ზაზა დავითაძე

# კომპიუტერის ელექტრონული და ლოგიკური საფუძვლები

ბათუმი
2021

## ვუძღვნი შვილიშვილებს. დარწმუნებული ვარ, წიგნი აქტუალური იქნება მათი შვილიშვილებისთვისაც

წიგნში განხილულია ციფრული ელექტრონიკის და ლოგიკის ძირითადი საკითხები: საბაზო და კომბინაციური ლოგიკური ელემენტები, მათი მოქმედების პრინციპები, მარტივი და რთული ლოგიკური სქემების აგების პრინციპები და მათი ჭეშმარიტების ცხრილებთან დაკავშირებული საკითხები, ევროპულ და ამერიკულ სტანდარტებში მოქმედი სქემატური აღნიშვნები, ლოგიკური ფუნქციების მინიმიზაციის სხვადასხვა მეთოდები, მათ შორის კარნოს ბარათების პოპულარული მეთოდი. ასევე განხილულია ლოგიკური სქემების სინთეზი მინიმიზაციის მეთოდების გამოყენებით, კომპიუტერის ძირითადი ციფრული კომპონენტები და კვანძები, მათი მოქმედების პრინციპები საბაზო ლოგიკურ ელემენტებზე დაყრდნობით. განხილული თემების საფუძვლიანად ათვისებისათვის და შესწავლილი მასალების შემოწმებისათვის წიგნში მოცემულია შესაბამისი ილუსტრაციები, მაგალითები და სავარჯიშოები.

წიგნი განკუთვნილია სტუდენტებისათვის და საკითხით დაინტერესებული მკითხველისათვის.

**რედაქტორი:** კომპიუტერულ მეცნიერებათა აკადემიური დოქტორი
**მიხეილ დონაძე**

**რეცენზენტი:** ფიზიკა-მათემატიკის მეცნიერებათა კანდიდატი
**ომარ ნაკაშიძე**

**გარეკანის გრაფიკა: ანა სურმანიძე**


ISBN 978-9941-8-3624-4

# ავტორის შესახებ

ზაზა ალექსანდრეს ძე დავითაძე, ფიზიკა-მათემატიკის მეცნიერებათა კანდიდატი, ბათუმის შოთა რუსთაველის სახელმწიფო უნივერსიტეტის

ასოცირებული პროფესორი. იგი 35 წელზე მეტია ეწევა პედაგოგიურ საქმიანობას ბათუმის შოთა რუსთაველის სახელმწიფო უნივერსიტეტში კომპიუტერული მეცნიერებების მიმართულებით, წლებია მიჰყავს სალექციო კურსები: კომპიუტერის არქიტექტურა და ორგანიზება, პერსონალური კომპიუტერის სისტემური და პროგრამული უზრუნველყოფა, რობოტოტექნიკის საფუძვლები, საგანთა ინტერნეტი (IOT), ელექტრონული კომერცია.

ზაზა დავითაძემ დაამთავრა თბილისის სახელმწიფო უნივერსიტეტის ფიზიკის ფაკულტეტი რადიოელექტრონიკის სპეციალობით და მ. ლომონოსოვის სახელობის მოსკოვის სახელმწიფო უნივერსიტეტის ასპირანტურა ქვანტური რადიფიზიკის მიმართულებით.

ინტერესის სფეროებია: კომპიუტერული სისტემები და არქიტექტურა, მიკროკონტოლერები, სენსორები და რობოტოტექნიკა.

დამატებითი ინფორმაცია ზაზა დავითაძის შესახებ შეიძლება იხილოთ ვებ-გვერდზე: bsu.edu.ge/sub-59/cv/320/index.html .

## სარჩევი

## ავტორის წინასიტყვაობა

წიგნში განხილული თემატიკა აქტუალური არის და იქნება კიდევ საკმაოდ დიდი დროის განმავლობაში, რადგან იგი ციფრული ელექტრონიკისა და კომპიუტერული სისტემების საფუძველს წარმოადგენს. როგორც არ უნდა შეიცვალოს კომპიუტერული სისტემების არქიტექტურა, თუნდაც მისი ელექტრონული კომპონენტები ჩანაცვლდეს ოპტიკურით, დღეისათვის კომპიუტერების მოქმედების ორობით პრინციპებს და ლოგიკას ალტერნატივა თითქმის არ გააჩნია. გამოთვლების ელექტრონული მეთოდების შემუშავების ისტორიაში იყო მცდელობა მოქმედების სხვადასხვა პრინციპებზე დამყარებული კომპიუტერების შექმნისა, მაგალითად, ანალოგური გამომთვლელი მანქანები, კვანტური და სხვა, მაგრამ საბოლოოდ ორობითმა სისტემებმა და მასზე დამყარებულმა ლოგიკამ მყარად დაიმკვიდრა ადგილი. ამასთან ერთად თვალნათლივ ვხედავთ, თუ რა ტემპებით ვითარდება კომპიუტერულ სისტემებთან დაკავშირებული მიმართულებები. თუ გასული საუკუნის 40-იან წლებში ელექტრონულ მილაკებზე აგებული გამომთვლელი მანქანა იჭერდა დიდი დარბაზის ტოლ ფართობს, დღეს მილიარდობით ტრანზისტორის მქონე მიკროპროცესორი დაახლოებით ერთ კვადრატულ სანტიმეტრზე, ერთი ჩიპის ფარგლებშია განთავსებული. არ არსებობს დარგი, რომელიც ასეთი სისწრაფით განიცდის განვითარებს. უამრავი თეორიული და პრაქტიკული კვლევა მიეძღვნა და ეძღვნება კომპიუტერულ სისტემების ფუნდამენტურ მიმართულებებს: კომპიუტერის არითმეტიკულ და ლოგიკურ საფუძვლებს, ანალოგურ და ციფრულ ელექტრონიკას, სქემოტექნიკას, დისკრეტულ მათემატიკას და სხვა. კომპიუტერულ სისტემებთან დაკავშირებული სამეცნიერო - ტექნიკური დარგის შესწავლა შეუძლებელია ისეთ დისციპლინების გარეშე, როგორებიცაა: თვლის სისტემები, ბულის ალგებრა, ლოგიკა, ლოგიკური ელემენტები და ელექტრონული სქემები და მათი ოპტიმიზაციის მეთოდები და სხვა.

წიგნის შექმნის მთავარ მიზანს წარმოადგენს სტუდენტებს და ზოგადად, კომპიუტერული მეცნიერებით დაინტერესებულ ადამიანებს მარტივი და გასაგები ენით ავუხსნათ ტექნიკის ამ პოპულარული მიღწევის მუშობის ზოგადი პრინციპები.

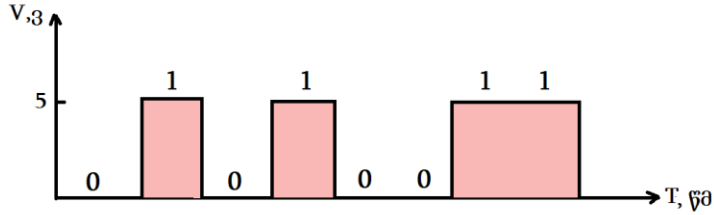
**ზაზა დავითაძე**

# თავი 1. ლოგიკური ელემენტები

## 1.1. საბაზო ლოგიკური ელემენტები

ისევე როგორც ნებისმიერი ნივთიერების თუ ორგანიზმის ფიზიკურ საფუძველს წარმოადგენს მოლეკულები, ხოლო მოლეკულები შედგება ატომებისაგან, ასევე ნებისმიერი კომპიუტერის ელექტრონულ საფუძველს წარმოადგენს სამი სახის ლოგიკური ელემენტი დიზუნქტორი, კონიუნქტორი და ინვერტორი. ამ სამი სახის ლოგიკური ელემენტებისა და მათი სხვადსხვა კომბინაციისაგან მიიღება კომპიუტერის ნებისმიერი ფუნქციონალური ბლოკი: პროცესორი, მეხსიერება, რეგისტრები და სხვა. ჩამოთვლილი თითოეული ლოგიკური ელემენტი ასრულებს მარტივ, მაგრამ მნიშვნელოვან ლოგიკურ ოპერაციას.

ლოგიკური ელემენტების მუშაობა ეყრდნობა ორობით თვლის სიტემას ანუ ბულის ალგებრას, მასთან დაკავშირებულ თავისებურებებსა და ოპერაციებს. როგორც კარგად ცნობილია ორობითი თვლის სიტემა ოპერირებს ორი ციფრით ორობითი 0-ით და ორობითი 1-ით, რაც კომპიუტერული ელექტრონიკისათვის (ციფრული ელექტრონიკა) გულისხმობს დროში ცვალებად ელექტრული იმპულსის არსებობას ან არ არსებობას. ნახ.1.1-ზე მოცემულია დროითი დიაგრამა V=5ვ სიდიდის ელექტრონული იმპულსისათვის, რომელიც შეესაბამება ორობით რიცხვს 11001010. რიცხვის წაკითხვას ვახდენთ მარჯვნიდან მარცხნივ იმ მიზეზის გამო, რომ T დროის მიხედვით მარჯვენა პირველი იმპულსი (ორობითი 1) უფრო ადრე გამოჩნდა ვიდრე დანარჩენები.



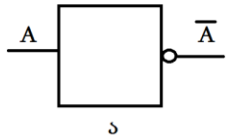

ნახ.1.1 11001010 ორობითი რიცხვის დროითი დიაგრამა 5 ვოლტიანი ელექტრული იმპულსისათვის

ეორე მხრივ ბინარული (ორობითი) ლოგიკისა და ელექტრონული იმპულსების კავშირს კარგად აღწერს ჩვეულებრივი ელექტრული ჩამრთველის მოქმედების პრინციპი, ანუ ოთახში ჩამრთველის 1 პოზიციაში გადაყვანა ჩართავს ელექტრულ დენს წრედში და ნათურა აინთება, ხოლო 0 პოზიციაში გადაყვანა დენს წრედიდან გამორთავს და ნათურა ჩაქრება. ასეთი ჩამრთველების როლს ციფრულ ელექტრონიკაში ასრულებს დიოდები და ტრანზისტორები, რომლებიც მიკროსქემების ძირითად შემადგენელ ნაწილებს წარმოადგენენ.

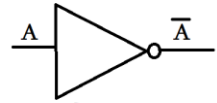
ციფრული ინფორმაციის ერთეული არის 1 ბიტი. **ბიტი** არის ელემენტარული ინფორმაციის ზომა და მას შეუძლია მიიღოს მხოლოდ ორი მნიშვნელობა ლოგიკური 1 ან ლოგიკური 0. რას ნიშნავს „ლოგიკური“? მიღებულია, რომ თუ ნახ.1.-ზე მოცემული იმპულსებისათვის 0 რეალურად იცვლება 0-დან 1 ვოლტამდე, მაშინ იგი ითვლება ლოგიკურ 0-ად, ხოლო თუ 1 რეალურად იცვლება 3 ვოლტიდან 5 ვოლტამდე ეს იმპულსი ითვლება ლოგიკურ 1-ად. ესე იგი თუ რომელიმე ელემენტის გამოსასვლელზე სიგნალი დავუშვათ 0,7 ვ-ია, მაშინ იგი შეესაბამება ლოგიკურ 0-ს, ხოლო თუ სიგნალი 3,8 ვ-ია, მაშინ იგი ლოგიკური 1-ია.

არსებობს სამი საბაზო (ძირითადი) ლოგიკური ელემენტი, რომლებიც ასრულებენ სამ ლოგიკურ ფუნქციას, ესენია: **„არა“** (ინვერსია), **„ან“** (დიზუნქცია) და **„და“** (კონიუნქცია). სხვაგვარად, ელემენტი „არა“ ასრულებს ლოგიკურ უარყოფას, ელემენტი „ან“ - ლოგიკურ შეკრებას და ელემენტი „და“ -ლოგიკურ გამრავლებას. ყველა დანარჩენი ლოგიკური ოპერაციები სრულდება ამ სამ საბაზო ფუნქციაზე დაყრდნობით.

**ლოგიკური ელემენტი „არა“.** ეს ელემენტი ლოგიკურ 0-ს გადაიყვანს ლოგიკურ 1-ში და პირიქით, ლოგიკურ 1-ს გადაიყვანს ლოგიკურ 0-ში. ანუ სხვაგვარად რომ ვთქვათ თუ ასეთი ელემენტის შესასვლელზე მოვდებთ ლოგიკური 0-ის შესაბამის დაბალ ძაბვას, მაშინ გამოსასვლელზე მივიღებთ ლოგიკური 1-ის შესაბამის მაღალ ძაბვა და პირიქით, თუ მის შესასვლელზე მოვდებთ მაღალ ძაბვას გამოსასვლელზე მივიღებთ დაბალს. სხვაგვარად რომ ვთქვათ ეს ელემენტი ასრულებს სიგნალის ინვერსიას და მას უწოდებენ **ინვერტორს.** ყოველ ინვერტორს გააჩნია ერთი შესავალი და ერთი გამოსავალი კონტაქტი. ინვერტორის სქემატური აღნიშვნა და ჭეშმარიტების ცხრილი მოცემულია ნახ.1.2-ზე.




