИЗВОЛНА ОСНОВА

Бидејќи дигиталните уреди се базираат на бинарниот броен систем, основниот принцип на работа, т.е. природната основа на броењето на бинарните бројачи е 2^n, каде *n* е цел број кој ги претставува употребените мемориски елементи (флип-флоповите). Според тоа, бинарните бројачи можат да бројат само со бинарни модули M_0, па така со еден флип-флоп се добива бројач со основа (модул) 2 ($M_0=2^1$), со два флип-флопови бројач со основа 4 ($M_0=2^2$), со три бројач со основа 8 ($M_0=2^3$), со четири бројач по модул 16 ($M_0=2^4$), итн. Но, од друга страна, поради различни практични потреби, многу често се бара бројачите да бидат изработени со произволна основа *M*, како небинарни бројачи, што се изведува со реализирање на соодветна повратна врска врз основната конфигурација на бинарниот бројач.

Бројачот со произволна основа *M* составен од *n* флип-флопови, ќе има основа која е помала од бинарниот бројач составен од ист толкав број (*n)* на флип-флопови кој брои според природната основа M_0, така што секогаш ќе важи релацијата:

$$M \leq M_0=2^n. \qquad (6\text{-}9)$$

Така на пример, бидејќи секој четиристепен бројач во својата структура има 4 флип-флопови (*n=4*), како основа може да ги има сите модули *M* помали од $M_0=2^4=16$, т.е. мора да важи $M \leq M_0=2^4=16$.

Ако претпоставиме дека треба да се проектира небинарен бројач со произволна основа *M*, тоа значи дека циклусот на броење треба да заврши по *M* влезни импулси, за веднаш потоа бројачот да се врати на својата почетна состојба. Знаејќи дека модулот на броење *M* е различен од степен на бројот 2 и се разликува од природната (бинарната) основа на броење M_0 ($M \neq M_0$), каде $M_0=2^n$, меѓу нив се јавува разлика ΔM што го претставува бројот на состојби кои бројачот треба да ги прескокне за да брои со основа *M*:

$$\Delta M=M_0 - M. \qquad (6\text{-}10)$$

Состојбите *M* во кои може да се најде бројачот кога брои, се викаат *дозволени* или *легални состојби*, додека оние состојби што треба да бидат прескокнати се *недозволени* или *илегални состојби* (*ΔM*).

За прескокнување на некорисните, т.е. забранетите *ΔM* состојби може во принцип да се примени следново решение. Најнапред, многу кусо време се генерира бинарниот код на изброениот импулс, а потоа се врши ресетирање на флип-флоповите, т.е. враќање во почетната состојба и продолжување со процесот на броење. Овде е карактеристично тоа што се јавува една недозволена квази-состојба помеѓу последната важечка (легална) состојба од претходниот циклус на броење и првата легална состојба од следниот циклус. Ако оваа меѓусостојба се прочита би се јавила грешка во броењето.

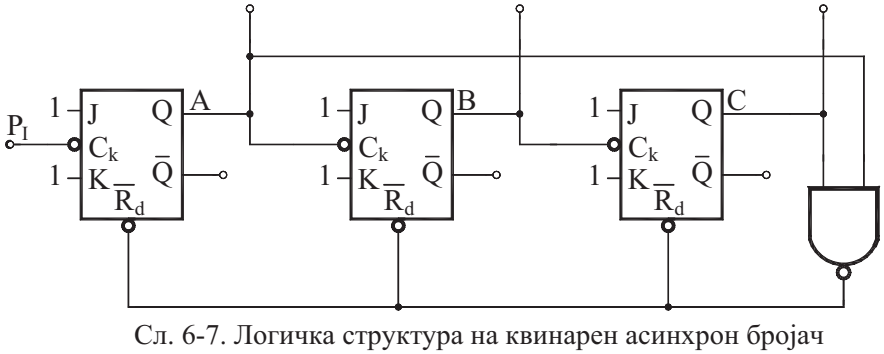
Пример: За бројач по модул 5, или т.н. *квинарен бројач*, според првиот начин, бројачот би ја повторувал низата *...000, 001, 010, 011, 100, 101/000...* т.е. во декадна нотација ... 0, 1, 2, 3, 4, 5/0 ... што значи дека квази-состојбата е 5, додека забранетите состојби се: *110* и *111*, т.е. 6 и 7 децимално. Меѓутоа, ако се примени второто решение, бројачот би ја генерирал секвенцата ... *000, 001, 010, 011, 100, 111/000* ..., или во декадно означување ... 0, 1, 2, 3, 4, 7/0, ... од каде се гледа дека сега квази-состојбата ќе биде 7-ка декадно, а забранети ќе се состојбите *101* и *110*, т.е. декадно 5 и 6. Во секој случај, една од недозволените состојби, како квази или меѓусостојба, се јавува по секој петти импулс кога завршува еден циклус на броење, а започнува нов.

За синтеза на бројач со однапред зададена произволна основа *M*, се применуваат флип-флопови со директен влез за ресетирање кои се активни на ниско ниво ($\overline{R_d}$). Логичката структура, т.е. потребниот број на мемориски елементи и начинот на нивното поврзување се добива со примена на следниве чекори:

1. Бројот на потребни флип-флопови *n* произлегува од условот (6-9), поточно треба да важи $2^{n-1} \leq M \leq 2^n$, каде што *M* е зададената основа (модулот) на бројачот, т.е. должината на циклусот на броење;
2. Флип-флоповите се поврзуваат во каскада и се формира логичка шема што реализира вообичаен *n*-степен асинхрон бројач;
3. Се врши претворување (конверзија) на зададениот модул на броење од декаден во бинарен број: $M_{(10)} \rightarrow M_{(2)}$ ($M_{DEC} \rightarrow M_{BIN}$);
4. Излезите од сите флип-флопови што се наоѓаат на високо ниво ($Q=1$) при $M_{(2)}$, треба да се приклучат на НИ коло;
5. Излезот од НИ колото се поврзува на сите влезови за директно ресетирање на флип-флоповите.

6.4.4.1. АСИНХРОН БРОЈАЧ СО ОСНОВА 5

Наведената постапка за дизајнирање на бројач со произволна основа *M* ќе ја илустрираме на претходниот пример за *квинарен бројач* чија основа на броење беше 5 (*M=5*).

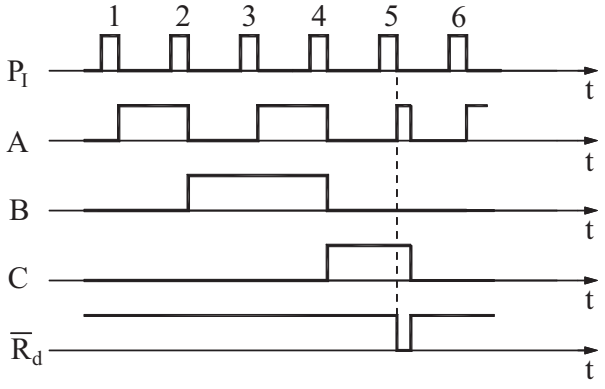


Сл. 6-7. Логичка структура на квинарен асинхрон бројач

Ќе претпоставиме дека на располагање имаме ЈК MS флип-флопови со директен влез за ресетирање $\overline{R_d}$. Бидејќи располагаме со ЈК флип-флопови кои треба да ја менуваат состојбата на секој влезен такт импулс, на нивните влезови J и K истовремено приклучуваме високо ниво т.е. ги држиме на фиксна *1*-ца (*J=1* и K=1), со што практично ЈК флип-флоповите ги трансформираме во T флип-флопови кај кои T=1. На тој начин тие работат во префрлувачки режим кога секој импулс на влезот за такт им ја менува состојбата во комплементарна.

Состојби		Флип-флопови
S_i	K_i	C B A
0	7	0 0 0
1	6	0 0 1
2	5	0 1 0
3	4	0 1 1
4	3	1 0 0
0	5 / 0	1 / 0 0 1 / 0

Таб. 6-3. Табела на вистинитост на квинарен асинхрон бројач



Сл. 6-8. Временски дијаграми на квинарен асинхрон бројач

Спроведувајќи ги редоследно чекорите од претходно наведената процедура, добиваме дека за реализација на квинарниот бројач ќе се потребни 3 флип-флопови (*n=3*) ($2^2=4\leq5\leq2^3=8$) при што $5_{DEC}=101_{BIN}$, од каде произлегува логичкиот дијаграм на бројачот прикажан на сл. 6-7, временските дијаграми дадени на сл. 6-8 и табелата на вистинитост таб. 6-3.

Од сл. 6-8 се гледа дека бројачот работи како вообичаен бинарен бројач до *5*-иот импулс, при што цело време излезот на НИ-колото е висок и не делува на директните влезови за ресетирање кои се активни на ниско ниво. Меѓутоа, по појавата на опаѓачкиот раб на *5*-иот бројачки импулс, бројачот влегува во меѓу-(квази)-состојбата што соодветствува на модулот на броењето во бинарна нотација ($M=5_{DEC}=101_{BIN}$) заради што излезите на флип-флоповите A и C се високи, додека на B е низок: $Q_A=1$, $Q_B=0$ и $Q_C=1$. Бидејќи излезите на *A* и *C* се врзани на влезовите на НИ-колото и истовремено и двата се на ниво на логичка единица, тоа предизвикува излезот на НИ-колото од високо да оди на ниско и со тоа да ги ресетира сите флип-флопови, со што бројачот се поставува во почетната (точната) состојба: $Q_A=0$, $Q_B=0$ и $Q_C=0$.

За реализација на *декаден бројач*, т.е. бројач со основа 10 (*M=10*), ќе се потребни 4 флип-флопови (*n=4*): A, B, C и D што произлегува од примена на првиот чекор, т.е. задоволување на релацијата $2^{n-1}\leq M\leq2^n$, бидејќи $2^3=8\leq10\leq2^4=16$. Поаѓајќи од тоа дека модулот на бројачот е 10 (*M=10*), ќе се изврши конверзија во бинарен број и ќе се добие 1010 ($10_{DEC}=1010_{BIN}$), што укажува на тоа дека влезовите во НИ колото треба да се земат од излезите на флип-флоповите B и D ($1010=Q_DQ_CQ_BQ_A$), по што лесно може да се комплетира логичкиот дијаграм на бројачот со примена на флип-флопови од ЈК или T тип, неговите временски дијаграми и табелата на вистинитост.

6.5. СИНХРОНИ БРОЈАЧИ

Ограничувањето на брзината во работата кај асинхроните бројачи може да се избегне ако бројачките импулси се носат паралелно до влезовите за такт кај сите флип-флопови во структурата на бројачот, а не само на првиот флип-флоп. На овој начин се овозможува синхроно побудување на флип-флоповите, заради што ваквите бројачи и се викаат *синхрони бројачи* или *бројачи со паралелен влез*. Кај овие бројачи, промената на состојбата за сите флип-флопови (степени) ќе може да се врши истовремено со појавата на тактот, т.е. со бројачката низа која на него се приклучува. Таа промена нема да зависи од редоследот на флип-флопот во однос на почетниот, т.е. нема да има чекање, со што преодниот режим се скратува и јасно доаѓа до зголемување на брзината на работа.

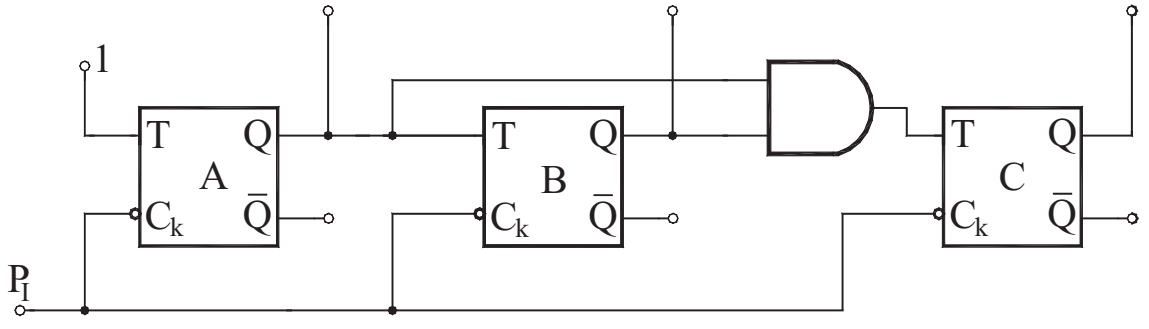
Како што ќе видиме во натамошното излагање, ваквото зголемување на брзината се остварува со внесување на дополнителни логички кола во структурата на бројачот што од друга страна, ќе доведе до усложнување на неговата изведба.

6.5.1. БИНАРЕН СИНХРОН БРОЈАЧ

Знаејќи дека сите бројачи изброените импулси ги прикажуваат во бинарен облик, јасно е дека за реализација на бинарни синхрони бројачи со основа (модул) M_0 која е цел степен на бројот 2 ($M_0=2^n$), ќе треба да се употребат n мемориски елементи (флип-флопови), исто како и во случајот на бинарните асинхрони бројачи.

Бидејќи кај синхрониот бројач бројачките импулси ќе се носат истовремено до влезовите за такт кај сите флип-флопови (степени), ќе треба дополнителна логика која нема да дозволи првиот импулс да им ја смени состојбата на сите флип-флопови одеднаш, што би довело до грешка во работата.

Да се потсетиме дека бројачот работи правилно тогаш кога првиот степен ја менува состојбата со секој влезен импулс, вториот ја менува состојбата само кога претходниот степен ќе се наоѓа во состојба на логичка единица, а за побудување на третиот степен и двата претходни флип-флопови треба истовремено да се во состојба на логичка единица. Четвртиот степен D ќе треба да ја менува состојбата само кога A и B и C се на високо ниво, итн. Во врска со кажаното станува јасно дека ќе треба да се применат И кола кои ќе имаат задача да обезбедат ускладување на работата на флип-флоповите имајќи го предвид такт-сигналот, т.е. бројачката низа која кај овие бројачи како такт-сигнал се носи истовремено до сите влезови за такт на флип-флоповите.



Сл. 6-9. Логичка структура на октален синхрон бројач со T MS флип-флопови

Во врска со ова, на сл. 6-9 е претставен еден пример на логичката шема на октален синхрон бројач (бинарен бројач со основа 8) ($M_0=8$), реализиран со T MS флип-флопови.

Првиот флип-флоп реагира на секој влезен импулс, вториот на секој втор, а третиот на секој четврт, заради што на Т влезовите од секој флип-флоп се доведува побудата:

- на А логичка единица ($T_A=1$),
- на В излезот од А ($T_B=A$, т.е. $T_B=Q_A$), и
- на С излезите од А и В (T_C = А В, т.е. $T_B=Q_AQ_B$).

(6-11)

Овие изрази покажуваат дека на влезот Т од флип-флопот А треба постојано да биде присутно високо ниво, степенот А директно се врзува на степенот В, додека за побуда на флип-флопот С е потребно И коло со два влеза на кои ќе се приклучени излезите од А и В.

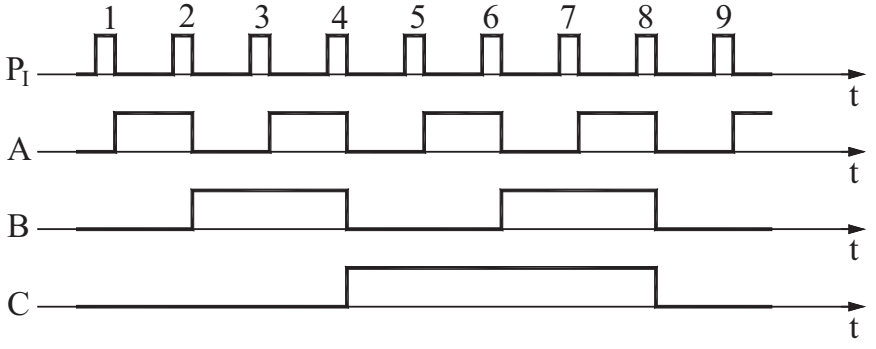
Ваквата конфигурација последователно го генерира природниот бинарен код со три бита со што добиваме идентична работа како кај асинхрониот октален бројач: ... *000, 001, 010, 011, 100, 101, 110, 111* ... односно декадно ... 0, 1, 2, 3, 4, 5, 6, 7 ... итн.

Состојби		Излези	Влезови
S_i	K_i	С В А	T_C T_B T_A
0	0	0 0 0	0 0 1
1	1	0 0 1	0 1 1
2	2	0 1 0	0 0 1
3	3	0 1 1	1 1 1
4	4	1 0 0	0 0 1
5	5	1 0 1	0 1 1
6	6	1 1 0	0 0 1
7	7	1 1 1	1 1 1
0	0	0 0 0	

Таб. 6-4. Табела на вистинитост на синхрон октален бројач нанапред

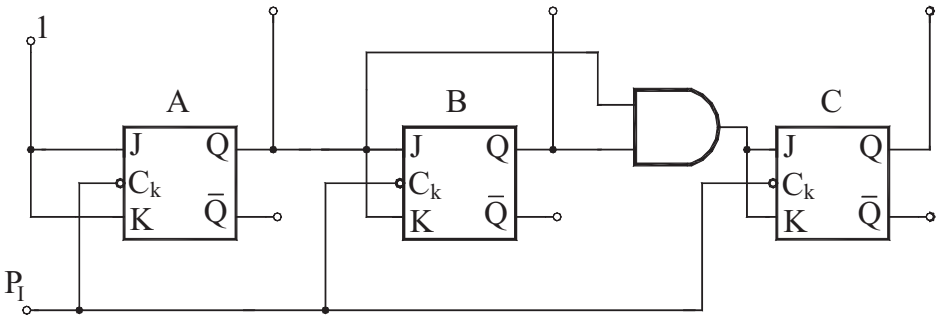
Комбинационата табела таб. 6-4 ги прикажува сите состојби на бројачот (S_i), нивните бинарни вредности, т.е. комбинациите на излезите на трите флип-флопови и нивните индекси (K_i) во декадна нотација. Секоја состојба е претставена со различна бинарна комбинација и соодветен индекс.

Состојбите на бројачот се менуваат според редоследот на бинарните броеви во природниот бинарен броен систем од *000* до *111* (од 0 до 7 декадно), па бројот на состојбата на бројачот е еднаков со индексот на секоја комбинација што таа состојба ја претставува ($S_i=K_i$). Временските дијаграми со кои дополнително се опишува и објаснува принципот на работа на бројачот се дадени на сл. 6-10.



Сл. 6-10. Временски дијаграми на октален синхрон бројач

На сл. 6-11 е прикажана уште една реализација на синхрон бинарен бројач со модул 8 ($M_0=8$), но сега со примена на ЈК MS флип-флопови.

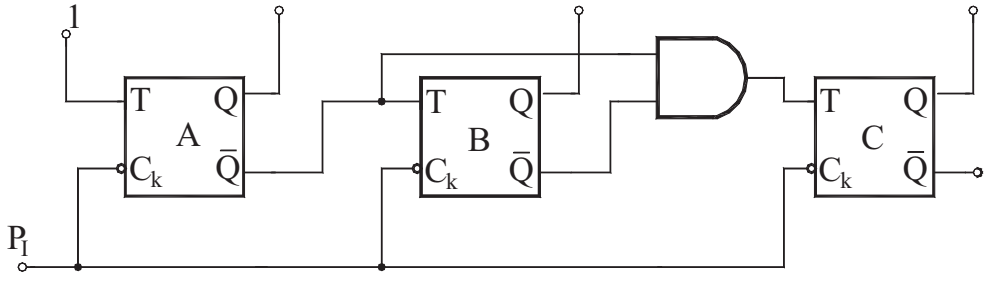


Сл. 6-11. Логичка структура на октален бројач со ЈК MS флип-флопови

6.5.2. БИНАРЕН СИНХРОН БРОЈАЧ НАНАЗАД

Поаѓајќи од принципот на работа на бројачите наназад, а знаејќи ги логичките структури на бинарните асинхрони бројачи наназад и на синхроните бројачи нанапред, може да се претпостави структурата на синхрониот бројач наназад. Една таква принципиелна шема на октален бројач наназад реализиран со T MS флип-флопови е прикажана на сл. 6-12.

Намалувањето на вредноста на бројачот за еден е овозможено со поврзувањето на комплементарниот излез од претходниот флип-флоп на влезот од следниот флип-флоп. Бидејќи почетната состојба е најголемата ($S_0=111$), броењето оди кон најмалата ($S_7=000$): *... 111, 110, 101, 100, 011, 010, 001, 000, ...,* или декадно ... 7, 6, 5, 4, 3, 2, 1, 0, ...

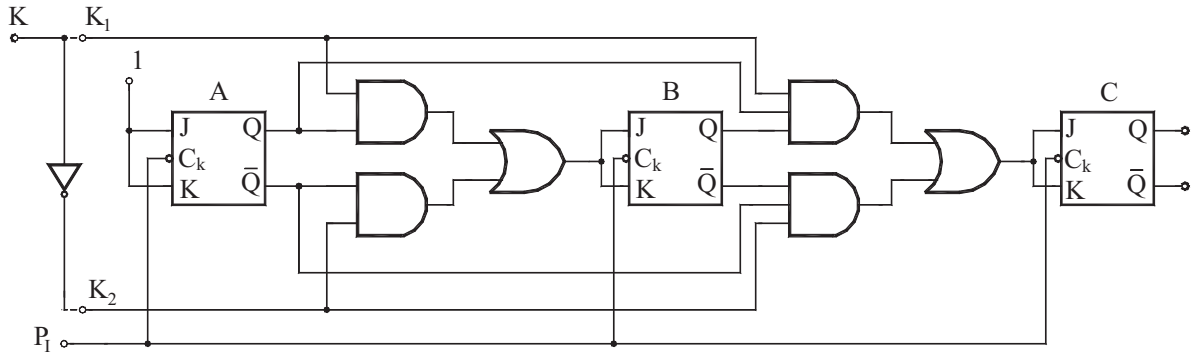


Сл. 6-12. Логичка структура на октален синхрон бројач наназад

6.5.3. БИНАРЕН СИНХРОН ДВОНАСОЧЕН БРОЈАЧ

Во продолжение, во куси црти ќе се задржиме и на билатералниот синхрон бројач кој може да брои во двете насоки: нанапред или наназад. Бидејќи и бројачите нанапред и оние наназад, имаат иста конструкција, но различно последователно поврзување, тоа значи дека пред секој флип-флоп ќе треба да се формира логичка мрежа која во основа ќе работи како 2-во-1 мултиплексер и ќе може да ги реализира и двете можности.

Бројачот од сл. 6-13 е еден пример на комбиниран октален бројач, т.е. бинарен бројач по модул 8 ($M_0=8$) со паралелен влез, реализиран со ЈК MS флип-флопови, што може да брои нанапред или наназад. За да се постигне тоа, побудата на секој флип-флоп оди преку логичка И–ИЛИ мрежа која овозможува селективност во насоката на броење преку контролната линија *K*. Имено, кога на неа се доведе високо ниво ($K=1$), таа ги отвора сите И кола на кои се поврзани директните излези од флип-флоповите кои преку ИЛИ колата ги побудуваат следните флип-флопови, па бројачот брои нанапред. Спротивно, кога нивото на *K* е ниско ($K=0$), преку инверторот се отвораат сите И кола на кои се поврзани комплементарните излези од флип-флоповите што преку ИЛИ колата се проследуваат до следните флип-флопови, па бројачот брои наназад.



Сл. 6-13. Логичка структура на октален двонасочен синхрон бројач

6.5.4. ПРОЕКТИРАЊЕ НА СИНХРОН БРОЈАЧ СО ПРОИЗВОЛНА ОСНОВА

Многу често, различните проблеми во практиката наметнуваат потреба од синхрони бројачи со произволна (небинарна) основа, меѓу кои најчесто се бара реализација на декаден бројач, т.е. бројач по модул 10 (*M=10*). Во врска со ова, однапред се познати основата на бројачот *M* кој треба да се проектира, различна од 2^i (*i=1,2,3, ...*), како и типот на флип-флоповите со кои тој бројач треба да се реализира. При синтезата на бројачките синхрони мрежи, треба да се определат дополнителни логички кола со кои ќе се реализираат повратни врски врз флип-флоповите заради исполнување на претходно поставените услови. Поконкретно, структурата на бројачот и формирањето на потребната комбинациска мрежа преку која ќе се реализираат повратните врски, зависи од модулот на бројачот *M* и типот на флип-флопови што треба да се употребат за негова градба.

Состојбите што треба да бидат прескокнати се несакани (некорисни, илегални или забранети) состојби и нивниот број е *ΔM (ΔM = M_0 - M)*. Во општ случај, постои опасност бројачот да влезе во некоја од овие илегални состојби и потоа да кружи само низ нив со што ќе работи погрешно. Заради претходно забележаното, проектирањето може да се врши и со дополнителен услов, а тоа е бројачот да излегува од секоја несакана состојба и да влегува во некоја легална, т.е. дозволена состојба. Обично при дизајнирањето на бројач со произволен модул се бара од секоја илегална состојба бројачот да влезе во почетната состојба која најчесто е ресетирана т.е. сите излези на флип-флоповите да се нули.

И во овој случај на дизајнирање синхрон бројач, како и во претходните реализации на бинарни или небинарни бројачи, најчесто се користат Т или ЈК флип-флопови. Бројот на потребни флип-флопови *n* се одредува од познатиот услов: $2^{n-1} \le M \le 2^n$, каде *M* е зададената основа на бројачот (должината на циклусот на броење). Разликата е во тоа што при проектирањето на синхрон бројач треба да се користат табелите на побуда (екситација) таб. 4-9 или таб. 4-11 на употребените флип-флопови.

Процесот започнува со формирање на табелата на вистинитост за бараниот бројач. По колони, како независни променливи се земаат излезите од флип-флоповите *A, B, C, ...*(Q_A, Q_B, Q_C, …) што стојат на располагање, додека функциите се претставени со влезовите Т, односно J и K, на секој од дадените флип-флопови.

Понатаму, се продолжува со пополнување на (*M+1*)-те редици за излезите од флип-флоповите наведувајќи ги сите состојби на бројачот (S_i), нивните бинарни комбинации и индекси (K_i), при што *i=0, 1, 2, 3, ...* итн. Се започнува од почетната, т.е. првата состојба на бројачот S_0 и нејзиниот индекс K_0, потоа втората (S_1), па третата (S_2), итн… сé до последната *M*-тата состојба (S_{M-1}), за конечно да се земе уште еднаш почетната состојба S_0, но сега како (*M+1*)-ва.

Користејќи ја соодветната табела на побуда, зависно од тоа кои флип-флопови ги имаме на располагање, таб. 4-9 за ЈК, односно таб. 4-11 за Т, една по една, се пополнуваат десните колони од комбинационата табела, т.е. вредностите на функциите, при што се споредува претходната со следната состојба на секој од флип-флоповите, дали таа се променила или не, се донесува заклучок за тоа каква треба да биде неговата побуда (вредноста на Т, односно на J и K влезовите) и истата се внесува во табелата во пресекот на редицата на анализираната состојба и колоната на разгледуваниот флип-флоп.

Конечно, со минимизација на секоја функција, поточно влезна променлива на секој флип-флоп (Т, односно J и K), се добива бараната комбинациона мрежа со која се остварува поврзувањето на *n*-те флип-флопови во единствена секвенцијална мрежа што ја претставува структурата на бараниот бројач со зададената основа *M*.

6.5.4.1. СИНТЕЗА НА СИНХРОН БРОЈАЧ СО ОСНОВА 10

Во практиката, од небинарните бројачи најголема примена имаат *декадните бројачи* чија основа на броење (модул) е 10 (*M=10*), заради што бројачкиот циклус минува низ 10 состојби, т.е. се состои од 10 различни бинарни состојби или секвенци. Толкав број на состојби може да се исполни со реализација на четиристепен бројач, односно со употреба на 4 флип-флопови (*n=4*) бидејќи $2^3 \le 10 \le 2^4$. Четиристепениот бројач овозможува $2^4=16$ различни состојби од кои ќе бидат искористени само 10, додека $\Delta M=16-10=6$ секвенци ќе останат неискористени, поточно ќе мора да се прескокнуваат бидејќи ќе се илегални. Враќањето на сите флип-флопови во почетната состојба треба да се случи по десеттиот импулс и со тоа да заврши циклусот на броење. Кои од состојбите ќе се одберат како непотребни или квазисостојби, зависи од тоа кој бинарен (BCD) код ќе се примени.

Состојби		Излези	Влезови
S_i	K_i	DCBA	$T_D\ T_C\ T_B\ T_A$
0	0	0000	0001
1	1	0001	0011
2	2	0010	0001
3	3	0011	0111
4	4	0100	0001
5	5	0101	0011
6	6	0110	0001
7	7	0111	1111
8	8	0101	0001
9	9	0101	1001
0	0	0000	

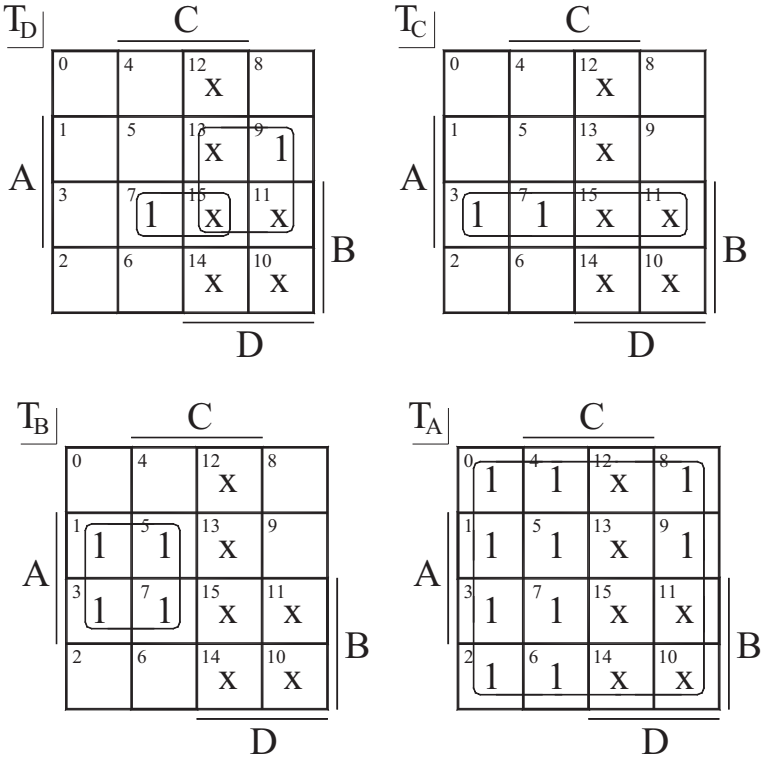
Таб. 6-5. Табела на вистинитост на декаден бројач

Најчесто се користи декаден бројач во природниот BCD код (NBCD, т.е. 8421 кодот), бидејќи редоследот на неговите состојби се поклопува со редоследот на комбинациите на конвенционалниот бинарен бројач кој ги опфаќа првите 10 комбинации од можните 16, почнувајќи од состојбата на логичка нула во сите степени $Q_AQ_BQ_CQ_D=0000$ (декадно 0), па сé до десеттата состојба кога $Q_AQ_BQ_CQ_D=1001$ (декадно 9). Последните шест состојби: $Q_AQ_BQ_CQ_D=1010,\ 1011,\ 1100,\ 1101,\ 1110,\ 1111$ (декадно 10, 11, 12, 13, 14 и 15) се забранети и како такви се неискористени.

Циклусот на броење завршува по десет импулси, кога повторно се воспоставува почетната состојба $Q_AQ_BQ_CQ_D=0000$.

При синтезата на декадниот NBCD бројач ќе претпоставиме дека на располагање стојат T MS флип-флопови. Имајќи ги во предвид состојбите на флип-флоповите по секој изброен импулс и при тоа користејќи ја табелата на екситација на T флип-флопот (т.10-10) треба да се пополни табелата на вистинитост таб. 6-5. Врз нејзина основа се продолжува со минимизација на секој од T-влезовите на употребените флип-флопови.

За минимизација на секоја функција, поточно влезна променлива на флип-флоповите се применува Карноовата метода која е прикажана на сл. 6-14 а), б), в) и г). Во процесот на минимизација се претпоставува дека шесте нелегални состојби 10, 11, 12, 13, 14 и 15 нема да настапат, заради што вредноста на било која функција за наведените комбинации е неважна. Таа може да се смета или за *0* или за *1* и како таква да се маркира со „x”.

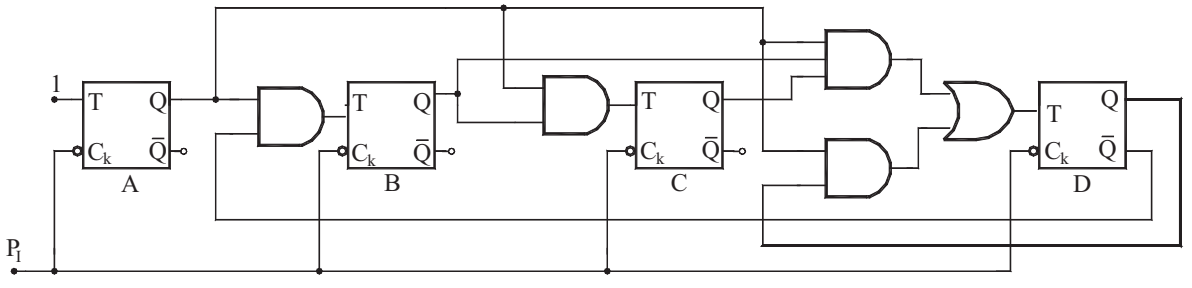


Сл. 6-14. Минимизација со метод на Карноови карти кај синхрон декаден бројач

По спроведената минимизација за влезовите на мемориските елементи се добиваат следниве логички равенки:

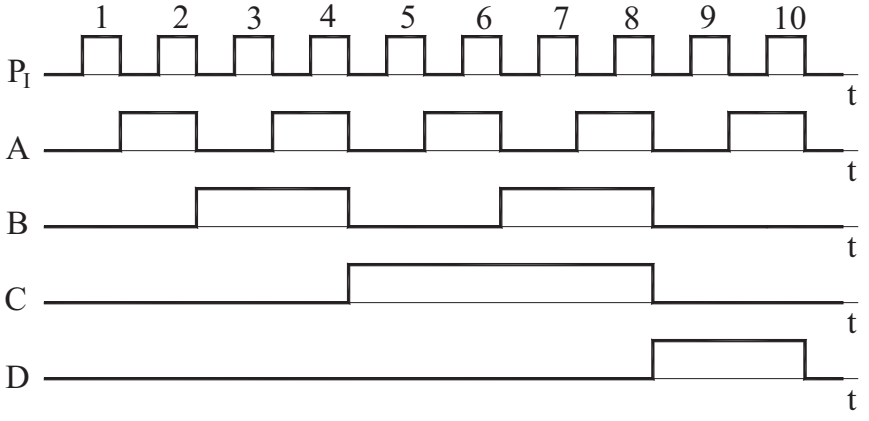
- за флип-флопот A: $T_A=1$,
- за флип-флопот B: $T_B= A\underline{D}$,
- за флип-флопот C: $T_C=AB$, и
- за флип-флопот D: $T_D=ABC+AD$.

(6-12)



Сл. 6-15. Логичка шема на декаден NBCD бројач

Со равенките 6-12 практично се одредени типот на логички кола и начинот на нивното за поврзување со флип-флоповите, од каде што произлегува и логичката шема на NBCD бројачот прикажана на сл. 6-15.



Сл. 6-16. Временски дијаграми на декаден NBCD бројач

Покрај логичката структура од сл. 6-15, заради подобро разбирање на процесот на работа, на сл. 6-16 се претставени и временските дијаграми на декадниот бројач. Од нив се гледа дека сè до 10-иот импулс бројачот работи како бинарен, а степените *A*, *B*, *C* се однесуваат како Т флип-флопови во префрлувачки режим на работа: степенот *A* реагира на секој импулс, степенот *B* на секој втор, а степенот *C* на секој четврт. Седмиот импулс ги сетира првите три флип-флопови, додека осмиот импулс го сетира последниот флип-флоп *D* и истовремено ги ресетира останатите *A*, *B* и *C*. По деветтиот импулс флип-флоповите *B* и *C* ќе бидат ресетирани, а сетирани се само првиот и последниот степен, *A* и *D*. Десеттиот импулс воспоставува почетна состојба на сите флип-флопови (*A, B, C, D = 0000*, т.е. $Q_A=0$, $Q_B=0$, $Q_C=0$ и $Q_D=0$) со што завршува еден циклус на броење.

6.6. КРУЖНИ БРОЈАЧИ

Логичката структура на кружните бројачи потсетува на поместувачките регистри бидејќи е составена од D тип флип-флопови. Ова е суштинска разлика во однос на бинарните бројачи што до сега ги анализиравме бидејќи тие главно се базираа на Т флип-флоповите, или на ЈК флип-флопови поврзани како Т.

Покрај наведеното, кај кружните бројачи се работи за едноставна бројачка мрежа која е затворена со повратна врска од излезот на влезот и тоа без употреба на логички кола. Ова е уште една разлика во однос на конвенционалните бројачи чии флип-флопови беа каскадно спрегнати (еден по друг) и имаа отворен или слободен излез со евентуална примена на одредени логички кола за меѓусебно поврзување и/или реализација на повратни врски што зависеше од основата (модулот) на броење.

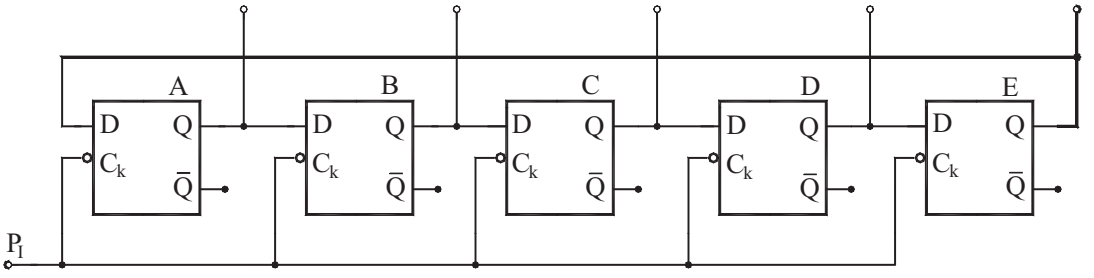
Дополнителна разлика е и основата на броење во однос на бројот на употребени мемориски елементи (*n)*. Имено, кога се употребуваат *n* флип-флопови за градба на бинарен бројач, неговата основа изнесува $M=M_0=2^n$, додека кај кружните бројачи модулот ќе биде *n* или *2n*, т.е. *M=n* или *M=2n*. Ова од една страна претставува недостаток на кружните бројачи во однос на бинарните, поради фактот што порастот на модулот кај кружните бројачи предизвикува брзо зголемување на бројот на флип-флопови *n*. Меѓутоа, од друга страна, кружните бројачи имаат предност во поедноставниот начин на поврзување и принципот на работа и броење.

6.6.1. КРУЖЕН БРОЈАЧ СО ОСНОВА 5

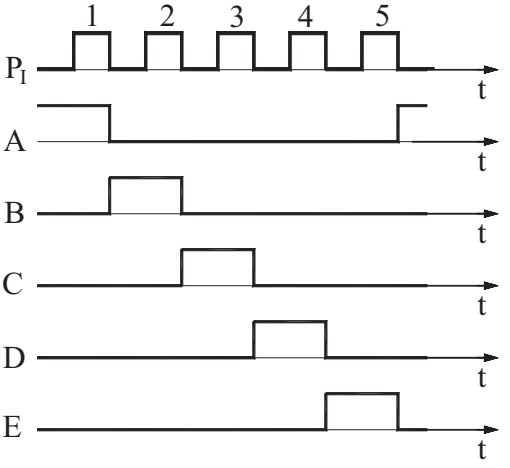
Како еден пример за кружен бројач со основа *n* ќе го анализираме квинарниот бројач кој има основа на броење 5 (*M=n=5*) и користи 5 флип-флопови (*n=5*) од D тип. Во табела таб. 6-6 се прикажани сите состојби на бројачот (излезните комбинации на флип-флоповите) со соодветните индекси во декадна нотација, како и комбинационите вредности во бинарен облик и екситационите вредности на D влезовите. Врз основа на таб. 6-6, на сл. 6-17 е нацртана логичката шема на оваа бројачка мрежа.

S_i	K_i	EDCBA	D_E D_D D_C D_B D_A
0	1	00001	00010
1	2	00010	00100
2	4	00100	01000
3	8	01000	10000
4	16	10000	00001
0	1	00001	

Таб. 6-6. Комбинациона табела на кружен квинарен бројач



Сл. 6-17. Логичка шема на квинарен кружен бројач



Сл. 6-18. Временски дијаграми на квинарен кружен бројач

За влезните функции на флип-флоповите можат да се напишат следниве равенки:

$D_A=EP_I$, $D_B=AP_I$, $D_C=BP_I$, $D_D=CP_I$ и $D_E=DP_I$. (6-13)

каде што *A, B, C, D* и *E* се излезите на флип-флоповите Q_A, Q_B, Q_c, Q_D и Q_E .

Временските дијаграми со кои попрегледно може да се објасни функционирањето на бројачот се дадени на сл. 6-18. Во почетната состојба на бројачот е сетиран само првиот степен (*A=1*), додека сите други степени се ресетирани (*B=C=D=E=0*). Кога ќе се доведе првиот бројачки импулс паралелно на сите влезови за такт, тој ја менува состојбата на првиот и вториот флип-флоп: на *A* и на *B*. Имено, влезот *D* на флип-флопот *A* што е приклучен на излезот од последниот флип-флоп *E* се наоѓа на ниско ниво ($D_A=Q_E=0$), додека излезот на *A* што е приклучен на влезот *D* на *B* е на високо ниво ($Q_A=D_B=1$). Заради ова се ресетира флип-флопот *A*, а се сетира *B*. Со донесување на вториот импулс се ресетира степенот *B*, а се сетира само степенот *C*, додека останатите излези и понатаму се на ниско ниво.

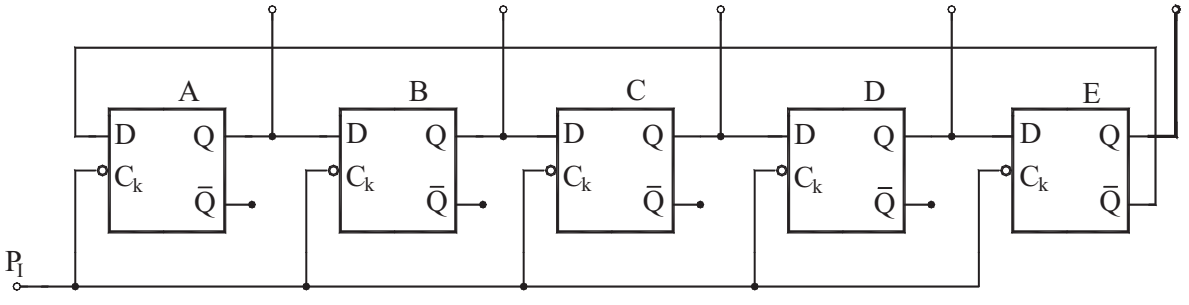
Процесот се повторува и со секој такт сигнал високото ниво што почетно беше присутно на излезот од првиот флип-флоп *A* последователно се пренесува кон влезовите на следните флип-флопови *B*, *C*, *D* и *E*. Заради повратната врска од последниот флип-флоп (E) до првиот (A), петтиот импулс ќе го сетира повторно флип-флопот *A*, а ќе го ресетира *E*, со што ќе се заврши циклусот на броење и ќе се воспостави почетната состојба.

Кружен бројач по модул *n* може да се реализира и со примена на JK или SR флип-флопови, така што излезот Q од секој претходен флип-флоп ќе оди на влезот S, односно J од секој следен флип-флоп, додека излезот Q ќе треба да се приклучи на влезот R, односно K. Ваквиот начин на поврзување ќе се примени и од последниот до првиот флип-флоп: од E на A. Практично, станува збор за истата конфигурација на D флип-флопови, бидејќи ваквото поврзување всушност е трансформација на JK или SR флип-флоповите во флип-флопови од D тип.

6.6.2. ДЕКАДЕН КРУЖЕН БРОЈАЧ

Кружниот бројач со модул *2n*, од конструктивна гледна точка претставува модификација на бројачот со основа *n*. Имено, промената е во формирање на повратна врска од комплементарниот, а не од директниот излез на последниот флип-флоп до влезот *D* на првиот степен. Кружниот бројач со основа *2n* најчесто се сретнува под името *Џонсонов бројач*, но и како *кружен бројач со вкрстена повратна врска*.

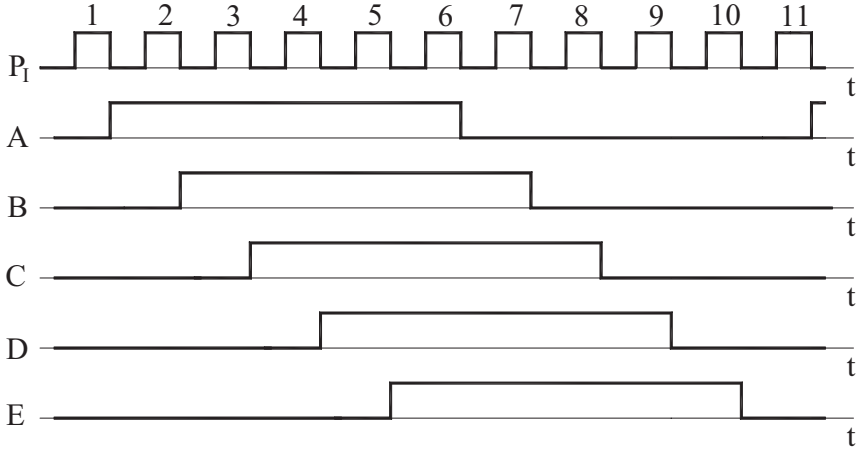
Како еден пример на бројач со модул *2n* ќе го претставиме кружниот бројач од слика сл. 6-19 чиј модул е 10 (*M=10*). Се работи за декаден бројач, составен од 5 флип-флопови (*n=5, M=2n=2x5=10*) од *D* тип кои се поврзуваат слично како кај бројачот по модул 5 од сл. 6-17, со единствена разлика во тоа што на влезот *D* во првиот флип-флоп *A* се поврзува комплементарниот, а не директниот излез од последниот флип-флоп *E*.



Сл. 6-19. Логичка шема на декаден кружен бројач со D флип-флопови

Принципот на работа на Џонсоновиот бројач ќе го објасниме преку неговите временски дијаграми прикажани на сл. 6-20. На почетокот сите флип-флопови се ресетирани па бројачот се наоѓа во ресетирана состојба: *ABCDE=0000*, т.е. $Q_AQ_BQ_CQ_DQ_E=00000$. Доведувањето на првиот бројачки импулс ќе го сетира првиот флип-

флоп *A* бидејќи само на неговиот влез е присутно високо ниво кое доаѓа од комплементарниот излез на последниот степен ($D_A=\overline{Q_E}$), додека останатите флип-флопови остануваат ресетирани. Вториот импулс не ја менува состојбата на *A*, неговиот излез останува висок, а истовремено го сетира и *B*. Ваквиот начин на работа, т.е. последователно сетирање на секој од флип-флоповите *C*, *D*, *E* продолжува сè до шестиот импулс кога се ресетира флип-флопот *A*, а сите други флип-флопови се сетирани. Сега почнува последователно ресетирање на секој од флип-флоповите, повторно почнувајќи од *A*, па преку *B*, *C* и *D* до *E*, што се случува со појавата на секој бројачки импулс, и тоа сè до десеттиот, кога повторно доаѓа до воспоставување на почетната состојба *ABCDE=0000*.



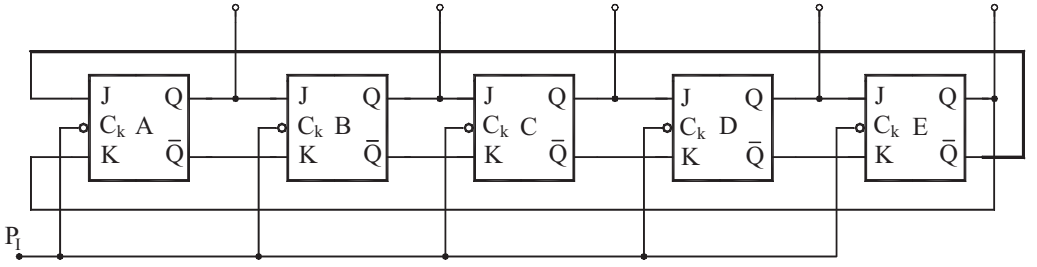
Сл. 6-20. Временски дијаграми на кружен декаден бројач

Преку комбинационата табела таб. 6-7 дополнително се разјаснува работата на бројачот. Во неа редоследно се презентирани сите 10 состојби во кои овој бројач може да се најде.

S_i	K_i	EDCBA	D_E D_D D_C D_B D_A
0	0	00000	00001
1	1	00001	00011
2	3	00011	00111
3	7	00111	01111
4	15	01111	11111
5	31	11111	11110
6	30	11110	11100
7	28	11100	11000
8	24	11000	10000
9	16	10000	00000
0	0	00000	

Таб. 6-7. Табела со комбинациони вредности на кружен декаден бројач

За конструкција на ваков бројач можат да се употребат ЈК или SR флип-флопови ако се трансформираат во D. Имено, секој директен излез од претходниот флип-флоп ќе треба да се приклучи на влезот J од следниот флип-флоп, комплементарниот излез ќе се поврзи на влезот К ($J=Q$, $K=\overline{Q}$), додека врската од последниот до првиот флип-флоп ќе се оствари така што директниот излез на последниот флип-флоп ќе се приклучи на влезот К на првиот, а комплементарниот излез на влезот J ($J_A=\overline{Q_E}$, $K_A=Q_E$), според логичкиот блок-дијаграм претставен на сл. 6-21.



Сл. 6-21. Логичка шема на декаден кружен бројач со ЈК флип-флопови

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

6-1. Која е основната намена на бројачите? Кои се основните градбени елементи што ја формираат структурата на бројачот?

6-2. Кои од наведените дигитални компоненти може да се користат како основен градбен елемент на бројачите; а) флип-флопови управувани со работ на такт-сигналот, б) MS флип-флопови; в) леч кола? Објасни го одговорот.

6-3. Што претставува една состојба на бројачот?

6-4. Каква е врската помеѓу бројот на употребени флип-флопови во структурата на бројачот и бројот на состојби во кои тој може да се најде?

6-5. Што претставува основа на броење (модул) на бројач? Што е бројачки циклус? Што е капацитет на бројачот? Како се тие поврзани со бројот на употребени флип-флопови во структурата на бројачот и неговата основа?

6-6. Која е а) основата на броење б) капацитетот на бројач составен од 1) n=2, 2) n=3, 3) n=4 флип-флопови?

6-7. Како се делат бројачите во однос на основата (модулот) на броење?

6-8. Кои типови флип-флопови најчесто се користат за конфигурирање на бројач? Во каков режим на работа? Како се поврзуваат?

6-9. Дали бројачите можат да се користат како делители на фреквенција? Образложи го одговорот!

6-10. Нацртај блок шема на бројач со а) бинарна б) декадна индикација.

6-11. Која е разликата помеѓу бинарните бројачи и оние со небинарна основа?

6-12. Како се делат бројачите според насоката на броење?

6-13. Какви бројачи постојат според начинот на приклучување на импулси?

6-14. Кои се разликите помеѓу асинхроните (паралелните) и синхроните (сериските) бројачи?

6-15. Кои се состојбите низ кои минува бинарен бројач реализиран со а) n=2, б) n=3, в) n=4 флип-флопови? Кои се тие комбинации, во бинарно и децимално означување, и по кој редослед се повторуваат?

6-16. Што опфаќа анализата на даден бројач? Што треба да е познато, а што се бара?

6-17. Што треба да е познато за да се проектира одреден бројач?

6-18. Нека е даден бинарен асинхрон бројач реализиран со: а) n=2, б) n=3, в) n=4, г) n=5 флип-флопови. Одреди ги: 1) модулот (M_0), 2) капацитетот (N_K), 3) почетната и 4) крајната состојба на овој бројач.

6-19. Нацртај логичка шема, комбинациона табела и временски дијаграми за бинарен асинхрон бројач со основа а) $M_0=4$, б) $M_0=8$, в) $M_0=16$ реализиран со 1) Т, 2) ЈК, 3) SR MS (master - slave) флип-флопови што имаат директен влез за ресетирање $\overline{R_d}$.

6-20. Даден е асинхрон бинарен бројач по модул а) $M_0=4$, б) $M_0=8$ и в) $M_0=16$ што се напојува со извор +Vcc=5V и е побуден со низа на импулси со периода $T_P=1[\mu sec]$. Нацртај ги временските дијаграми на бројачот и пресметај ја фреквенцијата на влезните импулси (f_P), како и фреквенциите на импулсите што се добиваат на излезот од првиот ($f1$), вториот ($f2$), ... итн., последниот флип-флоп.

6-21. Нацртај логичка шема, комбинациона табела и временски дијаграми за бинарен асинхрон бројач наназад по модул а) $M_0=4$, б) $M_0=8$, в) $M_0=16$ со примена на 1) Т, 2) ЈК, 3) SR MS флип-флопови што имаат директен влез за ресетирање $\overline{R_d}$.

6-22. Која е главната карактеристика на двонасочниот бројач? Кој е основниот принцип што овозможува две насоки на броење кај билатералниот бројач?

6-23. Ако претпоставиш дека на располагање ти стојат а) n=3; б) n=4; в) n=5 флип-флопови, определи ги сите можни модули на бројачи со небинарна основа $M\neq M_0$.

6-24. Која е основната разлика помеѓу легална (дозволена) и илегална (недозволена, забранета) состојба кај бројачите со произволна основа?

6-25. Посебно наведи ги и објасни ги, чекорите од постапката (процедурата) за добивање на асинхрони небинарни бројачи со произволна основа.

6-26. Проектирај асинхрон бројач со произволна основа M= а) 5; б) 6; в) 7; г) 9; д) 10; ѓ) 11; е) 6; ж) 13; з) 14; s) 15 ако на располагање ти стојат 1) Т, 2) ЈК MS флип-флопови со влез за директно ресетирање $\overline{R_d}$. Нацртај ги неговата логичка структура, временските дијаграми и комбинационата табела.

6-27. Во што меѓусебно се разликуваат асинхроните и синхроните (редните и паралелните) бројачи?

6-28. Нацртај логичка шема, табела на вистинитост и временски дијаграми за бинарен синхрон бројач со основа а) $M_0=4$, б) $M_0=8$ и в) $M_0=16$ со примена на MS 1) Т, 2) ЈК флип-флопови што имаат директен влез за ресетирање $\overline{R_d}$. На располагање ти стојат И кола со произволен број влезови.

6-29. Објасни го принципот на проектирање синхрони бројачи со произволен модул $M\neq M_0$.

6-30. Проектирај синхрон бројач со основа а) 5, б) 6, в) 9, г) 10, д) 11, ѓ) 6, ѓ) 14 со примена на 1) ЈК, 2) Т MS флип-флопови. Во процесот на решавање претпостави дека 1) бројачот никогаш нема да се најде во некоја илегална состојба; 2) ако се најде во илегална состојба, треба да премине во почетната (ресетирана) состојба.

6-31. Кој е основниот начин на поврзување на флип-флоповите во формирањето на логичката структура кај кружните бројачи?

6-32. Каква врска постои помеѓу бројот на употребени флип-флопови и основата на броење (модулот) на кружниот бројач реализиран со нив?

6-33. Кои се разликите помеѓу асинхроните и синхроните бројачи во однос на кружните?

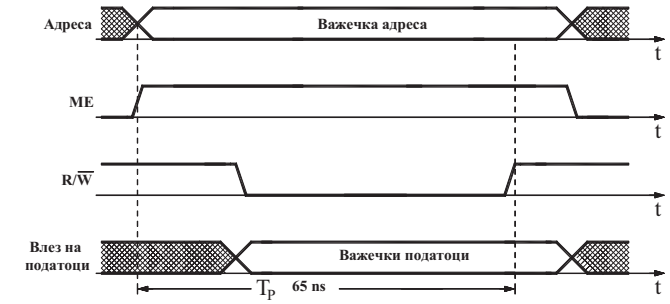
6-34. Ако на располагање ти стојат а) n=4, б) n=5, в) n=6 флип-флопови, одговори какви кружни бројачи во поглед на основата на броење можат да се реализираат?

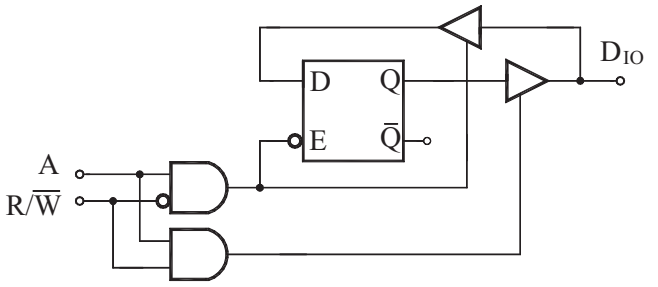
6-35. Анализирај го принципот на работа на квинарниот кружен бројач.

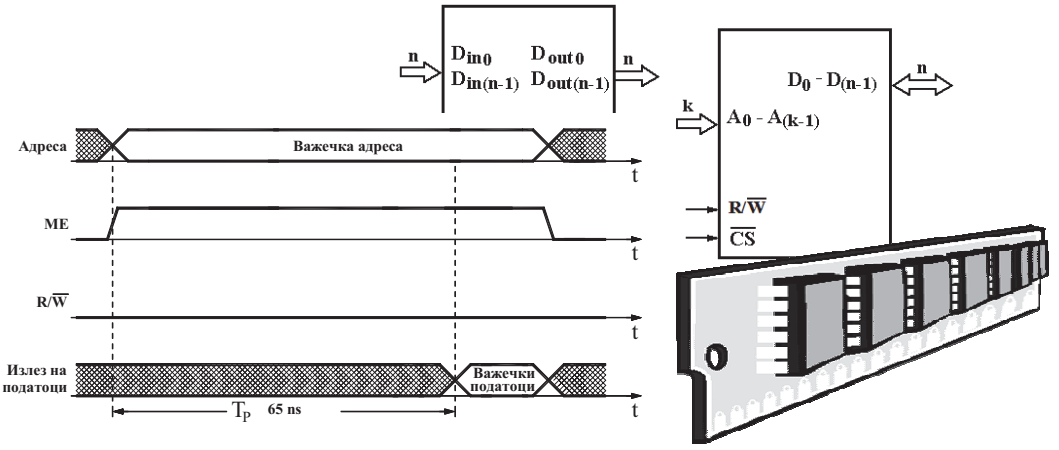
6-36. Нацртај логичка шема, табела на вистинитост и временски дијаграми за кружен бројач со основа а) M=4, б) M=5, в) M=6 со примена на 1) D, 2) ЈК, 3) SR MS флип-флопови. Објасни го принципот на работа.

6-37. Анализирај го принципот на работа на декадниот кружен бројач.

6-38. Нацртај логичка шема, табела на вистинитост и временски дијаграми за кружен бројач со основа а) M=8, б) M=10, в) M=6 со примена на 1) D, 2) ЈК, 3) SR MS флип-флопови. Објасни го принципот на работа.







7. МЕМОРИСКИ КОМПОНЕНТИ

По изучувањето на оваа тематска целина

- ќе ги познавате основните поими и концепти кои се однесуваат на мемориските компоненти и уреди;
- ќе се запознаете со организацијата на меморијата;
- ќе умеете да ги споредувате различните типови на мемории: ROM, PROM, EPROM, EEPROM, RAM;
- ќе можете да ги опишувате сличностите и разликите помеѓу мемориските интегрирани кола;
- ќе знаете да го објасните начинот на адресирање на меморијата;
- ќе ја познавате логичката структура на RAM мемориската ќелија и нејзиното функционирање;
- ќе разберете како се одвиваат процесите на читање и запишување во меморијата, а со тоа ќе умеете да го објасните и принципот на нејзината работа;

7.1. ВОВЕД

Мемориските компоненти се едни од основните и главни составни делови на дигиталните системи. Меморијата има фундаментална улога, посебно кај компјутерите, бидејќи нејзина улога е да чува (запаметува) најразлични типови на податоци, кои претставуваат кодирани информации во бинарен облик. Основната функција на меморијата е да овозможува читање на внесените податоци, како и нивно менување (ажурирање) преку запишување на нови. Мемориите се посебни компоненти составени од голем број на логички кола реализирани со електронски (полупроводнички) елементи: биполарни и униполарни транзистори.

Во оваа тема најнапред ќе се запознаеме со хиерахијата на мемориските компоненти и уреди, со терминологијата и дефинирањето на клучните поими кои се однесуваат на мемориите со посебен осврт на полупроводничките мемории, како и на нивната внатрешна организација. Потоа ќе се задржиме на поделбата на мемориите според различни критериуми. Притоа ќе ги споредиме сличностите и разликите помеѓу повеќето типови мемориски компоненти, донесувајќи соодветни заклучоци за нивните предности и слабости.

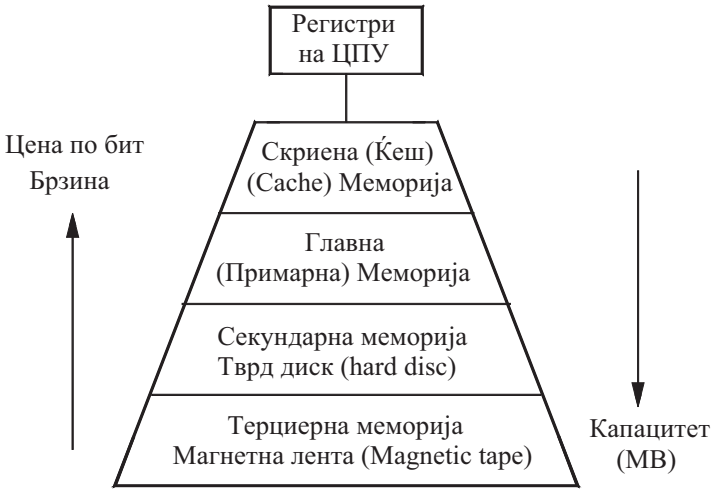
Заради полесно разбирање на функционирањето на мемориските компоненти ќе бидат претставени и анализирани: моделот на меморијата, принципиелната електрична шема - логичката структура, и блок-шемата на најмалата елементарна градбена единица на меморијата - мемориската ќелија. Во врска со ова ќе биде објаснет и принципот на работа на меморијата, со анализа на улогата на нејзините влезни и излезни линии и со користење на временски дијаграми при анализирање на двете основни операции: читање и запишување.

7.2. МЕМОРИСКА ХИЕРАРХИЈА

Кога се обработува меморијата како проблематика, неодминливо треба да се анализира и т.н. мемориска хиерархиска пирамида која на еден сликовит начин нè воведува во поделбата на мемориските компоненти и уреди и воедно нè запознава со односот помеѓу нивната цена на чинење, брзина на работа и капацитет.

Мемориската пирамида, која е прикажана на сл. 7-1, на својот врв ја носи т.н. скриена или ќеш (анг. Cache) меморија која во однос на другите типови мемории има најмал капацитет, но од друга страна има најголема брзина при работата. На второто ниво се наоѓа т.н. главна или примарна меморија, која е поголема по капацитет од ќеш меморијата, но е поспора од неа во однос на брзината. За овие две највисоки нивоа карактеристично е тоа што тие се изградени од полупроводнички елементи. Денес тие најчесто се изработуваат од униполарни NMOS транзистори или во комплементарна MOS технологија (CMOS). Како главни претставници на следните хиерахиски нивоа се единиците со тврди дискови (анг. hard-disk) кои се претставници на т.н. секундарна меморија, додека единиците со магнетна лента (анг. magnetic tape) припаѓаат на уште пониското ниво на т.н. терцијална меморија.

Последните два типа на мемориски уреди се всушност магнетни мемории, бидејќи читањето и запишувањето на податоците се врши на магнетен медиум: намагнетизирани тенки дискови, односно намагнетизирана лента, за чие движење се потребни механички елементи. Како последица на тоа, овие мемориски уреди имаат најмала брзина на работа, но затоа многу поголем капацитет во однос на полупроводничките мемории.



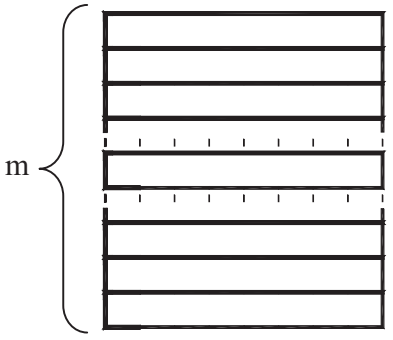
Сл. 7-1. Пирамида на мемориската хиерархија

Од сликата сл. 7–1 се гледа дека полупроводничките мемориски компоненти како предност ја имаат големата брзина на работа, но нивна слабост е тоа што имаат мал капацитет и висока цена на чинење по бит во однос на магнетните мемории. Од друга страна, мегнетните мемориски уреди, кои се наоѓаат на дното на пирамидата, се одликуваат со најниска цена на чинење по бит и најголем капацитет, но нивната голема слабост е малата брзина на работа во однос на полупроводничките мемории.

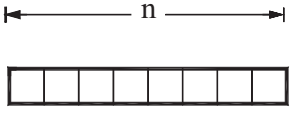
Понатаму во излагањето ќе се фокусираме на презентирање на карактеристиките и анализа на принципот на работа на мемориските полупроводнички интегрирани компоненти, кои припаѓаат на највисокото хиерархиско ниво, ќеш меморијата.

7.3. ИНТЕРНА ОРГАНИЗАЦИЈА НА МЕМОРИЈАТА И БАЗИЧНИ ПОИМИ И КОНЦЕПТИ

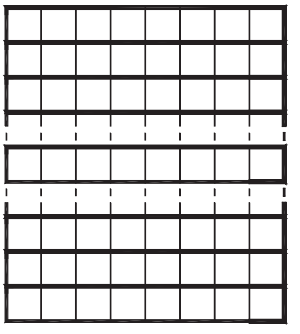
Од организациски аспект логичката структура на меморијата претставува конечно и уредено множесто на голем број мемориски локации (m) во форма на табела или матрица според сл. 7-2 која ја прикажува на наједноставен начин.



Сл. 7-2. Мемориска табела



Сл. 7-3. Мемориски збор



Сл. 7-4. Мемориска матрица

Во секоја локација може да се наоѓа одреден *мемориски збор* според сл. 7-3 кој претставува некаков податок кодиран во бинарен облик како низа (комбинација) од 0-и и 1-и со фиксна должина (n) изразена во битови (бинарен вектор). Мемориският збор може да се смести во одредена мемориска локација, која содржи конечен број на мемориски ќелии, еднаков со должината (бројот на битови) на зборот, според сл. 7-4. Имено, мемориската ќелија претставува најмала елементарна градбена единица на секоја мемориска компонента, бидејќи во неа може да се запамети (меморира) само еден бит

(еднобитен податок): 0 или 1. Најзначајниот бит (MSB) (битот со најголема тежина) од податокот, кој се наоѓа на најлевата позиција вообичаено се означува со $d_{(n-1)}$, следниот до него со $d_{(n-2)}$, и така сè до последниот најдесно поставен бит, кој е најмалку значаен (LSB) (бит со најмала тежина) и се означува со d_0. Бидејќи во практиката се користат меморији чии зборови имаат должина од 1, 2, 4 или 8 бајти, најчесто n ќе биде 8, 16, 32 или 64 бита. Имајќи предвид дека секој мемориски збор може да претставува одреден податок, кој е кодиран во бинарен облик и има должина n бита, конкретниот бинарен вектор, т.е. конкретната комбинација од 0-и и 1-и која е сместена во мемориската локација ја претставува нејзината *содржина*. Ова значи дека во рамките на меморијата секој *мемориски збор* претставува посебна целина (ентитет) со должина од n битови до која може да се пристапи со цел да се прочита нејзината содржина или да се измени преку запишување на нова. Токму преку реализација на овие две основни операции на *читање* и *запишување* мемориската компонента ја остварува својата функција.

Во врска со ова, кога се користи *терминот запишување* се мисли на запамтување (меморирање, внесување, анг. Write) на мемориски збор (нов податок) кој доаѓа од компонента или уред кој се наоѓа надвор од меморијата и се запишува во неа како нова содржина во една (било која) зададена мемориска локација од целото расположиво множество на мемориски локации. Старата содржина на специфицираната мемориска локација неповратно ќе се изгуби бидејќи ќе биде заменета со нова.

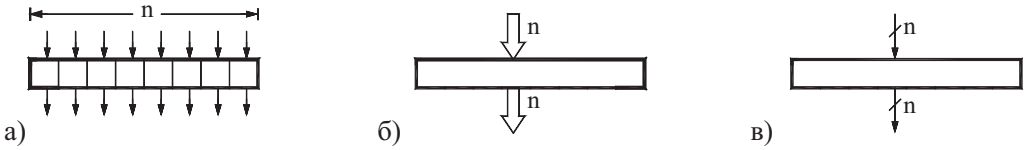
Од друга страна, кога се употребува *поимот читање* (анг. Read), поконкретно се мисли на изнесување на мемориски збор (постоечки запаметен податок) од меморијата кон некоја надворешна компонента или уред со што практично се чита содржината која е сместена во зададената мемориска локација. Постојат два различни начини на читање: деструктивно и недеструктивно. При *деструктивното читање* содржината на мемориската локација се губи, додека при *недеструктивното читање* нејзината содржина останува каква што била непосредно пред читањето. Во практиката многу почесто се сретнува недеструктивното читање и кога се користи терминот читање вообичаено се смета дека тоа е недеструктивно.

Согласно со претходното, содржината на мемориската локација може да се чита само како единствена целина со фиксна должина од n бита или во неа да се запишува нова содржина, со истата должина.

Реалните мемориски интегрирани кола располагаат со посебна влезна контролна линија која вообичаено се означува со $R/\overline{W}$, $\overline{WR}$ или $\overline{WE}$. Ако на неа се доведе 1 ($R/\overline{W}=1$) тоа значи дека од меморијата може да се чита, додека ако состојбата на оваа линија е 0 ($R/\overline{W}=0$), тоа значи дека во мемориската компонента може да се запишува.

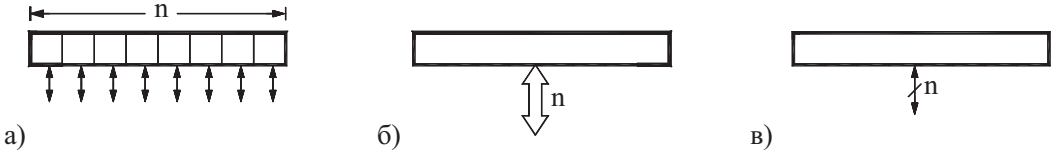
Имајќи во вид дека новата содржина најбрзо може да се прочита или промени на паралелен начин, со едновремено пристапување до сите мемориски ќелии, се наметнува заклучокот дека за секој бит од податокот ќе треба да се користи по една посебна т.н. податочна линија бидејќи во даден момент преку една линија може да се пренесува само еден бит. Имено, напонското ниво кое е присутно на неа може да биде ниско $V_L=V_{LOW}=V(0)$ што одговара на логичка 0, или високо $V_H=V_{HIGH}=V(1)$ што соодветствува на логичка 1.

Според ова, ако запишувањето на нов податок (содржина) во зададена мемориска локација се извршува преку едни водови, а читањето преку други, тоа може симболички да се претстави според сл. 7-5. На сликите множеството паралелни податочни линии (сл. 7-5 а) имаат иста намена и заради тоа поедноставено се означуваат со двојна широка линија (сл. 7-5 б) или со подебела прецртана линија (сл. 7-5 в) покрај која е напишан вкупниот број на водови (жици). Со ваквата нотација се поедноставува цртањето на логичките шеми и воедно им се се зголемува нивната прегледност.



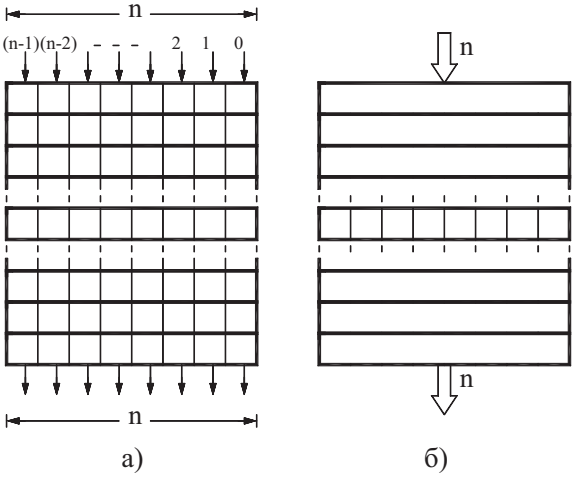
Сл. 7-5. Симболи на мемориска локација со посебни линии за читање и запишување

Имајќи во вид дека истовремено не може и да се чита и да се запишува, бидејќи во еден момент се пристапува на мемориската локација и нејзината содржина или се чита, или се запишува нова, наместо употреба на две групи податочни линии: едни за влез, а други за излез, може да се користи само едно множество на линии. Тие во моментот кога се чита ќе бидат излезни ($R/\overline{W}$=1), а тогаш кога ќе се запишува влезни ($R/\overline{W}$=0), што симболички се прикажува со сл. 7-6 а), б) и в). Во практиката кај реалните мемориски компоненти податочните линии најчесто се двонасочни (бидирекционални).

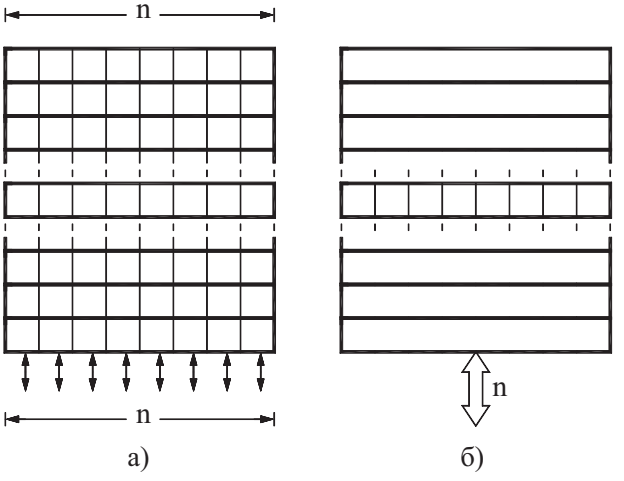


Сл. 7-6. Симболи на мемориска локација со двонасочни линии за читање/запишување

Бидејќи во даден момент се пристапува само до една мемориска локација, бројот на податочни линии со кои располага една мемориска компонента е идентичен со должината на меморискиот збор кој е сместен во неа, според сл. 7-7 а) б) и сл. 7-8 а) б). Во моментот кога се остварува комуникацијата помеѓу меморијата и некоја друга компонента, на единствените податочни линии од мемориското коло ќе се појават или постават битовите кои се однесуваат само на специфицираната локација. Ваквата структура и поврзување овозможува бројот на податочни линии да биде еднаков со должината само на еден мемориски збор, т.е. со бројот на мемориски ќелии во една мемориска локација (n).