ა



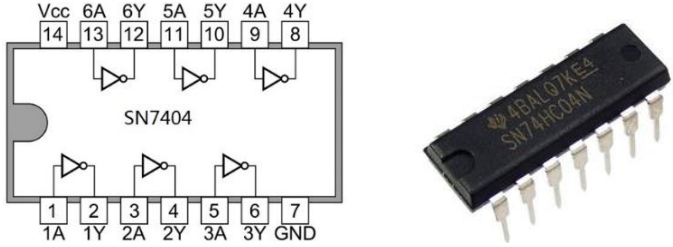

ბ

გ

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

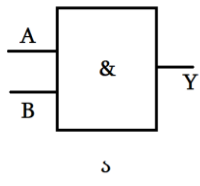
**ნახ.1.2. ინვერტორი. ა. ევროპული სტანდარტი, ბ. ამერიკული სტანდარტი, გ. ჭეშმარიტების ცხრილი**

პრაქტიკული თვალსაზრისით ლოგიკური ინვერტორები მზადდება მიკროსქემის სახით. მაგალითისათვის ნახ1..3-ზე მოცემულია ერთ-ერთი ყველაზე გავრცელებული მიკროსქემის 7404-ის კორპუსში 6 ინვერტორის განლაგების სქემა და მიკროსქემის ფოტო.

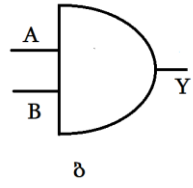



**ნახ.1.3. მიკროსქემა SN7404-ში ლოგიკური ინვერტორების განლაგება და მიკროსქემის ფოტო გამოსახულება**

**ლოგიკური ელემენტი „და“.** ლოგიკური ელემენტი „და“ (კონიუნქტორი) ლოგიკურად გაამრავლებს მის შესავლებზე მოდებულ სიგნალებს და გამოსასვლელზე გამოიტანს შედეგს. რადგან ორობით ალგებრაში გვაქვს მხოლოდ 0-ები და 1-ები, ამიტომ ოპერაციები ტარდება მხოლოდ ამ ორ ორობით რიცხვზე. კონიუნქტორის მუშაობის პრინციპი ერთი წინადადებით შემდეგნაირად შეიძლება ჩამოვაყალიბოთ: **„კონიუნქტორის გამოსასვლელზე ლოგიკური „1“ გვაქვს მაშინ და მხოლოდ მაშინ, როდესაც მის ორივე შესასვლელზე ერთდროულად გვაქვს „1“.** კონიუნქტორის სქემატური გამოსახულებები სხვადასხვა სტანდარტებში და მისი ჭეშმარიტების ცხრილი მოცემულია ნახ.1.4-ზე.




ა




ბ

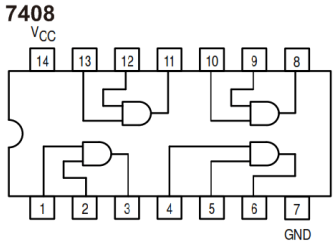
| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

გ

**ნახ.1.4. კონიუნქტორი. ა. ევროპული სტანდარტი, ბ. ამერიკული სტანდარტი,გ. ჭეშმარიტების ცხრილი**

ნახ.1.4-ზე მოცემული ჭეშმარიტების ცხრილი გვიჩვენებს კონიუნქტორის A და B შესასვლელზე ყველა შესაძლო ლოგიკური რიცხვების არსებობის დროს რა რიცხვი იქნება გამოსასვლელზე Y=A*B ანუ ლოგიკის სიმბოლოთი ჩაწერით Y=A∧B. მიუხედავად იმისა, რომ ლოგიკური გამრავლების პროცესი იგივეა რაც ათობითი რიცხვების გამრავლების პრინციპი, არ უნდა დაგვავიწყდეს, რომ აქ გვაქვს ლოგიკური გამრავლება.

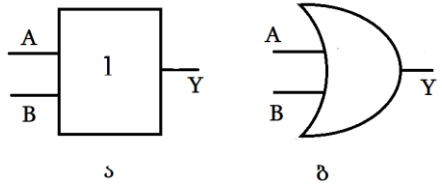
პრაქტიკული თვალსაზრისით ლოგიკური კონიუნქტორი მზადდება მიკროსქემის სახით. მაგალითისათვის ნახ.1.5-ზე მოცემულია ერთ-ერთი ყველაზე გავრცელებული მიკროსქემის 7408-ის კორპუსში 4 კონვენტორის განლაგების სქემა და მიკროსქემის ფოტო.





**ნახ.1.5. მიკროსქემა SN7408-ში ლოგიკური კონვენტორების განლაგება და მიკროსქემის ფოტო გამოსახულება**

**ლოგიკური ელემენტი „ან“.** ლოგიკური ელემენტი „ან“ (დიზუნქტორი) ლოგიკურად შეკრებს მის შესავლებზე მოდებულ ორობით სიგნალებს და გამოსასვლელზე გამოიტანს შედეგს. რადგან ორობით ალგებრაში გვაქვს მხოლოდ 0-ები და 1-ები, ამიტომ ოპერაციები ტარდება მხოლოდ ამ ორობით რიცხვებზე. დიზუნქტორის მუშაობის პრინციპი ერთი წინადადებით შემდეგნაირად შეიძლება ჩამოვაყალიბოთ: **„დიზუნქტორის გამოსასვლელზე ლოგიკური „1“ გვაქვს მაშინ, როდესაც მის ერთ-ერთ შესასვლელზე მაინც გვაქვს ლოგიკური „1“.** დიზუნქტორის სქემატური გამოსახულებები სხვადასხვა სტანდარტებში და მისი ჭეშმარიტების ცხრილი მოცემულია ნახ.1.6-ზე.




ა ბ

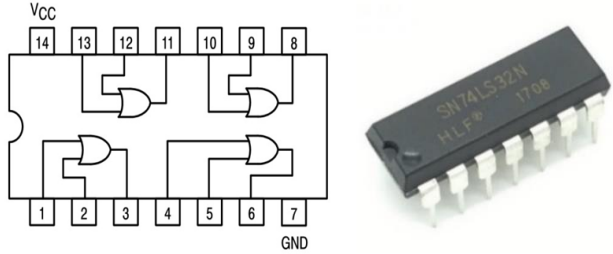
| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

გ

**ნახ.1.6. დიზუნქტორი. ა. ევროპული სტანდარტი, ბ. ამერიკული სტანდარტი, გ. ჭეშმარიტების ცხრილი**

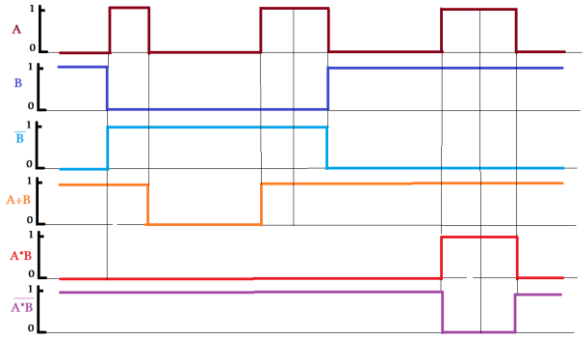
ნახ.1.6-ზე მოცემული ჭეშმარიტების ცხრილი გვიჩვენებს დიზუნქტორის A და B შესასვლელზე ყველა შესაძლო ლოგიკური რიცხვების არსებობის დროს რა რიცხვი იქნება გამოსასვლელზე Y=A+B, ანუ ლოგიკის სიმბოლოთი ჩაწერით Y=AVB. შეკრების პროცესი ძირითადში იგივეა რაც ათობითი რიცხვების შეკრების პრინციპი, გარდა შემთხვევისა 1+1=1. არ უნდა დაგვავიწყდეს, რომ აქაც საქმე გვაქვს ორი რიცხვის ლოგიკურ შეკრებასთან.

პრაქტიკული თვალსაზრისით ლოგიკური დიზუნქტორი მზადდება მიკროსქემის სახით. მაგალითისათვის ნახ.1.7-ზე მოცემულია ერთ-ერთი ყველაზე გავრცელებული მიკროსქემის 7432-ის კორპუსში 4 დიზუნქტორის განლაგების სქემა და მიკროსქემის ფოტო.




**ნახ.1.7. მიკროსქემა SN7432-ში ლოგიკური კონვენტორების განლაგება და მიკროსქემის ფოტო გამოსახულება**

ციფრულ ელექტრონიკაში ინფორმაციის ანალიზის ერთ-ერთი საუკეთესო საშუალებაა ლოგიკურ სქემებში გამავალი ორობითი იმპულსების დროითი დიაგრამა, რომლის მაგალითი ნახ.1.8-ზე მოცემული.




**ნახ.1.8. ლოგიკურ სქემებში გამავალი ორობითი იმპულსების დროითი დიაგრამა**

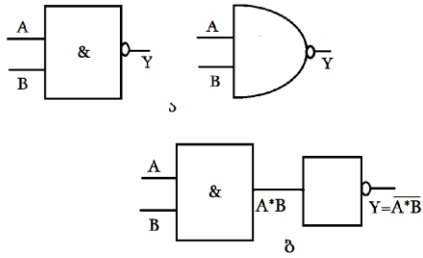
გრაფიკული გამოსახვის ასეთი ფორმა საშუალებას გვაძლევს თვალნათლივ დავინახოთ ლოგიკური სქემების მოქმედების პრინციპი. ნახაზზე ნაჩვენებია ორი A და B ორობითი ცვლადი იმპულსები და მათი სხვადასხვა შესაძლო კომბინაციები (ინვერსია - $\bar{B}$, დიზუნქცია - A+B, კონიუნქცია - A*B და კონიუნქციის ინვერსია - $\overline{A * B}$) სხვადასხვა საბაზო ლოგიკურ ელემენტებში.

## 1.2. კომბინირებული ლოგიკური ელემენტები.

საბაზო ლოგიკური ელემენტები გამოიყენება სხვადასხვა ფუნქციონალური დანიშნულების ლოგიკური ელემენტების მოდელირებისა და კონსტრუირებისათვის. ეს ელემენტები საბაზო ლოგიკურ ელემენტებთან ერთად რეალიზაციას უკეთებენ ბულის ალგებრის კონკრეტულ ფუნქციებს და პრაქტიკული თვალსაზრისითაც ხშირად გამოყენებადი არიან. მიუხედავად იმისა, რომ საბაზო და კომბინირებულ ლოგიკურ ელემენტები უკვე მზა ფუნქიონირებად მოდულად შეიძლება არსებობდეს კომპიუტერის ან ციფრული ელექტრონიკის რომელიმე ბლოკის (მეხსიერება, მიკროპროცესორი, მიკროკონტროლერი და სხვა) დიდ მიკროსქემებში, მიუხედავად ამისა მათ განცალკევებულად აწარმოებენ ცალკეული მიკროსქემების სახით სხვადასხვა მწარმოებლები მსოფლიოს სხვადასხვა ქვეყნებში.

**ლოგიკური ელემენტი „და-არა“ (შეფერის შტრიხი).** ამ ლოგიკური ელემენტის რეალიზებისათვის გამოიყენება ორი საბაზო ლოგიკური ელემენტი „და“ და „არა“. ანუ ჯერ

ლოგიკური ელემენტი „და“ ამრავლებს შესასვლელზე მოცემულ ორობით რიცხვებს, ხოლო შემდეგ მიღებული შედეგის ინვერსირებას ახდენს ელემენტი „არა“. ინგლისურენოვან ლიტერატურაში ამ ელემენტს ეძახიან ელემენტი „NAND” (Not And). ნახ.1.9-ზე მოცემულია ელემენტი „და-არა“-ს სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში, მისი კონსტრუირების პრინციპი და ჭეშმარიტების ცხრილი.

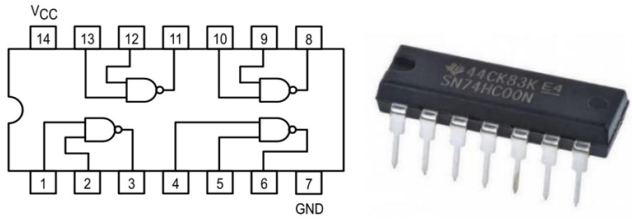



| A | B | A*B | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

გ

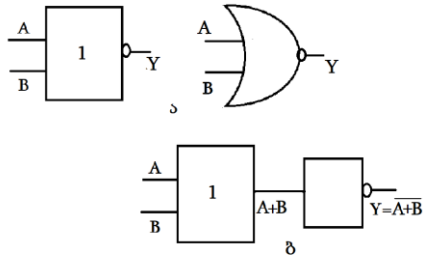
**ნახ.1.9. ა. ლოგიკური ელემენტი „და-არა“-ს სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში, ბ. მოქმედების პრინციპი, გ. ჭეშმარიტების ცხრილი**

პრაქტიკული თვალსაზრისით ლოგიკური ელემენტი „და-არა“ მზადდება მიკროსქემის სახით. მაგალითისათვის ნახ.1.10-ზე მოცემულია ერთ-ერთი ყველაზე გავრცელებული მიკროსქემის 7400-ის კორპუსში 4 „და-არა“ ელემენტის განლაგების სქემა და მიკროსქემის ფოტო. აღსანიშნავია, რომ ეს და სხვა კომბინირებული ლოგიკური ელემენტების რეალური დამზადება წარმოებს სხვადასხვა კომპანიების მიერ და ამიტომ მისი დასახელება შეიძლება განსხვავდებოდეს, ან სიმბოლოები ნუმერაციაში შეიძლება იყოს განსხვავებული, ხოლო მიკროსქემების ფუნქციონალური დანიშნულება კი იყოს ერთნაირი.