Сл. 7-7. Симболи на мемориски компоненти со посебни линии за читање и запишување

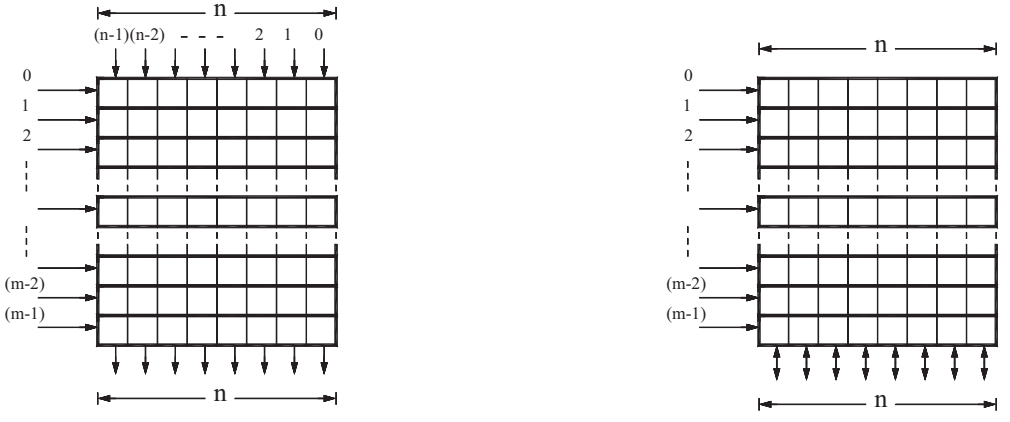


Сл. 7-8. Симболи на мемориски компоненти со двонасочни линии за читање/запишување

Реалните мемориски интегрирани кола располагаат со многу голем број локации. Вкупниот број на мемориски локации со кои располага дадена мемориска компонента се означува со m, при што m е цел број многу поголем од 1 (m>>>1) кој вообичаено се движи во опсегот од 2^{10}=1024=1K≈10^3, 2^{20}=1024x1024=1M≈10^6, до 2^{30}=1024x1024x1024=1G≈10^9, што зависи од тоа за каков дигитален систем станува збор: микропроцесорски, микроконтролерски, персонален компјутерски или друг поголем систем.

Бидејќи операциите читање или запишување можат да се однесуваат на било која мемориска локација, ќе треба да постои можност за пристап и пребарување на целата меморија од нејзината прва до нејзината последна локација. За да може ефикасно да се оствари пристап до било која мемориска локација, секој мемориски збор се нумерира (индексира) со посебен број кој претставува негова еднозначно доделена *адреса*. Адресата ќе се користи за одредување на мемориската локација во која даденият збор ќе се запише или од каде ќе се прочита. Бидејќи секој мемориски збор има посебна адреса и мемориска локација, која е еднозначно поврзана со него, тоа значи дека при операцијата читање или запишување најмалиот податок до кој може да се пристапи е токму еден мемориски збор кој се наоѓа во специфицираната мемориска локација. Во врска со ова, кога се пристапува до некоја мемориска локација за да се изврши зададена операција на читање или запишување се користи терминот *адресирање* или *специфицирање*. На овој начин, со користење на адреси, може да се обезбеди мемориските зборови да се движат од надворешните компоненти кон меморијата, и обратно. На тој начин секој податок може да се запише или да се прочита од мемориската локација чија адреса е зададена.

Од кажаното се наметнува заклучокот дека мемориското коло треба да има посебни влезни адресни линии со кои ќе може во даден момент да се врши адресирање (селектирање) на една, било која од вкупниот број на расположиви мемориски локации (m), што значи дека нивниот број треба да биде еднаков со вкупниот број на мемориски зборови. На овој начин, со активирање на некоја од адресните линии, додека останатите се држат пасивни, ќе и се пристапи токму на мемориската локација која е адресирана и во неа ќе може да се запишува нов збор (податок), или ќе може да се прочита нејзината содржина (зборот, податокот кој е зачуван во неа), додека останатите мемориски локации ќе бидат пасивни. Во тој момент на n–те податочни линии ќе се појави содржината на посочената локација ако истата се чита, односно на нив ќе треба да се постави нов податок ако во локацијата треба да се запише нова содржина. Вообичаено, првиот мемориски збор има нулта адреса, т.е. адресата на првата мемориска локација се означува со 0 (m=0), додека последниот збор, а со тоа и последната мемориска локација, како највисока адреса ќе ја има вредноста (m-1), според сл. 7-9 а) или б).



а) со посебни линии за читање и запишување б) со двонасочни линии за читање/запишување

Сл. 7-9. Означување на адресите во мемориската матрица

Бидејќи бројот на мемориски локации m е многу голем, огромен би бил и бројот на адресни линии, со кои надворешно би се адресирала меморијата, заради што адресирањето на мемориските локации (пристапот до целиот адресен простор, мемориската матрица) се врши преку адресен декодер, кој влегува во составот на мемориската компонента.

Со неговата примена бројот на надворешни адресни линии се намалува во голема мерка, поточно со логаритамска зависност, бидејќи за да се добијат на излезот од декодерот m линии, на неговиот влез се потребни само k линии при што треба да биде исполнет условот $k=\log_2 m$, т.е. $m=2^k$.

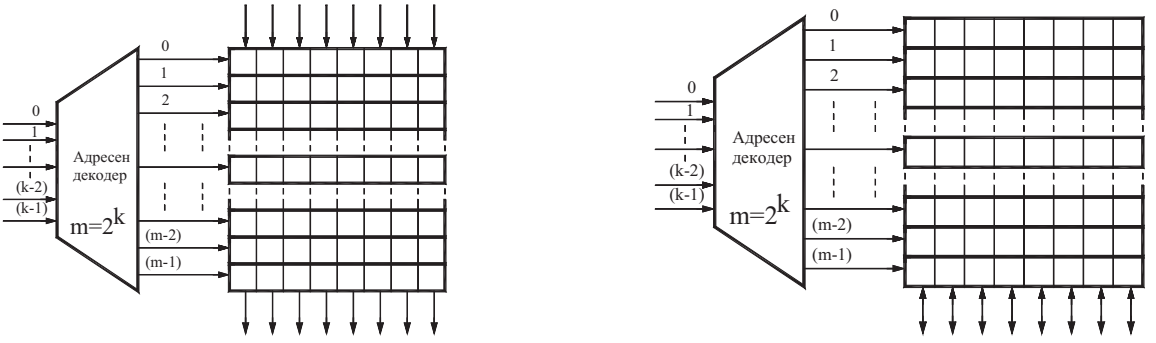
Во врска со последниот заклучок вкупниот број на мемориски локации m со кои располага дадена мемориска компонента се добива како степен на бројот 2, така што секогаш ќе важи равенката

$$m = 2^k \qquad (7\text{-}1)$$

каде што k е цел број поголем од 1 (k>>1), кој вообичаено се движи во опсегот од 10 до 30.

Заради примената на адресниот декодер, адресната информација ќе претставува низа битови (бинарен вектор) кои по декодирањето ќе активираат точно одредена мемориска локација. Поточно, ќе го активираат оној мемориски збор чија адреса во декадна нотација соодветствува на бинарно кодираната адреса донесена на адресните линии. Имајќи го предвид фактот дека вкупниот број на мемориски зборови е $m = 2^k$ и дека првиот мемориски збор има нулта адреса, (m=0) , последниот збор како највисока адреса ќе ја има вредноста (2^k-1) што одговара на (m-1).

Бидејќи сите битови на адресата се носат едновремено (паралелно) до влезот на декодерот, ќе биде потребна една група (множество) од k адресни линии. Ова значи дека адресата на секоја мемориска локација ќе претставува k-битен вектор (бинарен број) чија ширина е еднаква со бројот на адресни линии: $a_{(k-1)}a_{(k-2)}\ldots a_2a_1a_0$, според сл. 7-10 а) б).



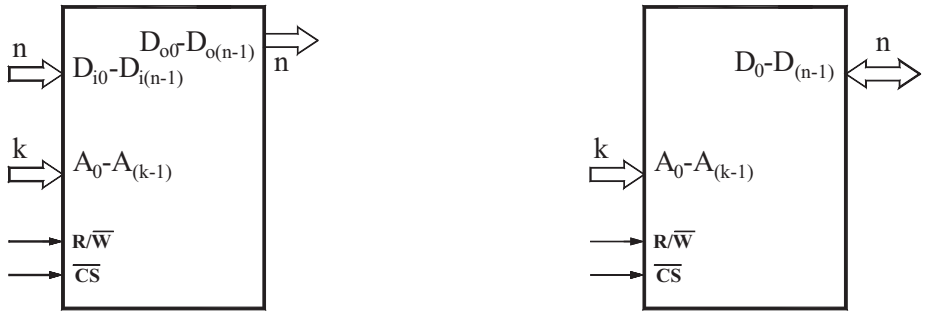
а) со посебни линии за читање и запишување б) со двонасочни линии за читање/запишување

Сл. 7-10. Еднодимензионално адресирање на мемориски локации со адресен декодер

Декодирањето на адресата се врши според природниот бинарен броен систем според кој битот што се наоѓа на најлевата позиција има најголема тежина $a^{(k-1)}$, додека најдесниот бит има најмала тежина 2^0=1. Ваквиот пристап до меморискиот простор кога за адресирање се користи единствен адресен декодер претставува *линарно* или *еднодимензионално адресирање.* Во практиката најчесто се сретнува т.н. дводимензионално адресирање кога до мемориската матрица се пристапува со два адресни декодери: еден посебен за редици еден посебен декодер за колони.

Покрај податочните ($d_{(n-1)}$ $d_{(n-2)}$ … $d_2d_1d_0$) и адресните линии ($a_{(k-1)}a_{(k-2)}\ldots a_2a_1a_0$), како и контролната линија за читање/запишување ($R/\overline{W}$ или $\overline{WE}$), мемориските компоненти без исклучок имаат уште еден многу важен контролен влез кој најчесто се означува со $\overline{CS}$ или $\overline{ME}$, според англиските термини Chip Select или Memory Enable. Станува збор за сигнал со кој се врши селекција и овозможување на работа на мемориската компонента (меморискиот чип, мемориското интегрирано коло). Овој сигнал $\overline{CS}$ (или $\overline{ME}$) е активен на ниско ниво, што значи дека тој има ефективно дејство врз мемориското интегрирано коло само ако е исполнет условот $\overline{CS} = 0$ (или $\overline{ME}$ =0). Само во овој случај меморискиот чип е селектиран и меморијата е ставена во функција: таа може да се адресира и со неа може да се извршуваат операциите читање или запишување. Меѓутоа, ако на овој влез се донеси високо логичко ниво $\overline{CS} = 1$ (или $\overline{ME}$ =1), мемориската компонента ќе биде пасивна (нема ништо да работи) бидејќи сите нејзини линии ќе одат во состојба на висока отпорност, така што меморијата ќе биде практично исклучена од дигиталниот систем во чиј што состав влегува.

Како заклучок, на сл. 7-11 а) б) се дадени симболичките ознаки на мемориски компоненти со сите влезни и излезни податочни, адресни и контролни линии.



а) со посебни линии за читање и запишување б) со двонасочни линии за читање/запишување

Сл. 7-11. Симболички ознаки на мемориски компоненти

Капацитет на меморија

Капацитетот на меморијата го претставува вкупниот број на битови (b) или бајти (B) со кој располага зададената мемориска компонента и него ќе го означиме со w. Капацитетот може лесно да се добие ако се помножи вкупниот број на мемориски локации (зборови) со нивната должина според следнава равенка

$$w = m \times n \qquad (7\text{-}2)$$

Капацитетот најчесто се изразува во единици што се многу поголеми од битовите (b) или бајтите (B) бидејќи реалните мемориски компоненти кои практично се применуваат располагаат со големи капацитети. Во врска со ова, најчесто ќе се сретнуваме со единици како што се кило (K), мега (M) и гига (G):

$\Rightarrow$ $1K=2^{10}=1024\approx10^3$,

$\Rightarrow$ $1M=2^{20}=2^{10}x2^{10}1024x1024\approx10^6$,

$\Rightarrow$ $1G=2^{30}=2^{10}x2^{10}x2^{10}=1024x1024x1024\approx10^9$,

Применувајќи ја равенката (7-1) во равенката за мемориски капацитет (7-2) се добива уште една, многу почесто користена равенка за одредување на капацитетот на меморијата

$$w = 2^k \times n \qquad (7\text{-}3)$$

Решени примери:

Заради појаснување на она што досега го презентиравме и анализиравме, во продолжение ќе решиме неколку примери со зададени конкретни вредности.

Пример 7-1: Содржината на некоја мемориска компонента со мал капацитет која располага со m=16 зборови е прикажана во облик на табела или матрица (Таб.7-1). Во матрицата зборовите се прикажани како редици, при што секој збор е составен од конечен број битови (n=8), кои ги претставуваат колоните. Зборовите на мемориското коло се со должина од по 8 бита (n=8), т.е. 1 бајт (B) што значи дека нејзиниот капацитет според равенката (7-1) изразен во битови е w=16 збора x 8b/збор = 128 b, додека изразен во бајти би бил w=16 збора x 1 B = 16 B.

Адреса на мемориска локација			Содржина на мемориска локација			
Бин.	Дек.	Хекса.	Бин.	Дек.	Хекса.	ASCII симбол
0000	0	1	01100001	97	61	a
0001	1	2	01100010	98	62	b
0010	2	3	01100011	99	63	c
0011	3	4	01100100	100	64	d
0100	4	5	01100101	101	65	e
0101	5	6	01100110	102	66	f
0110	6	6	01100111	103	67	g
0111	7	7	01101000	104	68	h
1000	8	8	01101001	105	69	i
1001	9	9	01101010	106	6A	j
1010	10	A	01101011	107	6B	k
1011	11	B	01101100	108	6C	l
1100	12	C	01101101	109	6D	m
1101	13	D	01101110	110	6E	n
1110	14	E	01101111	111	6F	o
1111	15	F	01110000	112	70	p

Таб. 7-1. Содржина на мемориско коло со 16 зборови од по 1 бајт

Во меморијата сите информации се чуваат во бинарен облик, токму како што и означено во првите две колони од адресата и податоците на табелата. Меѓутоа, ваквите ознаки се разбирливи за машината (компјутерот), но за човекот претставуваат проблем, како заради својата должина така и заради бинарната основа 2, заради што адресите и содржините на зборовите во меморијата се дадени во декаден облик, но и во хексадецимална нотација. Декадното означување е многу поблиско за разбирање од страна на човекот, додека со хексадецималното означување долгите бинарни низи (вектори) се прикажуваат на наједноставен и најкомпактен начин: секоја четворка битови (секој нибл) се заменува со единствена и соодветна хекса цифра. Од таб. 7-1 се гледа дека во меморијата се внесени ASCII кодовите на првите 16 мали букви од англиската абецеда.

Пример 7-2: Табелата Таб.7-2 ја прикажува содржината на првите четири мемориски локации од мемориска компонента која располага со поголем број на зборови во однос на претходната бидејќи, како што може да се види од дадената табела нејзиниот капацитет е $m=1024_{DEC}$ мемориски збора или 1 К зборови кои се означени со адреси почнувајќи од првата ($a=0_{DEC}$) до последната (1023_{DEC}) кои го претставуваат адресниот простор на меморијата. Покрај ова, од сликата се гледа дека секоја редица има по 8 полиња (мемориски ќелии) во кои може да се смести по 1 бит што нè упатува на заклучокот дека секој збор има должина n=8 бита или 1 бајт (1B).

Дополнително од табелата се гледа и тоа дека содржината на 0-та локација е $d_7d_6d_5d_4d_3d_2d_1d_0=01000001$, на 1-та е $01000010_{(2)}$, содржината на 2-та локација е 01000011, а во 3-та локација е сместена содржината 01000100. Ако пред себе ја имаме таблицата на стандарниот ASCII код, лесно ќе ги препознаеме податоците кои се запаметени во првите четири локации. Имено, според ASCII табелата во нултата локација е сместен кодот на буквата A, во првата се наоѓа кодот на B, во втората на C и во третата на D.

Адреса на мемориска локација			Содржина на мемориска локација			
Индекс		k=10 бита	ASCII симбол			n=8 бита
Hex	Ai(dec)	A_9A_8 … $A_2A_1A_0$	Ch	HEX	DEC	$D_7D_6D_5D_4D_3D_2D_1D_0$
000	0000	0 0 0 0 0 0 0 0 0 0	A	41	65	0 1 0 0 0 0 0 1
001	0001	0 0 0 0 0 0 0 0 0 1	B	42	66	0 1 0 0 0 0 1 0
002	0002	0 0 0 0 0 0 0 0 1 0	C	43	67	0 1 0 0 0 0 1 1
003	0003	0 0 0 0 0 0 0 0 1 1	D	44	68	0 1 0 0 0 1 0 0
004	0004	0 0 0 0 0 0 0 1 0 0	E	45	69	0 1 0 0 0 1 0 1
005	0005	0 0 0 0 0 0 0 1 0 1	F	46	70	0 1 0 0 0 1 1 0
006	0006	0 0 0 0 0 0 0 1 1 0	G	47	71	0 1 0 0 0 1 1 1
007	0007	0 0 0 0 0 0 0 1 1 1	H	48	72	0 1 0 0 1 0 0 0
008	0008	0 0 0 0 0 0 1 0 0 0	I	49	73	0 1 0 0 1 0 0 1
009	0009	0 0 0 0 0 0 1 0 0 1	J	4A	74	0 1 0 0 1 0 1 0
00A	0010	0 0 0 0 0 0 1 0 1 0	K	4B	75	0 1 0 0 1 0 1 1
...	...	. . .		...		. . .
...	...	. . .		...		. . .
3FE	1022	1 1 1 1 1 1 1 1 1 0	b	6 2	98	0 1 1 0 0 0 1 0
3FF	1023	1 1 1 1 1 1 1 1 1 1	a	6 1	97	0 1 1 0 0 0 0 1

Таб. 7-2. Содржина на мемориско коло со 1К зборови по 1 бајт

Знаејќи го вкупниот број на мемориски зборови и должината на секој мемориски збор изразена во битови, може да го определиме капацитетот w на зададената мемориска компонента согласно равенката (7-1). Пресметката за капацитетот на меморијата w изразен во бајти (B) и битови (b), последователно ќе биде:

w = m x n = 1024 збора x 8 b/збор = 2^{10} x 1 B = 1K x 1B, т.е. 1KB или 1024 B,

w = 1K x 8b, т.е. 8 Kb или 8096 b.

Меморијата прикажана на претходната таб. 7-2, можеме да ја искористиме и за претставување на операциите запишување и читање. Имено, да земеме дека компјутерот треба да изврши две инструкции: прво ја чита мемориската локација чија адреса е 3 и потоа прочитаната содржина ја запишува во почетната локација со адреса 0.

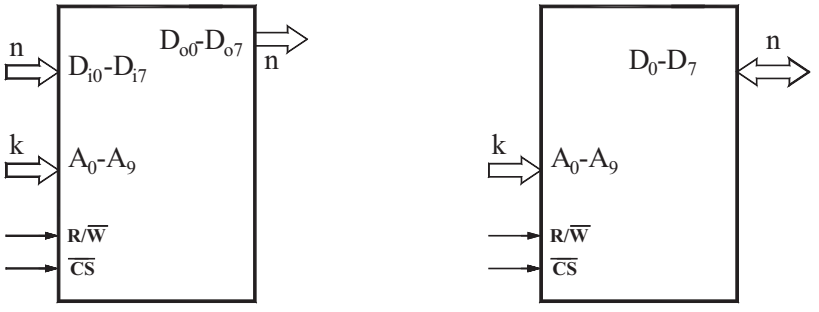
Ако претпоставиме дека читањето е недеструктивно, што се подразбира, тогаш по завршувањето на програмот содржината на мемориската локација со адреса 0 ќе биде иста со онаа на адреса 3 при што и во двете локации ќе биде сместен зборот $01000100_{(2)}$., т.е. во две мемориски локации ќе се чува ASCII кодот на симболот D. Од меморијата неповратно ќе се изгуби податокот $01000001_{(2)}$, т.е. ASCII кодот на буквата A кој беше сместен во првиот мемориски збор со нулта адреса според таб. 7-3.

Hex	Ai(dec)	A_9A_8… $A_2A_1A_0$	Ch	HEX	DEC	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
000	0000	0 0 0 0 0 0 0 0 0 0	D	44	68	0	1	0	0	0	1	0	0
001	0001	0 0 0 0 0 0 0 0 0 1	B	42	66	0	1	0	0	0	0	1	0
002	0002	0 0 0 0 0 0 0 0 1 0	C	43	67	0	1	0	0	0	0	1	1
003	0003	0 0 0 0 0 0 0 0 1 1	D	44	68	0	1	0	0	0	1	0	0

Таб. 7-3. Содржини на првите четири локации на мемориско коло со 1К зборови по 1 бајт

Пример 7-3: На сликата сл. 7-12 а) б) е прикажана блок-шема - модел на една мемориска компонента, која како и претходната има десет адресни линии (k=10) и осум податочни (n=8). Имајќи ја предвид претходната анализа и равенката (7-1) мемориското интегрирано коло располага со 1K мемориски локации (k=10 $\Rightarrow$ m=2^{10}=1024=1K) со должина од 8 бита (n=8 b =1 B), што значи дека нејзината внатрешна организација е m=1024 збора по n=8 бита од каде според равенките (7-2 и 7-3) произлегува дека капацитетот на меморијата изнесува w=1024 збора x 8 бита = 2^{10} x 1 бајт = 1 KB.

Првото интегрирано коло дадено на сл. 7-8 а) има две групи на податочни линии: влезни и излезни. На влезните линии се поставува податокот кој треба да се запише во меморијата, додека на излезните се добива податокот кој се чита од неа. Кај второто коло дадено на сл. 7-8 б) постои само едно множество податочни линии кои се двонасочни (бидирекционални): кога ќе се запишува податок во меморијата тие ќе бидат влезни, додека кога ќе се чита тие ќе бидат излезни.



а) со посебни линии за читање и запишување б) со двонасочни линии за читање/запишување

Сл. 7-12. Симболи на мемориска компонента организирана како 1 К зборови x 1 бајт

Пример 7-4: Запишување во меморија. Да се одредат логичките состојби на сите линии на зададеното мемориско коло со цел симболот „J“ (латиничната буква голема J според ASCII кодот) да се запиши во мемориска локација со адреса 5_{DEC}.

1. Избор на чип: $\overline{CS}$ =0;
2. Избор на адреса на мемориска локација: 5_{DEC}=a_{BIN}=0000000101. Оваа адресна информација се поставува на адресните линии (од A_9 до A_0);
3. Избор на операција запишување: $R/\overline{W}$ =0;
4. Внесување на податок „J“=d_{BIN}=01001010. Овој податок се поставува на податочните линии (од D_7 до D_0);

Пример 7-5: Читање. Да се одредат логичките состојби на сите линии на зададеното мемориско коло со цел да се прочита содржината на мемориската локација со адреса 6_{DEC}.

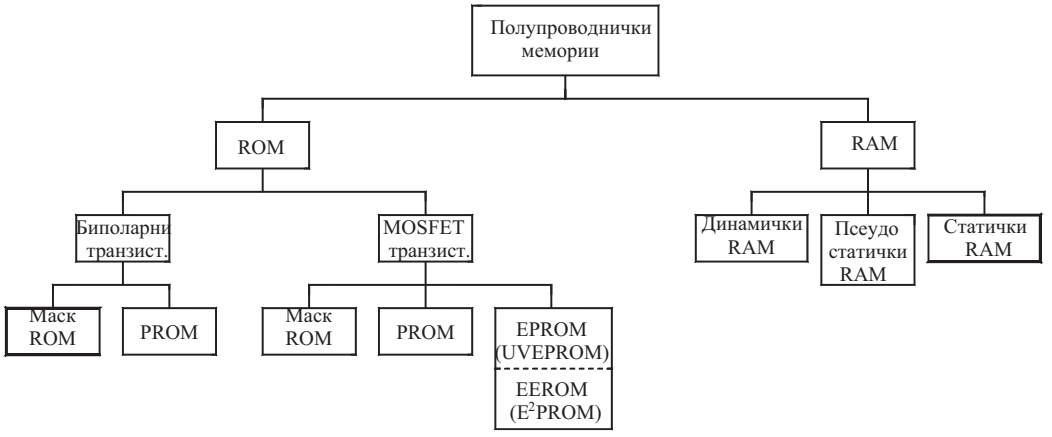
1. Избор на чип: $\overline{CS}$ =1;
2. Избор на адреса на мемориска локација: 6_{DEC}=a_{BIN}=0000001100. Оваа адресна информација се поставува на адресните линии (од A_9 до A_0);
3. Избор на операција читање: $R/\overline{W}$ =1;
4. Изнесување на податок „G“=d_{BIN}=01000111. Овој податок се поставува на податочните линии (од D_7 до D_0);

Голем број од мемориските чипови се произведуваат со две контролни линии: едната, $\overline{WE}$, овозможува запишување, додека втората, $\overline{OE}$ овозможува читање. При ова нивната логичка состојбата мора да биде комплементарна една на друга: се чита кога $\overline{WE}$ =1 и $\overline{OE}$ =0, додека се запишува ако е исполнет условот $\overline{WE}$ =0 и $\overline{OE}$ =1.

7.4. ПОДЕЛБА НА МЕМОРИСКИТЕ КОМПОНЕНТИ

Во натамошното излагање ќе се задржиме малку повеќе на поделбата на полупроводничките мемории, бидејќи тие ќе бидат во фокусот на излагањето што понатаму следува во рамките на оваа тема.

Во однос на тоа дали меморијата ја губи својата содржина со престанокот на напојувањето или не, мемориските компоненти се делат на две големи групи и тоа на деструктивни (анг. Volatile) и недеструктивни мемории (анг. Non/Volatile memories) како што е претставено на сл. 7-13.



Сл. 7-13 Поделба на полупроводничките мемориски компоненти.

Кај *недеструктивните мемории* податоците што се внесени не се губат со престанокот на напојувањето. Типичен и најпознат претставник на оваа група е т.н. ROM меморија, или меморија од која може само да се чита (анг. Read Only Memory). ROM-от се изработува како интегрирано коло во кое податоците се запишуваат со програмирање на внатрешните врски во структурата на меморијата. Во ROM мемориските компоненти содржината може да се внесе само еднаш и тоа во процесот на фабрикување на ваквото интегрирано коло. ROM интегрираните логички кола имаат само еден контролен сигнал за овозможено читање ($\overline{OE}$) бидејќи кај нив не е можно да се запишува нова содржина.

Програмабилната ROM меморија, која скратено се нарекува PROM, во основа претставува ROM компонента кај која корисникот може сам да ги програмира врските со употреба на посебен уред, PROM програматор, и со тоа да внесува содржина по свој избор. По програмирањето, PROM-от се однесува како ROM меморија бидејќи неговата содржина повеќе не може да се менува.

Посебна категорија на PROM мемориите се т.н. бришливи PROM интегрирани кола (анг. erasable programmable ROM) или скратено EPROM-и. Тие може да се програмираат исто како и PROM-мемориите, но покрај тоа нивната содржина може да се брише со нивно изложување на ултравиолетова (UV) светлина, со што постои можност за внесување на нова содржина во секоја од мемориските локации посебно и тоа по пат на повторно програмирање на врските.

EEPROM-от е уште една варијанта на PROM меморија, бидејќи неговата содржина може да се избрише и да се репрограмира, но бришењето се врши по електричен пат, со пуштање на струја со поголема јачина, а не со изложување на ултравиолетова (UV) светлина како во случајот на EPROM компонентите. Флеш (анг. Flash) мемориските интегрирани кола се посебна класа на EEPROM мемориски компоненти кај кои запишувањето се врши во блокови, а не бајт-по-бајт како кај класичниот EEPROM.

Во Таб. 7-4 е даден преглед на основните параметри на различните типови ROM мемории.

Тип	Технологија	Циклус на читање	Циклус на запишување	Коментар
Маск ROM	NMOS, CMOS	20-200 ns	4 недели	Еднократно запишување; мала моќност;
Маск ROM	Биполарна	<100 ns	4 недели	Еднократно запишување; голема моќност; мала густина;
PROM	Биполарна	<100 ns	5 минути	Еднократно запишување; голема моќност; без маска;
EPROM	NMOS, CMOS	25-200 ns	5 минути	Повеќекратно запишување; мала моќност; без маска;
EEPROM	NMOS	50-200 ns	10 μs/бајт	10,000 запишувања/локација;
FLASH	CMOS	25-200 ns	10 μs /блок	100,000 циклуси на бришење;

Таб. 7-4. Споредбени карактеристики на различни типови ROM мемории

Кај *деструктивните мемории* нивната содржина се брише со престанок на напојувањето, меѓутоа, додека постои напојување, нивната содржина може да се менува во секое време: да се читаат внесените податоци, да се бришат постоечките и да се запишуваат нови. Токму затоа нивната кратенка е RWM што произлегува од англискиот термин Read/Write Memory што би значело меморија која може да се чита и во која може да се запишува. Меѓутоа, за ваквите мемориски компоненти во практиката е нашироко распространета во употреба и веќе насекаде прифатена многу попозната кратенка RAM, која доаѓа од англискиот израз Random-Access Memory што значи меморија со случаен пристап, а се однесува на фактот што времето кое е потребно за да се прочита било кој бит не зависи од неговата местоположба во меморијата, поточно времето за пристап до било кој податок во меморијата е еднакво (униформно).

Статичките RAM мемории (SRAM) се посебна група на RAM компоненти, кај кои податоците се внесуваат и чуваат во мемориски ќелии реализирани со леч флип-флопови (кола за заклучување). Тие во нив може да се чуваат, читаат или да се менуваат сè додека мемориското интегрирано коло е приклучено на напојување. Со исклучување на напојувањето податоците неповратно се губат. SRAM мемориските ќелии се користат за изработка на ќеш (скриената) меморијата која има најголема брзина работа.

Динамичките RAM мемории (DRAM) се друга група на RAM компоненти, бидејќи кај нив податоците се чуваат како полнежи на многу мали кондензатори заради што со тек на времето тие се празнат, па се јавува потреба од често периодично обновување (освежување, анг. refresh) на нивниот полнеж што обезбедува зачувување на информациите кои се внесени во нив. Покрај асинхрониот DRAM во последно време се почесто кај персоналните сметачи се јавува и синхрониот DRAM (SDRAM) кај кој преносот се остварува синхроно со посебен такт сигнал и тоа со појавата на едниот раб на тактот. Како посебно брзи мемориски кола од ваков тип се познатите DDR (Double Data Rate) SDRAM интегрирани кола, кои работат со двојно поголема брзина на пренос на податоците, бидејќи пренесуваат податоци и на предниот раб и на задниот раб на такт сигналот. Со помош на DRAM мемориските интегрирани кола се реализира главната (оперативната, примарната) меморија, која е со поголем капацитет од ќеш меморијата, но од неа е поспора во однос на брзината на работа.

Псеудо статичката DRAM меморија (PSRAM) всушност е DRAM со интерни кола за освежување кои се однесуваат како SRAM.

7.5. ROM мемориски компоненти

ROM: ROM-от се изработува како интегрирано коло во кое податоците се запишуваат со програмирање на внатрешните врски во структурата на меморијата. Поточно, во процесот на програмирањето се воспоставуваат, односно се прекинуваат одредени интерни врски формирани преку транзистори изработени во биполарна или MOS технологија, кои внатрешно се поврзани како диоди. Во ваквите ROM мемории содржината може да се внесе само еднаш и тоа во процесот на фабрикување на ик според однапред изработена маска заради што и се викаат маск-програмабилни ROM мемории. Маската ја изработува производителот на ROM-от врз основа на барање што го поставува корисникот за тоа која треба да биде нејзината содржина.

PROM: PROM меморијата има голема предност во однос на ROM меморијата бидејќи корисникот може сам да ги програмира врските со употреба на PROM програматор. Во процесот на програмирање секоја од воспоставените врски може да се прекине ако низ неа протечат струјни импулси, што се прави преку програматорот, и со тоа да се внесе логичка 0, па на овој начин практично корисникот може да избира на кои позиции ќе ги остави 1-ците, а на кои ќе внеси 0-и. По програмирањето, PROM-от се однесува како ROM меморија бидејќи неговата содржина не може да се менува.

EPROM: EPROM-ите може да се програмираат исто како и PROM-мемориите, но дополнително нивната содржина може да се брише и постои можност пак да се внесе нова содржина. Ваквото однесување на EPROM-ите се должи на фактот што тие користат MOS транзистори со т.н. пливачки гејт бидејќи неговиот потенцијал може да се менува. Вообичаено EPROM-ите може да се препрограмираат 10 до 100 пати. Бришењето на содржината на EPROM-от се врши со негово изложување на ултравиолетова светлина заради што ваквите мемориски кола се викаат и UV EPROM-и. EPROM интегрираните кола на своето куќиште имаат прозирен дел преку кој се гледа чипот во внатрешноста на колото. По програмирањето прозирниот дел се покрива со заштитна фолија, која непосредно пред бришењето треба да се извади како би можел чипот да биде изложен на ултра-виолетовата светлина.

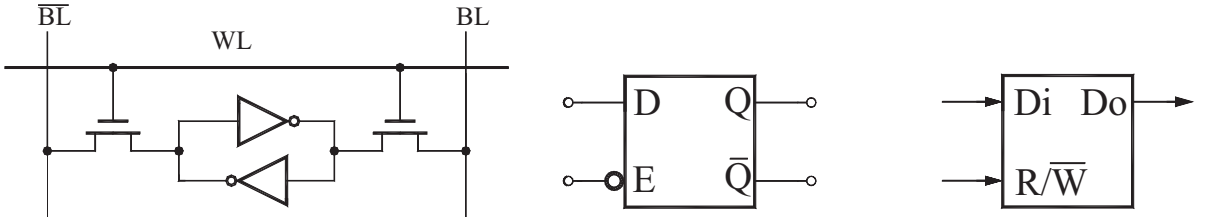
EEPROM: EEPROM-от (или E^2PROM-от) е уште една варијанта на PROM меморија, но кај овие мемориски компоненти бришењето се врши по електричен пат, со пуштање на струја со поголема јачина, а не со изложување на UV зраци. Циклусите на бришење и запишување може да се изведуваат и околу десетина илјади пати. Flash меморијата (FEPROM) е една варијанта на EEPROM бидејќи може електронски да се брише, но до 100.000 пати.

7.6. RAM

Концепцијата на SRAM меморијата е релативно едноставна за разбирање, бидејќи станува збор за матрица од мемориски ќелии формирани од леч флип-флопови, т.е. уредено множество на мемориски ќелии, организирани во редици и колони. Нив им претходи логика за контрола на процесите на читање/запишување, како и за декодирање на адресната информација. Во практиката се користат синхрони и асинхрони SRAM чипови. Асинхрониот SRAM е независен од тактот, кој ја синхронизира работата на компјутерот. Кај асинхрониот SRAM влезот и излезот на податоците се управуваат само со појавата на контролните сигнали за селекција на чип и читање/запишување. Кај синхрониот SRAM временското усогласување на адресата, влезните/излезните податоци и другите контролни сигнали зависат од такт сигналот и се иницирани со појавата на неговиот преден/заден раб. Тоа е овозможено со посложен интерфејс во однос на интерфејсот кај асинхрониот SRAM бидејќи синхрониот SRAM содржи интерни регистри кои ја овозможуваат синхронизацијата на неговата работа.

7.6.1. SRAM МЕМОРИСКА ЌЕЛИЈА

За да можеме во поголеми детали да ја разбереме работата на SRAM мемориските компоненти, најнапред ќе се задржиме на логичката структура на нејзината елементарна градбена единка која е прикажана на сл. 7-14. Станува збор за т.н. мемориска ќелија која може да запамти еднобитен податок. Всушност се користи бистабилен мултивибратор, односно за флип-флоп од D-тип применет како леч (коло за заклучување, задржување) кој може да биде реализиран и со флип-флоп од SR тип. Во практиката мемориската ќелија најчесто се реализира со шест MOSFET транзистори: два меѓусебно спрегнати инвертори (составени од по еден CMOS транзисторски пар), што го памтат податокот (битот) со дополнителни два транзистори, потребни за поврзување на линиите преку кои се врши негово читање/запишување.

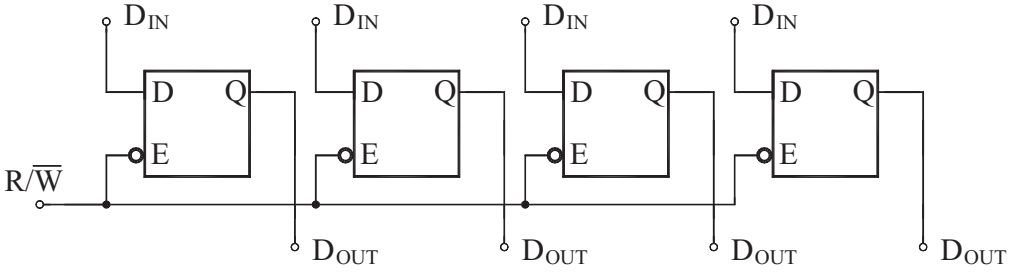


а) принципиелна електрична шема б) логичка структура в) симбол

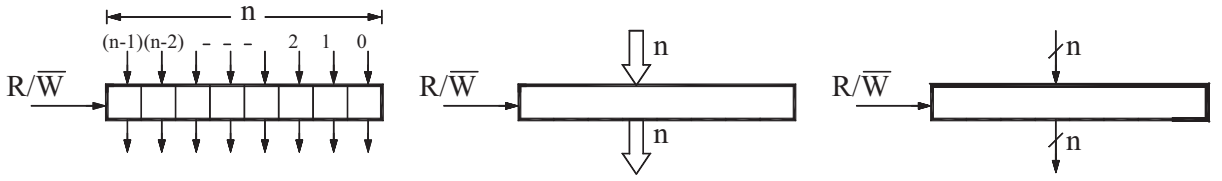
Сл. 7-14. SRAM мемориска ќелија

Новата содржина, т.е. новиот податок (новиот бит) се запишува преку влезната линија D, додека запамтената содржина, т.е. битот кој што се чува во ќелијата се чита од излезот на флип-флопот Q. При запишување на линијата за контрола E (за дозвола) се носи ниско ниво (E=0), додека новиот бит се поставува на влезот D. При читање на линијата за контрола E се носи високо ниво (E=1) со што на излезот Q се појавува тековната состојба на флип-флопот, т.е. битот што е во него запамтен. Комплементарниот излез на леч-колото $\overline{Q}$ не се користи.

Бидејќи еден бит е најмалото количество информација што може да го запомниме, на следната слика сл. 7-15 а) ќе прикажеме на кој начин можеме да поврземе четири мемориски ќелии за да добиеме еден мемориски збор со должина од 4 бита, додека на сл. 7-15 б) тоа е претставено симболички. Имајќи го во вид фактот дека сите 4 бита треба да бидат третирани како една целина, линијата за контролата на читање/запишување едновремено се доведува на истиот влез кај сите флип-флопови.



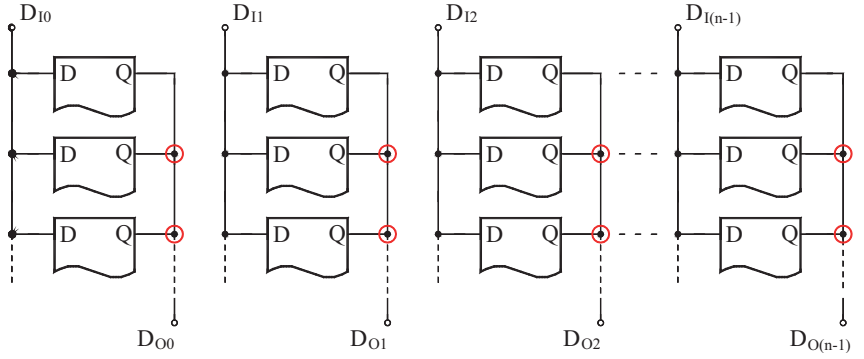
а) со четири мемориски ќелии



б) симболички приказ за n- мемориски ќелии

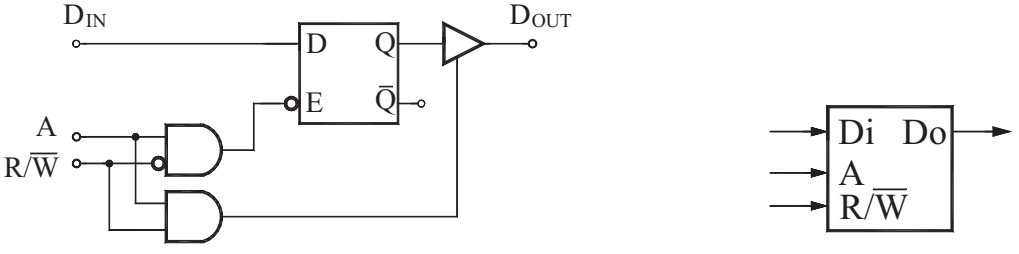
Сл. 7-15. Формирање на еден мемориски збор

Меѓутоа, при формирањето на меморија од повеќе вакви мемориски локации ќе се појави проблем бидејќи ваквите мемориски ќелии немаат линија за адресирање, а со тоа нема да може да се направи селекција на мемориските локации т.е. не може да се одлучи во која од нив да се внесе податокот кој во процесот на запишување е присутен на влезните линии. Покрај ова, и излезните податочни линии од ќелиите на различните локации НЕ можат директно да се поврзат во единствена точка заради формирање на податочни излезни линии од меморијата, во процесот на читање, бидејќи тогаш нивната состојба ќе биде конфликтна: потенцијалот во спојната точка, а со тоа и на секоја од податочните линии ќе биде недефиниран, според сл. 7-16.



Сл. 7-16. Недозволено поврзување на излезите од мемориските ќелии за формирање на податочни линии

Токму заради овие причини на следната слика сл. 7-17 е прикажана посложената и вообичаена варијанта на мемориска ќелија која содржи дополнителна линија за адресирање означена со A, како и излезно баферско коло со три состојби.



а) логичка структура б) симбол

Сл. 7-17 Мемориска ќелија со адресен влез, податочен влез и податочен излез со три состојби

Од дадената логичка шема се забележува дека состојбата на адресната линија A има доминантна улога врз работата на мемориската ќелија, која е презентирана со таб. 7-5.

A	$R/\overline{W}$	Режим (Операција)	Излез	Потрошувачка
0	X	Не е селектирана	HiZ	Неактивна (анг. standby)
1	0	Запишување	HiZ	Активна
1	1	Читање	D_{OUT}	Активна

Таб. 7-5. Функционална таблица на мемориската ќелија од сл. 7-17

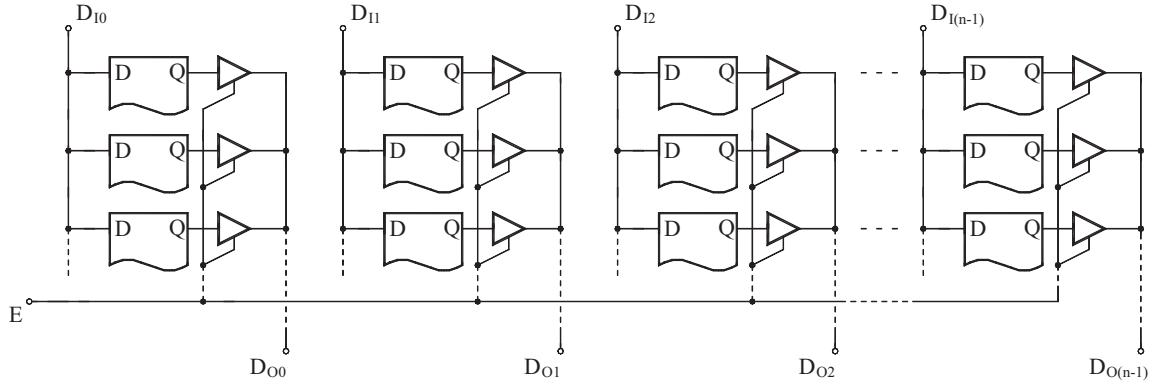
Доколку адресната линија A се наоѓа на ниско ниво (A=0), контролната линија на баферот ќе оди на ниско ниво, со што ќе биде оневозможен преносот на податокот од излезот на флип-флопот кон излезната податочна линија, бидејќи излезниот бафер ќе оди во трета состојба (HiZ). Едновремено на ниско ниво ќе оди и контролната линија E (E=0) заради што нема можност за менување на постоечката состојба на флип-флопот и тој ја зачувува претходно внесената содржина. Значи ако мемориската ќелија не е адресирана, во неа ниту може да се запишува нова содржина, ниту од неа може да се чита постоечката.

Ефективната работа на ќелијата (ќелијата е активна) само ако е адресирана, поточно само ако се активира линијата А и на неа се постави високо ниво (А=1) што ќе овозможи запишување или читање, зависно од сигналот $R/\overline{W}$ (или $\overline{WE}$) при што:

→ ако $R/\overline{W}$=0, се отвора И-колото кое ја активира контролната линија Е за запишувањето на податокот присутен на влезот D_{IN}, а преку второто И-коло се оневозможува излезниот бафер, бидејќи на неговата контролна линија се испраќа ниско ниво и тој оди во трета состојба (HiZ), додека

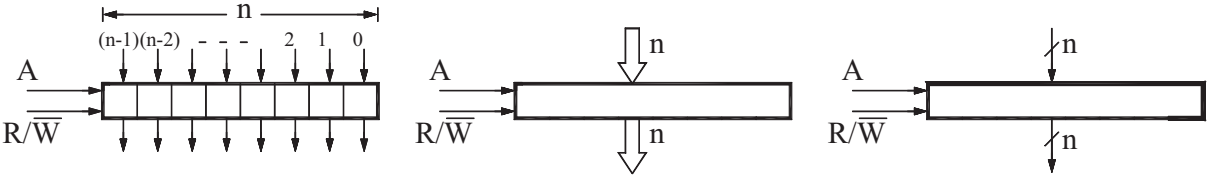
→ ако $R/\overline{W}$=1, се отвора И-колото, кое ја активира линијата за контрола на излезниот бафер и таа оди на високо ниво, така што состојбата на флип-флопот преку баферот се пренесува до излезот од мемориската ќелија. При ова контролната линија Е за запишувањето на податокот се наоѓа на ниско ниво (Е=0) и флип-флопот си ја задржува својата состојба.

Од дадената слика и претходното објаснување станува јасна и улогата на излезниот бафер кој овозможува излезот од ќелијата да ја рефлектира состојбата на флип-флопот (1 или 0) на излезната линија, но дополнително баферот може да се најде и во трета состојба на висока отпорност (HiZ, $R_D \to \infty$) што овозможува поврзување на излезите од "паралелните" мемориски ќелии од секоја локација во единствена заедничка точка и со тоа формирање на единствена излезна податочна линија од меморијата за соодветниот бит, според сликата сл. 7-18.



Сл. 7-18. Поврзување на излези од мемориските ќелии и формирање на податочни излезни линии

На сл. 7-19 а), б) и в) се дадени симболичките прикази на мемориска локација формирана од осум вакви ќелии при што поединечните адресни линии се врзуваат во единствена адресна линија, а исто така и поединечните водови за контролата на читање/запишување се врзуваат во единствена контролна линија.



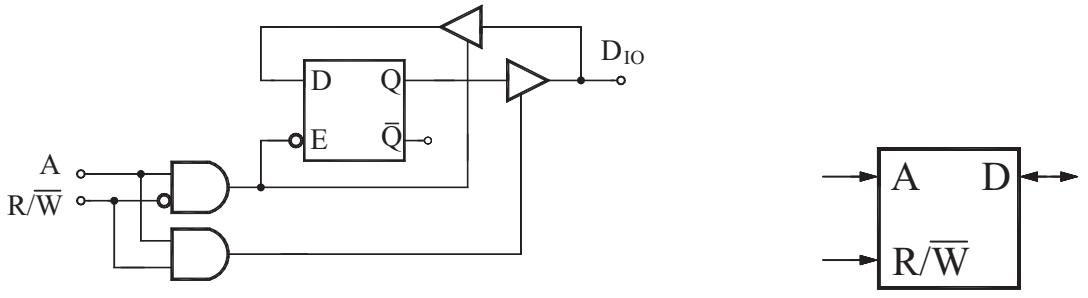
Сл. 7-19. Симболички ознаки на мемориска локација со посебни линии за адресирање и контрола на читање/запишување

Кај мемориската ќелија, чиј логички дијаграм е прикажан на сл. 7-17 карактеристично е постоењето на две посебни линии за битот на податокот: преку едната се запишува, додека преку другата се чита. Меѓутоа, ако се воведе уште еден бафер паралелно на постоечкиот, но во спротивна насока, според сл. 7-20 а) ќе може да се користи единствена линија за податок која ќе биде бидирекционална, т.е. ќе овозможува преку неа податокот да се запишува или да се чита.

Од сликата сл. 7-20 а) може да се извлече заклучок за принципот на работа на вака реализираната мемориска ќелија.

1. Ќелијата нема да биде адресирана ако на адресната линија A се постави ниско ниво (A=0). Тогаш и двата бафери се оневозможени заради што не постои комуникација помеѓу флип-флопот и излезот од ќелијата, кој ќе се наоѓа во трета состојба (HiZ). Флип-флопот е пасивен и си ја чува постоечката состојба.
2. Ќелијата се адресира со поставување на високо ниво на адресната линија (A=1), при што се можни две состојби кои зависат од нивото на сигналот $R/\overline{W}$:
   - При читање треба $R/\overline{W}$ да оди на високо ниво ($R/\overline{W}$ =1) со што се овозможува горниот (излезниот) бафер да ја проследи состојбата на флип-флопот до податочната линија, која во овој случај ќе биде излезна, додека долниот (влезниот) бафер е затворен;
   - При запишување треба $R/\overline{W}$ да оди на ниско ниво ($R/\overline{W}$ =0) со што се овозможува долниот (влезниот) бафер да се отвори/активира, а со тоа битот на податокот, кој се наоѓа на податочната линија, која сега ќе биде влезна, да се внесе како нова содржина во флип-флопот. При ова горниот (излезниот) бафер е затворен.

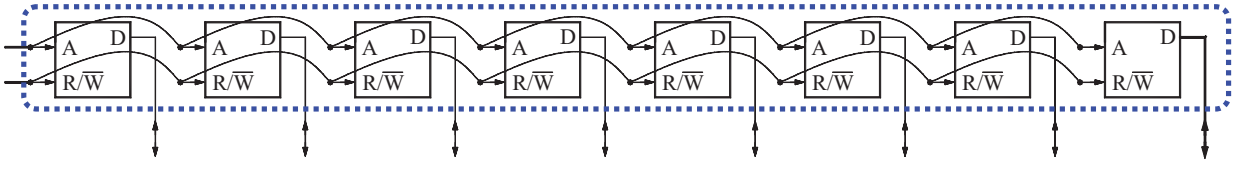
Симболичката ознака на оваа мемориска ќелија е прикажана на сликата сл.7-20 б).



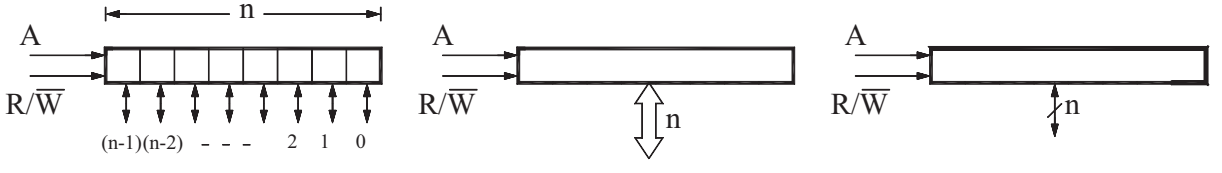
а) логичка структура б) симбол

Сл. 7-20 Мемориска ќелија со адресен влез и двонасочна податочна линија со три состојби

За формирање на 8 битен мемориски збор (локација) 8 мемориски ќелии треба да се поврзат според сл.7-21 а) и сл. 7-21 б) на која е прикажана нејзината симболичка ознака.



а) логичка структура

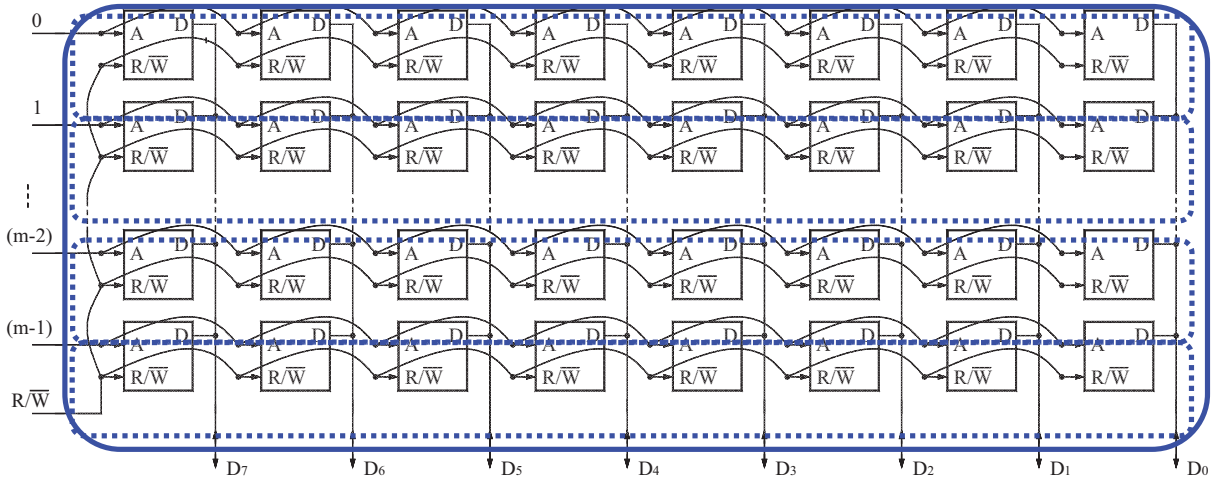


б) симболи

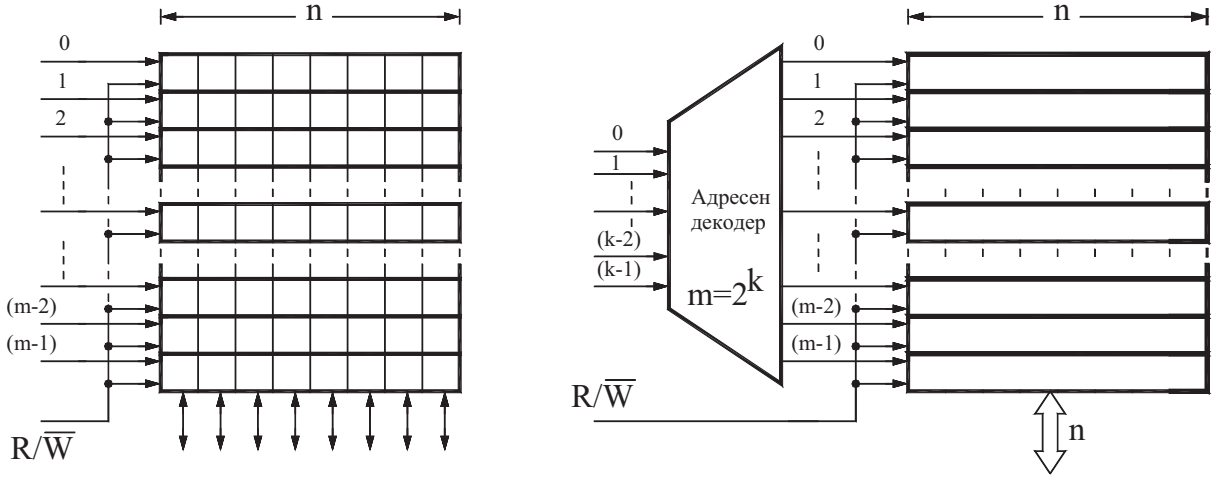
Сл. 7-21. Осум битна мемориска локација со адресен влез, контролен влез за читање/запишување и двонасочна податочна линија

Од сликата се гледа дека адресната линија за меморискиот збор е единствена. Таа се приклучува паралелно на сите влезови за селекција на ќелијата. Исто така паралелно во единствена линија се поврзуваат и влезовите за читање/запишување. Само ваквото спојување на мемориски ќелии обезбедува формирање на единствена мемориска локација како најмала адресибилна формација на која ќе може да и се пристапи во меморијата.

Заради формирање на мемориска матрица треба да се изврши поврзување на поголем број вакви локации според следната слика (сл.7-22). Имајќи во вид дека ваквиот приказ е прилично оптоварен со линии и доста непрегледен, многу почесто се користи симболичката ознака од сл. 7-23 а) или ознаката од 7-23 б), која ја прикажува мемориската матрица заедно со декодерот. Адресната линија за секој меморискиот збор е посебна, додека сите поединечни влезови за читање/запишување паралелно се поврзуваат во единствена линија за контрола на читање/запишување.



Сл.7-22. Поврзување на SRAM мемориски ќелии за формирање на 8 битна мемориска матрица со m-адресни влезови, единствен контролен влез за читање/запишување и двонасочна податочна линија

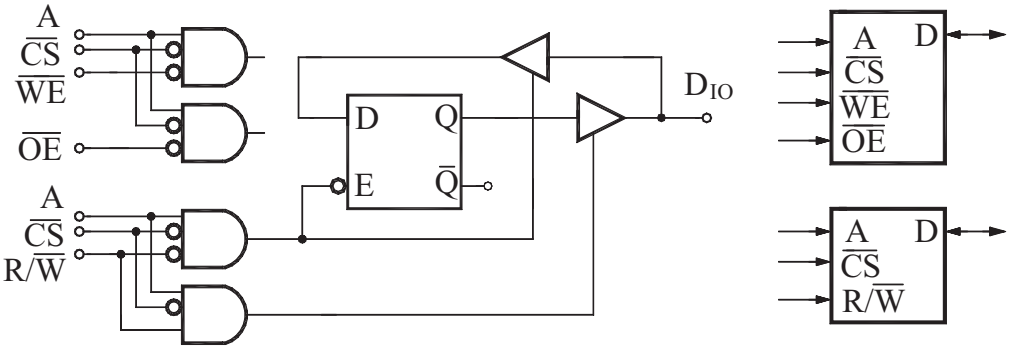


а) без адресен декодер б) со k/m адресен декодер за кој важи $m=2^k$

Сл. 7-23. Логичка структура на 8 битна мемориска матрица со m-адресни влезови, контролен влез за читање/запишување и двонасочна податочна линија

Покрај екстерните k, односно интерните $m=2^k$ адресни водови и податочните n линии, како и влезот за читање/запишување $R/\overline{W}$ (или $\overline{WE}$), меморискиte RAM компоненти располагат и со два дополнителни контролни влеза според сл. 7-24. Со едниот влез се селектира мемориското интегрирано коло заради што тој се означува со CS (селекција на чип, анг. chip select), додека вториот влез OE го контролира и овозможува или оневозможува излезот на податоците (читањето) (анг. output enable), според сл. 7-24 а) и б). Кај реалните мемориски компоненти овие сигнали најчесто се активни на ниско ниво.

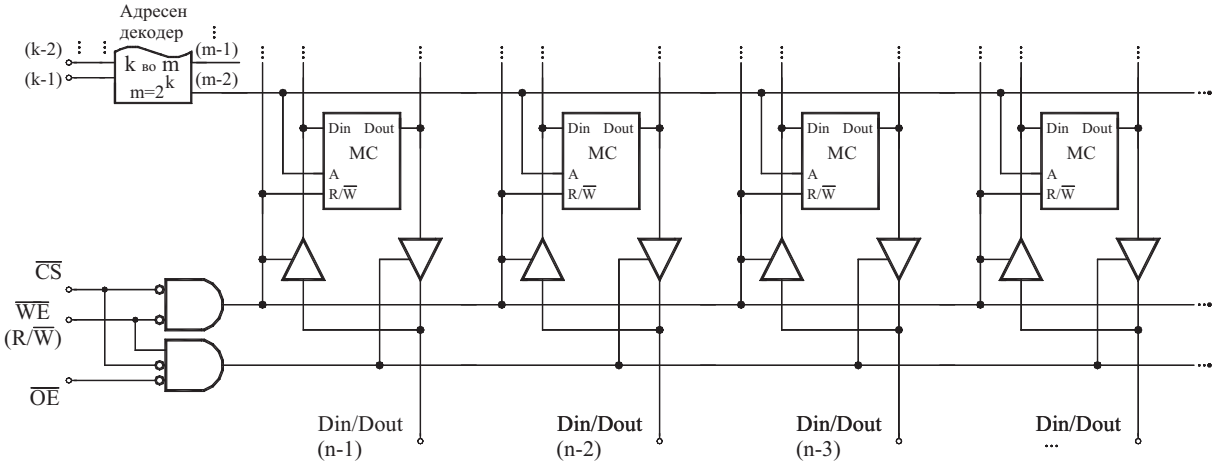
Ако чипот не е селектиран, тогаш $\overline{CS}=1$, и сите податочни водови се наоѓаат во трета состојба (HiZ). За да се чита или запишува во меморијата таа мора да биде селектирана со доведување активно (ниско) ниво на линијата за селекција на чип ($\overline{CS}=0$). При ова, на адресните линии, а со тоа и на влезот од адресниот декодер треба да биде поставена адресната информација на специфицираната мемориска локација со што ќе се активира соодветната излезна адресна линија од декодерот.



а) логичка структура б) симболички ознаки

Сл. 7-24. Мемориска ќелија со контролната логика за селекција, адресирање и читање/запишување

Во овие услови за да се чита треба да биде исполнет условот $R/\overline{W}$ =1 ако читањето и запишувањето се управуваат со еден контролен сигнал $R/\overline{W}$, односно ќе треба да важи $\overline{WE}$ =1 и $\overline{OE}$ =0 кога мемориските чип располага со два контролни сигнали за управување на овие процеси. Меѓутоа ако се запишува, логичките состојби на сигналите треба да бидат спротивни од претходно: во случај на единствен контролен сигнал за читање/запишување треба да важи $R/\overline{W}$ =0, односно $\overline{WE}$ =0 и $\overline{OE}$ =1 ако станува збор за два контролни сигнали за контрола на читањето/запишувањето во меморијата.



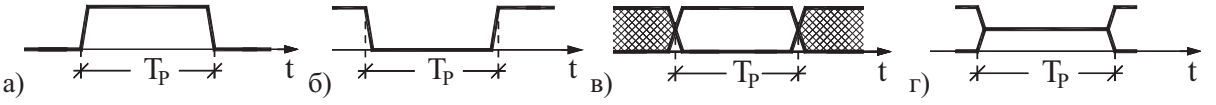
Сл. 7-25. Мемориски збор формиран во рамките на SRAM мемориска матрица заедно со контролните сигнали

Сликата 7-25 го прикажува поврзувањето на четири мемориски ќелии од некоја SRAM мемориската матрица во единствен мемориски збор вклучувајќи ја контролната логика и адресниот декодер. Според сликата станува збор за последната локација на меморијата која има највисока адреса $a_{(m-1)}=2^k-1$ која ќе биде адресирана ако сите влезни адресни линии се наоѓаат на високо ниво, т.е. ако важи $a_{(k-1)}$, $a_{(k-2)}$, $a_{(k-3)}\ldots$ a_1, a_0 = 111…11.

7.6.2. АСИНХРОНО ЧИТАЊЕ И ЗАПИШУВАЊЕ

Работата на меморијата, а со тоа и процесите на нејзино читање и запишување ги контролира и управува централниот процесор. Додека работата на процесорот е синхронизирана од неговиот внатрешен такт сигнал, меморијата може да функционира и асинхроно. Нејзината работа се управува со промената на логичките нивоа на нејзините влезни контролни сигнали за селекција на чип и оние сигнали што треба да се постават на двете контролни линии за читање/запишување. Сигналите присутни на линиите за адресирање ја специфицираат мерморискa локација чија содржина се појавува на податочните линии.

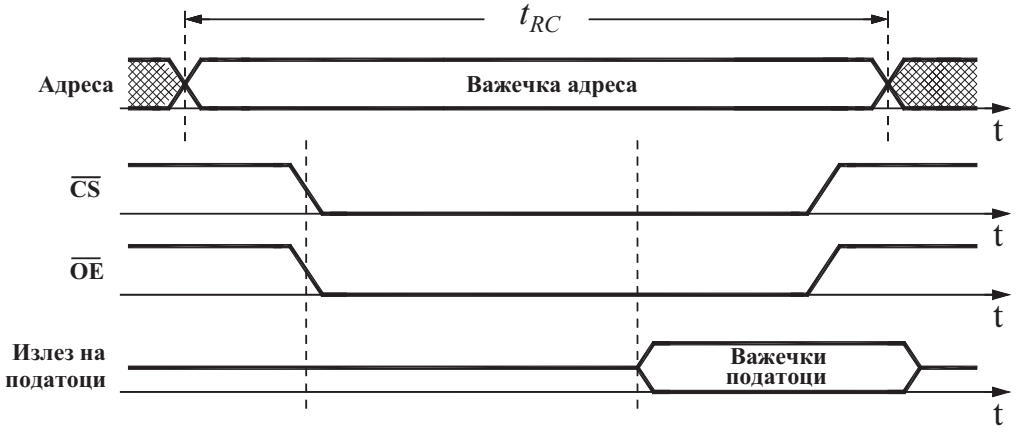
За објаснување на операциите читање од адресирана мемориска локација, или запишување во неа, ќе користиме временски дијаграми дадени на сл.7-26 кои ги прикажуваат напонските нивоа, а со тоа и логичките состојби, на влезните и излезните водови, преку кои комуницираат компонентите на дигиталниот систем. За презентирање на високо ниво (ниво на логичка 1), ќе користиме напонски правоаголен импулс со одредено времетраење Тр (сл.7-26 а). За ниско ниво (ниво на логичка 0) напонот е нула заради што на временската линија во периодот Тр нема да има никаква промена на напонот и нивото ќе биде нула (сл.7-26 б). Доколку во периодот Тр на водот може да се појави податок со било каква вредност, високо или ниско (1 или 0), на почетокот и во крајниот момент на перидот Тр се поставуваат две накрсни линии во облик на буквата "X" (сл.7-26 в). Помеѓу нив се повлекуваат две паралелни линии со t-оската за ниско и за високо ниво кои означуваат појава на бинарен податок чие времетраење е еднакво на Тр. Доколку за време на периодот Тр водот се најде состојба на бесконечно голема отпорност (висока импенданса, HiZ) се повлекува паралелна линија на временската оска со напонско ниво на половина помеѓу високото и ниското (сл.7-26 г).



Сл.7-26. Бранови облици на напоните кои одговараат на различни логички состојби

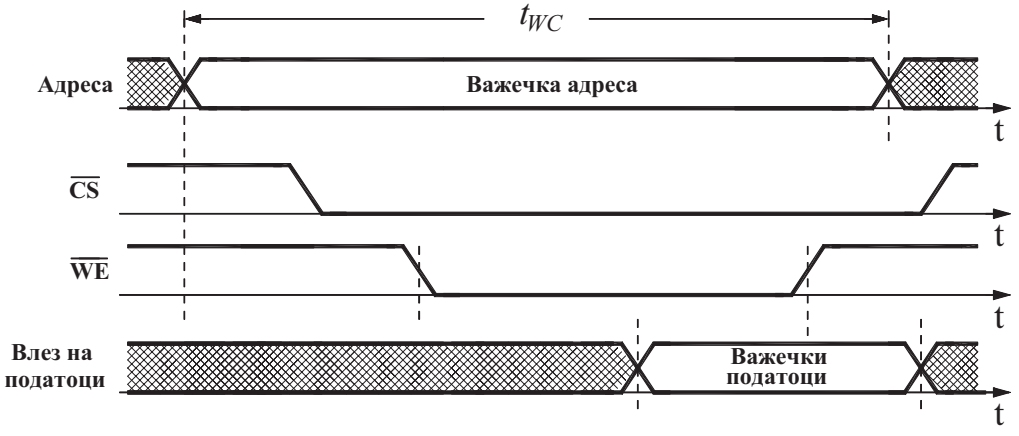
На следните временски дијаграми визуелно се презентирани логичките состојби на мемориските линии кои се дадена една под друга со што лесно се воочува редоследот на појавување и времетраење на сигналите присутни на тие водови. И во двата случаи за релизација на операциите, сигналот за селекција на чип треба да биде низок, додека до меморијата се пристапува преку позната адреса чии адресни битови се поставуваат на адресните линии. Состојбите на линиите за контрола на читањето/запишувањето ќе ги менуваат своите вредности во зависност од операцијата која ќе треба да се изврши.

Процесот на читање се реализира кога е исполнет условот $\overline{WE}=1$ и $\overline{OE}=0$. Податокот кој се чита и кој е запамтен во специфицираната мемориска локација не може да се појави на податочната магистрала едновремено со појавувањето на адресната информација на адресните водови, туку за тоа е потребен одреден временски интервал. Имајќи го во вид претходното, се дефинира *време на пристап,* прикажано на временските дијаграми од сл. 7-27 како максимално време, кое ќе помине од моментот на поставување на адресната информација на адресните линии, па сè до моментот кога на излезните податочни линии ќе се појави содржината на адресираната локација. Временскиот период во кој адресната информација се доведува на адресните линии и се здржува одреден временски период потребен за да се прочитаат податоците се дефинира како *време на циклусот на читање* и се означува со t_{RC}.



Сл.7-27. Циклус на читање и време на пристап

Кога се извршува операција запишување логичката состојба на контролните линии треба да биде $\overline{WE}$ =0 и $\overline{OE}$ =1. Слично како кај читањето, кај запишувањето се дефинира *време за циклус на запишување* означено со t_{WC}, кое е прикажано на временските дијаграми од сл. 7-28. Овај временски интервал го претставува максималното време, кое е потребно од поставување на адресната информација до завршување на сите интерни (внатрешни) мемориски операции, со кои во специфицираната адресна локација се запишува новиот податок (меморискиот збор).



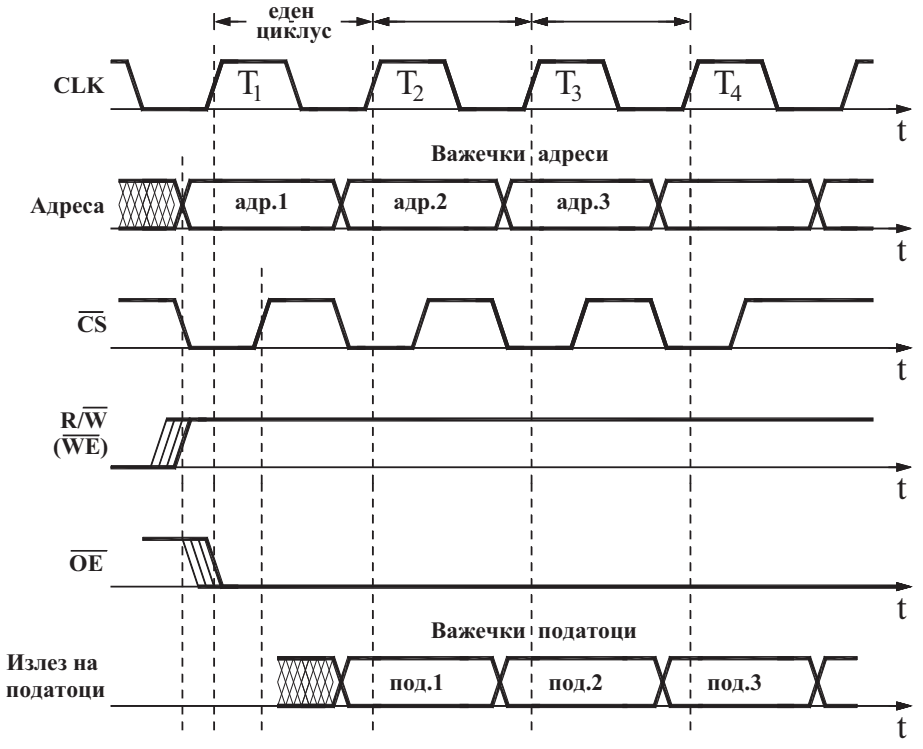
Сл.7-28. Циклус на запишување и време на циклус на запишување

Имајќи го во вид претходното објаснување како и временските дијаграми на сл. 7-27 и сл. 7-28 треба да се знае дека при читање или запишување во меморијата, процесорот е kодигиталната компонента која управува со состојбата на контролните линии. При ова, заради правилна работа на дигиталниот систем, тој треба да го зема предвид времето на доцнење помеѓу последователното појавување на контролните сигнали во рамките на времето на циклусот на читање, односно во рамките на циклусот за запишување.

7.6.3. МЕМОРИСКИ ЦИКЛУС НА СИНХРОНО ЧИТАЊЕ

Функционирањето на синхрониот SRAM е временски усогласено со системскиот такт според кој работи и процесорот, поточно со појавата на неговиот активен раб кога започнува конкретниот процес на читање или на запишување. Во поглед на мемориската матрица и контролните сигнали не постои принципиелна разлика помеѓу синхрониот и асинхрониот SRAM. Меѓутоа, основната разлика е во тоа што синхрониот SRAM во себе содржи различни регистри во кои се заклучуваат сите сигнали: податоците, адресата и контролните сигнали и тоа синхроно со појавата на предниот раб на системскиот такт.

Операцијата читање на податок од зададена мемориска локација се извршува во неколку базични чекори, кои се попрегледно објаснети со временските дијаграми презентирани на сл. 7-29.



Сл. 7-29. Временски дијаграми на карактеристични сигнали кога се реализира циклус на читање од меморијата

(1) Непосредно пред појавата на растечкиот (активниот) раб на такт сигналот микропроцесорот на адресните линии ја поставува адресата што ја специфицира мемориската локација од која треба да биде прочитан меморискиот збор (бинарниот податок),

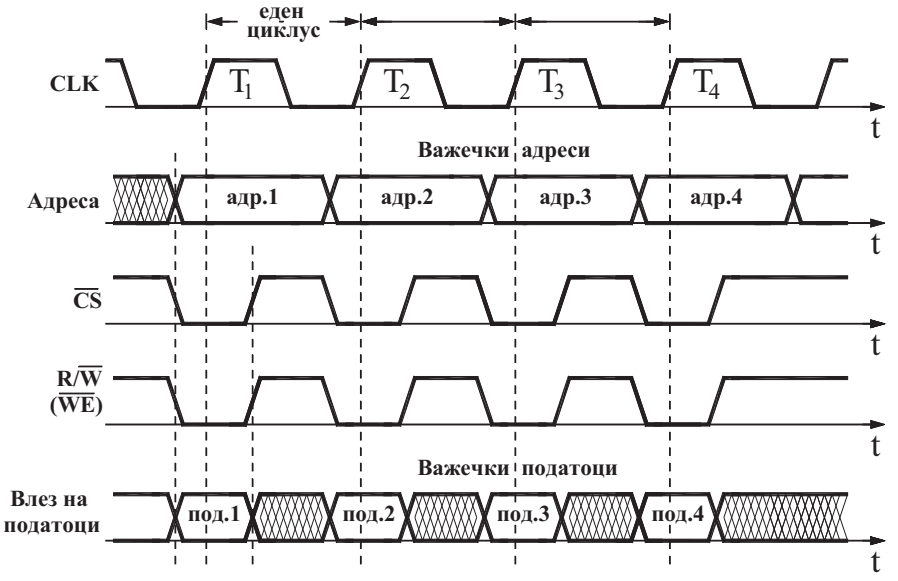
(2) Едновремено со тоа процесорот ја активира линијата за селекција на меморискиот чип $\overline{CS}$ поставувајќи на неа ниско ниво ($\overline{CS}$ =0), додека процесот на читање го овозможува така што на контролната линија за читање/запишување $\overline{WE}$ поставува високо ниво и на линијата за овозможен излез $\overline{OE}$ ниско ниво ($\overline{WE}$ =1 и $\overline{OE}$ =0),

(3) Почнувајќи од моментот кога на адресните водови ќе се постави адресната информација и доцнење кое е еднакво со времето на пристап, на мемориските податочни линии се поставуваат осумте битови на податокот (содржината) на адресираната мемориска локација. Тие се појавуваат во истиот такт интервал, или во следниот што зависи од тоа за каков начин на читање станува збор.