**ნახ.1.10. მიკროსქემა SN7400-ში "და-არა" ლოგიკური ელემენტების განლაგება და მიკროსქემის ფოტო გამოსახულება**

**ლოგიკური ელემენტი „ან-არა" (პირსის ისარი).** ამ ლოგიკური ელემენტის რეალიზებისათვის გამოიყენება ორი საბაზო ლოგიკური ელემენტი „ან" და „არა". ანუ ჯერ ლოგიკური ელემენტი „ან" ლოგიკურად შეკრებს შესასვლელზე მოცემულ ოროობით რიცხვებს, ხოლო შემდეგ მიღებული შედეგის ინვერსირებას ახდენს ელემენტი „არა". ინგლისურენოვან ლიტერატურაში ამ ელემენტს ეძახიან ელემენტი „NOR" (Not Or). ნახ.1.11-ზე მოცემულია ელემენტი „ან-არა"-ს სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში, მისი კონსტრუირების პრინციპი და ჭეშმარიტების ცხრილი.

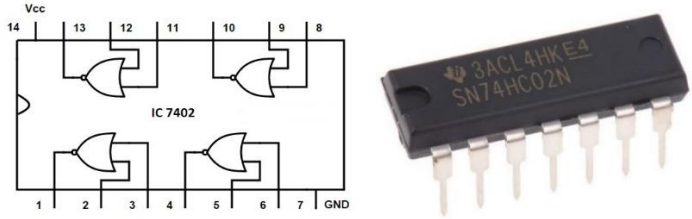



| A | B | A+B | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 |

გ

**ნახ.1.11. ა. ლოგიკური ელემენტი „ან-არა"-ს სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში**
**ბ. მოქმედების პრინციპი, გ. ჭეშმარიტების ცხრილი**

პრაქტიკული თვალსაზრისით ლოგიკური ელემენტი „ან-არა“ მზადდება მიკროსქემის სახით. მაგალითისათვის ნახ.1.12-ზე მოცემულია ერთ-ერთი ყველაზე გავრცელებული მიკრო-სქემის 7402-ის კორპუსში 4 „და-არა“ ელემენტის განლაგების სქემა და მიკროსქემის ფოტო.




**ნახ.1.12. მიკროსქემა SN7402-ში “ან-არა“ ლოგიკური ელემენტების განლაგება და მიკროსქემის ფოტოგამოსახულება**

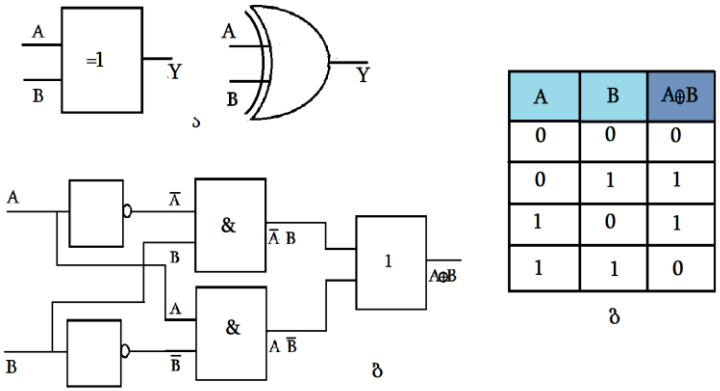
**ლოგიკური ელემენტი „ჯამი 2-ის მოდულით“.** ეს ლოგიკური ფუნქცია გამოისახება ფორმულით:

$$Y = \overline{A} * B + \overline{B} * A = A \oplus B \qquad (1.1)$$

სადაც სიმბოლო $\oplus$ აღნიშნავს ლოგიკურ ფუნქციას „ჯამი 2-ის მოდულით“.

ლოგიკური ფუნქციის რეალიზებისათვის გამოიყენება ორი საბაზო ლოგიკური ელემენტი „არა“, ორი ელემენტი „და“ და ერთი ელემენტი „ან“. ანუ ყველა გამოყენებული ლოგიკური ელემენტები საბაზო ლოგიკური ელემენტებიდანაა. ინგლისურ-ენოვან ლიტერატურაში ამ ელემენტს ეძახიან ელემენტი „XOR” (Exclusive Or). ნახ.1.13-ზე მოცემულია ელემენტის სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში, მისი კონსტრუირების პრინციპი და ჭეშმარიტების ცხრილი. ნახ.1.14.-

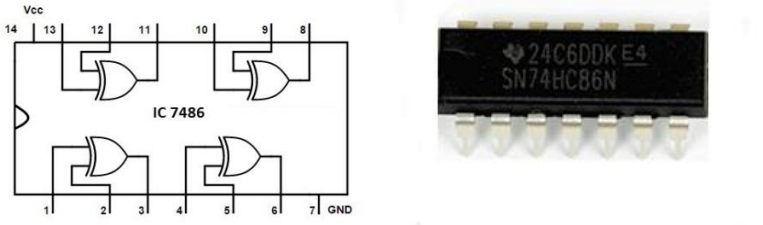
ზე მოცემულია 4 ლოგიკური ელემენტის „ჯამი 2-ის მოდულით“ განლაგება რეალურ მიკროსქემაში და მიკროსქემის ფოტო.



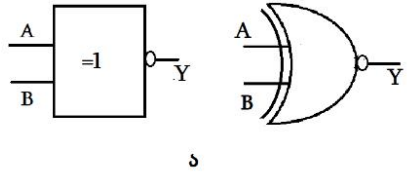

| A | B | A⊕B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

ბ

ნახ.1.13. ა. ლოგიკური ელემენტი „ჯამი 2-ის მოდულით“-ს (XOR) სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში, ბ. მოქმედების პრინციპი, გ. ჭეშმარიტების ცხრილი



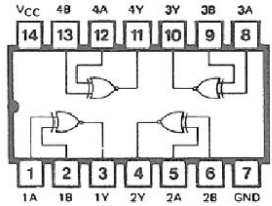

ნახ.1.14. მიკროსქემა SN7486-ში “ჯამი 2-ის მოდულით“-ში ლოგიკური ელემენტების განლაგება და მიკროსქემის ფოტო გამოსახულება



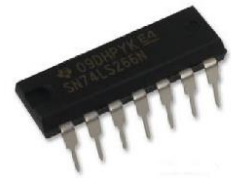

ა

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

ბ




გ



დ

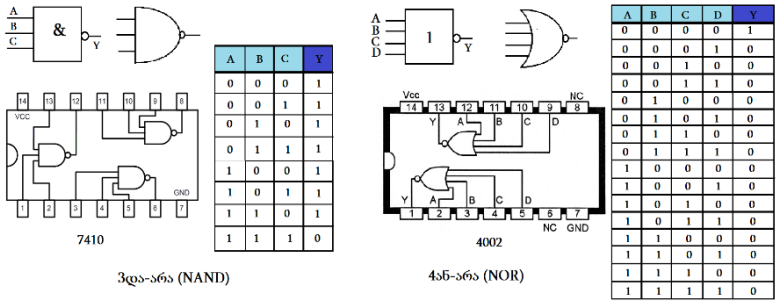
**ნახ.1.15. ელემენტი „ჯამი 2-ის მოდულით ინვერსიით“ (XNOR)**
**ა. სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში,**
**ბ. ჭეშმარიტების ცხრილი, გ. რეალურ მიკროსქემაში 4 ლოგიკური ელემენტის განლაგება,**
**დ. მიკროსქემის SN74266 ფოტო.**

პრაქტიკაში საკმაოდ ფართოდ გამოიყენება ლოგიკური ფუნქცია „ჯამი 2-ის მოდულით ინვერსიით“, რომელიც სქემატურად აღინიშნება „ჯამი 2-ის მოდულით“ მსგავსად გამოსასვლელზე დამატებით ინვერსიის ნიშნით. ინგლისურენოვან ლიტერატურაში ამ ელემენტს ეძახიან ელემენტი „XNOR”.

ნახ.1.15-ზე მოცემულია ელემენტის სქემატური აღნიშვნა ევროპულ და ამერიკულ სტანდარტებში, ჭეშმარიტების ცხრილი, რეალურ მიკროსქემაში 4 ლოგიკური ელემენტის „ჯამი 2-ის მოდულით ინვერსიით“ განლაგება და შესაბამისი მიკროსქემის SN 74266 ფოტო.

ჩვენ განვიხილეთ საბაზო და კომბინირებული ლოგიკური ელემენტები, რომლებსაც ძირითადად ორი შესავალი და ერთი გამოსასვლელი გააჩნიათ, გარდა ინვერტორისა ერთი შესასვლელით და ერთი გამოსასვლელით. პრაქტიკული თვალსაზრისით, ბევრ შემთხვევაში, მოსახერხებელია ლოგიკური ელემენტები რამდენიმე შესასვლელით და რეალურად ასეთი ლოგიკური ელემენტები იწარმოება სხვადასხვა მწარმოებლის მიერ. მაგალითისათვის, ლოგიკური ელემენტი „და“ 3 შესასვლელით - „3და“ (3AND) წარმოდგენილია მიკროსქემით სერიით 7411; 3-შესასვლელიანი ლოგიკური ელემენტი „3ან-არა“ (3NOR) – მიკროსქემა 7427; 4 შესასვლელიანი ლოგიკური ელემენტი „4და-არა“ (4NAND) - მიკროსქემა 7420; 8-შესასვლელიანი ლოგიკური ელემენტი „8და-არა“ (8NAND) -მიკროსქემა 7430 და სხვა, რომელთა ფართო არჩევანი ინტერნეტის საშუალებთ შეიძლება მოვიპოვოთ. ნახ.1.16-ზე მოცემულია ასეთი ელემენტების ორი მაგალითი: 1. ელემენტი „3და-არა“, 3-შესასვლელიანი ლოგიკური ელემენტი „და-არა“, რომელიც რეალიზებულია მიკროსქემით სერიით 7410. ერთ 14 კონტაქტიან (პინი) მიკროსქემაში განთავსებულია სამი აღნიშნული ლოგიკური ელემენტი. აქვე ნახაზზე მოცემულია ამ ელემენტის ჭეშმარიტების ცხრილი საიდანაც ჩანს, რომ „იმისათვის რომ ელემენტის გამოსასვლელზე გვქონდეს ლოგიკური ერთი საკმარისია მის ნებისმიერ ფეხზე იყოს ლოგიკური ერთი“. იგივე ნახ.1.16-ზე აგრეთვე მოცემულია ელემენტი „4ან-არა“, 4-შესასვლელიანი ლოგიკური ელემენტი „ან-არა“, რომელიც რეალიზებულია მიკროსქემით სერიით 4002. ერთ 14 კონტაქტიან (პინი) მიკროსქემაში განთავსებულია ორი აღნიშნული ლოგიკური ელემენტი. აქვე ნახაზზე მოცემულია ამ ელემენტის ჭეშმარიტების ცხრილი საიდანაც ჩანს, „იმისათვის რომ ელემენტის გამოსასვლელზე

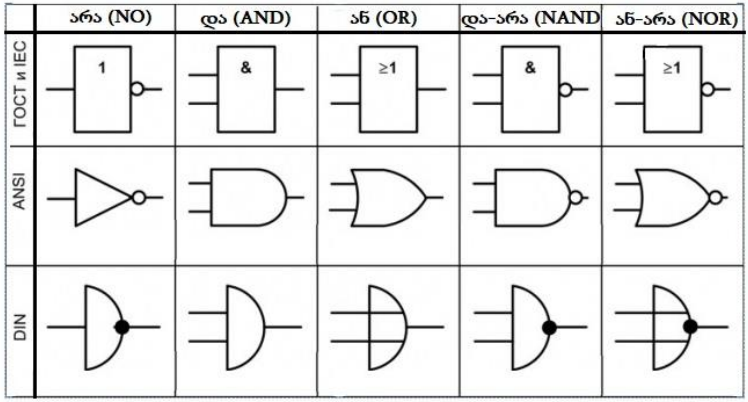
გვქონდეს ლოგიკური ერთი აუცილებელია მის ყველა შესასვლელზე იყოს ლოგიკური ნული“. სხვადასხვა მრავალშესასვლელიანი ლოგიკური ელემენტების გრაფიკული სქემატური აღნიშვნა და ჭეშმარიტების ცხრილი გამოისახება ზემოთ მოცემული ელემენტების ანალოგიურად და ბულის ალგებრის კანონების გამოყენებით.




**ნახ.1.16. ელემენტების „3და-არა“ და „4ან-არა“ სქემატური აღნიშვნა, ელემენტების განლაგება მიკროსქემის კორპუსში და ჭეშმარიტების ცხრილები**

უნდა აღინიშნოს, რომ სხვადასხვა მწარმოებლის მიერ მიკროსქემის დასახელება (ან დასახელებაში სხვადასხვა ასოების ცვლილება) და შესასვლელ/გამოსასვლელი კონტაქტების განლაგება შეიძლება იცვლებოდეს, რომლის შესახებაც ინფორმაცია ფიზიკურად მოყვება მიკროსქემას შეძენის დროს (ქაღალდზე ნაბეჭდი ინფორმაციის სახით) ან იძებნება ინტერნეტში სხვადასხვა ვებ გვერდებზე, სადაც ასევე მოცემულია კონკრეტული მიკროსქემის ტექნიკური პარამეტრები. მაგალითად ვებ-გვერდზე https://www.alldatasheet.com/ საძიებო მიკროსქემის

დასახელების შეყვანის შემდეგ მოგვეწოდება ვრცელი ინფორმაცია მიკროსქემის სხვადასხვა ტექნიკური და ფიზიკური პარამეტრების შესახებ, რაც გამოიყენება მისი პრაქტიკული ექსპლოატაციისა და მოდელირების პროცესში.




**ნახ.1.17. ლოგიკური ელემენტების სქემატური სიმბოლოების სხვადასხვა სტანდარტები**

არსებობს ლოგიკური ელემენტების სქემატური სიმბოლოების სხვადასხვა სტანდარტები, როგორიცაა: DIN, ANSI, ГОСТ და სხვა, რომელთაგან ზოგიერთი მათგანი ნახ.1.17-ზე არის მოცემული. ამასთან ცხადია, რომ თვით ლოგიკა არ არის დამოკიდებული თუ რომელი სტანდარტით არის მოცემული სქემა, განსხვავება მხოლოდ ნახაზზე გამოსახვით სიმბოლოებშია.

# თავი 2. ლოგიკური სქემების აგების პრინციპები

საბაზო ლოგიკური ელემენტებისა და კომბინაციური ლოგიკური ელემენტების საფუძველზე იქმნება კომპიუტერის ძირითადი ბლოკები. იმისათვის, რომ ლოგიკური ელემენტების ბაზაზე შესრულდეს რომელიმე ლოგიკური სქემა საჭიროა შესაბამისი ალგორითმი. ლოგიკური ფუნქციის შესაბამისი ლოგიკური სქემის შესადგენად პირველ რიგში საჭიროა დადგინდეს ცვლადების რაოდენობა ლოგიკურ ფუნქციაში. ეს რიცხვი ტოლი იქნება ლოგიკური სქემის შესასვლელთა რაოდენობისა. შემდგომ უნდა დადგინდეს საბაზო ლოგიკური ოპერაციების რაოდენობა და მათი თანმიმდევრობა ფუნქციაში. ეს რიცხვი განსაზღვრავს ლოგიკური ელემენტების რაოდენობას სქემაში.

ლოგიკაში ფუნქციის შესასრულებელ **ოპერაციათა პრიორიტეტები** (უპირატესობები), ისევე როგორც მათემატიკაში, შემდეგია: 1. ოპერაციები ფრჩხილებში; 2. ინვერსია (ლოგიკური უარყოფა); 3. კონიუნქცია (ლოგიკური გამრავლება); 4. დიზუნქცია (ლოგიკური შეკრება); 5. იმპლიკაცია და ეკვივალენტობა.

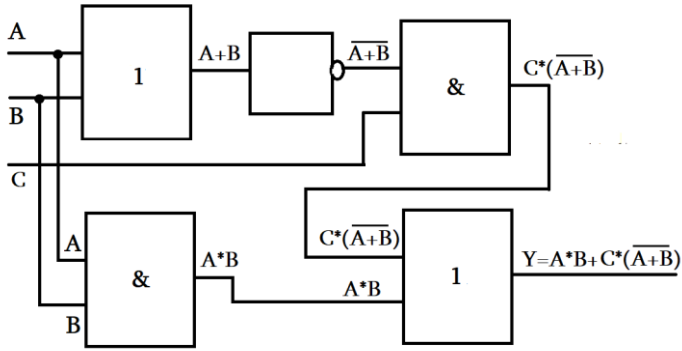
განვიხილოთ კონკრეტულ მაგალითი. ვთქვათ მოცემული გვაქვს ლოგიკური ფუნქცია:

$$Y = A * B + C * (\overline{A + B}) \qquad (2.1)$$

მოცემულ ფუნქციაში ცვლადთა რაოდენობა სამია A, B და C. ოპერაციების დასათვლელად ფუნქციაში გავითვალისწინოთ ის, რომ როგორც მათემატიკაში აქაც პირველ რიგში სრულდება ოპერაციები ფრჩხილებში **(A+B)**, შემდეგ ინვერსია $\overline{(A + B)}$, შემდეგ გამრავლება $C * \overline{(A + B)}$ და **A * B,** ბოლოს საბოლოო შეკრება $A * B + C * \overline{(A + B)}$. დავთვალოთ ოპერაციათა

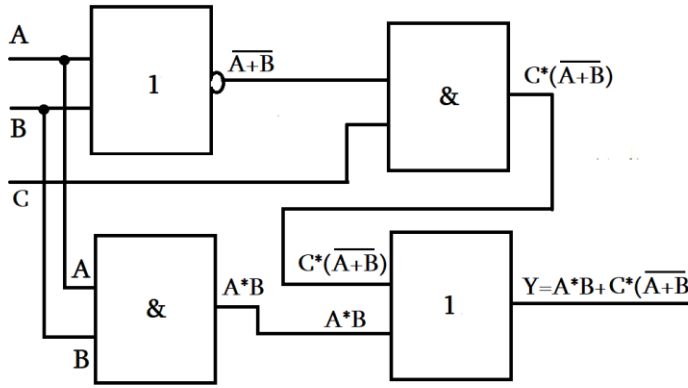
რაოდენობა: პირველი ოპერაცია შეკრება ფრჩხილებში მოთავსებულ წევრებისა - ოპერაცია ერთი, მიღებული ჯამის ინვერსია - ოპერაცია ორი, შედეგის ნამრავლი C-ზე - ოპერაცია სამი, **A * B** გამრავლება - ოპერაცია ოთხი და საბოლოო შეკრება - ოპერაცია ხუთი. ესე იგი ფორმულა (2.1) შეიცავს 3 ცვლადსა და 5 ოპერაციას.

ლოგიკური სქემის შესადგენად საჭიროა სქემატურად გამოვსახოთ ყველა შესასრულებელი ოპერაციის შესაბამისი ლოგიკური ელემენტი და შევაერთოთ ისინი ფორმულის შესაბამისად. მაგალითად ფორმულა (2.1)-ის შესაბამის ლოგიკურ სქემას ექნება ნახ.2.1-ზე მოცემული სახე.

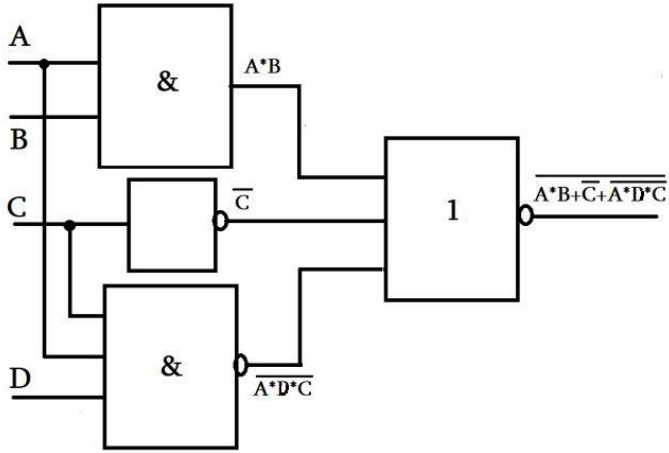



**ნახ.2.1. სამი ცვლადის ფუნქციის Y= A * B + C * ($\overline{A + B}$) ლოგიკური სქემა.**

ნახ.2.1-ზე მოცემულ სქემაში თუ გავითვალისწინებთ, რომ $\overline{A + B}$ წარმოადგენს ორ ლოგიკურ ოპერაციას, შეკრებასა და ინვერსიას, რომლის რეალიზება შესაძლებელია ერთი კომბინირებული ლოგიკური სქემით ან-არა (NOR), მაშინ შესაძლებელი იქნება ნახ.2.1-ის უფრო კომპაქტური სახით წარმოდგენა (ნახ.2.2)




ნახ.2.2. სამი ცვლადის ფუნქციის $Y= A * B + C * (\overline{A + B})$ ლოგიკური სქემის აგების მაგალითი




ნახ.2.3 ოთხი ცვლადის ფუნქციის $Y= \overline{A * B + \bar{C} + (\overline{A * D * C})}$ ლოგიკური სქემის აგების მაგალითი

# თავი 3. ჭეშმარიტების ცხრილი

## 3.1. რთული ლოგიკური სქემების ჭეშმარიტების ცხრილების შედგენის პრინციპები

რთული ლოგიკური ფუნქციის ჭეშმარიტების ცხრილის შედგენის პრინციპები განვიხილოთ კონკრეტული მაგალითის საფუძველზე. განვიხილოთ 3.1. ფორმულით მოცემული ლოგიკური გამოსახულება:

$$Y = A * ( B + \overline{B} * \overline{C} ) \qquad (3.1)$$

ჭეშმარიტების ცხრილის შესადგენად პირველ რიგში დავადგინოთ ცხრილში საჭირო სტრიქონების რაოდენობა, რომელიც ზოგადად გამოითვლება ფორმულით:

$$N = 2^n + 1 \qquad (3.2)$$

სადაც, N - სტრიქონების რაოდენობაა ჭეშმარიტების ცხრილში, ხოლო n - ცვლადების რაოდენობა ლოგიკურ გამოსახულებაში. „+1“ არის დამატებითი სტრიქონი ცვლადების, ოპერაციების და შედეგების სიმბოლურად მისათითებლად. 3.1 ფორმულით განსაზღვრულ ლოგიკურ გამოსახულებაში გვაქვს სამი ცვლადი A, B და C, ამიტომ გვექნება $N = 2^3 + 1 = 9$ სტრიქონი. აქვე ავღნიშნოთ, რომ $\overline{B}$ და $\overline{C}$ ცალკე ცვლადებად არ დაითვლება.

მეორე ეტაპზე უნდა მოხდეს საჭირო რაოდენობის სვეტების დათვლა. ფორმულა შემდეგია:

**M = ცვლადების რაოდენობა + ოპერაციების რაოდენობა (3.3)**

სადა, **M** - ჭეშმარიტების ცხრილში საძიებო სვეტების რაოდენობაა. 3.1-ფორმულაში გვაქვს 3 ცვლადი და 5 ლოგიკური ოპერაცია (2 ლოგიკური ინვერსია, 2 - ლოგიკური გამრავლება (კონიუნქცია) და 1 ლოგიკური შეკრება (დიზუნქცია)). აქედან გამომდინარე სვეტების რაოდენობისთვის მივიღებთ M=3 + 5 = 8.

მესამე ეტაპზე დავხაზოთ 5X8 ზომის ცხრილი, დავაწეროთ შესაბამისი სიმბოლოები და შევიტანოთ საწყისი ორობითი რიცხვები ცვლადებისათვის (ცხრ.3.1.)

| A | B | C | $\bar{B}$ | $\bar{C}$ | $\bar{B}*\bar{C}$ | $B+\bar{B}*\bar{C}$ | $A*(B+\bar{B}*\bar{C})$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | | | | | |
| 0 | 0 | 1 | | | | | |
| 0 | 1 | 0 | | | | | |
| 0 | 1 | 1 | | | | | |
| 1 | 0 | 0 | | | | | |
| 1 | 0 | 1 | | | | | |
| 1 | 1 | 0 | | | | | |
| 1 | 1 | 1 | | | | | |

**ცხრილი 3.1. ლოგიკური ფუნქციის ჭეშმარიტების ცხრილის შედგენის პრინციპი**

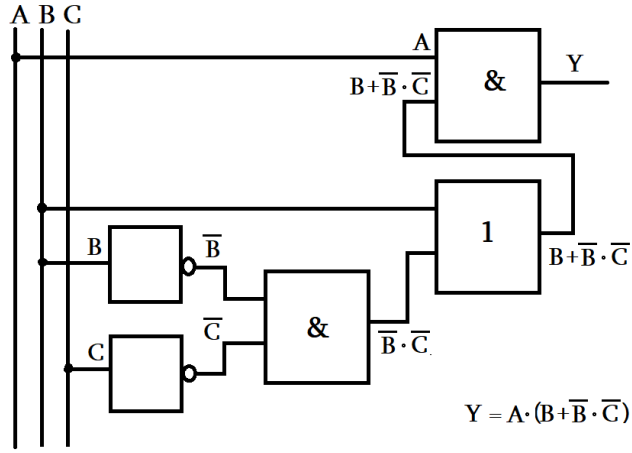
ცხრილში სიმბოლოთა და ოპერაციათა ჩაწერის თანმიმდევრობა შემდეგია "ჯერ ჩაიწერება ცვლადები (A,B,C), შემდეგ პირველ რიგში შესასრულებელი ოპერაციები, ჩვენს შემთხვევაში $\bar{B}$ და $\bar{C}$ ინვერსიები, შემდეგ მათი ნამრავლი $\bar{B} * \bar{C}$, შემდეგი ლოგიკური ჯამი $B + \bar{B} * \bar{C}$ და ბოლოს ფინალური ოპერაცია, შედეგის A-ზე გამრავლების ოპერაცია $A * ( B + \bar{B} * \bar{C})$.

ბოლო ეტაპზე ჭეშმარიტების ცხრილში ცვლადების საწყისი მნიშვნელობებისა და შესასრულებელი ფუნქციების მიხედვით

თანმიმდევრულად ვასრულებთ ლოგიკურ ოპერაციებს და შეგვაქვს მასში შესაბამისი ოპერაციების შესრულების შემდეგ მიღებული მნიშვნელობები (ცხრილი 3.2).

| A | B | C | $\overline{B}$ | $\overline{C}$ | $\overline{B} * \overline{C}$ | $B + \overline{B} * \overline{C}$ | $A * (B + \overline{B} * \overline{C})$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

ცხრილი 3.2. რთული ლოგიკური ფუნქციის ჭეშმარიტების ცხრილის მაგალითი




ნახ. 3.1. გამოსახულება $Y = A * ( B + \overline{B} * \overline{C} )$ ლოგიკური სქემა.