Прочитаниот податок (бајт) од специфицираната (адресираната) мемориска локација процесорот го запишува во својот интерен регистер како негова нова содржина со што завршува операцијата читање и процесорот може да извршува друг циклус на читање или на запишување.

7.6.4. МЕМОРИСКИ ЦИКЛУС НА СИНХРОНО ЗАПИШУВАЊЕ

И операцијата запишување на податок во конкретно адресирана мемориска локација се изведува во неколку основни чекори, според временските дијаграми дадени на сл. 7-30.

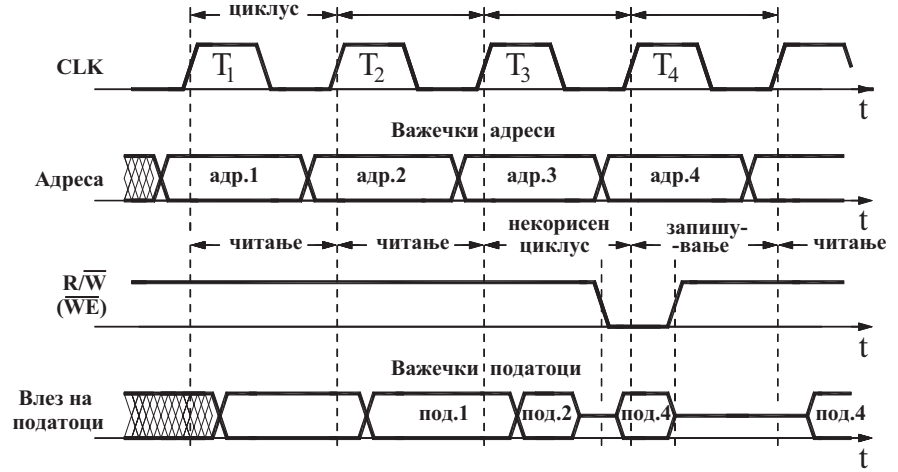


Сл. 7-30. Временски дијаграми на карактеристични сигнали при циклус на запишување

(1) Непосредно пред појавата на активниот (предниот) раб на такт сигналот адресната информација, што ја специфицира мемориската локација, во која треба да биде запишан (внесен) меморискиот збор (бинарниот податок), се поставува од страна на микропроцесорот на адресните линии. Едновремено со тоа процесорот го селектира и меморискиот чип, т.е. овозможува тој да функционира преку активирање на соодветната линија за селекција на чип $\overline{CS}$ која ја спушта на ниско ниво ($\overline{CS}$ =0),

(2) Во истиот момент процесорот на контролната линија за читање/запишување $\overline{WE}$ поставува активен сигнал за запишување, а тоа е ниско ниво: $\overline{WE}$ =0. Време-траењето на овој сигнал треба да биде доволно долго за податоците присутни на податочните линии да се запишат во мемориската компонента по што овој сигнал станува пасивен, т.е. се враќа на високо ниво.

На овој начин процесот на запишување на нов збор (податок), кој е поставен на податочната магистрала, се запишува како нова содржина на посочената (адресираната) мемориска локација во еден такт на процесорот, или во следниот такт интервал, што зависи од тоа за каков начин на запишување станува збор.



Сл. 7-31. Временски дијаграми на карактеристични сигнали при читање и запишување

Пример: За примерот што во продолжение ќе го презентираме со таб. 7-6 ќе претпоставиме дека процесите на читање и запишување се однесуваат на комуникацијата помеѓу централниот процесор и SRAM мемориски чип кај некој едноставен микрокомпјутер. Ако усвоиме дека неговиот микропроцесор располага со k=16 адресни и n=8 податочни линии тоа значи дека меморискиот простор кој тој може да го адресира е 2^{16} мемориски зборови, т.е. 2^6x2^{10}=64x1024=64K зборови со должина осум бита (еден бајт). Заради ова се користи соодветно мемориско SRAM интегрирано коло исто така со k=16 адресни и n=8 податочни линии со капацитет 2^{16}=64KB.

Контролни линии			Адресни линии		Податочни линии		
Операција	$\overline{CS}$	$R/\overline{W}$	Бин.	Хекса.	Бин.	Хекса.	Насока (Смер)
Читање	0	1	0000000011001010	00CA	11110000	F0	Излезни
Пишување	0	0	0000000011001111	00CF	01111001	79	Влезни
Неактивна	1	X	xxxxxxxxxxxxxxxx	xxxx	xxxxxxxx	xx	HiZ (Висока отпорност)

Таб. 7-6 Пример на основните мемориски операции

Во таб. 7-6 е прикажана состојбата на сите линии кај RAM мемориската компонента кога а) се чита податокот 240_{DEC} = 11110000_{BIN} што се наоѓа во мемориска локација чија адреса е 202_{DEC} = 0000000011001010_{BIN}; б) во мемориска локација со адреса 207_{DEC} = 0000000011001111_{BIN} се запишува податокот 121_{DEC} = 01111001_{BIN}; в) процесорот не комуницира со меморијата.

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

7-1. Која е основната улога (задача) на мемориските компоненти и уреди?

7-2. Спореди ги полупроводнички мемории и тврдите дискови и наведени ги нивните предности и слабости во поглед на брзина на работа, капацитет и цена на чинење.

7-3. Од организациски аспект каква логичка структура претставува меморијата?

7-4. Меморискиот збор е …

7-5. Мемориска локација е …

7-6. Мемориската ќелија претставува …

7-7. Должината на зборот претставува … и се изразува во …

7-8. Содржина на зборот е …

7-9. Која е улогата на податочните линии? Каква информација се појавува на нив? Од што зависи нивниот број?

7-10. Најмалата адресибилна целина во меморијата е …

7-11. Кои се двете основни операции со кои меморијата ја остварува својата функција?

7-12. Читањето претставува . . .

7-13. Запишувањето претставува . . .

7-14. Која е разликата помеѓу деструктивното и недеструктивното читање?

7-15. Адресата претставува . . .

7-16. Која е улогата на адресните линии? Каква информација се појавува на нив? Од што зависи нивниот број?

7-17. Која е улогата на адресниот декодер?

7-18. Капацитетот на меморијата претставува . . .

7-19. Колку адресни и податочни линии треба да има бајт-организирано мемориско коло ако има капацитет од а) 512 MB б) 8KB в) 64 KB? Одговори го истото прашање ако мемориските зборови имаат должина два бајти (1 збор = 2 B).

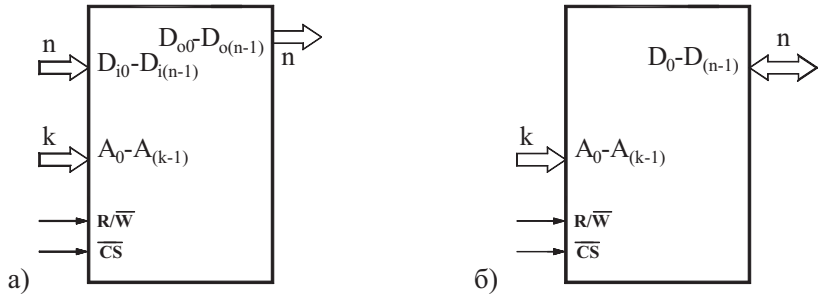
7-20. Која е намената на контролната линија $R/\overline{W}$?

7-21. Која е намената на контролната линија $\overline{WE}$?

7-22. Која е намената на контролната линија $\overline{OE}$?

7-23. За што служи контролната линија $\overline{CS}$ ($\overline{ME}$)?

7-24. Која е улогата на секоја од влезните/излезните линии за мемориските компоненти чии симболички ознаки се прикажани на следните слики?



Слики за прашање 7-22.

7-24. Како е организирана и колкав капацитет изразен во бајти има мемориска компонента со k адресни линии при што а) k=10 б) k=16 в) k=20 в) k=30 ако истата располага со n податочни линии при што а) n =8 б) n =16 в) n =32 г) n =64. Нацртај нејзин симбол.

7-25. Колку адресни и податочни линии треба да има мемориска компонента која располага со а) m=1024 б) m=4096 в) m=65536 г) m=536870912 мемориски зборови со должина од а) n=8 б) n=16 в) n=32 г) n =64 бита. Нацртај нејзин симбол.

7-26. Да претпоставиме дека треба да купиме мемориско интегрирано коло во кое треба да сместиме една книга од а) 100 б) 250 б) 320 страни чиј текст ќе го внесуваме кодиран според 8-битниот ASCII код. Колкав капацитет изразен во бајти треба да има мемориската компонента што ќе треба да ја набавиме за да може да се внесе текстот на посочената книга? Каква треба да е организацијата на меморијата за да може посебно да му се пристапи на секој симбол од текстот? Колку вакви книги можат да се сместат во оперативната меморија од (1) 1 GB (2) 2 GB на Пентиум базиран персонален сметач?

7-27. Која е основната разлика и сличност помеѓу ROM и RAM мемориските компоненти во поглед на пристапот до податоците и времетраењето на нивното памтење?

7-28. Наведи ја поделбата на ROM мемориските компоненти. Која е разлика помеѓу нив?

7-29. Наведи ја поделбата на RAM мемориските кола. Која е разлика помеѓу нив?

7-30. Кои се разликите помеѓу SRAM и DRAM мемориските компоненти?

7-31. Наведи ги разликите помеѓу асинхрониот и синхрониот SRAM.

7-32. Нацртај ја принципиелната електрична шема на SRAM мемориска ќелија.

7-33. Нацртај ја логичката структура на SRAM мемориска ќелија со две податочни линии. Објасни го нејзиниот принцип на работа користејќи функционална табела.

7-34. 7-33. Нацртај ја логичката структура на SRAM мемориска ќелија со една податочна линија. Објасни го нејзиниот принцип на работа користејќи функционална табела.

7-35. Во дадената табела прикажи ја содржината на првите и последните три мемориски локации ако во нив се сместени ASCII кодовите на буквите: a, b, A и B, како и цифрите 1 и 2 според 8-битниот ASCII код. Претпостави дека мемориската компонента располага со а) 16 б) 32 мемориски зборови со должина (1) 1 бајт (2) 2 бајти.

7-36. Мемориска компонента со капацитет а)1К б) 4К в) 16К г) 64К бајти чии мемориски зборови имаат должина од 1) 1 бајт 2) 2 бајти. Во мемориските локации со адреси а) $255_{(10)}$ б)$256_{(10)}$ в)$1023_{(10)}$ г)$1024_{(10)}$ д) $4095_{(10)}$ ѓ) $4096_{(10)}$ последователно се запамтени буквите a, b, A и B, како и цифрите 1 и 2 според 8-битниот ASCII код. Пополни ја приложената табела.

Адреса		Содржина		
Бин.	Хекса.	Бин.	Хекса.	ASCII симбол

Табела за прашање бр. 7-35 и 7-36

7-37. Каква е состојбата на контролните, адресните и податочните линии а) за да се прочита податокот кој се наоѓа запамтен во мемориска локација со адреса а) 4095 б) 4096 и потоа да се запише во мемориска локација со адреса а) 1023 б) 1024. Пополни ја приложената табела.

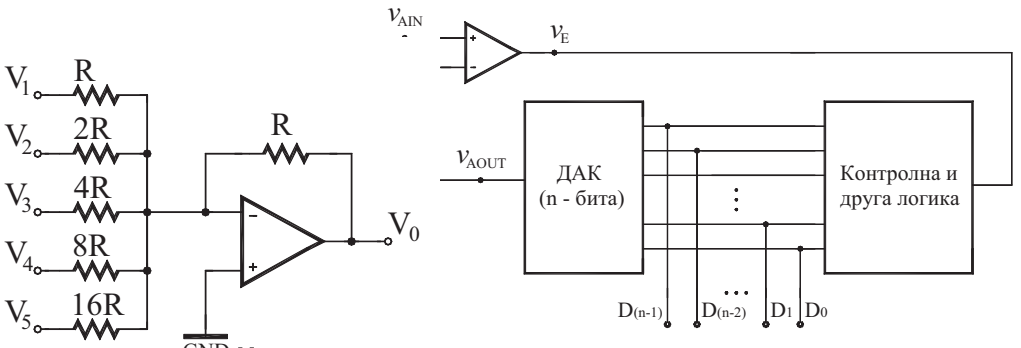
Контролни линии			Адресни линии		Податочни линии		
Операција	$\overline{CS}$	$R/\overline{W}$	Бин.	Хекса.	Бин.	Хекса.	Насока

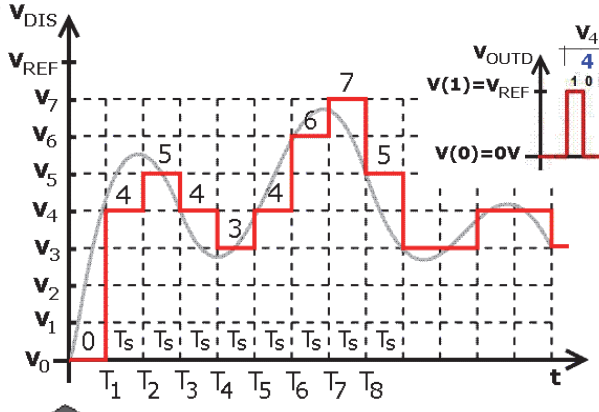
Табела за прашање бр. 7-37

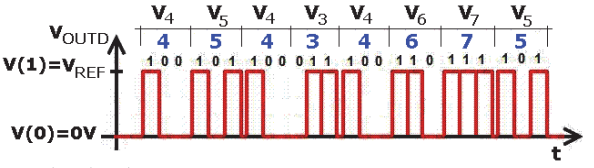
7-38. Објасни ја улогата и временскиот редослед на појавување на сигналите кои се однесуваат на процесот на а) читање б) запишување кај асинхрон RAM чип според временските дијаграми прикажани на сл. 7-26 и сл. 7-27.

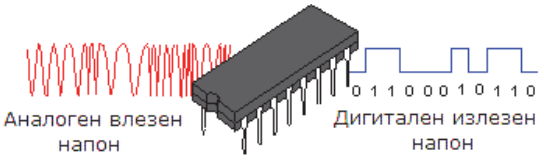
7-39. Објасни ја улогата и временскиот редослед на појавување на сигналите кои се однесуваат на процесот на а) читање б) запишување кај синхрон RAM чип според временските дијаграми прикажани на сл. 7-28 и сл. 7-29. Претпостави дека фреквенцијата на системскиот такт изнесува 5 MHz.

7-40. (*) Потруди се преку интернет да го најдеш, разгледаш и анализираш документот со деталните технички карактеристики (анг. datasheet) за мемориското интегрирано коло 74LS189. Потоа поконкретно објасни а) за каква меморија станува збор, б) која е улогата на секој од пиновите на колото, в) каков е неговиот принцип на работа, и г) прикажи еден пример со кој ќе го демонстрираш функционирањето на колото.









8.

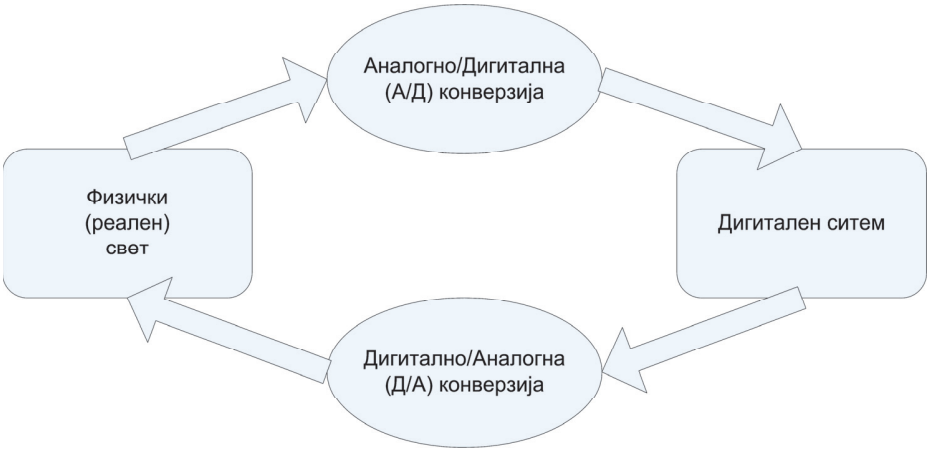
ДИГИТАЛНО – АНАЛОГНА и АНАЛОГНО – ДИГИТАЛНА КОНВЕРЗИЈА

По изучувањето на оваа тематска целина

- ќе ги познавате базичните поими кои се однесуваат на Д/А и А/Д конверзија;
- ќа ги разберете основните концепти на процесите на Д/А и А/Д конверзија;
- и процесот на Д/А и А/Д конверзија;
- ќе ги познавате и принципиелно ќе знаете да ги објаснете различните методи на Д/А и А/Д конверзија;
- ќе ги познавате различните типови на Д/А и А/Д конвертори и ќе ја разберете нивната работа.

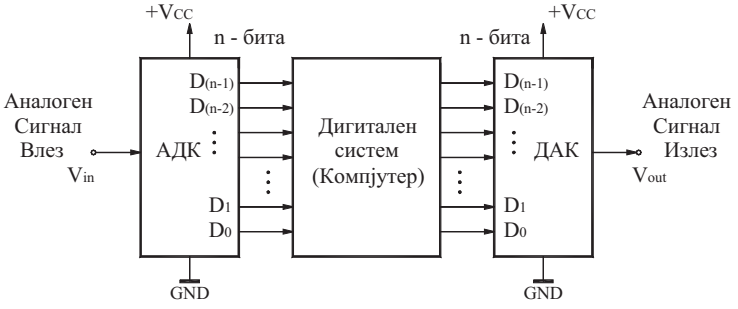
8.1. ВОВЕД

Глобалната и воедно најопштата слика на проблематиката на аналогно-дигиталната (А/Д) и дигитално-аналогната (Д/А) конверзија е прикажана на сл. 8-1. На едната страна се наоѓа реалниот (физичкиот) свет, а на другата страна се дигиталните системи. Тие може меѓусебно да комуницираат само со помош на А/Д и Д/А конвертори кои како посебни компоненти имаат задача да извршат прилагодување. Имено, во основа постои фундаментална разлика: појавите во реалниот свет во општ случај се континуални, тие примаат бесконечно многу вредности, заради што сигналите со кои тие се опишуваат се аналогни. Од друга страна, дигиталните системи работат само со дигитални сигнали. Тоа се низи од бинарни зборови од кои секој претставува комбинација од само две напонски нивоа: високо $V(1)=+V_{CC}$ и ниско $V(0)=0$ кои соодветствуваат на логичка 1 и логичка 0 кои претставуваат бинарно кодирани репрезенти на сигнали кои имаат дискретни вредности со конечен број на нивоа.

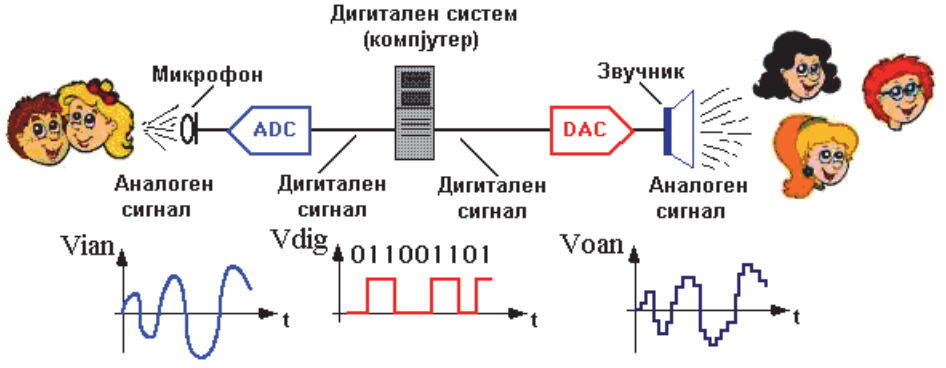


Сл. 8-1. Општа блок-шема за местото и улогата на А/Д и Д/А конверторите во интеракцијата помеѓу реалниот свет и дигиталните системи

А/Д и Д/А конверторите се неопходно присутни во дигиталните уреди, па може да се каже и пошироко, бидејќи многу често се користат скоро во сите електронски уреди. Заедничката употреба на А/Д и Д/А конверторите обезбедува целосно поврзување (анг. interface) со секакви сензори (анг. sensors) на влезот и излезни претворувачки елементи (анг. transducers) на излезот со цел реализација на контролна и управувачка улога кај електронските системи, процесите и појавите од реалниот свет. На сл. 8-2 А/Д и Д/А конверторите се прикажани на наједноставен начин како влезен и излезен блок кај некој дигитален систем за општа намена, додека на сл. 8-3 е даден пример за А/Д и Д/А конверзија на аудио сигнал со персонален компјутер.



Сл. 8-2. Наједноставна блок-шема на поврзување на компјутер со А/Д и Д/А конвертор



Сл. 8-3. А/Д и Д/А конверзија на аудио сигнал со персонален компјутер

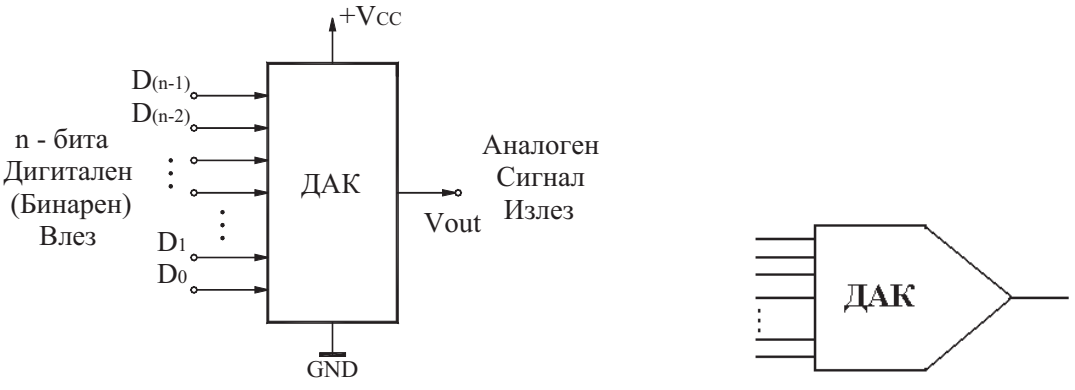
Кога треба на влезот од дигитален уред да се приклучи аналоген уред, работата станува прилично комплексна, бидејќи директното поврзување на аналогните излези на дигитални влезови е невозможно бидејќи аналогните сигнали можат да имаат бесконечно многу вредности, додека дигиталните конечно многу дискретни вредности во ограничен опсег. Она што треба да се направи е да се најде начин да се изврши електронска трансформација на аналогниот сигнал во дигитален облик, како низи од логички 1-и и 0-и со конечна должина (бинарни зборови, вектори). Во врска со ова, А/Д конверторот (анг. analog-to-digital converter, ADC) е електронска компонента или уред кој на својот единствен влез добива аналоген сигнал, најчесто напон или струја и врши негово претворување (конверзија) во дигитален сигнал, т.е. зборови од кои секој претставува низа од битови според одреден бинарен код или во бинарни броеви според природниот бинарен броен систем. За секој бит на излезот од А/Д конверторот постои посебна линија по која тој се проследува на понатамошна обработка до процесорот на дигиталниот систем.

Од друга страна, Д/А конверторот (анг. digital-to-analog converter, DAC) ја извршува обратната функција. Имено, по завршувањето на обработката на податоците, резултатите се добиваат во дигитален облик како комбинации од битови кои по посебни линии се испраќаат до влезовите на Д/А конверторот, кој истите ги претвора во аналоген сигнал и го пренесува на својот единствен излез. Поврзувањето на излезите од логичките кола со претворувачки елементи (анг. tranducers) е прилично едноставно, бидејќи тие самите по својата природа се дигитални елементи. Имено, излезите од логичките кола лесно се поврзуваат на транзистори, релеа, итн. заради нивниот принцип на работа со две положби: вклучен - исклучен, каква што е и природата на сигналите кои се добиваат на излезите од дигиталните компонети.

Во оваа тематска целина фокусот ќе биде ставен токму на А/Д конверзијата и нејзиниот обратен процес, Д/А конверзијата, со посебна анализа и објаснување на различните постапки на конверзија, како и на принципот на работа на различните видови конвертори.

8.2. ДИГИТАЛНО-АНАЛОГНА КОНВЕРЗИЈА

Претходно веќе е истакнато дека Д/А конверзијата е процес кој е поедноставен за реализација во однос на инверзниот процес на А/Д конверзија. Што е уште поважно, во практиката се користат А/Д конвертори, кои во својот состав содржат Д/А конвертор, заради што најнапред ќе биде обработен процесот на Д/А конверзија, а потоа и А/Д конверзијата.



а) наједноставна блок-шема б) симболичка ознака

Сл. 8-4. Д/А конвертор

Д/А конверторот има задача дигиталниот сигнал, кој е претставен како множество од зборови, од кои секој има должина од конечен број битови n кои доаѓаат на неговите влезови да го конвертира во аналоген облик. На сл. 8-4 а) и б) се прикажани наједноставна блок шема и симболичка ознака на Д/А конвертор.

8.3. БАЗИЧНИ РАВЕНКИ, ПОИМИ И ПРЕНОСНА КАРАКТЕРИСТИКА

Д/А конверторот за секој од n-те битови D_i во зборот каде i=0, 1, 2, …, (n-1) треба да генерира аналогна вредност соодветна на реципрочната вредност од тежината G_i на тој бит во бинарниот збор и потоа сите овие аналогни вредности треба да ги собере и на својот излез да даде единствена аналогна вредност V_{ODAC} (V_{OUT}) според следнава равенка:

$$V_{ODAC} = K \cdot V_{REF} (D_{n-1}G_{n-1} + D_{n-2}G_{n-2} + ... + D_1G_1 + D_0G_0) \qquad (8\text{-}1)$$

Во равенката секој бит D_i од зборот што се конвертира може да има вредност само логичка 0 или 1, додека тежините G_i се претставени како степени на бројот 2 почнувајќи од најголемиот $2^{(n-1)}$ кон најмалиот 2^0=1: $2^{(n-1)}$, $2^{(n-2)}$, …, 8, 4, 2, 1, додека $+\mathbf{V_{REF}}$ е референтниот напон на кој е приклучен Д/А конверторот.

Во претходната равенка константата на пропорционалност K треба да има таква вредност, која ќе обезбеди излезниот напон да добива вредности во границите помеѓу 0 и V_{REF}. Тоа ќе биде исполнето само ако константата K е дропка, која го дели референтниот напон V_{REF} на помали делови, така што треба да важи $K=1/2^n$. Заменувајќи ја последната констатација во равенката (8-1), таа се трансформира и го добива следниов облик:

$$V_{ODAC} = V_{REF}/2^n \cdot (D_{n-1}2^{n-1} + D_{n-2}2^{n-2} + ... + D_12^1 + D_02^0) \qquad (8\text{-}2)$$

$$V_{ODAC} = V_{REF} \cdot (D_{n-1}1/2^0 + D_{n-2}1/2^1 + ... + D_11/2^{n-1} + D_01/2^n) \qquad (8\text{-}3)$$

Со ваков избор на константата $K=1/2^n$ (која важи за т.н. униполарна кодна шема или униполарно кодирање за најмалата дигитална вредност кога сите битови се 0-и се добива најмалото, нултото излезно ниво V_0=0V, додека за најголемата дигитална вредност кога сите битови се 1-и се добива најголемото излезно ниво $V_{n-1}=[(2^n-1)/2^n]\cdot V_{REF}$.

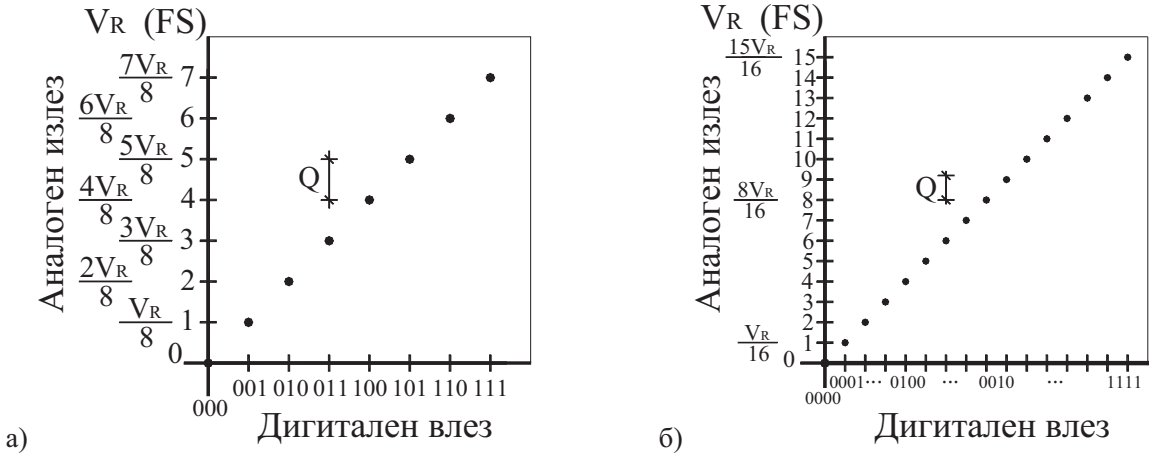
Имајќи ја предвид равенката (8-3) станува јасно дека колку е бројот на битови n поголем, толку попрецизно ќе биде добиеното излезно аналогно ниво бидејќи секој новододаден бит внесува дополнителна помала дропка во конечниот збир со кој се дефинира вредноста на излезното напонско ниво и со тоа го дели V_{REF} на помали можни делови.

Равенката (8-2) може да се напише и во следниов облик:

$$V_{ODAC} = V_{REF}/2^n \cdot d \qquad (8\text{-}4)$$

Во последната равенка со d е означена декадната вредност на бинарниот вектор изразена преку збирот ($D_{n-1}G_{n-1} + D_{n-2}G_{n-2} + ... + D_1G_1 + D_0G_0$).

На следната слика сл. 8-5 а) е прикажана идеализирана преносна карактеристика на трибитен Д/А конвертор. Ваквиот конвертор на својот влез може да добие дигитален сигнал претставен со било која од осумте можни комбинации: 000, 001, 010, ..., 110, 111 бидејќи со 3 бита се кодираат 2^3=8 различни комбинации. За секоја од нив Д/А конверторот треба на својот излез да продуцира соодветно напонско ниво во однапред зададени граници. Опсегот кој го претставува максималното можно ниво на излезниот аналоген сигнал се вика полн (целосен) износ на скалата и се означува со FS или FSR (анг. full scale range). Неговата вредност зависи од референтиот напон V_{REF} кој е приклучен на А/Д конверторот.



Сл. 8-5. Идеализирани преносни карактеристики на а) 3-битен б) 4-битен Д/А конвертор

Доколку станува збор за 4-битен Д/А конвертор, тогаш на неговиот влез ќе можат да се појават 2^4=16 различни 4-битни вектори, според сликата сл. 8-5 б), на која е претставена неговата преносна карактеристика. Ако претпоставиме дека полниот износ на скалата FS е ист и за двата случаи, јасно е дека за вториот пример излезниот сигнал ќе биде со поголема прецизност и подобар кавалитет, бидејќи целиот опсег FS се дели на два пати повеќе нивоа. Во врска со наведеното, како мерка за прецизност и квалитет на Д/А конверторот, се воведува поимот резолуција (анг. Resolution) кој се однесува на бројот на битови n на зборот, што тој може да ги конвертира, а со тоа и на вкупниот број различни напонски нивоа N што може да се добијат на излезот од Д/А конверторот. Имајќи во вид дека на влезот од Д/А конверторот се појавува бинарен збор со должина n–бита, вкупниот број на различни нивоа N се добива од познатата равенка $N = 2^n$.

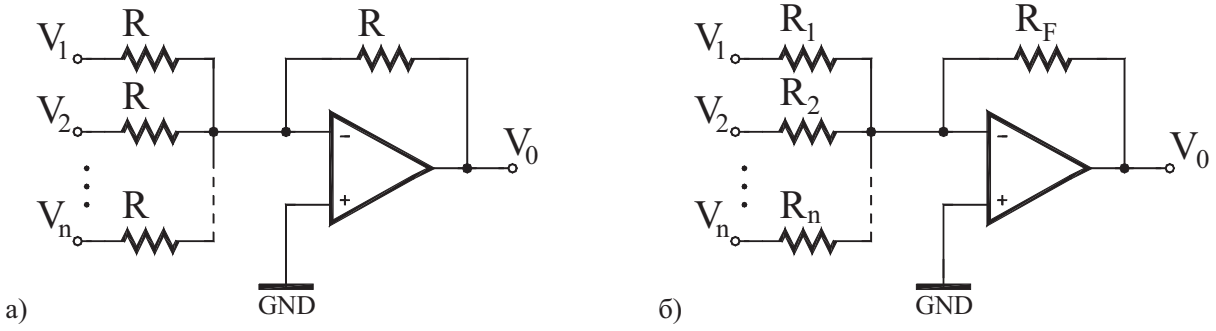
8.4. Д/А КОНВЕРТОРИ

Д/А конверторите во основа претставуваат кола, базирани на инвертирачки засилувачки степени, чиј излезен напон претставува сума на повеќе влезни напони. При ова излезниот напон, кој има максимална вредност, ограничена на однапред зададено референтно ниво $+V_{REF}$, се добива како збир од влезни дискретни напонски нивоа, чии вредности се цел број пати помали од референтното ниво. Ваквите делови имаат бинарна основа. Поточно, тие претставуваат количници на референтното ниво $+V_{REF}$ со цели степени на бројот 2, така што излезниот напон од Д/А конверторот се добива како збир од $V_{REF}/2$, $V_{REF}/4$, $V_{REF}/8$, $V_{REF}/16$, ... итн. Овие напонски нивоа кај најголемиот број Д/А конвертори се добиваат со помош на мрежа од прецизни отпорници со различни вредности, поврзани на влезот од суматорот, на чии влезови се појавува нулто напонско ниво ако битот на соодветниот влез е 0, или $+V_{REF}$, ако битот на тој влез е 1.

8.4.1. Д/А КОНВЕРТОР СО R/2^{n}R ТЕЖИНСКА ОТПОРНИЧКА МРЕЖА

Д/А конверторот со тежинска R/2^{n}R мрежа или со бинарна тежинска отпорничка мрежа, како што уште се вика, претставува една варијанта на познатата шема на суматор реализиран со операциски засилувач. Станува збор за инвертирачка конфигурација на операциски засилувач, побуден од повеќе влезни напони, според принципиелната електрична шема прикажана на сл. 8-6 а). Кај суматорот влезните сигнали се доведени преку отпорници со меѓусебно еднакви вредности $R_1=R_2=\ldots=Ri=\ldots=Rn=R$, со цел секој поединечен влезен напон V_i да има еднакво влијание врз излезениот напон V_0. Имајќи во вид дека негативната повратна спрега е изведена со отпорникот R_F кој има иста вредност со влезните отпорници ($R_F=R$), напонот на излезот V_0 ќе го претставува нивниот збир во инвертиран облик според рав. (8-5).

$$V_0 = -(V_1 + V_2 + \ldots + V_n) = -\Sigma V_i \qquad (8\text{-}5)$$



а) б)

Сл. 8-6. Електрични шеми на суматор со операциски засилувач а) со еднакви отпорници, б) со различни вредности на отпорниците

Доколку влезните отпорници се разликуваат еден од друг, според сл. 8-6 б), различно ќе биде и влијанието на поединечните напони врз излезниот напон. Имено, излезниот напон повторно ќе биде пропорционален со нивниот збир, но повеќе нема да ја претставува точната вредност на сумата. Уште повеќе, со промена на отпорникот R_F во повратната врска се контролира и се одредува засилувањето според следнава равенка:

$$V_0 = -R_F\left(\frac{1}{R_1}V_1 + \frac{1}{R_2}V_2 + \ldots + \frac{1}{R_n}V_n\right) \qquad (8\text{-}6)$$

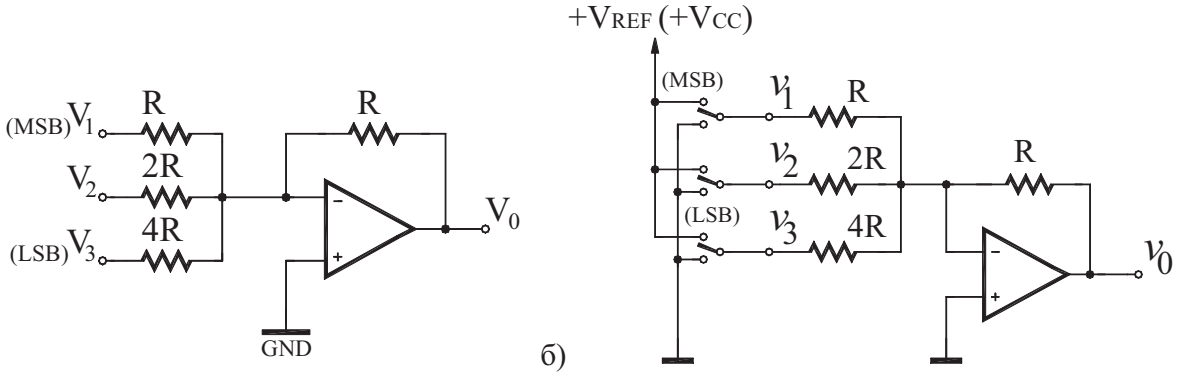
Токму оваа особина на суматорот е основа за правилна работа на Д/А конверторот, кај кој било кое од влезните напонски нивоа V_1, V_2, …, V_n може да биде високо ($+V_{REF}$) или ниско (0V) што зависи од тоа дали битот на соодветната позиција е 1, или 0. Од тука произлегува следнава равенка:

$$V_0 = -R_F \cdot V_{REF} \cdot \left(\frac{1}{R_1}D_1 + \frac{1}{R_2}D_2 + \ldots + \frac{1}{R_n}D_n\right) \qquad (8\text{-}7)$$

Во врска со последната забелешка да претпоставиме дека конфигурираме инвертирачко суматорско коло чии вредности на влезни отпорници меѓусебно треба да се разликуваат за два пати почнувајќи од најмалиот кон најголемиот. Поконкретно, ако отпорникот R_1 преку кој се носи првиот напон V_1 има вредност $R_1=R$, следниот отпорник ќе треба да има отпорност 2R, и секој нареден отпорник ќе се зголемува за два пати во однос на својот претходник. Според ова, ако станува збор за коло со три влеза (n=3), т.е. три битен Д/А конвертор чија шема е прикажана на сл. 8-7 а) или б) отпорниците ќе бидат: $R_1=2^0R=R$, $R_2=2^1R_1=2R$ и $R_3=2^2R_1=4R$.