## 3.2. ლოგიკური სქემების სინთეზი ჭეშმარიტების ცხრილის მიხედვით

ლოგიკაში ერთ-ერთი მნიშვნელოვანი ამოცანაა ლოგიკური ფუნქციის ან/და ლოგიკური სქემის სინთეზი (შედგენა) ჭეშმარიტების ცხრილის მიხედვით. ფაქტიურად ამოცანა ე.წ. "შავი ყუთის" ამოცანაა, როდესაც ცნობილია მოწყობილობის (შავი ყუთის) შესასვლელზე მიწოდებული ინფორმაცია და გამოსასვლელზე მიღებული შედეგი, ხოლო დასადგენია რა ფუნქციას ასრულებს იგი. ფაქტიურად ლოგიკური გამოსახულების ჭეშმარიტების ცხრილი შეიცავს როგორც შესავალ ინფორმაციას ორობით კოდში, ასევე გამოსავალ ინფორმაციას. ამოცანა მოითხოვს ლოგიკური გამოსახულებისა და/ან ლოგიკური სქემის დადგენას. პრაქტიკულად იყენებენ ამოცანის ამოხსნის ორ პოპულარულ მეთოდს: „სრულყოფილი დიზუნქციური ნორმალური ფორმის" მეთოდს (სდნფ) და სრულყოფილი კონიუნქციური ნორმალური ფორმის" მეთოდს (სკნფ). „სდნფ" მეთოდს იყენებენ იმ შემთხვევაში, როდესაც ჭეშმარიტების ცხრილში საბოლოო გამოსავალი ორობითი ინფორმაცია შეიცავს ლოგიკური „1"-ის ნაკლებ რაოდენობას ვიდრე ლოგიკურ „0"-ს და პირიქით „სკნფ" მეთოდს იყენებენ მაშინ, როცა ჭეშმარიტების ცხრილში საბოლოო გამოსავალი ორობითი ინფორმაცია შეიცავს ლოგიკური „0"-ის ნაკლებ რაოდენობას ვიდრე ლოგიკურ „1"-ს.

გამოსახულების სრულყოფილი **დიზუნქციური** ნორმალური ფორმით ჩაწერის მაგალითისათვის განვიხილოთ 3 ცვლადის შემცველი ლოგიკური გამოსახულება:

$$(A * B * \bar{C}) + (\bar{A} * B * C) + (\bar{B} * \bar{A} * C) = Y \qquad (3.4)$$

ხოლო დიზუნქციური ნორმალური ფორმით ჩაწერილი გამოსახულება შესაძლოა შემდეგნაირად გამოიყურებოდეს:

$$(A * B) + (\bar{A} * B * C) + (\bar{B} * \bar{A}) = Y \qquad (3.5)$$

დიზუნქციური ნორმალური ფორმით გამოსახულების ჩანაწერი აუცილებლად უნდა შეიცავდეს **ლოგიკურ ჯამს** (დიზუნქციას) ფრჩხილებში მოთავსებული ცვლადების ლოგიკურ ნამრავლსა (კონიუნქციას).

თუ შევადარებთ 3.4 და 3.5 გამოსახულებებს დავინახავთ, რომ სრულყოფილი დიზუნქციური ნორმალური ფორმით ჩაწერის შემთხვევაში ფრჩხილებში განთავსებული ცვლადების რაოდენობა აუცილებლად უნდა იყოს საერთო ცვლადების რაოდენობის ტოლი და რადგანც ჩვენს კონკრეტულ შემთხვევაში ცვლადების რაოდენობა არის 3, ამიტომ ყოველ ფრჩხილში უნდა გვქონდეს 3 ცვლადი (ფორმულა 3.4) რაც არ არის აუცილებელი დიზუნქციური ნორმალური ფორმით ჩაწერილი გამოსახულებისათვის (ფორმულა 3.5) სადაც პირველ და მესამე შესაკრებში მხოლოდ ორი ცვლადი გვაქვს.

გამოსახულების სრულყოფილი **კონიუნქციური** ნორმალური ფორმით ჩაწერის მაგალითისათვის განვიხილოთ 3 ცვლადის შემცველი ლოგიკური გამოსახულება:

$$(A + B + \bar{C}) * (\bar{A} + B + C) * (\bar{B} + \bar{A} + C) = Y \qquad (3.6)$$

ხოლო დიზუნქციური ნორმალური ფორმით ჩაწერილი გამოსახულება შესაძლოა შემდეგნაირად გამოიყურებოდეს:

$$(A + B) * (\bar{A} + B + C) * (\bar{B} + \bar{A}) = Y \qquad (3.7)$$

კონიუნქციური ნორმალური ფორმით გამოსახულების ჩანაწერი აუცილებლად უნდა შეიცავდეს **ლოგიკურ ნამრავლს** (კონიუნქციას) ფრჩხილებში მოთავსებული ცვლადების ლოგიკური ჯამისა (დიზუნქციას).

თუ შევადარებთ 3.6 და 3.7 გამოსახულებებს დავინახავთ, რომ სრულყოფილი კონიუნქციური ნორმალური ფორმით ჩაწერის შემთხვევაში ფრჩხილებში განთავსებული ცვლადების რაოდენობა აუცილებლად უნდა იყოს საერთო ცვლადების რაოდენობის ტოლი და რადგან ჩვენს კონკრეტულ შემთხვევაში ცვლადების რაოდენობა არის 3, ამიტომ ყოველ ფრჩხილში უნდა გვქონდეს 3 ცვლადი (ფორმულა 3.6) რაც არ არის აუცილებელი კონიუნქციური ნორმალური ფორმით ჩაწერილი გამოსახულებისათვის (ფორმულა 3.7), სადაც პირველ და მესამე ლოგიკური თანამამრავლებისათვის მხოლოდ ორი ცვლადი გვაქვს.

აღწერილი მეთოდები განვიხლოთ კონკრეტულ მაგალითებზე.

**სრულყოფილი დიზუნქციური ნორმალური ფორმით ჩაწერის მეთოდი.** განვიხილოთ ჭეშმარიტების ცხრილი 3.3 რომლის მიხედვითაც საჭიროა ლოგიკური გამოსახულების დადგენა.

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

**ცხრილი 3.3. უცნობი ლოგიკური ფუნქციის ჭეშმარიტების ცხრილი**

პირველ რიგში უნდა შემოწმდეს Y გამოსავალ სვეტში რომელი ორობითი რიცხვი სჭარბობს. ჩვენს შემთხვევაში გვაქვს 3 ლოგიკური „1“ და 5 - ლოგიკური „0“, ამიტომ ვირჩევთ სრულყოფილი დიზუნქციური ნორმალური ფორმით ჩაწერის მეთოდს, რადგან ამ მეთოდით მიღებულ საბოლოო გამოსახულებაში შესაკრებთა რაოდენობას განსაზღვრავს „1“-ბის რაოდენობა ჭეშმარიტების ცხრილის გამოსავალ სვეტში.

ყურადღება გავამახვილოთ მხოლოდ იმ სტრიქონებზე სადაც გვაქვს ლოგიკური „1“ (ცხრილი 3.4). ასეთი სტრიქონების რაოდენობა არის 3, რაც ნიშნავს იმას, რომ საძიებელ ლოგიკურ გამოსახულებაში გვექნება 3 ლოგიკური შეკრების (დიზუნქციის) ოპერაცია. გავაანალიზოთ აღნიშნულ სტრიქონებში ცვლადების მნიშვნელობები. მონიშნულ პირველ სტრიქონში გვაქვს A=0, B=0, C=1. ამ მონაცემებისთვის სამართლიანი იქნება შემდეგი გამონათქვამი: „გამოსავალი არის „1“ როდესაც $\overline{A}$ და $\overline{B}$ და C ტოლია „1“-ს“, ანუ მივიღებთ გამოსახულების პირველ წევრს (შესაკრებს) $\overline{A} * \overline{B} * C$ . ესე იგი იქ სადაც ცვლადის მნიშვნელობა ლოგიკური „0“-ია შევცვალეთ ამ ცვლადის ინვერსიით, ხოლო იქ სადაც ცვლადი ლოგიკური „1“-ია დავტოვეთ უცვლელად და ეს ცვლადები ლოგიკურად გადავამრავლეთ.

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

**ცხრილი 3.4. უცნობი ლოგიკური ფუნქციის ჭეშმარიტების ცხრილი**

ანალოგიური მსჯელობით მეორე მონიშნული სტრიქონისათვის (A=0, B=1, C=1) მივიღებთ მეორე შესაკრებს $\bar{A} * B * C$ და მესამე სტრიქონისათვის (A=1, B=1, C=0) გვექნება ლოგიკური გამოსახულება $A * B * \bar{C}$. თუ მიღებულ სამ გამოსახულებას შევკრებთ მივიღებთ საძიებო ლოგიკურ გამოსახულებას (ფორმულა 3.7).

გამოსახულება 3.8 წარმოადგენს ჭეშმარიტების ცხრილი 3.3-ის მიხედვით სრულყოფილი დიზუნქციური ნორმალური ფორმით ჩაწერილ გამოსახულებას.

$$(\bar{A} * \bar{B} * C) + (\bar{A} * B * C) + (A * B * \bar{C}) = Y \qquad (3.8)$$

**სრულყოფილი კონიუნქციური ნორმალური ფორმით ჩაწერის მეთოდი.** განვიხილოთ ჭეშმარიტების ცხრილი 3.5 რომლის მიხედვითაც საჭიროა ლოგიკური გამოსახულების დადგენა.

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**ცხრილი 3.5. უცნობი ლოგიკური ფუნქციის ჭეშმარიტების ცხრილი კონიუნქციური მეთოდის გამოყენების პროცესში**

პირველ რიგში უნდა შემოწმდეს Y გამოსავალ სვეტში რომელი ორობითი რიცხვი სჭარბობს. ჩვენს შემთხვევაში გვაქვს 3 ლოგიკური „0“ და 5 - ლოგიკური „1“, ამიტომ რადგან ლოგიკური „0“-ების რაოდენობა ნაკლებია ვირჩევთ სრულყოფილი კონიუნქციური ნორმალური ფორმით ჩაწერის მეთოდს, რისთვისაც საჭიროა ყურადღება გავამახვილოთ მხოლოდ იმ სტრიქონებზე სადაც გვაქვს ლოგიკური „0“ (ცხრილი 3.6). ასეთი სტრიქონების რაოდენობა არის 3, რაც ნიშნავს იმას, რომ საძიებელ ლოგიკურ გამოსახულებაში გვექნება 3 ლოგიკური გამრავლების (კონიუნქციის) ოპერაცია.

გავაანალიზოთ აღნიშნულ სტრიქონებში ცვლადების მნიშვნელობები. მონიშნულ პირველ სტრიქონში გვაქვს A=0, B=0, C=1. ამ მონაცემებისთვის სამართლიანი იქნება შემდეგი გამონათქვამი: „გამოსავალი არის „0“ როდესაც $\overline{A}$ ან $\overline{B}$ ან C ტოლია „0“-ს“, ანუ მივიღებთ გამოსახულების პირველ წევრს $A + B + \bar{C}$ . ესიგი იქ სადაც ცვლადის მნიშვნელობა ლოგიკური „1“-ია შევცვალეთ ამ ცვლადის ინვერსიით, ხოლო იქ სადაც ცვლადი ლოგიკური „0“-ია დავტოვეთ უცვლელად და ეს ცვლადები ლოგიკურად შევკრიბეთ. ანალოგიური მსჯელობით მეორე მონიშნული სტრიქონისათვის (A=0, B=1, C=1) მივიღებთ საბოლოო გამოსახულების მეორე თანამამრავლს $A + \overline{B} + \bar{C}$ და მესამე სტრიქონისათვის (A=1, B=1, C=0) გვექნება ლოგიკური გამოსახულება $\overline{A} + \overline{B} + C$. თუ მიღებულ სამ გამოსახულებას ლოგიკურად გადავამრავლებთ ერთმანეთზე მივიღებთ საძიებო ლოგიკურ გამოსახულებას (ფორმულა 3.9)

$$(A + B + \bar{C}) * (A + \overline{B} + \bar{C}) * (\overline{A} + \overline{B} + C) = Y \qquad (3.9)$$

გამოსახულება 3.9 წარმოადგენს ჭეშმარიტების ცხრილი 3.5-ის მიხედვით სრულყოფილი კონიუნქციური ნორმალური ფორმით ჩაწერილ გამოსახულებას.

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**ცხრილი 3.6. უცნობი ლოგიკური ფუნქციის ჭეშმარიტების ცხრილი კონიუნქციური მეთოდის გამოყენების პროცესში**

ცხადია, რომ ცხრილი 3.3-ს მიმართ რომ გამოგვეყენებინა სრულყოფილი კონიუნქციური ნორმალური ფორმით ჩაწერის მეთოდი ან ცხრილი 3.5-ს მიმართ რომ გამოგვეყენებინა სრულყოფილი დიზუნქციური ნორმალური ფორმით ჩაწერის მეთოდი ორივე შემთხვევაში მივიღებდით ლოგიკურად იდენტურ გამოსახულებას, მაგრამ თითოეულ მათგანში გვექნებოდა 5 შესაკრები ან 5 თანამამრავლი ნაცვლად 3-სა, რაც გაცილებით გაართულებდა ლოგიკურ გამოსახულებას და ლოგიკურ სქემას. ამ შემთხვევაში რთული გამოსახულების გამარტივება (მინიმიზაცია) შესაძლებელია სხვადასხვა მეთოდებით, რომლებიც შემდეგ თავშია აღწერილი.

# თავი 4. ლოგიკური ფუნქციების მინიმიზაცია

რამდენიმე ცვლადიანი ლოგიკური ფუნქცია გარკვეული თვალსაზრისით შეიძლება საკმაოდ რთულად გამოიყურებოდეს და შესაბამისად მასზე დაყრდნობით შესრულებული ლოგიკური სქემაც მრავალკომპონენტიანი და რთული იქნება. ცვლადების რაოდენობის ზრდა ართულებს ფუნქციასა და შესაბამისად სქემასაც. მეორეს მხრივ არსებობს რიგი მეთოდებისა, რომელთა დახმარებით ხშირ შემთხვევაში საგრძნობლად შეიძლება რთული გამოსახულების გამარტივება. ამ დროს შესაბამისად მარტივდება სქემაც. ამ პროცეს სხვაგვარად მინიმიზაციას უწოდებენ. მაგალითად, ქვემოთ მოყვანილი ორი ფუნქცია (4.1) და (4.2) ლოგიკურად იდენტურნი არიან.

$$\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}\bar{C} \qquad (4.1)$$

$$\bar{A}B + \bar{B}\bar{C} \qquad (4.2)$$

ლოგიკური ფუნქციების მინიმიზაციის, ანუ გამარტივების სხვადასხვა მეთოდები არსებობენ, რომელთა ნაწილი პოპულარობით სარგებლობენ სიმარტივისა და პრაქტიკული მოსახერხებლობის თვალსაზრისით, ზოგიც სპეციალური კონკრეტული ამოცანებისთვის გამოიყენება. განვიხილოთ ლოგიკური ფუნქციების მინიმიზაციის რამდენიმე პოპულარული მეთოდი.

## 4.1. ალგებრული მეთოდი

ალგებრული მეთოდი ემყარება ლოგიკური ფუნქციების თვისებებს, ბულის ალგებრის კანონებს და ძალიან გავს ალგებრული გამოსახულებების გამარტივების მეთოდს. მაგალითად,

შეიძლება გამოვიყენოთ ერთი და იგივე ცვლადის ფრჩხილებს გარეთ გამოტანის ან სხვა მსგავსი მეთოდი. ისევე, როგორც ლოგიკურ გამოსახულებაში გამრავლების ნიშანი რეალობაში წარმოადგენს „კონიუნქციას“, შეკრების ნიშანი წარმოადგენს „დიზუნქციას“ და ა.შ, აქაც, ლოგიკური ფუნქციის გამარტივების დროს ყოველთვის უნდა გვახსოვდეს, რომ საქმე გვაქვს არა მათემატიკურ, არამედ ლოგიკურ გამოსახულებასთან და ყოველ პროცედურას მივუყენოთ მისი შესაბამისი ბულის ლოგიკის კანონები. განვიხილოთ კონკრეტული მაგალითი, გავამარტივოთ გამოსახულება:

$$\bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC \qquad (4.3)$$

ცხადია, რომ პირველი და მეორე შესაკრებიდან ფრჩხილებს გარეთ შეიძლება გამოვიტანოთ $\bar{A}B$ და ფრჩხილებში დარჩება $\bar{C} + C$, რაც ბულის ალგებრის თანახმად ლოგიკური 1-ის ტოლია, ასე რომ პირველი ორი შესკრების ადგილას დაგვრჩება დაგვრჩება მხოლოდ $\bar{A}B$. მეოთხე და მეხუთე შესაკრებიდან ფრჩხილებს გარეთ გამოვა $AB$ და ფრჩხილებში კვლავ დარჩება $\bar{C} + C$, რაც ლოგიკური 1-ის ტოლია. ასე, რომ მეოთხე და მეხუთე შესაკრების ნაცვლად დაგვრჩება $AB$. ერთი შეხედვით გაუმარტივებელი გვრჩება შესაკრები $A\bar{B}C$, მაგრამ ასე იქნებოდა ალგებრული გამოსახულების დროს. ლოგიკურ გამოსახულებებში კი შესაძლებელია $A\bar{B}C$ დავაწყვილოთ სხვა შესკრებელთან თუ ასეთი არსებობს, მაგალითად კვლავ გამოვიყენოთ $ABC$, რადგან მათ საერთო აქვთ $AC$, რომელთა ფრჩხილებს გარეთ გამოტანის შემდეგ ფრჩხილებში დაგვრჩება $B + \bar{B}$, რაც ლოგიკური 1-ის ტოლია. ამგვარად დაგვრჩა სამი მარტივი შესაკრებელი:

$$\bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC = \bar{A}B + AB + AC = B + AC \quad (4.4)$$

როგორც ვხედავთ, გამოვიყენეთ რა ბულის ალგებრის კანონები, საკმაოდ რთული (4.3) ლოგიკური გამოსახულებიდან მივიღეთ ძალიან მარტივი (4.4) გამოსახულება. აქვე აღვნიშნოთ, რომ იმ შემთხვევაში თუ ვერ დავინახავდით ბოლო გამარტივებას, ანუ $A\bar{B}C$ და $ABC$ -დან $AC$-ს გამოტანის ვარიანტს, მაშინ შედარებით უფრო რთული გარდაქმნებით კვლავ მივიღებთ საბოლოო (4.4) შედეგს. ჩავატაროთ ეს გარდაქმნები, ვთქვათ:

$$\bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC = \bar{A}B + A\bar{B}C + AB = B + A\bar{B}C \quad (4.5)$$

აქ პირველი და მესამე შესაკრებლებიდან ფრჩხილებს გარეთ გავიტანეთ $B$ და ფრჩხილებში დარჩენილი $\bar{A} + A = 1$. გავამარტივოთ (5) შემდეგნაირად:

$$B + A\bar{B}C = (B + A)(B + \bar{B})(B + C) = (B + A)(B + C) = B + AC \quad (4.6)$$

აქ $(B + A)$ $(B + C)$-სთვის გამოვიყენეთ ბულის ლოგიკის დისტრიბუტიულობის კანონი. ესე იგი საბოლოოდ მივიღეთ იგივე შედეგი, რაც (4.4) გამოსახულებაში გვაქვს, მაგრამ გამოვიყენეთ მეტი პროცედურები. ასე, რომ მეთოდი რომელიც უშუალოდ იქნა გამოყენებული (4.4) გამოსახულებაში მნიშვნელოვნად აიოლებს ლოგიკური გამოსახულების გამარტივების მეთოდს, მაგრამ მეორეს მხრივ, საჭიროა ბულის ალგებრის კანონების ცოდნა და მისი მორგება, დანახვა კონკრეტული გამოსახულებისათვის.

## 4.2. კარნოს ბარათები

მეთოდი, რომელიც ერთ-ერთ ყველაზე პოპულარულად ითვლება ლოგიკური გამოსახულების გამარტივებისათვის, ცნობილია კარნოს ბარათების სახელით. ეს მეთოდი პირველად შემოთავაზებული იყო 1952 წელს ედუარდ ვეიჩის მიერ, ხოლო 1953 წელს, მოდერნიზების შემდეგ, წარმოადგინა მორის კარნომ ციფრული სისტემების პროექტირების გამარტივების მიზნით. მეთოდი ემყარება გარკვეული სტანდარტული პროცედურების ჩატარების ლოგიკურ გამოსახულებაზე. კარნოს ბარათის წარმოდგენა შესაძლებელია კლასიკური სახით, მაგრამ თვალსაჩინოების და მარტივად გაგების მიზნით მოსახერხებელია მეთოდის ახსნა კონკრეტულ რაოდენობა ცვლადებზე დაყრდნობით.

**ორი ცვლადის შემთხვევა**. განვიხილოთ ლოგიკური ფუნქცია და მისი შესაბამისი ჭეშმარიტების ცხრილი:

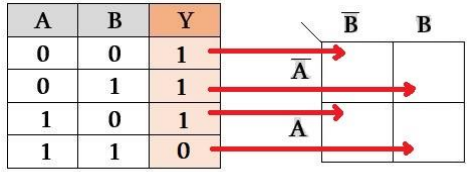
$$\bar{A}B + A\bar{B} + AB = Y \qquad (4.7)$$

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**ცხრილი.4.1. ლოგიკური ფუნქცია და ჭეშმარიტების ცხრილი**

კარნოს ბარათის ასაგებად უნდა შევადგინოთ სპეციალური ცხრილი ნახ.4.1-ზე მოტანილი სახით. ცხრილის მარჯვნივ დახაზულ ოთხ უჯრაში ცხრილიდან უნდა მოხდეს Y მონაცემების

გადატანა ისრებით მითითებულ უჯრებში. აქვე ძალიან მნიშვნელოვანია, რომ აქ და შემდეგაც, ცხრილზე $\overline{A}$, A, $\overline{B}$ და B ცვლადები ზუსტად ისე უნდა იყოს მითითებული როგორც ნახაზზეა. დაუშვებელია მათი ადგილების შეცვლა. ამ პროცედურის ჩატარების შემდეგ მივიღებთ კარნოს ბარათს ჩვენი კონკრეტული შემთხვევისათვის (ნახ.4.1). მოვნიშნოთ და გავაერთიანოთ ერთ კონტურში მეზობლად (არა დიაგონალურად) განლაგებული „1“-ინაბი. მივიღებთ ნახ.4.2.-ზე გამოსახულ ნახაზს.




| | $\overline{B}$ | B |
|---|---|---|
| $\overline{A}$ | 1 | 1 |
| A | 1 | 0 |

ნახ.4.1. კარნოს ბარათის შედგენის პროცედურა და ბარათი კონკრეტული ამოცანისათვის

| | $\overline{B}$ | B |
|---|---|---|
| $\overline{A}$ | 1 | 1 |
| A | 1 | 0 |

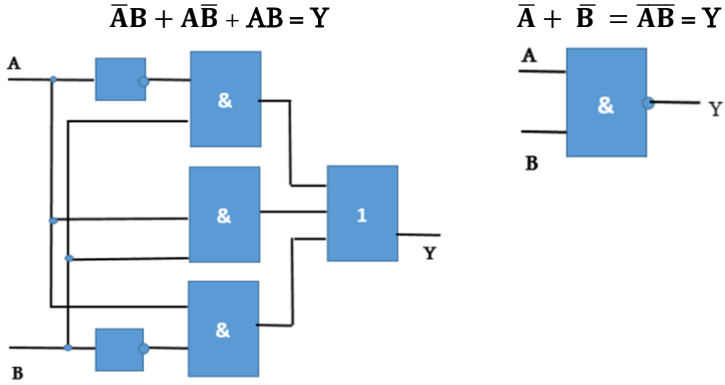
$$\overline{A} + \overline{B} \qquad (4.8)$$

ნახ.4.2. კარნოს ბარათში მეზობლად განლაგებული 1-ების მონიშვნის პროცედურა.

მიღებული კარნოს ბარათიდან შეგვიძლია დავწეროთ მინიმიზირებული ფუნქცია, სადაც ვერტიკალურად განლაგებული 1-ებისთვის გვექნება $\bar{B}(\bar{A}+A) = \bar{B}$ და ჰორიზონტალურად განთავსებული 1-ბისთვის გვექნება $\bar{A}(\bar{B}+B) = \bar{A}$. მიღებული შედეგები უნდა შევკრიბოთ, ანუ საბოლოოდ მივიღებთ $\bar{A} + \bar{B}$ ან თუ ამ გამოსახულებას მივუყენებთ დე მორგანის კანონს მივიღებთ $\bar{A} + \bar{B} = \overline{AB}$, რაც პრაქტიკული თვალსაზრისითაა ხელსაყრელი, რადგან $\overline{AB}$ უშუალოდ იწარმოება ლოგიკური ელემენტის (მიკროსქემის) სახით.

იმისათვის რომ თვალნათლივ დავინახოთ ლოგიკური ფუნქციის პრაქტიკული რეალიზების შემთხვევაში რა ეფექტი აქვს კარნოს ბარათის გამოყენებას, დავხაზოთ ლოგიკური სქემები ორივე შემთხვევისთვის, მინიმიზაციამდე და მინიმიზაციის შემდეგ (ნახ.4.3.).

ეფექტი ცხადია: პირველ შემთხვევაში სქემა მოითხოვს ექვს კომპონენტს და მეორე შემთხვევაში კი მხოლოდ ერთს.




**ნახ.4.3. ლოგიკური სქემები მინიმიზაციამდე და მინიმიზაციის შემდეგ**

**სამი ცვლადის შემთხვევა**. დასაწყისისთვის განვიხილოთ შედარებით მარტივი შემთხვევა. ვთქვათ მოცემულია ლოგიკური ფუნქცია (4.9) ან მისი შესაბამისი ჭეშმარიტების ცხრილი 4.2.