Покрај ова, да избереме отпорник во повратната врска R_F чија вредност е два пати помала од онаа на отпорникот R_1=R, така што ќе важи $R_F=R_1/2=R/2$, заради што согласно равенките (8-6) и (8-7), излезниот напон ќе биде:

$$V_0 = -\left(\frac{1}{2}V_1 + \frac{1}{4}V_2 + \frac{1}{8}V_3\right) \qquad (8\text{-}8)$$



Сл. 8-7. Три битен Д/А конвертор со $R/2^nR$ тежинска отпорничка мрежа

Имајќи ги предвид равенките (8-6) и (8-7), станува јасно дека ваквиот избор на отпорници овозможува секој влезен напон V_1, V_2 и V_3, да има точно една половина помало влијание врз излезниот напон, во однос на претходниот, согласно равенката (8-8). Кажано со други зборови, влезниот напон V_1 ќе има најголем ефект и ќе влијае врз излезниот напон со една половина од максималниот референтен напон, следниот напон V_2 ќе влијае врз излезот за половина пати помалку од првиот, т.е. ќе допринесува со една четвртина од V_{REF}, додека V_3 со уште една половина помалку од него, што значи ќе делува со една осмина. Ваквите вредности на отпорниците, а со тоа и на влијанијата врз излезниот напон не се случајно избрани бидејќи ги претставуваат истите тежински односи кои одговараат на соодветните места на битовите кај приридниот бинарен броен систем како би се реализирала равенката (8-3). Според оваа анализа, во шемата од сл. 8-7 а) б), на влезот каде е приклучен најмалиот отпорник треба да се доведе битот со најголема тежина 2^{n-1} (MSB), додека на влезот каде што е најголемиот отпорникот треба да се приклучи битот со најмала тежина 2^0 (LSB). Конечно може да заклучиме дека ако на влезот од колото доведеме дигитални излези од три логички кола, а со тоа било која комбинација од три бита, излезниот напон изразен во волти ќе биде аналоген репрезент во декаден облик на бинарната вредност на овие три бита, според следнава равенка:

$$V_0 = -V_{REF} \cdot \left(\frac{1}{2}D_2 + \frac{1}{4}D_1 + \frac{1}{8}D_0\right) \qquad (8\text{-}9)$$

Во последната равенка (8-9) V_{REF} го претставува нивото на логичка 1, $V(1)=V_{REF}$, додека D_2, D_1 и D_0 се влезните битови чии вредности може да бидат 0 или 1, при што MSB е D_2 кој се доведува на првиот влез V_1, додека LSB е D_0 кој го побудува третиот влез V_3.

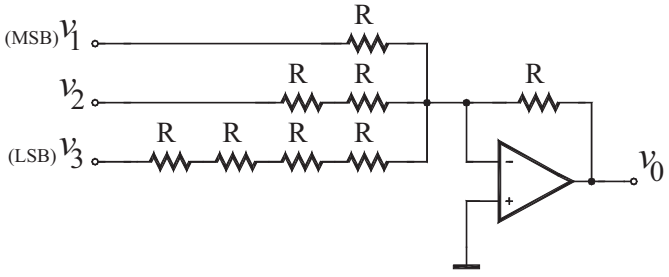
Во општ случај, за n-влезови (битови), равенката на овој Д/А конвертор може да се напише и во следниов облик

$V_{ODAC} = - R_F/R_0 \times V_{REF} \times d = - R_F/R_0 \times V_{REF} \times (2^{n-1}D_{n-1} + ... + 2^0D_0)$;

За точно пресметување на вредноста на излезните напони за секоја од осумте три-битни комбинации, од 000 до 111, кои може да се јават на влезот од колото, ќе ја примениме равенката (8-9) и ќе ја добиеме таб. 8-1. Во примерот како референтно ниво на Д/А конверторот е претпоставено напојување од 5V (V_{REF} = +5V), заради што ќе важи V(1)=+5V и V(0)=0V.

Декадна вредност	Бинарен еквивалент	Излезен напон [V]
0	000	0.000
1	001	- 0.625
2	010	- 1.250
3	011	- 1.875
4	100	- 2.500
5	101	- 3.125
6	110	- 3.750
7	111	- 4.375

Таб. 8-1. Еквивалентни вредности кај трибитен ДАК



Сл. 8-8. Трибитен ДАК со $R/2^nR$ тежинска отпорничка мрежа од еднакви отпорници

Дадената анализа може да се генерализира и врз основа на позната вредност на еден од отпорниците на Д/А конверторот да се изврши правилен избор на останатите отпорници. При ова, ако е познат

- најголемиот отпорник R_0 кај битот со најниска тежина LSB, тогаш отпорникот во повратната врска R_F треба да има вредност $R_F = R_0/2^n$;
- најмалиот отпорник R_{n-1} кај битот со највисока тежина MSB, тогаш отпорникот во повратната врска R_F треба да има вредност $R_F = R_{n-1}/2$;
- отпорникот во повратната врска R_F, тогаш најголемиот отпорник R_0 кај битот со најниска тежина LSB, треба да има вредност $R_0 = 2^n \cdot R_F$;

Да не заборавиме дека во секој случај вредностите на влезните отпорници треба со секој нареден отпорник последователно да се зголемуваат за два пати почнувајќи од најмалиот R_{n-1} кон најголемиот R_0.

Доколку се постави барање Д/А конверторот да има поголема прецизност, ќе треба да се зголеми бројот на битови n, а со тоа и бројот на влезови. При ова, за секој новододаден дигитален влез ќе се приклучи по еден дополнителен отпорник кој ќе има вредност два пати поголема од онаа на отпорникот преку кој се доведуваше LSB-битот, т.е. отпорник кој е два пати поголем од најголемиот отпорник што постои во мрежата. Ваквото барање ја покажува слабата страна на овој конвертор бидејќи бројот на прецизни отпорници со различни вредности станува сè поголем.

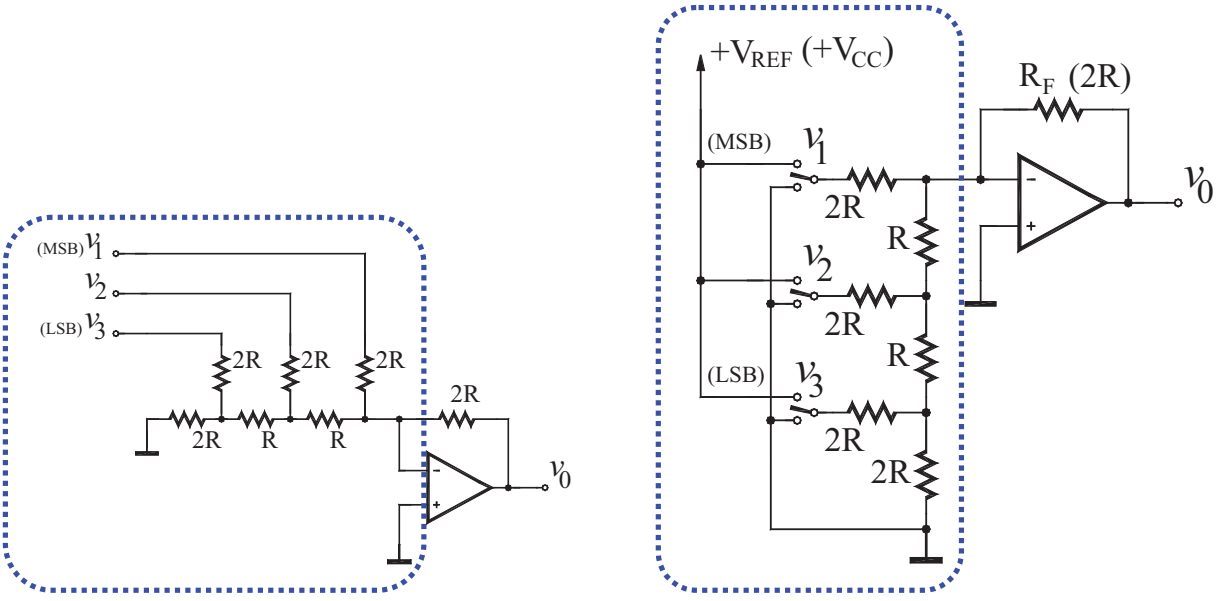
Д/А конверторот со тежинска $R/2^nR$ отпорничка мрежа може да се реализира и со примена на повеќе сериски поврзани отпорници за секој влез посебно, кои ќе имаат иста вредност. Притоа на секој нов влез ќе треба да се приклучат по два пати повеќе отпорници отколку на претходниот, според сл. 8-8. Ваквото поврзување го решава претходно наведениот проблем, но за жал, се јавува нов, а тоа е многу покомплексната конфигурација на отпорничката мрежа.

8.4.2. Д/А КОНВЕРТОР СО R/2R СКАЛЕСТА ОТПОРНИЧКА МРЕЖА

Како најголема слабост на претходниот Д/А конвертор со бинарна тежинска мрежа беше потребата од сè поголем број на прецизни и единствени отпорници со различни и сè поголеми вредности. За надминување на оваа слабост во практиката многу често се

користи едно прилично ефикасно решение со отпорничка мрежа чија топологија наликува на скала. Токму заради тоа, вака реализираниот конвертор и го добил името конвертор со скалеста мрежа, а заради вредностите на употребените отпорници тој популарно уште се нарекува и R/2R Д/А конвертор (анг. R/2R Ladder DAC).

И кај оваа конфигурација повторно се користи суматор, но сега на неговиот влез се формира отпорничка мрежа која користи отпорници само со две вредности според сл. 8-9 а) или б). Ако се направи споредба со Д/А конверторот со тежинска мрежа може да се забележи дека сега бројот на отпорници е незначително зголемен.



Сл. 8-9. а) б) Трибитен Д/А конвертор со скалеста R/2R отпорничка мрежа

Математичката анализа на применетата скалеста мрежа е нешто посложена од претходниот случај за што треба да се примени Тевененовата теорема. Овде само ќе го превземеме конечниот резултат без изведувањето. Тој покажува дека левиот дел од шемата со тежинската $R/2^nR$ мрежа може да се замени со еквивалентен Тевененов извор V_T со внатрешна отпорност R_T, чии вредности се идентични со оние кај претходниот Д/А конвертор со тежинска мрежа.

$$V_T = \frac{1}{2}V_1 + \frac{1}{4}V_2 + \frac{1}{8}V_3 \text{ и } R_T=R \qquad (8\text{-}10)$$

Со ова се добива и крајната равенка која се однесува на нивоата на излезниот напон соодветно на таб. 8-1.

$$V_0 = -\frac{R_F}{R_T}V_T = -\frac{R_F}{R_T}\left(\frac{1}{2}V_1 + \frac{1}{4}V_2 + \frac{1}{8}V_3\right) = -\frac{R_F}{R}\left(\frac{1}{2}V_1 + \frac{1}{4}V_2 + \frac{1}{8}V_3\right) \qquad (8\text{-}11)$$

Ако се избере R_F=R, тогаш за излезниот напон се добива изразот:

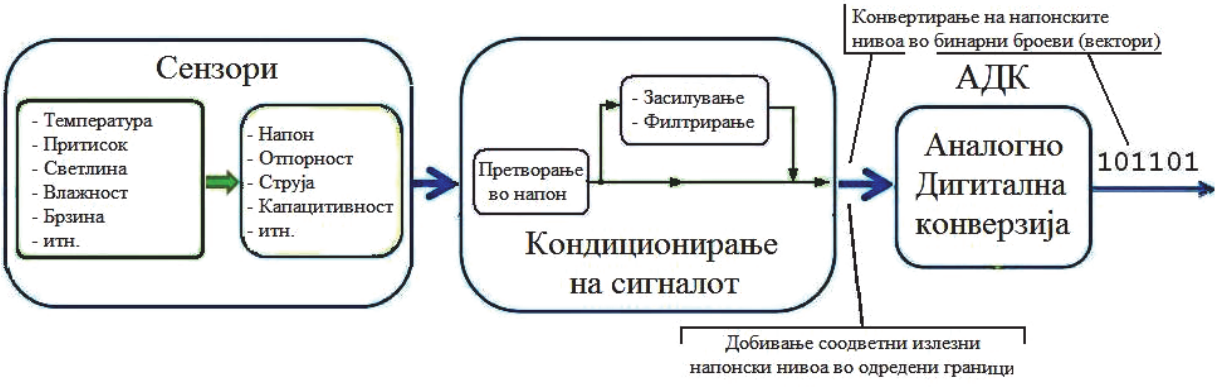
$$V_0 = -\left(\frac{1}{2}V_1 + \frac{1}{4}V_2 + \frac{1}{8}V_3\right) = -\left(\frac{1}{2}\cdot D_2 + \frac{1}{4}D_1 + \frac{1}{8}D_0\right)\cdot V_{REF} \qquad (8\text{-}12)$$

Бидејќи вредноста на влезните напони V_1, V_2, и V_3 може да биде само ниско ниво (0V) или високо ($+V_{REF}$) што зависи од тоа дали соодветниот бит е 0 или 1, добивме идентичен израз со веќе познатата равенка (8-9).

8.5. АНАЛОГНО-ДИГИТАЛНА КОНВЕРЗИЈА

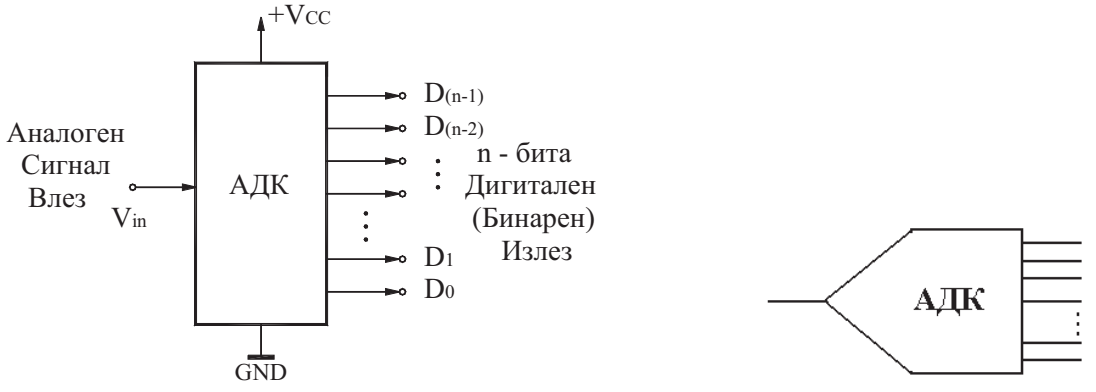
Дигиталните системи работат со дигитални сигнали, податоци кои се претставени во дигитален облик како бинарни зборови. Секој збор претставува комбинација од конечен број битови што системот ги добива на својот влез, а се однесува на одредена информација од надворешниот свет. Кога станува збор за комуникацијата помеѓу луѓето, таа се остварува на различни начини, но секогаш се поставува барање за претворување, поточно за кодирање на речениците и броевите во дигитален облик. Станува збор за информации кои се добиваат како комбинации од различни графички симболи (цифри и букви) и знаци (анг. characters). При извршувањето на пресметките се применуваат декадните цифри во комбинација со знаците на аритметичките и логичките операции, додека за зборовите и речениците тоа се буквите во комбинација со знаците за интерпункција. Конверзијата на ваквите податоци во дигитални е прилично едноставна бидејќи и декадниот броен систем и азбуките располагаат со конечен број цифри, букви и знаци, па тие претставуваат дискретни величини кои се конвертираат со примена на природниот бинарен броен систем, или некој друг бинарен систем или код.

Меѓутоа, природните појави и процеси во основа се континуални величини, како на пр. звукот, светлината, притисокот, температурата, брзината, и многу други. Имено, тие се менуваат континуирано со тек на времето и заради тоа можат да добијат било која вредност. Покрај ова, секоја природна појава има свои карактеристики, а соодветно на тоа и посебни единици според кои тие се мерат, така што секој континуален сигнал може дигитално да се обработува само ако се претвори во аналогна електрична величина, напон или струја, и потоа се дигитализира. Проблемот на претворање на континуалните сигнали во дигитални, кои се погодни за обработка на компјутер, или друг дигитален систем, е прилично комплексна работа што треба да ја изврши системот за аквизиција на податоци (анг. data acquisition system) чија поедноставена блок шема е прикажана на сл. 8-10.



Сл. 8-10. Блок-шема на ситем за аквизиција на податоци

Прилагодувањето се изведува во повеќе чекори почнувајќи од трансформација на континуалната појава во електрична величина аналогна на неа, како на пр. струја, отпорност, и сл., потоа нејзино претворување во напон и по потреба негово кондиционирање, и на крај конвертирање на тој напон во дигитален (бинарен) облик. Претворувањето на физичките величини како што се на пр. топлината, светлоста, притисокот, итн. во електрични се врши со помош на сензорски елементи анг. (sensors). Кондиционирањето претставува подобрување и прилагодување на напонскиот сигнал на различни нивоа во точно дефинирани граници. Крајниот чекор е претворувањето на аналогниот напонски сигнал во дигитален што го извршува најважниот блок, а тоа е А/Д конверторот чија наједноставна блок шема е прикажана на сл. 8-11 а), додека неговата симболичка ознака е дадена на 8-11 б).



а) Наједноставна блок-шема б) симболичка ознака

Сл. 8-11. А/Д конвертор

А/Д конверторот на својот влез добива аналоген напон V_{INA} со амплитуда која се менува континуирано со тек на времето, а на излезите генерира бинарно кодирани зборови (вектори), поточно одредени комбинации (низи) битови со конечна должина, кои одговараат на избрани конкретни (селектирани) вредности од влезниот напон (примероци, одбироци). Амплитудата на аналогниот сигнал треба да биде ограничена и да припаѓа во границите одредени со минимално и максимално дозволено влезно ниво (V_{AMAX} и V_{AMIN}). За А/Д конверторите кои најчесто се сретнуваат во праксата влезниот опсег е или позитивен и тогаш важи $V_{AMAX}=V_{REF}$, додека $V_{AMIN}=0$ V, или влезниот сигнал може да биде и позитивен и негативен во границите на симетрично дефиниран опсег кога важи $V_{AMAX}=+V_{REF}$ и $V_{AMIN}=-V_{REF}$. Најчесто овие нивоа всушност се напојувањата на конверторот.

8.6. ОСНОВНИ ПОИМИ И КОНЦЕПТИ

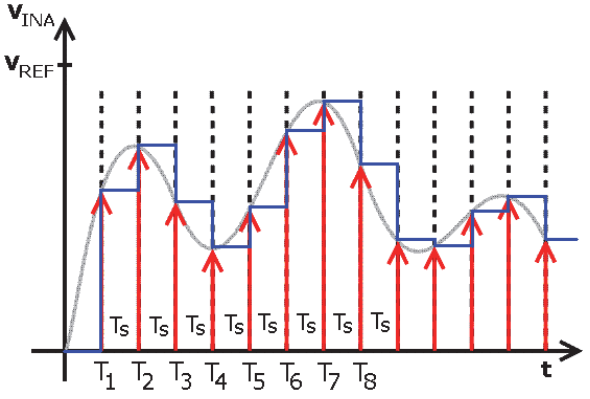
Процесот на А/Д конверзија се извршува во две фази. Најнапред, во првата фаза аналогниот сигнал се дискретизира по време, а потоа, во втората фаза се дискретизира по ниво (по амплитуда), т.е. се квантизира и се кодира.

Дискретизацијата по време претставува земање на одбироци (примероци), поточно одредени напонски нивоа од сигналот во регуларни временски интервали, со периода T_S, односно со одредена фреквенција на одбирање f_S, заради што се користи и терминот одбирање или семплирање (анг. sample).

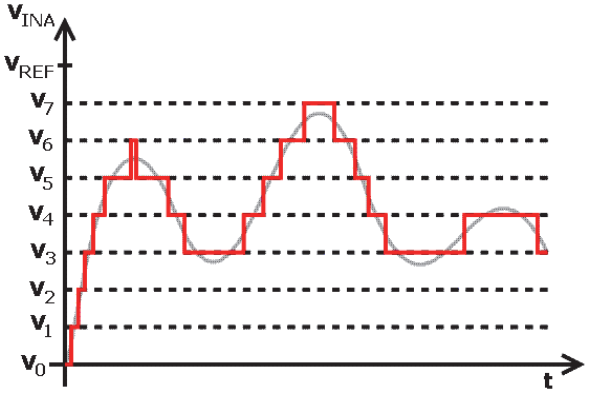
Дискретизацијата по ниво или квантизацијата, (термин кој најчесто се користи во практиката), врши претворување на амплитудата (нивото) на секој одбирок од континуална (бесконечно прецизна, точна вредност) во конкретен број кој претставува вредност на дискретно напонско ниво со одредена конечна прецизност.

По квантизацијата секое дискретно ниво кое има конкретна декадна вредност (d) се претворува во посебен бинарен збор (вектор) според одреден код, т.е. комбинација од битови со конечна должина (n). Можеството на сите зборови го претставува конечниот облик на дигиталниот сигнал кој може да се добие со различни техники на кодирање. Униполарното кодирање се однесува на природниот бинарен броен систем и се применува кога се работи за позитивен опсег на влезниот аналоген напонски сигнал. Кога станува збор за симетричен опсег на влезниот напон се применува биполарното кодирање. Тоа се извршува според некој бинарен систем со знак како што е на пр. вториот комплемент (RC системот).

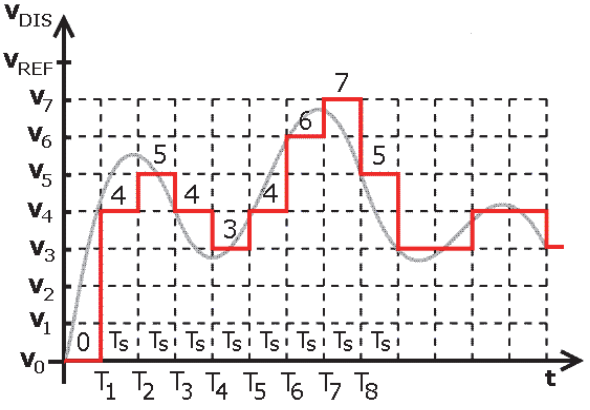
На следните слики, сл. 8-12 а), б), в) и г) последователно е прикажан еден пример на влезен аналоген сигнал, неговиот изглед кога е дискретизиран по време, кога е дискретизиран по ниво (квантизиран) во осум нивоа, кога е дискретизиран и по време и по ниво и конечниот изглед на дигиталниот сигнал како низа од битови кој може да биде обработуван од страна на дигиталниот систем. Секој одбирок има едно од осум можни нивоа кои се кодирани со три бита со што е дефинирана и должината на кодниот збор.



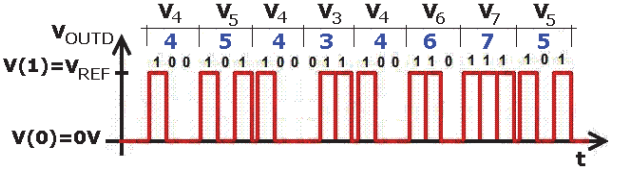
Сл. 8-12. а) *Временско дискретизирање* - земање на одбироци од влезен аналоген напон (временските интервали се дискретни, додека напонските нивоа се континуални)



Сл. 8-12. б) *Дискретизирање по ниво* или *Квантизирање* – Разделување на напонскиот опсег на конечен број напонски нивоа (временскиот период е континуиран, додека напонските нивоа се дискретизирани)



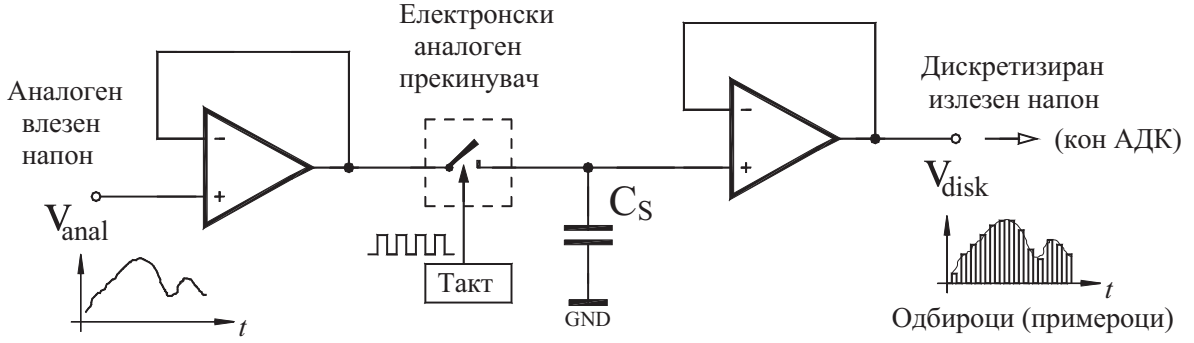
Сл. 8-12. в) *Дискретизиран и квантизиран сигнал* – дискретни временски интервали и дискретни вредности на напонските нивоа



Сл. 8-12. г) *Дигитален напонски сигнал* добиен во процесот на А/Д конверзија еквивалентен на влезниот аналоген напон

На сл. 8-13 е прикажана блок шема на процесот дискретизација, т.е. на земање на одбироци на некој напонски сигнал. Влезниот аналоген напон преку напонски следител со единечно напонско засилување се доведува на коло за одбирање и задржување (анг. Sample-and-Hold, SH) на влезот од А/Д конверторот. Ова коло, наједноставно земено, е комбинација од аналоген прекинувач и кондензатор. Прекинувачот периодично се отвора и се затвора согласно фреквенцијата на одбирање f_S која зависи од времетраењето на периодата на такт импулсите T_S кои го контролираат неговото вклучување и исклучување. За време на едната полупериода на тактот прекинувачот е затворен и тогаш се одбира едно ниво на влезниот напон со што се остварува дискретизација по време. Во следната полупериода прекинувачот се отвора и последната моментална вредност на влезниот напон,која беше присутна непосредно пред отворањето на прекинувачот на кондензаторот и го имаше наполнето до тоа ниво, се задржува.

Според кажаното, на кондензаторот, а со тоа и на излезот од вториот напонски следител кој има единечно напонско засилување, исто како и првиот, се добива временски дискретизиран сигнал со скалест облик формиран од низа на последователни одбироци, чии амплитуди треба понатаму да се дискретизираат по ниво, т.е. да се квантизираат.



Сл. 8-13. Процес на земање на одбироци (примероци) од аналоген сигнал

Времето кое ќе измине додека повторно да се затвори прекинувачот се користи за напонското ниво на тековниот одбирок, (кое е присутно на кондензаторот и на излезот од напонскиот следител), да се проследи на понатамошна обработка, поточно на квантизација. Во врска со ова, циклусот на конверзија го претставува временскиот период кој е потребен на А/Д конверторот за да ја одреди дискретната вредност на нивото на земениот напонски одбирок од влезниот аналоген сигнал, т.е. да изврши негова конверзија во соодветен дигитален облик на излезот кој претставува збор (бинарен вектор, комбинација) чија должина n типично изнесува од 4 до 16 бита, зависно од потребите.

Помеѓу фреквенцијата на одбирање f_S и брзината на промена на нивото на аналогниот влезен сигнал, а со тоа и на неговиот максимален фреквентен опсег $\mathbf{f_{MAX}}$, во процесот на А/Д конверзија, постои едноставен однос за да не дојде до изобличување на аналогниот сигнал при неговата рекострукција во инверзниот процес на Д/А конверзија. Поконкретно, аналогниот сигнал може целосно да се реконструира од земените примероци ако се задоволи т.н. теорема за одбирање (анг. *Sampling Theorem).* Според неа, за да не дојде до изобличување и појава на шум, од континуален сигнал чија највисока фреквенција во неговиот спектар е f_{MAX}, треба да се земаат одбироци со фреквенција f_S, која е барем два пати поголема од неа според равенката (8-13), позната како Најквистов критериум.

$$f_S \geq 2f_{MAX} \qquad (8\text{-}13)$$

Доколку фреквенцијата на одбирање f_S не го исполнува горниот услов, односно важи обратниот $f_S<2f_{MAX}$, ќе дојде до појавување на лажни фреквенции (анг. aliasing). Тоа е појава на непостоечки фреквенции во спектарот на реконструираниот сигнал, кои ги нема во оригиналниот сигнал, а кои се јавуваат како последица од пресликувањето на некоја реална (постоечка) фреквенција од влезниот сигнал.

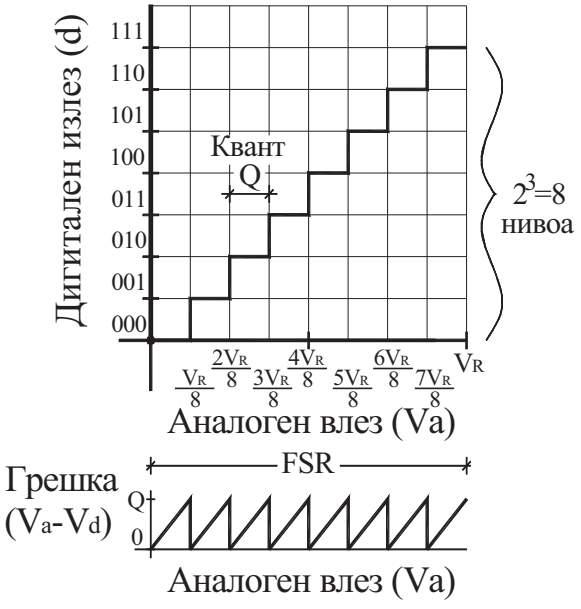
Од до сега изнесеното можеме да извлечеме заклучок дека за правилна работа на Д/А конверторот времетраењето на циклусот на конверзија треба да биде помало од периодата на одбирање T_S, за да не дојде нов одбирок, а претходниот да остане неконвертиран.

Што се однесува до процесот на квантизација, тој ќе биде обработен во поголеми детали во понатамошниот текст.

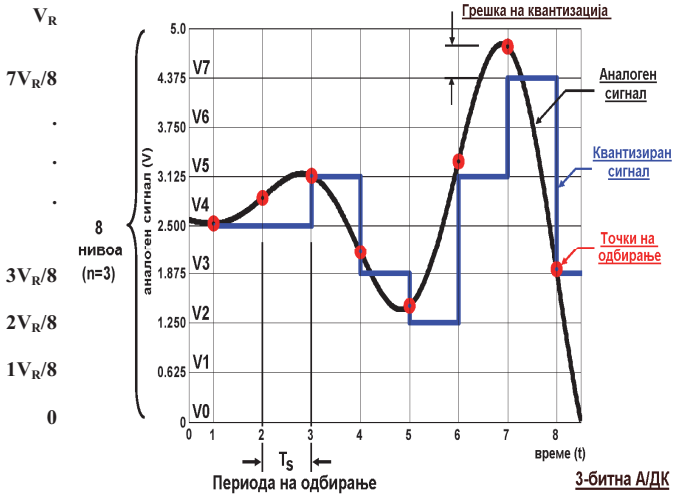
8.7. КАРАКТЕРИСТИЧНИ ПАРАМЕТРИ И ПРЕНОСНА КАРАКТЕРИСТИКА

Од сè она што претходно наведовме, може да заклучиме дека во А/Д конверторот при процесот на квантизација во период на еден такт интервал Ts се врши сведување на даден напонски интервал Q од влезниот аналоген сигнал (множество напонски вредности) кој се наоѓа помеѓу две соседни референтни напонски нивоа (V_i;V_{i+1}) каде што i=0, 1, 2, …, n во околината на дадено референтно напонско ниво V_i во него како единствена дискретна вредност. Секое следно референтно ниво V_{i+1} е поголемо од претходното V_i за една иста вредност која се нарекува чекор на квантизација или квант и се означува со Q, т.е. важи $V_{i+1}=V_i+Q$.

Процесот на А/Д конверзија најчесто се остварува преку коло кое реализира преносна функција со скалест облик како на сл. 8-14 на која е прикажана конкретна преносна карактеристика која овозможува извршување на процес на А/Д конверзија со три бита. (За да се реализира процесот на А/Д конверзија ќе биде потребна преносна функција со облик како на сл. 8-14). Заради дополнително појаснување на процесот на квантизирање на сл. 8-15 е прикажан еден пример на аналоген влезен сигнал и соодветниот дискретизиран излезен сигнал за дадената преносна карактеристика.



Сл. 8-14. Идеализирана преносна карактеристика на три битен А/Д конвертор



Сл. 8-15. Аналоген влезен сигнал и соодветен излезен дискретизиран излезен сигнал

Од сликите може да се забележи дека амплитудата на аналогниот сигнал е ограничена помеѓу две нивоа: најниското нулто напонско ниво од V_0=0 V и највисокото референтно напонско ниво $+V_{REF}$, заради што овој опсег се нарекува и полна или целосна скала и се означува со FS или FSR (анг. Full Scale Fange). Во општ случај, полната скала FSR ја претставува разликата помеѓу најголемата (V_{INAmax}) и најмалата вредност (V_{INAmin}) што може да ја добие влезниот аналоген напон, т.е. $FSR = V_{INAmax} - V_{INAmin}$. Ако влезниот напон е позитивен и се движи во границите помеѓу 0V и одредено референтно ниво $+V_{REF}$, како во нашиот случај, тогаш последната равенка се поедноставува и го добива следниов облик: $FSR = V_{REF} - 0\ V = V_{REF}$. Бидејќи во примеров од сликите сл. 8-14 и сл. 8-15 референтно напонско ниво е V_{REF}=+5V, а најниското 0V, и полната скала е FSR =5V.

Покрај ова, од сликите се гледа дека полниот опсег е поделен на осум напонски опсези што произлегува од фактот дека станува збор за А/Д конверзија со три бита. Имено, бројот на напонски опсези N_U зависи од бројот на битови n со кои се врши конверзијата и се пресметува според познатата равенка:

$$N_U=2^n \qquad (8\text{-}14)$$

За конкретниов пример n=3, така што $N_U=2^3=8$. Од тука може да се добие ширината на напонскиот интервал на секој напонски опсег Q ако износот на полната скала FS се подели со вкупниот број на напонски опсези N_U.

$$Q=FSR/N_U= FSR/2^n \qquad (8\text{-}15)$$

За конкретниот пример $Q=5\ V /2^3 = 5\ V /8 = 0{,}625\ V$.

Сите напони кои се наоѓаат во овој напонски интервал се заменуваат со единствена вредност зависно од тоа каде се наоѓа конкретниот интервал. Имено, за конкретниот пример, сите влезни напони помеѓу 0 и влезно напонско ниво помало од 0,625 V ќе се конвертираат во нулто референтно ниво V_0=0 V кое се кодира како 000, влезни напони помеѓу 0,625 V и 1,250 V ќе се конвертираат во прво референтно ниво V_1=0,625 V кое се кодира како 001, итн. сè до последниот опсег помеѓу 4,325 V и 5 V кој ќе се конвертира во последото седмо референтно ниво V_7=4,325 V кое се кодира како 111. Може да се забележи дека бројот на референтни напонски нивоа N_Q е еднаков со бројот на опсези N_U ($N_Q=N_U=2^n$) започнувајќи од нултото V_0 до последното V_{n-1} (во случајов $V_{n-1}=V_7$), со меѓусебна напонска разлика еднаква на опсегот Q. Врската помеѓу декадната вредност на секое напонско ниво V_i и неговата вредност изразена во волти се пресметува според следнава равенка:

$$V_i = i \cdot \frac{V_{REF}}{N_Q} = i \cdot \frac{V_{REF}}{2^n}\text{, каде што i=0, 1, 2, …, (n-2), (n-1)} \qquad (8\text{-}16)$$

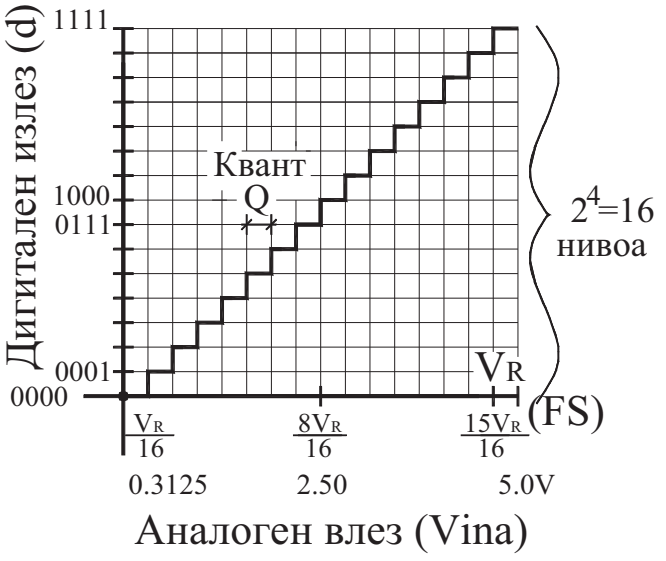
Големината на напонскиот интервал кој ги оддалечува дискретните референтни напонски нивоа едно од друго се вика квант и како што видовме се означува со Q. Во нашиот случај при рефрентен напон V_{REF}=+5V и А/Д конверзија со три бита, големината на квантот изнесуваше $1/8 \cdot V_{REF} = 0{,}625$ V, што може да се види и од таб. 8-2.