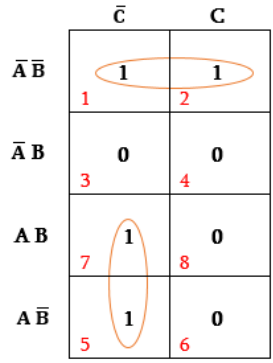
სამი ცვლადისათვის კარნოს ბარათის ასაგებად უნდა შევადგინოთ სპეციალური ცხრილი ნახ.4.4-ზე მოტანილი სახით. 4.2 ცხრილიდან მის მარჯვნივ დახაზულ რვა უჯრაში მოცემული ცვლადების გადატანა კარნოს ბარათზე უნდა მოხდეს N ნუმერაციის და კარნოს ბარათზე დატანილი ნუმერაციის შესაბამისად და არავითარ შემთხვევაში სხვა სახით! ასევე სამი ცვლადისათვის იგივე ჩაწერის მეთოდი უნდა გამოვიყენოთ ყველა სხვა შემთხვევაშიც. აქვე აღვნიშნოთ ძალიან მნიშვნელოვანი მომენტი: აქ და შემდეგაც, ცხრილზე $\bar{A}$, **A**, $\bar{B}$ , **B**, $\bar{C}$ და **C** ცვლადები ზუსტად ისე უნდა იყოს მითითებული როგორც ნახაზზეა. დაუშვებელია მათი ადგილების შეცვლა. ამ პროცედურის ჩატარების შემდეგ მივიღებთ კარნოს ბარათს ჩვენი კონკრეტული შემთხვევისათვის (ნახ.4.4).

$$\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}C = Y \qquad (4.9)$$

| N | A | B | C | y |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 1 |
| 3 | 0 | 1 | 0 | 0 |
| 4 | 0 | 1 | 1 | 0 |
| 5 | 1 | 0 | 0 | 1 |
| 6 | 1 | 0 | 1 | 0 |
| 7 | 1 | 1 | 0 | 1 |
| 8 | 1 | 1 | 1 | 0 |

**ცხრილი 4.2. სამი ცვლადის ლოგიკური ფუნქცია და ჭეშმარიტების ცხრილი**

მოვნიშნოთ და გავაერთიანოთ ერთ კონტურში მეზობლად (არა დიაგონალურად) განლაგებული 1-ინაბი. მივიღებთ ნახ.4.4.-ზე გამოსახულ ნახაზს. მნიშვნელოვანია ის რომ შეიძლება მხოლოდ ვერტიკალურად ან/და ჰორიზონტალურად განლაგებული მეზობლად განლაგებული 1, 2, 4, 8 და ა.შ (2-ის ხარისხის მიხედვით) ერთიანის გაერთიანება ერთ კონტურში




$$Y = \bar{A}\,\bar{B} + A\,\bar{C} \qquad (4.10)$$

**ნახ.4.4. კარნოს ბარათი სამი ცვლადის ფუნქციისთვის**

ცხრილზე ჰორიზონტალურად მონიშნული „1“-ებისთვის იცვლება მხოლოდ $\bar{C}$ და **C,** ამიტომ ამ შემთხვევაში დაგვრჩება მხოლოდ $\bar{A}\,\bar{B}$**.** ვერტიკალურად განლაგებული „“1-ბისთვის იცვლება მხოლოდ **B** და $\bar{B}$**,** ამიტომ ამ შემთხვევისთვის დაგვრჩება მხოლოდ $A\,\bar{C}$. აქედან გამომდინარე საბოლოოდ ფუნქციისთვის მივიღებთ $\bar{A}\,\bar{B} + A\,\bar{C}$**.** როგორც ვხედავთ მე-9 ფორმულით მოცემული ლოგოკური ფუნქცია კარნოს ბარათის გამოყენების შემდეგ მნიშვნელოვნად გამარტივდა და ნაცვლად 10 ლოგიკური ოპერაციისა დაგვრჩა მხოლოდ 6.

ნახ.4.5-ზე მოცემულია კონკრეტული მაგალითი იდენტური ლოგიკური ფუნქციებისა, სადაც თვალნათლივ ჩანს თუ როგორი გავლენა შეიძლება მოახდინოს კარნის ბარათის გამოყენებამ ფუნქციის მინიმიზაცისთვის. ცხადია, რომ მეორე გამოსახულება, რომლის მინიმიზაციაც მოხდა კარნოს ბარათის მიხედვით, გაცილებით მარტივია ვიდრე პირველი, რომლის მინიმიზაციაც არ წარმოებულა.

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

**ნახ.4.5. იდენტური ლოგიკური ფუნქციები მინიმიზაციის შემდეგ**

ეხლა განვიხილოთ შედარებით რთული შემთხვევა. ვთქვათ მოცემულია ჭეშმარიტების ცხრილი (ცხრილი.4.3) და უნდა დავადგინოთ მისი ლოგიკური ფუნქცია.

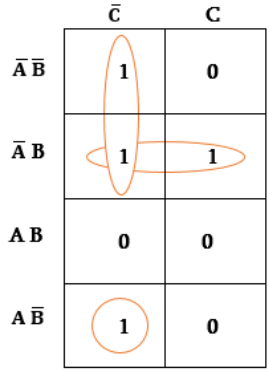
| A | B | C | y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

**ცხრილი 4.3. ჭეშმარიტების ცხრილი კონკრეტული მაგალითისათვის**

4.3 ჭეშმარიტების ცხრილის მიხედვით შეგვიძლია პირდაპირ დავწეროთ ლოგიკური ფუნქცია ან ჯერ მოვახდინოთ მინიმიზაცია კარნოს ბარათის გამოყენებით და შემდეგ ჩავწეროთ ფუნქცია ცხადი სახით. თვალსაჩინოებისათვის შევასრულოთ ორივე პროცედურა. ჭეშმარიტების ცხრილიდან გამომდინარე ფუნქციას ექნება შემდეგი სახე:

$$\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + \bar{A}\bar{C}B + \bar{A}BC = Y \qquad (4.11)$$

იგივე ჭეშმარიტების ცხრილის მიხედვით შევადგინოთ კარნოს ბარათი და მეზობლად განთავსებული „1“-ები მოვაქციოთ ერთიან კონტურებში. ერთ შემთხვევაში, ეს კონტურები ნახ.4.4-ზე განთავსებული კარნოს ბარათით შეიძლება გამოვსახოთ, რომლისთვისაც მივიღებთ ფორმულა 4.12-ს.

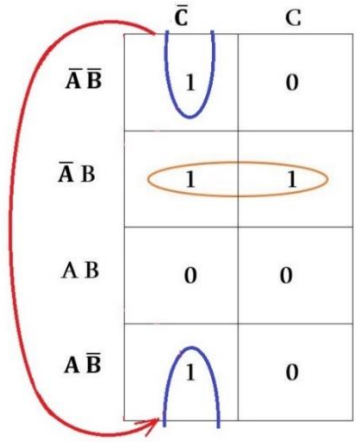



$$Y = \bar{A}\,\bar{C} + \bar{A}\,B + A\,\bar{B}\,\bar{C} \qquad (4.12)$$

**ნახ.4.4. კარნოს ბარათი და შესაბამისი ფუნქცია კონკრეტული ამოცანისთვის**

მეორეს მხრივ ასეთი ქმედება იქნება არაოპტიმალური, რადგან კარნოს ბარათს გააჩნია მნიშვნელოვანი **თვისება**, რომელიც კიდევ უფრო ამარტივებს საძიებო ფუნქციას:

**კარნოს ბარათში უკიდურესად ზემოთ და უკიდურესად ქვემოთ რიგში განთავსებული 1-ები შეიძლება მოვაქციოთ ერთ კონტურში თუ ისინი ერთ სვეტში არიან განთავსებულნი და უკიდურესად მარცხენა და უკიდურესად მარჯვენა სვეტებში განთავსებული 1-ები შეიძლება მოვაქციოთ ერთ კონტურში თუ ისინი ერთ რიგში არიან განთავსებულნი.**




$$Y = \bar{A}\,B + \bar{B}\,\bar{C} \qquad (4.13)$$

**ნახ. 4.5. კარნოს ბარათი და შესაბამისი ფუნქცია კონკრეტული ამოცანისთვის**

მარტივად გაგებისთვის წარმოვიდგინოთ, რომ კარნოს ბარათი გვაქვს განცალკევებულ ქაღალდის ფურცელზე. თუ ამ ქაღალდის ფურცელს ისე გადავხრით, რომ მისი ზედა კიდე დაუკავშირდეს ქვედა კიდეს (ცილინდრული ფორმა) და თუ ამ

შემთხვევაში 1-ები აღმოჩნდებიან განთავსებულნი მეზობლად, მაშინ ისინი შეიძლება მოვაქციოთ ერთ კონტურში და გამოვიყენოთ კარნოს მეთოდი. მივიღებთ 4.13 ფორმულას (ნახ.4.5). საბოლოოდ მივიღებთ, რომ თუ საწყის ფუნქციაში გვჭირდებოდა 10 ლოგიკური ოპერაცია კარნოს ბარათით მინიმიზაციის შემდეგ დაგვჭირდება მხოლოდ 6 და შესაბამისი სქემაც მნიშვნელოვნად გამარტივდება.

$$Y= \bar{A}\,\bar{B}\,\bar{C} + A\,\bar{B}\,\bar{C} + \bar{A}\,\bar{C}\,B + \bar{A}\,B$$
$$C= \bar{A}\,\bar{C} + \bar{A}\,B + A$$
$$\bar{B}\,\bar{C} = \bar{A}\,B + \bar{B}\,\bar{C} \qquad (4.14)$$

კარნოს ბარათის ზემოთ აღნიშნული თვისებების მეორე ნაწილი სამ ცვლადიანი ფუნქციისთვის არააქტუალურია, რადგან ამ შემთხვევაში მხოლოდ ორი სვეტი გვაქვს, მაგრამ იგი აქტუალურია 4 და მეტი ცვლადიანი ფუნქციისათვის.

**ოთხი ცვლადის შემთხვევა.** ეს შემთხვევა მაქსიმალურად ავლენს კარნოს ბარათის უპირატესობებს და პრაქტიკულობას ლოგიკური ფუნქციების მინიმიზაციის პროცესში. განვიხილოთ კონკრეტული მაგალითი 4 ცვლადისათვის. ვთქვათ მოცემული გვაქვს ფუნქცია შემდეგი სახით (ცხრილი 4.4). დავადგინოთ მინიმიზებული ლოგიკური ფუნქცია.

აუცილებლობას არ წარმოადგენს, მაგრამ შემდგომი თვალსაჩინოებისათვის დავწეროთ როგორი სახე აქვს მინიმიზაციამდე ლოგიკურ ფუნქციას:

$$Y = \bar{A}\,\bar{B}\,C\,\bar{D} + \bar{A}\,\bar{B}\,CD + \bar{A}\,B\,C\,\bar{D} + \bar{A}\,B\,C\,\bar{D} + A\,\bar{B}\,\bar{C}\,\bar{D} + A\,\bar{B}\,\bar{C}\,D +$$
$$+ A\,\bar{B}\,C\,\bar{D} + A\,\bar{B}\,C\,D + A\,B\,\bar{C}\,D + A\,B\,C\,\bar{D} + A\,B\,C\,D \qquad (4.15)$$

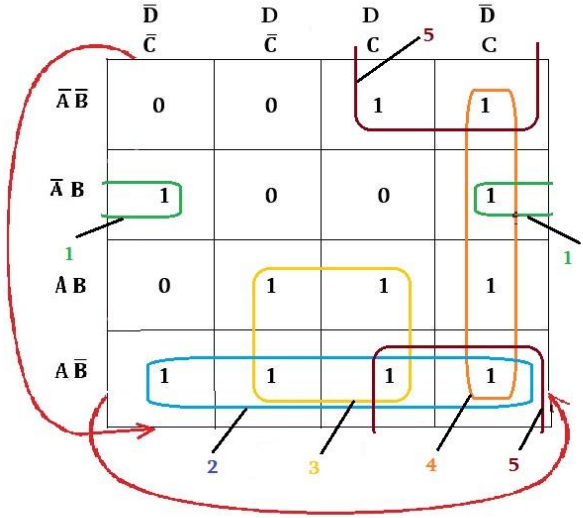
დავხაზოთ კარნოს ბარათი და გადმოვიტანოთ მასზე ცვლადები ზუსტად მითითებული თანმიმდევრობით ნუმერაციის შესაბამისად.

| N | A | B | C | D | y |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 0 | 1 |
| 4 | 0 | 0 | 1 | 1 | 1 |
| 5 | 0 | 1 | 0 | 0 | 1 |
| 6 | 0 | 1 | 0 | 1 | 0 |
| 7 | 0 | 1 | 1 | 0 | 1 |
| 8 | 0 | 1 | 1 | 1 | 0 |
| 9 | 1 | 0 | 0 | 0 | 1 |
| 10 | 1 | 0 | 0 | 1 | 1 |
| 11 | 1 | 0 | 1 | 0 | 1 |
| 12 | 1 | 0 | 1 | 1 | 1 |
| 13 | 1 | 1 | 0 | 0 | 0 |
| 14 | 1 | 1 | 0 | 1 | 1 |
| 15 | 1 | 1 | 1 | 0 | 1 |
| 16 | 1 | 1 | 1 | 1 | 1 |

**ცხრილი 4.4. ოთხცვლადიანი ლოგიკური ფუნქციის ჭეშმარიტების ცხრილი**

ყველა სხვა შემთხვევაში ცვლადების გადმოტანა კარნოს ბარათზე მოვახდინოთ მხოლოდ ამ თანმიმდევრობით და არა სხვანაირად, ეს მნიშვნელოვანია. გავაერთიანოთ 1-ები ზემოთ მოცემული კარნოს ბარათის თვისებების შესაბამისად (ნახ.4.6).

| | $\bar{D}\bar{C}$ | $D\bar{C}$ | $DC$ | $\bar{D}C$ |
|---|---|---|---|---|
| $\bar{A}\bar{B}$ | 0 (1) | 0 (2) | 1 (4) | 1 (3) |
| $\bar{A}B$ | 1 (5) | 0 (6) | 0 (8) | 1 (7) |
| $AB$ | 0 (13) | 1 (14) | 1 (16) | 1 (15) |
| $A\bar{B}$ | 1 (9) | 1 (10) | 1 (12) | 1 (11) |




ნახ. 4.6. კარნოს ბარათი ოთხცვლადიანი ფუნქციის კერძო მაგალითისათვის

დაწვრილებით განვიხილოთ მე-4.6 ნახაზზე მიცემული კარნოს ბარათი.

როგორც ვხედავთ კარნოს ბარათის ძირითადი თვისებების გამოყენებით, „1“-ბის გაერთიანების შედეგად მიღებულია 5 კონტური. პირველი კონტური (მწვანე) აერთიანებს ორ „1“-ს, რომლებიც განლაგებულია ცხრილის უკიდურეს მარჯვენა და მარცხენა სვეტებში და მდებარეობენ ერთ რიგში, ამიტომ შესაძლებელია მათი გაერთიანება. ამ კონტურის მარჯვენა ნაწილში მითითებულია ცვლადები $\bar{A}\ B$, ამასთან მარჯვენა „1“ მდებარეობს სვეტში ცვლადებით $\bar{C}\,\bar{D}$ ხოლო მარცხენა „1“ მდებარეობს სვეტში ცვლადებით $C\,\bar{D}$. ამ შემთხვევაში $\bar{C}\,\bar{D} + C\,\bar{D} = \bar{D}$, ანუ ვიტოვებთ მხოლოდ იმ ცვლადს რომელიც არ იცვლება. $\bar{D}$ უცვლელია, ხოლო $C$ იცვლება $\bar{C}$-ით, ე.ი ამოვარდება. საბოლოოდ ამ კონტურისთვის დაგვრჩება $\bar{A}\,B\,\bar{D}$.