Аналоген напонски интервал		Напонска разлика	Рефрентно напонско ниво [V]		Декадна вредност	Бинарен еквивалент
$V_0 – V_1$	0.000 – 0.625	Q	$V_0=0V_R/8$	0.000	0	000
$V_1 – V_2$	0.625 – 1.250	Q	$V_1=1V_R/8$	0.625	1	001
$V_2 – V_3$	1.250 – 1.875	Q	$V_2=2V_R/8$	1.250	2	010
$V_3 – V_4$	1.875 – 2.500	Q	$V_3=3V_R/8$	1.875	3	011
$V_4 – V_5$	2.500 – 3.125	Q	$V_4=4V_R/8$	2.500	4	100
$V_5 – V_6$	3.125 – 3.750	Q	$V_5=5V_R/8$	3.125	5	101
$V_6 – V_7$	3.750 – 4.375	Q	$V_6=6V_R/8$	3.750	6	110
$V_7 – V_R$	4.375 – 5.000	Q	$V_7=7V_R/8$	4.375	7	111

Таб. 8-2. Напонски нивоа кај А/Д конверзија според преносната карактеристика од сл. 8-14

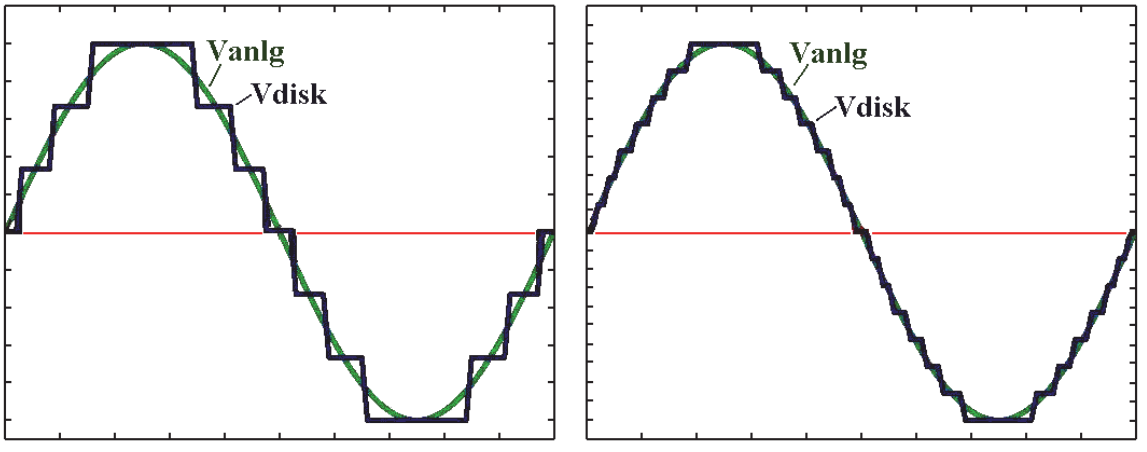
Од сè она што досега изложивме станува јасно дека ако се зголеми бројот на битови n ќе се намали големината на квантот, а со тоа ќе се зголеми бројот на референтни нивоа и ќе се зголеми прецизноста на квантизерот бидејќи помал напонски опсег ќе се

конвертира во единствено напонско ниво. Така на пр. ако примениме А/Д конверзија со n=4 бита, бројот на референтни нивоа ќе биде два пати поголем $N_U=2^4=16$, од $V_0=0$ V па до $V_{15}=4{,}6875$ V во однос на конверзијата со три бита, додека квантот ќе биде два пати помал, $Q=5\ V/2^4=5\ V/16=0{,}3125$ V, според преносната карактеристика дадена на сл. 8-16 со што два пати ќе се зголеми и прецизноста на квантизерот.



Сл. 8-16 Идеализирана преносна карактеристика на четирибитен А/Д конвертор

Така на пр. ако земеме една периода на простопериодичен сигнал и истата ја дискретизираме еднаш до три битен А/Д конвертор, а еднаш со четирибитен ќе ги добиеме следните две слики означени со сл. 8-17 а) и сл. 8-17 б).



Сл.8-17. Квантизација на простопериодичен сигнал а) со три бита б) со четири бита

Од сликите 8-17 а) и б) лесно се забележува дека квалитетот во вториот случај е многу подобар, што укажува на тоа дека прецизноста е поголема ако се користат поголем број на битови за квантизирање. Во врска со наведеното се дефинира поимот резолуција (анг. resolution) на конверторот кој се изразува како број на битови (n) на бинарниот вектор, (должина на зборот) со кој се кодира било кое влезно дискретно напонско ниво.

Од тука произлегува дека резолуцијата го покажува вкупниот број на дискретни вредности што може да се генерираат од А/Д конверторот, за даден целосен напонски опсег на влезни вредности FSR. Според ова, резолуцијата може да се изрази и во волти.

Вредноста на резолуцијата се добива кога полната скала на влезниот аналоген опсег FSR се подели со вкупниот број на дискретни напонски нивоа (N_Q) според равенката:

$$Q = \frac{FSR}{N_Q} = \frac{FSR}{2^n} \tag{8-17}$$

Практично земено резолуцијата го претставува најмалиот напон чие појавување на влезот од А/Д конверторот сигурно ќе предизвика промена на излезното ниво кодирано во битови кај излезниот дигитален сигнал, а тоа значи промена на битот со најмала тежина LSB. Од тука резолуцијата се нарекува квант, LSB напон, резолуционен напон или чекор на квантизација и покрај Q се означува и со Δ.

Како математичка поддршка на изложеното ќе ја презентираме т.н. равенка на А/Д конверторот од каде попрецизно математички се гледа влијанието на бројот на битови за конверзија врз прецизноста на А/Д конверторот.

Да претпоставиме дека разгледуваме А/Д конвертор со резолуција од n бита, каде V_{REF} е референтниот напон, и дека V_{INA} е аналогниот влезен напон кој ќе се конвертира. Излезниот напон од А/Д конверторот треба да се добие во облик на бинарен збор (вектор) од n бита ($D_{n-1}D_{n-2}\ldots D_1D_0$), со својата еквивалентна декадна вредност d (во декадна нотација). Декадната вредност d еквивалентна на влезниот аналоген сигнал конвертиран во дигитален облик, т.е. на бинарниот вектор се пресметува како цел дел од количникот помеѓу конкретната вредност на аналогниот влезен напон V_{INA} и квантот Q (Δ) на компараторот според равенката

$$d = \left[\frac{Vin}{Q} \right] \tag{8-18}$$

Во оваа равенка квантот (резолуцијата) Q (Δ) ја претставува најмалата промена на влезниот напон која може да биде откриена (детерминирана) од страна на А/Д конверторот, со што практично се дефинира и неговата прецизност.

$$Q = \frac{Van\max - Van\min}{N_Q} = \frac{V_{REF} - 0}{N_Q} = \frac{V_{REF}}{2^n} \tag{8-19}$$

каде што n е бројот на битови на излезниот дигитален сигнал.

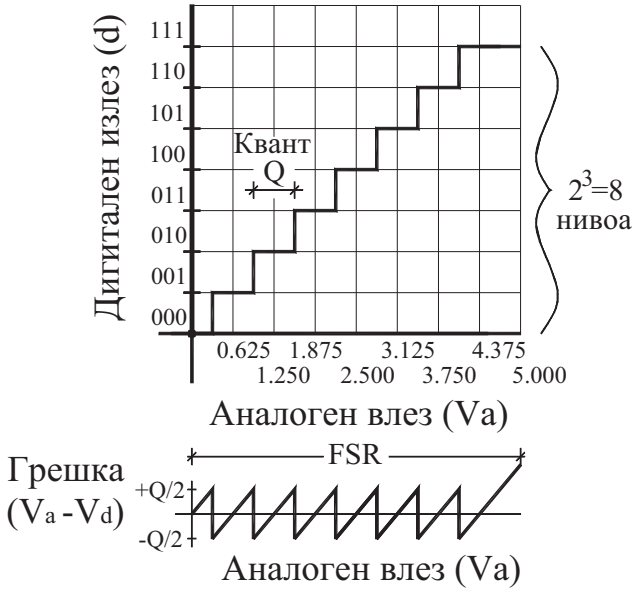
Како мерка за прецизноста на квантизерот се дефинира и грешката на квантизација (анг. Quantization Error), која претставува разлика помеѓу точното ниво на одбирокот на аналогниот влезен сигнал и дискретната вредност на излезот од квантизерот, т.е. отстапување во однос на одредено референтно ниво V_i. Тоа е средната вредност на разликата помеѓу аналогниот влез и неговата квантизирана вредност. Во идеален случај на А/Д конверзија, која се однесува на квантизер, чија преносна карактеристика е дадена на сл. 8-14, грешката се движи во границите од 0 до Q: од V_i до (V_i+Q).

Во практиката вообичаено се врши поместување на преносната карактеристика на А/Д конверторот од сл. 8-14 или 8-16 за Q/2 така што преносната карактеристика на А/Д конверторот најчесто го има обликот прикажан на сл. 8-18 за која одговара таб. 8-3 со напонски нивоа која се однесува на три битен А/Д конвертор, а заради споредување и полесно воочување на разликите во однос на претходниот пример од сл. 8-14.

Аналоген напонски интервал		Напонска разлика	Рефрентно напонско ниво [V]		Декадна вредност	Бинарен еквивалент
$V_0 - V_1$	0.0000–0.3125	0,5·Q	$V_0=0V_R/8$	0.000	0	000
$V_1 - V_2$	0.3125–0.9375	Q	$V_1=1V_R/8$	0.625	1	001
$V_2 - V_3$	0.9375–1,5625	Q	$V_2=2V_R/8$	1.250	2	010
$V_3 - V_4$	1,5625–2,1875	Q	$V_3=3V_R/8$	1.875	3	011
$V_4 - V_5$	2,1875–2,8125	Q	$V_4=4V_R/8$	2.500	4	100
$V_5 - V_6$	2,8125–3,4375	Q	$V_5=5V_R/8$	3.125	5	101
$V_6 - V_7$	3,4375–4,0625	Q	$V_6=6V_R/8$	3.750	6	110
$V_7 - V_R$	4,0625–5.000	1,5·Q	$V_7=7V_R/8$	4.375	7	111

Таб. 8-3. Напонски нивоа кај А/Д конверзија според преносна карактеристика од сл. 8-18

Ваквата промена овозможува заокружување на вредноста во околина на конкретно референтно напонско ниво што е математички поопрадано. Инаку и во овој случај даден на сл. 8-18 се гледа дека најголемата грешка на квантизација по апсолутна вредност изнесува колку што е и вредноста на еден квант Q, како на сл. 8-14, но сега таа се наоѓа во околината на референтното ниво плус/минус половина квант: од (V_i -Q/2) до (V_i +Q/2).



Сл. 8-18. Идеализирана преносна карактеристика и грешка кај три битен А/Д конвертор

8.8. ПОДЕЛБА И ВИДОВИ НА А/Д КОНВЕРТОРИ

Постојат повеќе базични постапки врз чија основа се градат А/Д конверторите. Најголемиот број од нив може да се поделат на следниве поголеми групи:

1. Паралелни (Флеш) А/Д конвертори (анг. Flash или Parallel ADC),
2. А/Д конвертори базирани на Д/А конверзија (анг. DAC based ADC),
3. А/Д конвертори со интегрирање (анг. Integrating ADC) и
4. Делта-сигма (сигма-делта) А/Д конвертори (анг. delta-sigma или sigma-delta ADC).

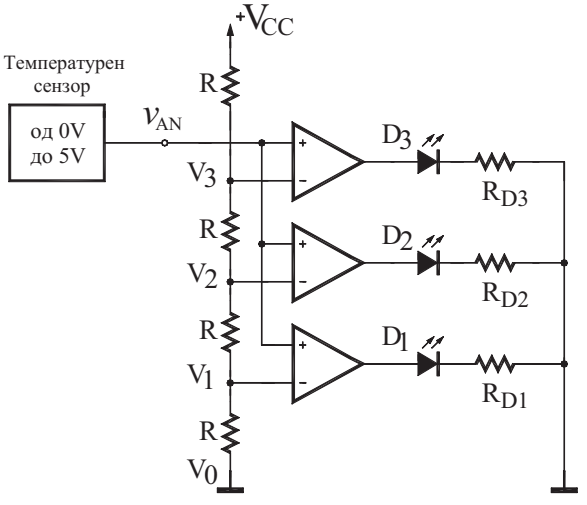
А/Д конверторите од секоја група меѓусебно се разликуваат во поглед на перформансите како што се на пр. брзината на работа, прецизноста и цената на чинење, од што зависи и нивната практична примена. Во продолжение посебно ќе се задржиме на конфигурацијата и анализата на принципот на работа на типичните претставници за секоја од наведените групи. Имајќи во вид дека секоја група А/Д конвертори има различна практична примена, во продолжение посебно ќе се задржиме на секоја од нив.

8.8.1. ПАРАЛЕЛЕН АДК

Паралелниот А/Д конвертор или А/Д конверторот со директна конверзија заради својата речиси молскавична брзина на работа популарно се нарекува и флеш или блиц (анг. flash) А/Д конвертор. Овој А/Д конвертор функционира на прилично едноставен начин, така што е доста лесен за разбирање. Имено, конверторот преку компаратори врши споредување на влезниот аналоген напон со однапред поставени и дефинирани фиксни напонски нивоа кои се добиени со разделување на зададениот референтен напон V_{REF}. Неговата вредност е еднаква со максималното напонско ниво кое може да го достигне влезниот аналоген напон. Така на пр. ако референтниот напон е 5 V, тогаш и највисокото ниво на влезниот аналоген напон треба да биде 5 V. Референтниот напон се разделува преку соодветна отпорничка мрежа која игра улога на напонски делител составен од еднакви отпорници. При тоа напонот од секој отпорник се зема како одделен референтен напон за секој компаратор на кој е поврзан, така што моменталната вредност на влезниот аналоген напон се компарира со конечен број на посебни референтни напони.

На следната слика сл. 8-19 е прикажан еден многу едноставен пример на флеш А/Д конвертор со референтен напон V_{REF}=+5V и четири поединечни компараторски нивоа, побуден од напонот кој го генерира температурниот сензор. Бидејќи сите отпорници имаат еднаква отпорност (R) вредностите на посебните напонски нивоа ќе бидат дадени со следниве равенки:

$$V_3 = \frac{V_{CC}}{4R} \cdot R = 3.75V\ ,\ V_2 = \frac{V_{CC}}{4R} \cdot 2R = 2.5V\ ,\ V_1 = \frac{V_{CC}}{4R} \cdot 3R = 1.25V\ \ ,\ V_0 = 0V \quad (8\text{-}20)$$



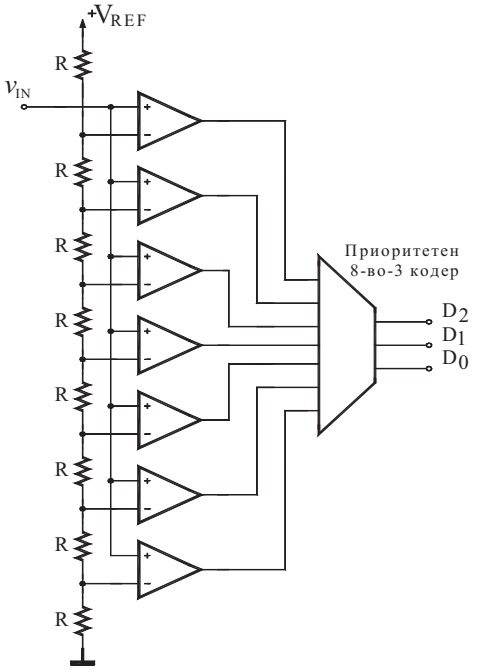
Сл. 8-19. Флеш А/Д конвертор со четири нивоа

На излезите од секој компаратор се наоѓаат светлечки LED диоди при што одредена диода ќе засвети во моментот кога излезното напонско ниво од температурниот сензор го достигне соодветното компараторско ниво на компараторот на чиј што излез е таа поврзана. Нејзиното светнување ќе покаже дека на тој излез е присутно високо ниво, 1.

Така на пр. ако влезниот аналоген напон е под V_1=1.25 V ниту еден компаратор не реагира и јасно ниту една од LED диодите нема да свети. Ако влезниот напон го надмине првиот референтен праг од V_1=1.25 V ќе засвети првата LED диода D_1, ако го надмине второто ниво од V_2=2.5 V покрај првата, ќе засвети и втората LED диода D_2, а кога ќе го надмине третото ниво V_3=3.75 V, покрај првите две диоди ќе засвети и третиот LED D_3. Сигнализацијата е прилично корисна и предупредувачка бидејќи со порастот на температурата расте и напонот, а со тоа и бројот на диоди што светат. Меѓутоа ваквиот принцип на работа не е најсоодветен за процесирање од страна на компјутер или некој друг дигитален систем бидејќи на излезот се добива сигнал кој не е кодиран во соодветен бинарен облик.

За конкретниов пример, за да се добие бинарно кодиран излезен сигнал, излезите од компараторите ќе треба да се приклучат на 4-во-2 кодер. Со приклучување на влезовите од кодерот на излезите од операциските засилувачи, при секое достигнување на едно од трите референтни нивоа на излезот од кодерот ќе се генерира единствена бинарна комбинација од два бита. Бидејќи температурата континуирано расте или се намалува, вакво однесување ќе може да се обезбеди само ако се примени кодер со приоритет.

Заради полесно разбирање на принципот на работа на ваквиот А/Д конвертор, ќе ја анализираме принципиелната електрична шема дадена на сл. 8-20 на која е прикажан еден прилично едноставен, трибитен флеш А/Д конвертор со 8-во-3 приоритетен кодер при што како компаратори се користат идеални операциски засилувачи. Бидејќи претпоставивме дека на излезот ќе треба да се генерира трибитен излезен вектор, тоа значи дека неговата најмала вредност ќе биде $0_{(10)}$ или $000_{(2)}$, односно како најголема вредност ќе се појави $111_{(2)}$, или $7_{(10)}$.



Сл. 8-20. Принципиелна електрична шема на 3-битен флеш АДК

Токму заради тоа што се работи за трибитен А/Д конвертор отпорничкиот делител е формиран од 2^3=8 отпорници со еднакви вредности и седум компаратори. Излезите од компараторите се доведуваат како влезови на кодерот. Бидејќи се работи за кодер кој реагира со приоритет на нивото на влезниот напон, на неговиот излез ќе се појави оној коден збор кој одговара на влезното аналогно ниво кое е најблиско до најголемото од седумте референтни нивоа различни од нултото. Наједноставна реализација на 8-во-3 приоритетниот кодер е примената на ик со ознака 74x148.

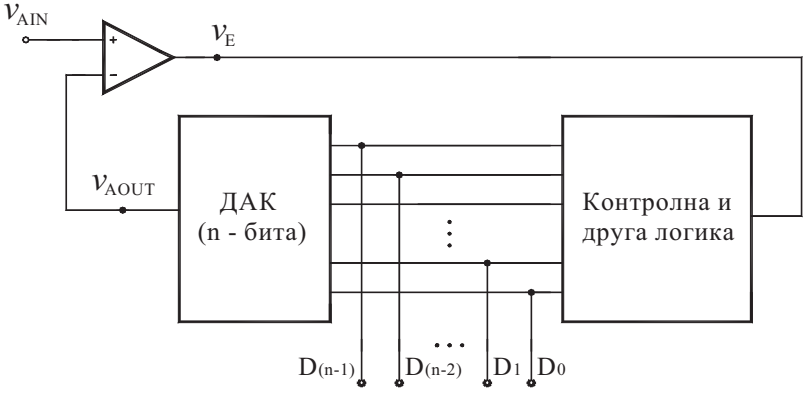
Имајќи во вид дека разгледуваниот А/Д конвертор е трибитен, најмалата разлика, т.е. квантниот интервал Q помеѓу излезните напонски нивоа, ќе претставува зголемување или намалување од 625 mV. Имено, бидејќи отпорниците се приклучени помеѓу нултото референтно ниво (заземјувањето, "масата") и напојувањето од 5 V (V_{REF}=+5V) ова ниво е практично и максималното ниво што може да се појави на влезот од кодерот. Така целосниот опсег FSR, ќе биде еднаков со напојувањето на отпорниците од 5 V. Бидејќи со три бита се кодираат $N=2^3=8$ различни нивоа со примена на равенката (8-17) се добива квантна разлика Q од 625 mV (Q=5 V/8=0.625 V). Така првото референтно ниво ќе биде V_1=625 mV, второто ниво ќе биде V_2=1.25 V, итн. на секое следно и поголемо референтно ниво ќе одговара претходната референтна вредност на напонот зголемена за 625 mV, па сè до последното, осмото ниво V_7=4.375 V. Се разбира дека најниското референтно ниво е нултото, потенцијалот на масата, V_0=0 V.

Од сл. 8-20 се забележува дека електричната шема на флеш А/Д конверторот е прилично едноставна. Меѓутоа, за посериозна примена, кога се бара поголема прецизност од осум или повеќе бита, ваквиот А/Д конвертор ќе биде доста скап бидејќи во неговиот состав ќе влезат голем број на компаратори, или поточно (2^n-1), каде што n е бројот на битови во излезниот коден збор. Така на пр. за 8-битен флеш А/Д конвертор ќе бидат потребни $2^8-1=255$ компаратори, додека за професионален 16-битен А/Д конвертор дури $2^{16}-1=65.535$! Покрај ова, кога има потреба за зголемување на резолуцијата, а со тоа и на прецизноста, потрошувачката на моќност кај флеш А/Д конверторот значително расте за секој новододаден бит бидејќи со тоа двојно се зголемува бројот на компаратори.

Од друга страна, овој А/Д конвертор се одликува со многу брза работа бидејќи бинарниот еквивалент на излезот од А/Д конвертор, кој одговара на аналогниот напон доведен на неговиот влез, се добива директно и скоро моментално на излезот. Имено, доцнењето се сведува само на времето кое е потребно сигналот да помине преку операциските засилувачи и логичките кола на кодерот, време кое е навистина многу кратко. Токму заради ваквата брзина на работа овој А/Д конвертор го добил и своето име флеш, блиц или молскавичен А/Д конвертор.

8.8.2. А/Д КОНВЕРТОРИ БАЗИРАНИ НА Д/А КОНВЕРЗИЈА

Главната одлика на овие А/Д конвертори е фактот што влезниот аналоген сигнал се компарира со дигитална вредност што него најмногу му одговара, и со тоа се поставува, или не поставува (брише), одреден бит во излезниот дигитален сигнал. Имено, принципот на работа на ваквите А/Д конвертори се базира на остварување на повратна врска преку интерен Д/А конвертор, според блок шемата прикажана на сл. 8-21. Основната идеја е контролната логика да генерира бинарен збор со должина n-бита: D_{n-1}, D_{n-2}, ..., D_1, D_0, кој преку Д/А конверторот се претворува во интерен аналоген напон V_{ADAC} со декадна вредност d која потоа се споредува со влезниот аналоген напон V_{AIN}.



Сл. 8-21. Блок-шема на А/Д конвертор базиран на Д/А конвертор

На едниот влез во компараторот се доведува токму излезниот V_{ADAC} од вградениот Д/А конвертор што е составен дел на ваквиот тип на А/Д конвертор, додека на другиот влез се носи влезниот аналоген напон V_{AIN}. Разликата која се јавува како сигнал на грешка го детерминира однесувањето на контролната логика која врши корекција на дигиталниот сигнал со цел да се минимизира разликата (грешката) помеѓу двата сигнали. Во оној момент кога излезниот напон од Д/А конверторот ќе го надмине нивото на влезниот аналоген напон за вредност на грешката која ќе биде помала од вредноста на квантот (резолуцијата), компараторот ја менува својата излезна вредност и го ресетира контролното коло да почне од почеток. Во истиот момент се запамтува дигиталната вредност во облик бинарен збор со должина n–бита на излезниот напон од Д/А конверторот. Со ова процесот на конверзија на одбирокот е завршен и повторно се почнува со конвертирање на следниот примерок од влезниот аналоген сигнал.

Постојат неколку различни начини за изведба на А/Д конвертори кои во нивниот блок за компарација како основен составен дел применуваат Д/А конвертор. Во продолжение ќе бидат анализирани два од нив кои припаѓаат на оваа група конвертори:

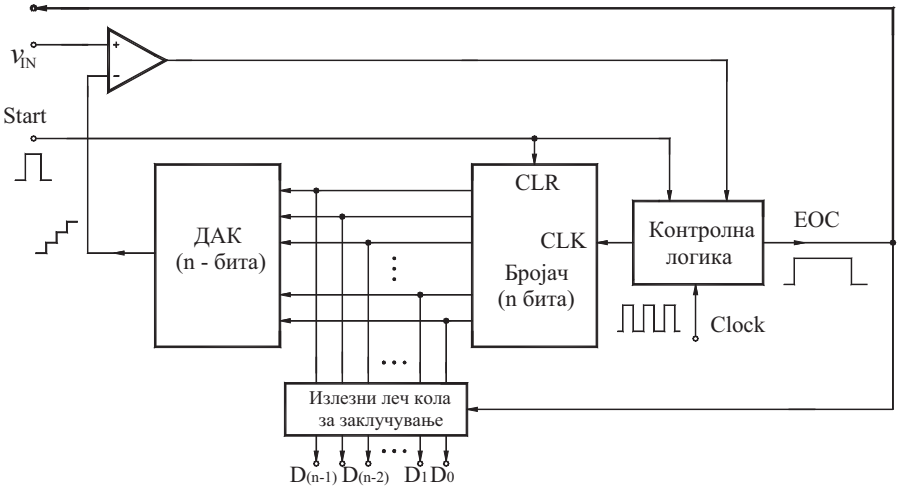
⇒ А/Д конверторот со броење (со бројачка конверзија) или со дигитална рампа (анг. ADC with ramp counter или digital ramp ADC) и

⇒ А/Д конверторот со последователно приближување (анг. successive-approximation ADC).

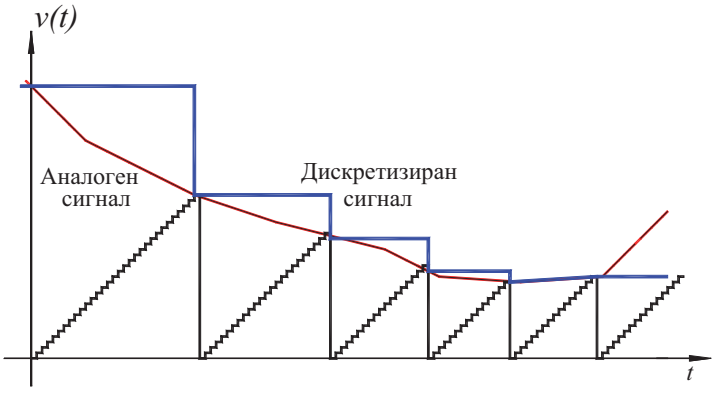
8.5.2.1. АДК СО БРОЈАЧКА РАМПА

Основен елемент со кој се изведува конверзијата кај овој А/Д конвертор е внатрешниот бинарен бројач. Тој почнува да брои од нула кон својата најголемата вредност и тоа сè до оној момент кога неговата состојба преку внатрешниот А/Д конвертор не се изедначи со точната вредност која соодветствува на одбирокот на влезниот аналоген напон Vin, или најблиската дигитална вредност поголема од тоа ниво. Најголемата вредност на бројачот изнесува (2^n-1) од каде се гледа дека таа зависи од резолуцијата на конверторот, т.е. од бројот на битови n.

Заради ваквиот принцип на работа овој А/Д конвертор се вика конвертор со бројачка, скалеста или дигитална рампа (анг. ramp counter ADC или digital ramp ADC). Приципиелната блок-шема на овој конвертор е прикажана на следната слика означена како сл. 8-22, додека временските дијаграми на напоните во карактеристичните точки на колото кои се прикажани на сл. 8-23 дополнително го појаснуваат неговото однесување.



Сл. 8-22. Блок-шема на А/Д конвертор со бројачка рампа



Сл. 8-23. Временски дијаграми на влезниот аналоген напон и споредбениот излезен напон генериран од интерниот Д/А конвертор кај А/Д конверторот со бројачка рампа

На сликата со Vin е прикажан аналогниот влезен напон, додека n–те дигитални излези (битовите) се означени од D_{n-1} до D_0. Работата на бројачот се контролира преку влезната линија START. Во моментот кога на неа ќе се доведе високо ниво, таа го ресетира бројачот и преку контролната логика овозможува тактните импулси (Clock) да дојдат на неговиот влез со што тој почнува да брои. Со секој импулс од такт сигналот бројачот ја зголемува вредноста за 1. Излезите од бројачот го побудуваат Д/А конверторот и бидејќи овие влезни комбинации континуирано се зголемуваат за 1, на излезот од Д/А конверторот се добива напонски сигнал со скалест облик кој постојано расте, или т.н. дигитална рампа.

Главната идеја на овој А/Д конвертор е зголемување на вредноста на бројачот сè додека вредноста што тој ја дава преку интерниот Д/А конвертор не го достигне нивото на одбирокот од влезниот аналоген сигнал. На почетокот од секој циклус на конверзија, излезот од компараторот, кој ги споредува влезниот аналоген сигнал и интерно генерираниот аналоген сигнал, чија амплитуда постојано расте, се наоѓа на ниско ниво. Во моментот кога излезот од Д/А конверторот ќе го достигне нивото на влезниот напон, бинарната комбинација, која се наоѓа на неговиот влез и која е иста со вредноста на излезот од бројачот, се зема како дигитален еквивалент на аналогниот сигнал. Во тој момент излезот од компараторот се менува од ниско на високо, што предизвикува реакција на логиката за контрола, која испраќа сигнал за крај (анг. End), т.е. дека процесот на конверзија на одбирокот е завршен EOC (анг. End Of Conversion) и дека може да се конвертира следен примерок од влезниот аналоген напон. Едновремено сигналот EOC ги активира излезните леч кола кои ја заклучуваат (задржуваат) тековната дигитална вредност на бројачот за таа да може да се прочита и обработи.

Излезниот бафер, во чиј состав се леч-колата, не мора да е составен дел на овој конвертор, но неговото присуство значи голема предност бидејќи задржувањето на дигиталната вредност на одбирокот му овозможува на колото, кое ги добива конвертираните вредности од А/Д конверторот, истите да ги прочита или процесира додека самиот А/Д конвертор веќе работи на следниот одбирок од влезниот аналоген напон.

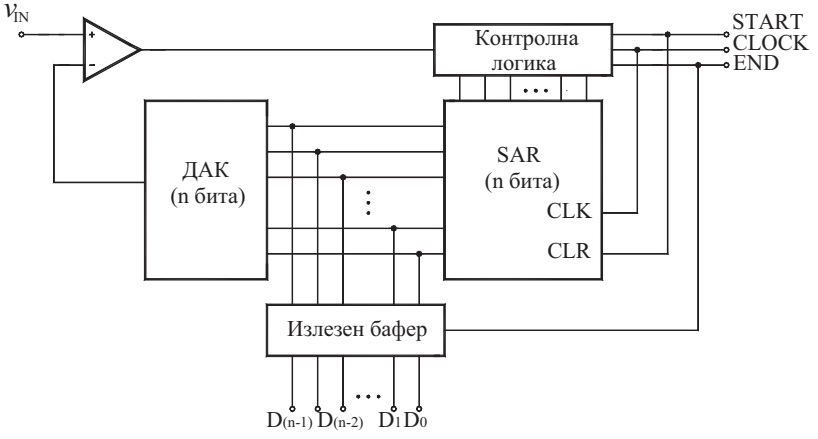
По добивањето на сигналот EOC, влезните кола на Д/А конверторот на влезот од компараторот го пренесуваат следниот одбирок од аналогниот сигнал и едновремено со него испраќаат сигнал START за почеток на негово конвертирање во дигитална вредност. Ова значи дека за секој одбирок на влезниот аналоген напон, што треба да биде конвертиран, се испраќа по еден START импулс со фреквенција, која е еднаква со фреквенцијата на одбирање f_S.

Накратко кажано бројачот брои почнувајќи од 0 до својата максимална вредност ($2^n - 1$) додека не ја достигне точната или најсоодветната дигитална вредност на нивото на одбирокот од аналогниот напон, кој е присутен на влезот од А/Д конверторот. Кога тоа ќе се случи, се генерира сигналот EOC и дигиталниот облик на влезното ниво на аналогниот напон (одбирок) Vin се добива преку бинарниот вектор присутен на излезите од бројачот од D_{n-1} до D_0. Во врска со ова, главен проблем кај овој АДК е неговата брзина. Имено, за одбироци кои имаат мала вредност на напонското ниво времето за конверзија е мало, но за поголеми влезни нивоа времето на конверзија драстично се зголемува. Така овој конвертор е прилично бавен бидејќи за секој примерок што треба да се конвертира и има вредност блиска до најголемата, може да поминат и до (2^n-1) такт интервали (циклуси). Така на пр. за 8 битен АДК за одбирок кој има максимално ниво ќе бидат потребни 2^8-1=256-1=255 такта, но ако се работи за АДК со 12 бита, тогаш за конвертирање на овој одбирок ќе требаат 2^{12}-1=(4x1024)-1=4096-1=4.095 тактни интервали. Покрај ова, ваквиот принцип на работа имплицира примена на такт-сигнал со фреквенција f_{CLK} која треба да биде 2^n пати повисока од фреквенцијата на одбирање f_S.

8.8.2.2. АДК СО ПОСЛЕДОВАТЕЛНО ПРИБЛИЖУВАЊЕ

А/Д конверзијата со последователно приближување (анг. successive approximation ADC) е еден од најчесто користените принципи за конвертирање на аналогните сигнали во дигитални. На следната слика, сл.8-24 е прикажана една принципиелна блок-шема од која се гледа дека изведбата на овој тип А/Д конвертор, слично како и А/Д конверторот со броење, се базира на интерен Д/А конвертор кој генерира сигнал за споредба со нивото на одбирокот од аналогниот влезен сигнал.

Аналогниот влезен напон е означен со Vin, додека дигиталните излези на кои се добива дигиталниот еквивалент на моменталното влезно напонско ниво на примерокот, се n-те битови, означени како D_{n-1}, D_{n-2}, ..., D_1, D_0. Блокот означен со SAR се однесува на т.н. регистер за последователно приближување (анг. Successive Approximation Register). Контролната логика, слично како и кај А/Д конверторот со бројачка рампа, ги испраќа истите контролни сигнали: START за пуштање во работа на регистерот, со што се стартува конверзијата на било кој одбирок, EOC (END) сигналот кој укажува дека конверзијата на конкретниот примерок е завршена, како и такт сигналот CLOCK. Од сликата се гледа дека и во овој случај се користи излезен бафер, така што дигиталниот податок (бинарниот збор) стои на располагање на логичката мрежа што треба да го обработува додека конвертерот го процесира следниот одбирок од влезниот аналоген напон. Сепак, овде постои голема разлика во принципот на работа во однос на А/Д конверторот со скалеста рампа.

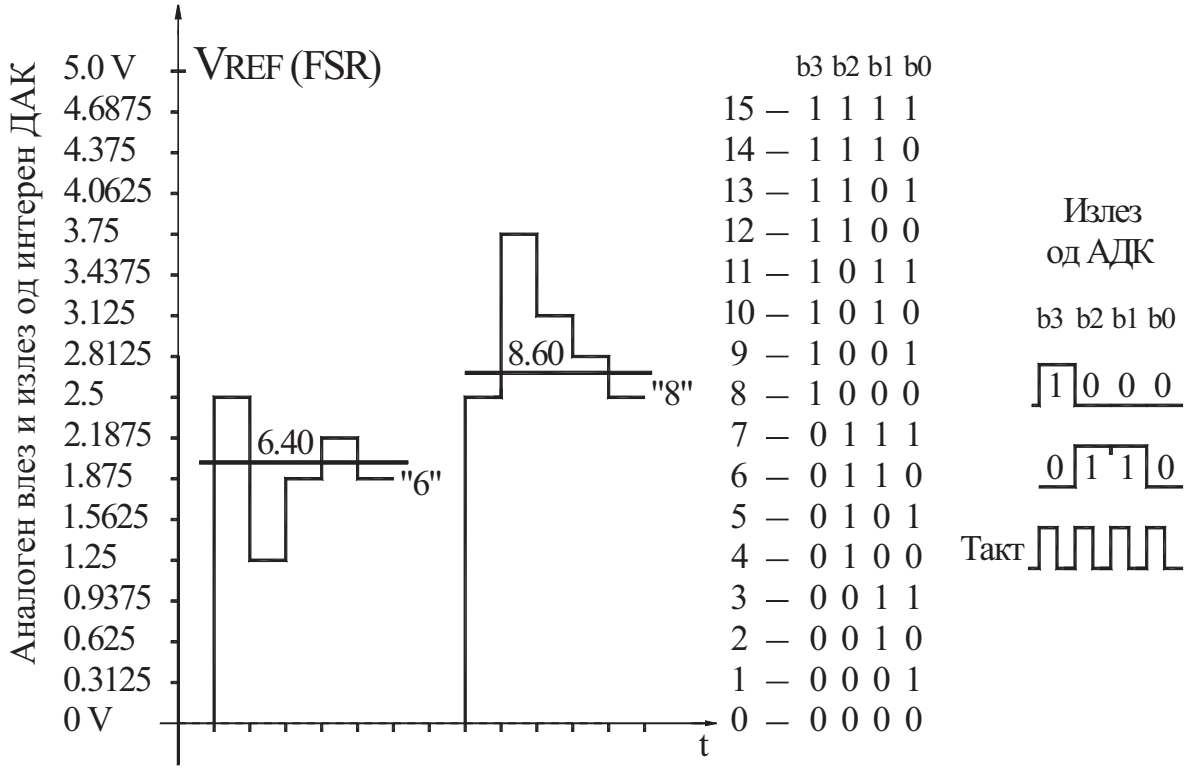


Сл. 8-24. Блок-шема на А/Д конвертор со последователно приближување

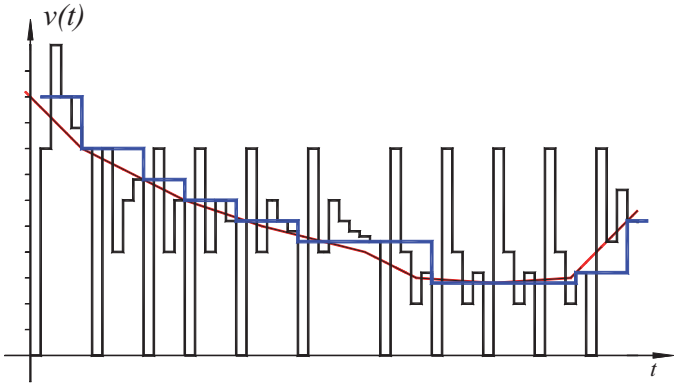
Поконкретно, овој конвертор последователно се приближува кон моменталната вредност на аналогниот влезен одбирок, започнувајќи со поставување на MSB битот D_{n-1} кој има најголема тежина (2^{n-1}) на ниво на логичка 1. Со ова практично на излезот од внатрешниот Д/А конвертор се појавува сигнал кој има напонско ниво еднакво на една половина од максималниот опсег кој е еднаков со референтниот напон $+V_{REF}$. Овој напон компараторот го споредува со влезниот аналоген напон. Врз основа на споредбата компараторот генерира соодветен сигнал до контролниот блок со кој му укажува:

- дали битот треба да остане поставен на 1 (што се случува кога излезниот напон од интерниот Д/А конвертор е помал од нивото на одбирокот, т.е. нивото на влезниот одбирок е поголемо од аналогниот напон генериран од страна на внатрешниот Д/А конвертор) или
- поставениот бит треба да го избрише, т.е. да се постави на 0 (што ќе се случи ако излезниот напон од интерниот Д/А конвертор е со поголемо ниво од она на влезниот напон, т.е. од одбирокот).

Ваквиот принцип на работа се повторува и за следниот бит D_{n-2} кој има два-пати помала тежина од претходната (2^{n-2}) со што аналогната вредност на излезот од Д/А конверторот сега ќе изнесува една четвртина од максималното ниво на А/Д конверторот. Во компараторот повторно се врши споредување помеѓу ова ново генерирано ниво со нивото на одбирокот. По извршената споредба излезното ниво од компараторот повторно преку контролната логика детерминира дали овој бит треба да остане на 1, или треба да се ресетира, итн. компарирањето продолжува сè до последниот LSB бит D_0 кој има најниска тежина (2^0). Целата оваа постапка е сликовито претставена на сл. 8 -25 и сл. 8-26.



Сл. 8-25. Временски дијаграм на процесот на последователно приближување за два одбироци



Сл. 8-26. Временски дијаграми на влезниот аналоген напон и споредбениот излезен напон генериран од интерниот Д/А конвертор кај А/Д конверторот со последователно приближување

Главната предност на овој тип на А/Д конвертор е неговата брзина на работа. Имено, за одредувањето на соодветната дигитална вредност за нивото на аналогниот одбирок, во најлош случај, ќе бидат потребни n циклуси на тактот, каде што n е бројот на битови кој што се користи за дигитална презентација на аналогниот влез, а не 2^n како кај А/Д конверторот со бројачка рампа. Така на пр. ако станува збор за ваков 8 битен А/Д конвертор со сукцесивна апкроксимација, дигиталната вредност на еден одбирок ќе биде одредена за најмногу 8 такта, што споредено со 2^8-1=255-такта кои би се потрошиле во случајот на А/Д конверторот со бројачка рампа, е навистина големо подобрување. Во случај на 12 битна А/Д конверзија, дигиталната вредност на примерокот може да се добие за 12 такт циклуси, а кај претходниот конвертор со скалеста рампа за тоа можеа да се потрошат и до 2^{12}-1=4.095 временски интервали. Од изложеното станува јасно дека може да се примени такт-сигнал со фреквенција f_{CLK} која треба да биде само n пати повисока од фреквенцијата на одбирање f_S, а не 2^n пати како кај конверторот со дигитална рампа.

8.8.3. АДК БАЗИРАНИ НА ИНТЕГРАТОРСКО КОЛО

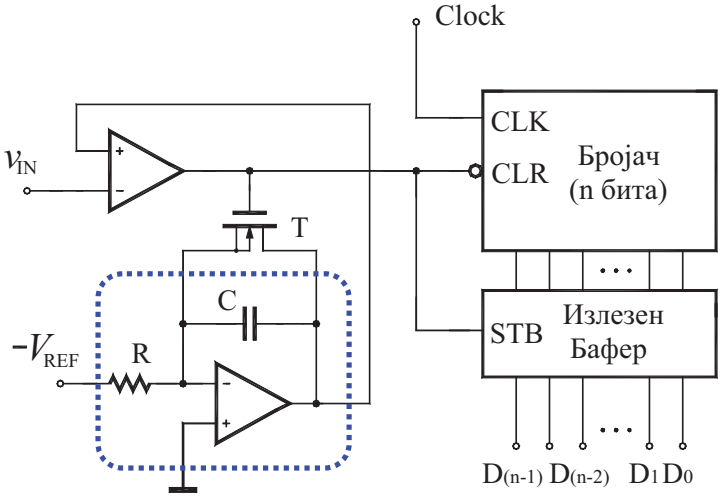
Постојат различни начини за дизајнирање на АДК со помош на интегрирање (анг. Integrating ADC). Во продолжение ќе ги разгледаме А/Д конверторот со единечен наклон. (анг. Single-slope ADC) и А/Д конверторот со двоен наклон (анг. Dual-slope ADC).

Како што ќе видиме во понатамошното излагање, заедничка каратеристика и на двата претходно наведени А/Д конвертори е сличноста во принципот на работа со А/Д конверторот со бројачка рампа. Имено, и кај овој тип на конверзија се применува бројач, со таа разлика што за генерирање на интерниот аналоген напон кој се споредува со вредноста на одбирокот на аналогниот влезен сигнал, се користи интеграторско коло, а не Д/А конвертор. Само во овој случај на излезот од интеграторот се добива напон за споредба кој има растечки линеарен, а не скалест облик.

8.8.3.1. А/Д КОНВЕРТОР СО ЕДИНЕЧЕН НАКЛОН

Принципиелната блок-шема на А/Д конверторот со интегрирање е претставена на сл. 8-27. Заради принципот на работа кој во продолжение ќе го објасниме во поголеми детали, овој А/Д конверторот се вика и конвертор со единечен наклон (анг. single-slope ADC). Имено, ако обрнеме поголемо внимание на сликата, ќе видиме дека оваа шема е многу слична со онаа на А/Д конверторот со бројачка рампа бидејќи и во овој случај се користи бројач. Меѓутоа, сега за генерирање на интерниот аналоген напон кој се

компарира со аналогниот влез, се користи интеграторско коло, поточно генератор на пилест напон (анг. integrator, saw-tooth generator) наместо Д/А конверторот кој на својот излез исто така продуцираше растечки сигнал, но со скалест, а не со линеарен облик. Интеграторското коло во основа се состои од операциски засилувач со капацитивна негативна повратна врска, при што побудниот напон се носи на инвертирачкиот влез преку отпорник. Кога на влезот од интеграторот се доведи константен еднонасочен негативен напон $-V_{REF}$, на излезот од интеграторот ќе се добие линеарен растечки сигнал (напонски облик) бидејќи колото врши инверзија на знакот. MOSFET-от има улога на прекинувачки елемент со кој се контролира почетокот на секој циклус на конверзија.

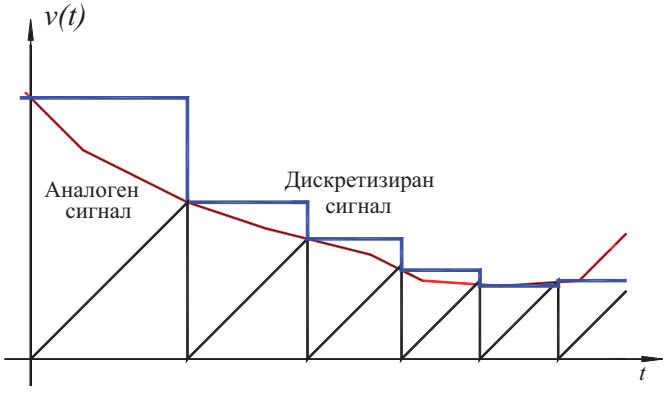


Сл. 8-27. Блок-шема на А/Д конвертор со интеграторско коло и единечен наклон

Пилестиот напон, кој се добива на излезот од интеграторот линеарно расте, тргнувајќи од најниското нулто ниво, кон својата најголема вредност која по апсолутна вредност е еднаква со максималното можно ниво на влезниот сигнал $|-V_{REF}|=+V_{REF}$. Во овој временски период влезниот аналоген напон е поголем од пилестиот и на излезот од компараторот се добива ниско ниво. Излезот од компараторот се доведува на гејтот од MOSFET-от така што неговата ниска вредност го поларизира MOSFET-от во запирното подрачје и со тоа го држи исклучен заради што тој не проведува. Бидејќи кондензаторот е поврзан паралелно на спојот дрејн-сорс, а низ него не тече струја ($I_D=0$), целата струја тече низ кондензаторот C кој се полни. Оваа струја на полнење има константна јачина ($I_C=V_{REF}/R$), па на излезот од операциското засилувач се јавува растечки пилест сигнал.

Едновремено со моментот кога почнува да се генерира пилестиот импулс, бројачот почнува да брои од 0 кон (2^n-1), каде n е бројот на битови кои ги користи А/Д конверторот. Кога пилестиот напон генериран од интеграторот го достигне нивото на одбирокот на аналогниот влезен сигнал Vin, компараторот го менува своето излезно ниво со што ја детектира вредноста што во тој момент ја има бројачот, т.е. неговата последна вредност. Во тој момент се активира контролниот влез на излезниот бафер и во него се сместува оваа дигиталната вредност која кореспондира на аналогното ниво на одбирокот од влезниот напон кој се конвертираше. Едновремено промената на излезното ниво на компараторот го ресетира бројачот и преку MOSFET-от го празни кондензаторот од интеграторот. Имено, во моментот на изедначување на линеарниот напон, кој има тенденција на пораст, и влезниот аналоген напон, компараторот го менува своето излезно ниво од ниско на високо кое го побудува гејтот на MOSFETот заради што тој се вклучува. Бидејќи напонот дрејн-сорс се однесува како куса врска ($U_{DS}=0V$) потенцијалот на излезот од интеграторот паѓа на нула и скоро моментално доаѓа до празнење на кондензаторот C со што веднаш се овозможува започнување на нов (следен) циклус за конверзија.

Временските дијаграми на напоните во карактеристичните точки на А/Д конверторот со интеграторско коло и единечна стрмина кои се прикажани на сл. 8-28 го илустрираат и визуелно го појаснуваат неговиот принцип на работа.



Сл. 8-28. Временски дијаграми на влезниот аналоген напон и споредбениот излезен напон генериран од интеграторот кај А/Д конверторот со бројачка рампа

Имајќи ја предвид презентираната анализа, како и блок-шемата на овој А/Д конвертор, може да се заклучи дека и сега се користи излезно баферско коло, што значи дека додека се чита последната конвертирана вредност, конверторот веќе го работи следниот примерок од влезниот аналоген напонски сигнал.

Иако оваа реализација на конверзија е поедноставна од онаа базирана на бројачка рампа, сепак и овде станува збор за последователно броење кое се зголемува за 1, така што и овој А/Д конвертор го има истиот проблем, а тоа е малата брзина на работа бидејќи за да се конвертира одбирок со големо напонско ниво може да се потребни и до (2^n-1) тактни интервали. Покрај ова, кај ваквиот тип на А/Д конверторот со единечен наклон се јавува уште еден проблем, а тоа е т.н. калибрациско поместување. Имено, фактот што интеграторот не е поврзан на тактниот сигнал на бројачот најчесто со тек на времето доаѓа до одредени отстапувања.

8.8.3.2. А/Д КОНВЕРТОР СО ДВОЕН НАКЛОН

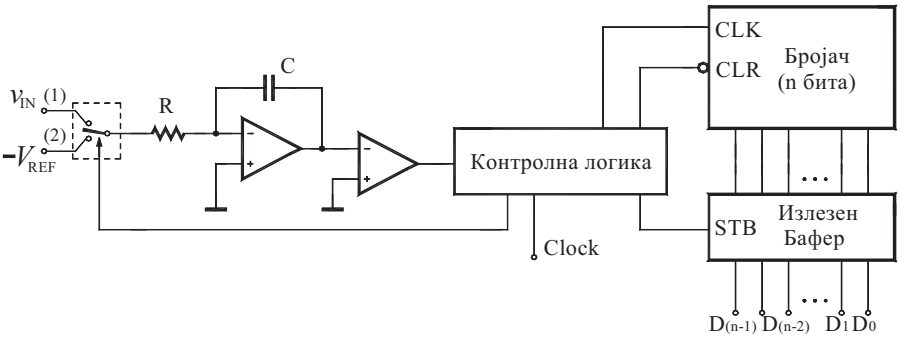
Вториот популарен дизајн, базиран на интеграторско коло е познат како А/Д конвертор со двоен наклон (анг. Dual-Slope ADC). Неговата стандардна, но поедноставена блок шема е прикажана на сл. 8-29. Овој А/Д конвертор го решава системскиот проблем на калибрациското поместување кај А/Д конверторот со единечен наклон кој предизвикуваше појава на грешки затоа што генерирањето на пилестиот сигнал не беше синхронизирано со тактот на бројачот.

Кај А/Д конверторот со двоен наклон, циклусот на конверзија се извршува во два чекори. Во првата фаза контролната логика го става прекинувачот S во положба 1 (S=1) со што го приклучува на аналогниот влезен напон V_{INA}, додека во втората фаза го преспојува во положба 2 (S=2) на константниот референтен напон кој има негативна вредност $-V_{REF}$.

На почетокот на процесот на конверзија кога прекинувачот S се наоѓа во положба (1) позитивното ниво на одбирокот од аналогниот сигнал се носи до влезот на интеграторот (генераторот на пилест напон), заради што на неговиот излез почнува да се генерира пилест напон кој има спротивен, негативен наклон. Овој линеарен опаѓачки сигнал се генерира за одреден фиксен (константен) временски период T_1. За ова време компараторот е пасивен, додека контролната логика управува со работата на бројачот кој почнува да брои од нула точно одреден број на такт-импулси со што практично се

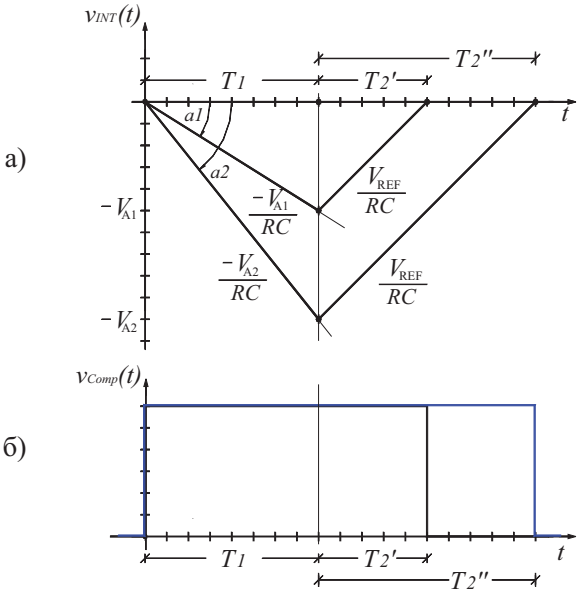
дефинира константна должината на временскиот интервал T_1. Кога овој фиксен временски интервал ќе помине, контролната логика реагира и ја менува положбата на прекинувачот од состојба (1) во (2), со што овозможува на влезот од интеграторот да се приклучи референтниот напон $-V_{REF}$. Едновремено таа испраќа два контролни импулси до бројачот: со едниот го ресетира бројачот (го враќа на броењето на почеток), а со вториот ги пропушта тактните импулси со што истиот го побудува веднаш по ресетирањето повторно да почне да брои. Компараторот и понатаму сèуште останува пасивен.

Бидејќи сега на влезот од интеграторот е приклучен негативен референтен напон $-V_{REF}$, на излезот од интеграторот ќе се добие пилест напон кој линеарно ќе почне да расте кон нултото ниво почнувајќи од негативното ниво до кое што предмалку стигнал кога на неговиот влез беше приклучен аналогниот сигнал. По одреден временски период T_2 кога излезниот сигнал од интеграторот ќе го достигне нултото ниво, компараторот ќе реагира, а состојбата на бројачот ќе ја претставува дигиталната вредност на аналогниот влезен сигнал која ќе се запамти во излезниот бафер.



Сл. 8-29. Блок-шема на А/Д конвертор со интегрирање и двоен наклон

Кажаното подобро ќе го разбереме ако уште еднаш се навратиме на циклусот на конверзија. Во првата положба на прекинувачот (S=1) за време на интервалот T_1 кој беше фиксен се генерираше пилест напон чија стрмнина зависеше од нивото на аналогниот влез V_{INA}: ако нивото беше со поголема вредност и стрмнината ќе беше поголема, и обратно. За разлика од тоа, во втората положба (S=2) побудата на интеграторот беше константа ($-V_{REF}$), па временскиот интервал T_2 за кој излезниот напон од интеграторот ќе ја стигне нулата зависи од тоа до кое претходно ниво тој тргнал.



Сл. 8-30. Бранови облици на излезите од а) интеграторот и б) компараторот

Оваа постапка подобро може да се разбере ако се погледне сл. 8-30, на која се прикажани временските дијаграми (брановите облици) на сигналите во карактеристичните точки од шемата и тоа на излезите од интеграторот и компараторот.

Времетраењето T_1 е константно и неменливо, додека периодот T_2 е пропорционален со вредноста на одбирокот на влезниот напон V_{INA}. Практично моменталното ниво на аналогниот сигнал V_{INA} ја поставува вредноста на наклонот (стрмнината) на линеарниот напон: колку што е тоа поголемо, толку ќе биде поголема и стрмнината, што значи дека поголем ќе биде и аголот, а со тоа и времето T_2 потребно да се достигне нултото ниво, бидејќи стрмнината со која сигналот се стреми кон нула е константна. Равенката (8-21) ја презентира зависноста помеѓу претходно наведените напонски нивоа и временски интервали.

$$\frac{V_{INA}}{T_2} = \frac{V_{REF}}{T_1} \Rightarrow V_{INA} = V_{REF} \cdot \frac{T_2}{T_1} \qquad (8\text{-}21)$$

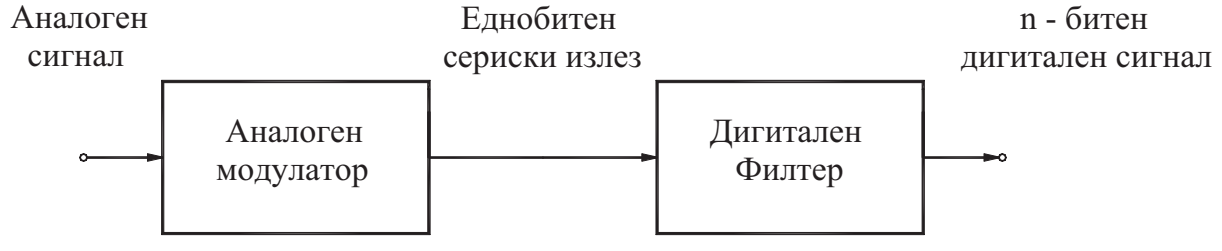
Токму заради ова, периодот T_2 се однесува на вториот чекор од конверзациониот циклус кога бројачот брои сè до моментот кога излезот од интеграторот не го достигне нултото ниво, кога компараторот реагира (го променува нивото на својот излез), со што повторно се активира контролното коло кое сега ги испраќа следниве управувачки сигнали:

- за ресетирање на бројачот,
- за пропуштање на импулсите од такт-сигналот до бројачот и отпочнување со броење,
- за задржување на бинарниот збор – дигитално конвертираниот аналоген одбирок во баферот и
- за враќање на аналогниот прекинувач S во положба (1).

Со ова може да се земе нов одбирок и да се отпочне следниот циклус на конверзија.

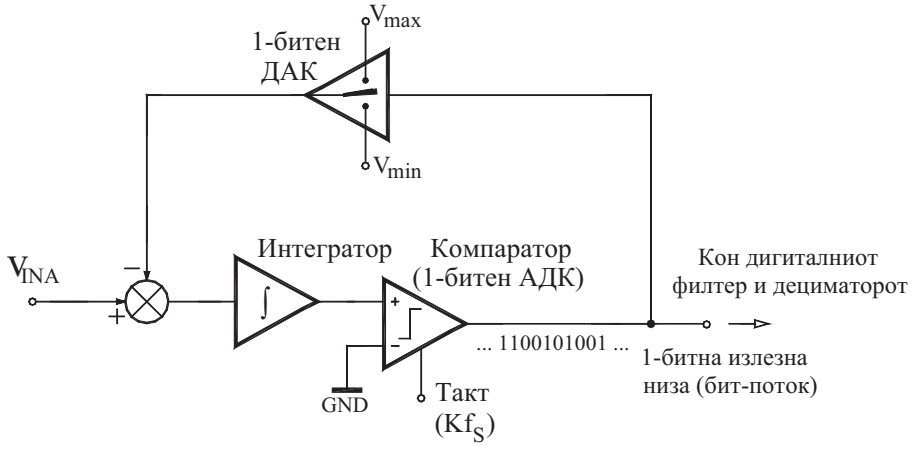
8.5.4. ДЕЛТА-СИГМА А/Д КОНВЕРТОРИ

Од принципиелна гледна точка, однесувањето на делта-сигма (ΔΣ) или сигма-делта (ΣΔ) А/Д конверторите (анг. delta-sigma или sigma-delta ADC) прилично се разликува од другите А/Д конвертори, иако малку потсетува на А/Д конверторот со двоен наклон. Наједноставната блок шема на делта-сигма конверторот која е прикажана на сл. 8-31 во основа содржи два блока. Првиот блок е аналоген модулатор кој го прифаќа аналогниот влезен сигнал и го конвертира во долга сериска низа од многу голем број битови, т.е. генерира едно-битен поток (анг. stream), заради што делта-сигма (ΔΣ) конверторите се познати и како еднобитни конвертори. Вториот блок е дигитален НФ филтер кој содржи и т.н. дециматор. Овој блок врши конверзија на долгите низи битови кои доаѓаат од модулаторот во дигитален излез, т.е. во облик на бинарни зборови со должина n-бита.



Сл. 8-31. Блок дијаграм на сигма – делта А/Д конвертор

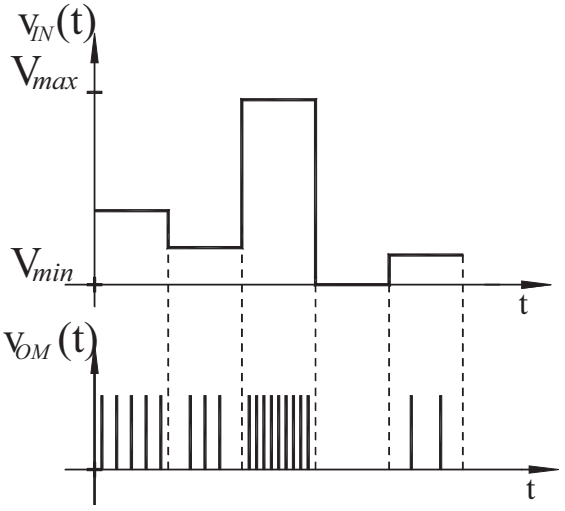
Принципот на работа на аналогниот делта-сигма модулатор ќе го објасниме со помош на неговата блок-шема која е прикажана на сл. 8-32. Компараторот ја препознава разликата (Δ) помеѓу излезот од интеграторот и нултото ниво, додека интеграторот ги интегрира (ги собира, Σ) разликите помеѓу излезните напони од компараторот (едно-битниот Д/А конвертор) и аналогниот влезен сигнал. Ова е овозможено со негативната повратна врска која е остварената преку едно-битниот Д/А конвертор. Неговото излезно ниво може да има само две различни вредности кои се означуваат со V_{Rmax} и V_{Rmin} со кое е ограничено максималното ниво на влезниот сигнал. Така на пр. ако V_{Rmin}=0V и V_{Rmax}=+5V опсегот на влезните напонски нивоа ќе биде помеѓу 0V и 5V, меѓутоа ако V_{Rmin}=–10V и V_{Rmax}=+10V, тогаш влезниот опсег на напони ќе биде помеѓу –10V и +10V.



Сл. 8-32**.** Блок дијаграм на модулаторот кај сигма – делта А/Д конвертор

За делта-сигма конверторот е карактеристична големата фреквенција на тактот со која работи компараторот, т.е. еднобитниот Д/А конвертор така што од аналогниот влезен сигнал се земаат прекумерен број на примероци заради што овие конвертори се познати и како конвертори со прекумерен број на одбирања (анг. over-sampling ADC). Заради ова, потокот од битови кој се добива на излезот од аналогниот модулатор е сериски едно-битен сигнал чија фреквенција f_{CLK} е повеќекратно зголемена во однос на фреквенцијата на одбирање $f_S=2\cdot f_{MAX}$. Таа е цел број пати (К) поголема од Најквистовата фреквенција f_S: $f_{CLK}=K\cdot f_S= K\cdot 2\cdot f_{MAX}$. Главна карактеристика на ваквата долга и густа низа од битови за кус временски период е фактот што нивното средно ниво го претставува средното ниво на влезниот аналоген сигнал за тој период. Поголемиот број на 1-и во единица време индицира поголемо ниво на влезниот анлоген сигнал, и обратно. Бидејќи излезот од А/Д конверторот е дигитален најголемото ниво ќе биде претставено со сите 1-и во излезниот дигитален збор, додека најниското ниво со сите 0-и. Колку што е бројот на 1-и поголем, т.е. колку се тие погусти, толку ќе биде поголемо и нивото на влезниот аналоген сигнал. Обратно, ако влезното ниво опаѓа, и бројот на 1-и ќе се намалува, како што е прикажано на примерот од сл. 8-33. Од тука станува јасно дека колку што е фреквенцијата на тактот поголема, толку ќе биде поголема и прецизноста на делта-сигма модулаторот.

Еден прилично едноставен начин за да се утврди нивото на влезниот сигнал е да се употреби бројач кој ќе ги брои 1-те во одреден фиксен временски интервал. Кога ќе заврши броењето, излезите на бројачот ќе го претставуваат дигиталниот еквивалент на аналогниот сигнал за тој временски интервал. Меѓутоа, кај делта-сигма конверторите кои практично се реализираат за да го претворат еднобитниот поток од битови во дигитален излез, т.е. бинарно кодирани зборови, се применува т.н. техника на десеткување (децимација) (анг. Decimacion) со која се редуцира бројот на одбироци кај временски дискретизираниот влезен сигнал.



Сл. 8-33. Бранови облици на неколку одбироци и излезниот сигнал од модулаторот

Бидејќи со децимирањето практично се намалува фреквенцијата на земање на одбироци треба да се внимава да не биде нарушен Најквистовиот критериум. Заради ова на излезот од А/Д модулаторот најнапред се поврзува дигитален НФ филтер, а потоа дециматор. НФ филтерот се користи како филтер за елиминирање на лажните фреквенции (анг. anti-aliasing filter) со што се редуцира фреквентниот опсег на сигналот, а потоа со дециматорот се врши намалување на фреквенцијата на одбирање (анг. downsampling) на сигналот, со што конечно се добива средното ниво на влезниот сигнал во даденият временски интервал.

Делта-сигма А/Д конверторите се применуваат за прецизни мерења и заради тоа се користат кај инструментите, во индустријата и сл.

ПРАШАЊА И ЗАДАЧИ ЗА ПОВТОРУВАЊЕ

8-1. Аналогните сигнали претставуваат ...

8-2. Дигиталните сигнали претставуваат ...

8-3. Нацртај наједноставна блок шема на А/Д конвертор и коментирај ја неговата улога.

8-4. Под поимот А/Д конверзија се подразбира ...

8-5. Под поимот Д/А конверзија се подразбира ...

8-6. Нацртај наједноставна блок шема на Д/А конвертор и коментирај ја неговата улога.

8-7. Нацртај наједноставна блок шема за поврзување на компјутер со А/Д и Д/А конвертор.

8-8. Напиши ја равенката за излезниот напон кај идеален Д/А конвертор а) во општ случај со резолуција n-бита, б) со резолуција n=3 бита, в) со резолуција n=4 бита.

8-9. Нацртај ја преносната карактеристика на идеален Д/А конвертор со резолуција а) n=3 бита, б) n=4 бита.

8-10. (*) Проектирај а) 3-битен, б) 4-битен Д/А конвертор со чекор од 1 V претпоставувајќи дека високото логичко ниво (нивото на 1) е (1) 5 V, (2) 10 V. Кој ќе биде излезниот напонски опсег за Д/А конверторот кој ќе го реализираш?

8-11. (*) Проектирај а) 3-битен, б) 4-битен Д/А конвертор со чекор од 0,2 V претпоставувајќи дека високото логичко ниво (нивото на 1) е а) 5 V, б) 10 V. Кој ќе биде излезниот напонски опсег за Д/А конверторот кој ќе го реализираш?

8-12. Нацртај електрична шема на Д/А конвертор со тежинска $R/2^nR$ отпорничка мрежа и резолуција а) n=3 бита, б) n=4 бита, в) (*) n=5 бита.

8-13. Нацртај електрична шема на Д/А конвертор со скалеста R/2R отпорничка мрежа и резолуција а) n=3 бита, б) n=4 бита, в) (*) n=5 бита.

8-14. Спореди ги добрите и лошите страни (предностите и слабостите) на Д/А конвертор со тежинска $R/2^nR$ и скалеста R/2R отпорничка мрежа.

8-15. Што претставува дискретизацијата по време на аналогниот сигнал?

8-16. Што претставува квантизацијата (дискретизацијата по ниво) на аналогниот сигнал?

8-17. Нацртај еден дел од временски дијаграм на аналоген напонски сигнал во опсег од 0 до 5 V. Изврши негова дискретизација по време така што од него ќе земеш 5 одбироци кои се дискретизирани по ниво (квантизирани) со а) n=3 бита, б) n=4 бита. Нацртај го и неговиот конвертиран дигитален сигнал.

8-18. Циклус на конверзија е …

8-19. Фреквенција на одбирање е …

8-20. Зошто е важен Најквистовиот критериум?

8-21. Да претпоставиме дека се дадени два аналогни сигнали: едниот е аудио сигнал кој зафаќа фреквентен опсег од 0 до 20 KHz, додека другиот е видео сигнал во опсег од 0 до 5 MHz. Која е најниската фреквенција на одбирање за секој од сигналите?

8-22. Нацртај преносна карактеристика на идеален А/Д конвертор со резолуција а) n=3 бита, б) n=4 бита.

8-23. Што претставува опсегот на целосна (полна) скала FS(R)?

8-24. Што е резолуција на квантизерот? Од што зависи? Во кои единици се изразува? Според која равенка се пресметува?

8-25. Што е квант (чекор на квантизација) (Q)? Од што зависи? Според која равенка се пресметува?

8-26. Зошто е важно чекорот на квантизација (квантот) да биде мал? Коментирај!

8-27. Што претставува грешката на квантизација? Во кои граници се движи?

8-28. Да претпоставиме дека на влезот од идеален А/Д конвертор доаѓа напонски аналоген сигнал во опсегот од 0 до 10 V. Пресметај го бројот на квантизациони нивоа и резолуциониот напон (големината на квантот) и грешката на квантизација ако А/Д конверторот врши конверзија со а) n=3 бита, б) n=4 бита, в) n=12 бита, г) n=16 бита.

8-29. Наброи ги главните групи на постапки според кои се градат А/Д конвертори.

8-30. Имајќи ја во вид принципиелната шема на паралелниот (флеш) А/Д конвертор објасни го принципот на неговата работа.

8-31. Колкав број на отпорници е потребен за да се реализира а) 8-битен, б) 12-битен, в) 16-битен флеш А/Д конвертор? Колку компаратори ќе бидат потребни?

8-32. Што е заедничко за А/Д конверторите со бројачка рампа, со последователно приближување и со следење?

8-33. Имајќи ја во вид принципиелната шема на А/Д конвертор со бројачка рампа објасни го принципот на неговата работа.

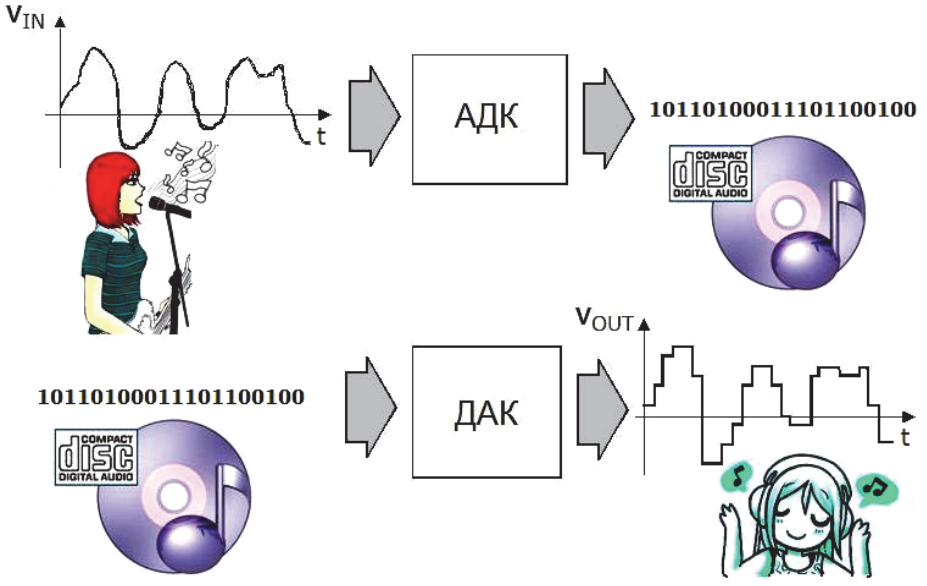
8-34. Имајќи ја во вид принципиелната шема на А/Д конвертор со последователно приближување објасни го принципот на неговата работа.

8-35. Кои А/Д постапки се базираат на примената на интеграторско коло?

8-36. Имајќи ја во вид принципиелната шема на А/Д конвертор со единечен наклон објасни го принципот на неговата работа.

8-37. Имајќи ја во вид принципиелната шема на А/Д конвертор со двоен наклон објасни го принципот на неговата работа.

8-38. Спореди ги и коментирај ги предностите и слабостите на различните постапки за А/Д конверзија и на А/Д конверторите што нив ги реализираат.



Слика за прашање 8-39

8-39. (*) Во некое музичко студио дигитално се снима музика за CD со фреквенција на одбирање 44.1 kHz. а) Колкав мемориски простор изразен во битови (b) и бајти (B) е потребен за да се сними песна која трае 1) 3 минути, 2) 4 минути? б) Ако капацитетот на едно CD е 700 MB колку минути музика може да се сними на него? в) Ако амплитудата на аналогниот аудио сигнал се менува во границите од 0 до 5V, колку изнесува бројот на квантизациони нивоа, чекорот на квантизација (резолуциониот напон, квантот) и грешката на квантизација?

ЛИТЕРАТУРА

1. Balch, M. (2003). Complete Digital Design, *McGraw-Hill Companies, Inc.*
2. Floyd, T. (2006). Digital Fundamentals, Ninth Edition, *Pearson Prentice Hall*.
3. Holdsworth, B., Woods, C. (2003). Digital Logic Design, Fourth Edition, *Elsevier Private Ltd*.
4. Maini, A. (2007). Digital Electronics, *John Wiley & Sons Ltd*.
5. M. Morris, Mano, Charles Kime (2008). Logic and Computer Design Fundamentals, Fourth Edition, *Prentice Hall*.
6. Saha, A., Manna, N. (2007). Digital Principles and Logic Design, *Laxmi Publications Ltd.*
7. Сервини, Ј. (2008). Импулсна и дигитална електроника I (прв дел), трето издание, *Просветно дело*.
8. Сервини, Ј., Дужевиќ, М. (2008). Импулсна и дигитална електроника II (втор дел), четврто издание, *Просветно дело*.