მე-2 კონტური აერთიანებს ოთხ წევრს. მათი რიგის გასწვრივ გვაქვს $A\,\bar{B}$, რომლის არცერთი ცვლადი არ იცვლება და ამიტომ ის რჩება უცვლელად, ხოლო სვეტებში გვაქვს $\bar{C}\,\bar{D}$, $\bar{C}\,D$, $CD$ და $C\,\bar{D}$, რომელთაგან ყველა იცვლება და ამიტომ ოთხივე ამოვარდება, ანუ საბოლოოდ ამ კონტურისთვის დაგვრჩება $A\,\bar{B}$.

მე-3 კონტურშიც გაერთიანებული 4 წევრს, რომელთათვისაც რიგებში გვაქვს $A\,B$ და $A\,\bar{B}$. ამ კომბინაციაში იცვლება $B$ და $\bar{B}$ რომელიც ამოვარდება და დარჩება მხოლოდ $A$. კონტურის სვეტებში მითითებულია $CD$ და $\bar{C}\,D$, რომელთაგანაც დაგვრჩება $D$. საბოლოოდ მე-3 კონტურის გამარტივება მოგვცემს $A\,D$ -ს.

მე-4 კონტურიც აერთიანებს ოთხ წევრს, რომელთათვისაც სვეტში გვაქვს ცვლადთა მხოლოდ ერთი კომბინაცია $\bar{C}\,D$, რომელიც უცვლელად დარჩება, ხოლო რიგებში გვაქვს $A$-სა და $B$-ს და მათი ინვერსიების ნამრავლის ყველა ვარიანტი, რის გამოც

ეს ორივე ცვლადი განიცდის ცვლილებას და ამიტომ ისინი სრულად ამოვარდება. გვრჩება მხოლოდ კომბინაცია $\bar{C}\,D$.

მე-5 კონტური აერთიანებს 4 წევრ, რომლებიც განლაგებულნი არიან ცხრილის უკიდურეს ზედა და უკიდურეს ქვედა რიგებში, ამასთან ისინი მდებარეობენ ორ შესაბამის სვეტში, ამიტომ კარნოს ცხრილის თვისების თანახმად ისინი შეიძლება გაერთიანდნენ ერთ კონტურში. ამ კონტურის შესაბამისი რიგებია $\bar{A}\,\bar{B}$ და $A\,\bar{B}$, ამ ორ კომბინაციაში იცვლება $A$ და $\bar{A}$ და შესაბამისად იგი ამოვარდება. დარჩება მხოლოდ $\bar{B}$. კონტურს აქვს ორი სვეტი $CD$ და $\bar{C}\,D$, სადაც იცვლება $C$ და $\bar{C}$, იგი ამოვარდება და დარჩება მხოლოდ $D$. საბოლოოდ მე-5 კონტურისთვის დაგვრჩება $\bar{B}\,D$.

გავაერთიანოთ ხუთივე კონტურისათვის მიღებული შედეგები და მივიღებთ მოცემული ლოგიკური ფუნქციის მინიმიზებულ გამოსახულებას (ფორმულა 16)

$$Y = \bar{A}\,B\,\bar{D} + A\,\bar{B} + A\,D + \bar{C}\,D + \bar{B}\,D \qquad (4.16)$$

თუ შევადარებთ ერთმანეთს 4.15 და 4.16 გამოსახულებებს აშკარად დავინახავთ მინიმიზაციის შედეგს. თუ 4.15 ლოგიკური გამოსახულების რეალიზაციისათვის გვჭირდებოდა 25 ლოგიკური ოპერაცია მინიმიზაციის შემდეგ საკმარისია მხოლოდ 13.

## 4.3. რთული ლოგიკური სქემების სინთეზი მინიმიზაციის გამოყენებით

როგორც გავარკვიეთ რთული ფუნქციების მიმართ მინიმიზაციის ალგებრული მეთოდისა ან/და კარნოს ბარათების მეთოდის გამოყენება მნიშვნელოვნად ამარტივებს რთული ლოგიკური გამოსახულების სინთეზს. იგივე შეიძლება ითქვას

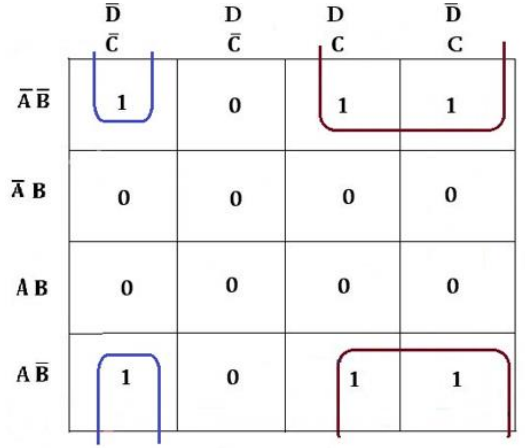
ჭეშმარიტების ცხრილების მიმართაც, რომლებიც ერთი ან რამდენიმე გამოსასვლელით არიან მოცემული. ასეთ შემთხვევაში მინიმიზაციისთვის ჭეშმარიტების ცხრილის მიმართ იყენებენ ჯერ კარნოს ბარათს (ან სხვა), შემდეგ კიდევ უფრო ამარტივებენ მიღებულ გამოსახულებას ალგებრული მეთოდით და საბოლოო ლოგიკური გამოსახულების მიხედვით გამოსახავენ ლოგიკურ სქემას.

გავაანალიზოთ კონკრეტული ამოცანა. მოცემული გვაქვს ჭეშმარიტების ცხრილი 4.5 ოთხი ცვლადითა (A, B, C, D) და ორი გამოსავლით (F1, F2). საჭიროა შევადგინოთ შესაბამისი ლოგიკური სქემა.

| A | B | C | D | F1 | F2 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 |

**ცხრილი 4.5. ჭეშმარიტების ცხრილი ოთხი ცვლადითა და ორი გამოსასვლელით.**

პრობლემის გადაწყვეტისათვის პირველ რიგში უნდა შევარჩიოთ მინიმიზაციის რომელი მეთოდი მივუყენოთ ჭეშმარიტების ცხრილს. როგორც ვხედავთ F1 გამოსასვლელის სვეტში გავქვს ექვსი „1“ და ათი „0“, ამიტომ მიზანშეწონილია აქ გამოვიყენოთ კარნოს ბარათი და სრულყოფილი **დიზუნქციური** ნორმალური ფორმა. ავაგოთ კარნოს ბარათი F1-სათვის (ნახ.4.7) და დავაჯგუფოთ „1“-ბი შესაბამისი წესის მიხედვით (იხილეთ პარაგრაფი 5.2.). მივიღებთ ნახ.4.7-ზე მოცემულ კარნოს ბარათს.




**ნახ.4.7. კარნოს ბარათი სრულყოფილი დიზუნქციური ნორმალური ფორმისათვის**

ნახ.4.7-ზე მოცემულ ცხრილში „1“-ების დაჯგუფება გვაძლევს 2-ელემენტიან და 4-ელემენტიან ორ ჯგუფს, საიდანაც 2-ელემენტიანისთვის გვექნება გამოსახულება: $\bar{B}*\bar{C}*\bar{D}$ და 4-ელემენტიანი ჯგუფისთვის გვექნება: $\bar{B}*C$. ამ ორი გამოსახულების დიზუნქცია გვაძლევს გამოსახულებას:

$$F1 = \bar{B}*\bar{C}*\bar{D} + \bar{B}*C \qquad (4.17)$$

4.17 გამოსახულებისათვის შეიძლება გამოვიყენოთ გამარტივების ალგებრული მეთოდი და გამოვიტანოთ ფრჩხილებს გარეთ $\overline{B}$. საბოლოოდ მივიღებთ:

$$F1 = \overline{B}(\overline{C} * \overline{D} + C) \qquad (4.18)$$

ცხადია ის უპირატესობა რაც მინიმიზაციის გამოყენებამ მოგვიტანა, რადგან პირდაპირი მეთოდით გამოსახულების ჩანაწერის შემთხვევაში გვექნებოდა ექვსი შესაკრები თითოეული ოთხი ცვლადის თანამამრავლებით ნაცვლად 4.18 მარტივი გამოსახულებისა.

გადავიდეთ F2 გამოსასვლელზე. მის სვეტში ვხედავთ, რომ გვაქვს ექვსი „0“ და ათი „1“, ამიტომ მიზანშეწონილია გამოვიყენოთ კარნის ბარათი და სრულყოფილი კონიუნქციური ნორმალური ფორმა. ავაგოთ კარნოს ცხრილი F2-სათვის (ნახ.4.8) და დავაჯგუფოთ „0“-ბი შესაბამისი წესის მიხედვით.

| | $\overline{D}$ $\overline{C}$ | D $\overline{C}$ | D C | $\overline{D}$ C |
|---|---|---|---|---|
| $\overline{A}\,\overline{B}$ | 0 | 0 | 1 | 1 |
| $\overline{A}$ B | 0 | 0 | 1 | 0 |
| A B | 1 | 1 | 1 | 0 |
| A $\overline{B}$ | 1 | 1 | 1 | 1 |

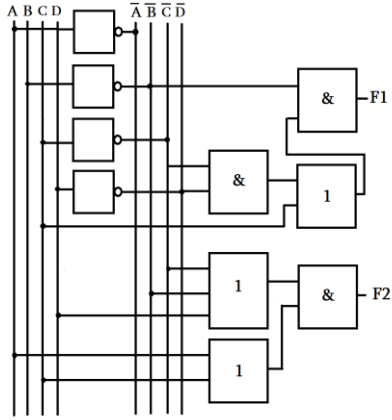
**ნახ.4.8. კარნოს ბარათი სრულყოფილი კონიუნქციური ნორმალური ფორმისათვის.**

ნახ.4.8-ზე მოცემულ ცხრილში „0“-ების დაჯგუფება გვაძლევს 2-ელემენტიან და 4-ელემენტიან ორ ჯგუფს. გამოსახულების შედგენის დროს უნდა გამოვიყენოთ შესაბამისი ცვლადების ინვერსია. 2-ელემენტიანი ჯგუფისთვის გვექნება გამოსახულება: $\overline{B}+\overline{C}+D$ და 4-ელემენტიანი ჯგუფისთვის გვექნება: A+C. საბოლოო კონიუნქცია გვაძლევს:

$$\mathbf{F2 = (\overline{B} + \overline{C} + D) * (A + C)} \qquad \mathbf{(4.19)}$$

ამ გამოსახულების მიხედვითაც ცხადია ის უპირატესობა რაც მინიმიზაციის გამოყენებამ მოგვცა **F2** გამოსახულებისათვის, რადგან პირდაპირი მეთოდით გამოსახულების ჩანაწერის შემთხვევაში გვექნებოდა ექვსი თანამამრავლი თითოეული ოთხი ცვლადის ჯამით, რასთან შედარებით 4.19 გამოსახულების უპირატესობა აშკარაა.

ბოლო ეტაპზე ავაგოთ 4.18 და 4.19 გამოსახულებების შესაბამისი ლოგიკური სქემა (ნახ.4.9)




**ნახ. 4.9. 4.18 და 4.19 გამოსახულებების შესაბამისი ლოგიკური სქემა**

## 4.4. ლოგიკური სქემების სინთეზი არასრული ჭეშმარიტების ცხრილის მიხედვით.

კომპიუტერული ლოგიკისა და ციფრული ელექტრონიკისათვის საინტერესოა კიდევ ერთი მნიშვნელოვანი პრობლემა, შემთხვევა როდესაც რომელიმე ლოგიკური სქემის გამოსასვლელზე ნაწილი სიგნალებისა განსაზღვრულია, ხოლო რაღაც ნაწილი არ არის განსაზღვრული ანუ უცნობია. სხვანაირად რომ ვთქვათ გვაქვს ჭეშმარიტების ცხრილი როდესაც გამოსავალი ინფორმაციის სვეტში ნაწილი მონაცემები კონკრეტულად არის მოცემული - ან „1“ ან „0“, ხოლო ნაწილი მონაცემებისა კი არ არის მოცემული კონკრეტულად (იხილეთ ცხრ.4.6.). ან მოცემული გვაქვს ჭეშმარიტების ცხრილის მხოლოდ ნაწილი.

| A | B | C | D | F1 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | x |
| 0 | 0 | 1 | 1 | x |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | x |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | x |
| 1 | 0 | 0 | 1 | x |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | x |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**ცხრილი 4.6. ჭეშმარიტების ცხრილი არასრული მონაცემებით**

ჭეშმარიტების ცხრილში უცნობი გამოსავალი ინფორმაცია x-ით არის აღნიშნული. სხვაგვარად რომ ვთქვათ გვაქვს არასრული ჭეშმარიტების ცხრილი. ამოცანა მდგომარეობს იმაში, რომ აღნიშნული არასრული მონაცემებით უნდა ვიპოვოთ ლოგიკური გამოსახულება ან/და შევადგინოთ ლოგიკური სქემა.

პრობლემის გადაჭრისათვის უნდა განვსაზღვროთ, რომ ცხრილში მოცემული კონკრეტული მონაცემები ზუსტად უნდა აისახებოდეს საძიებო ლოგიკური გამოსახულებით, ხოლო განუსაზღვრელი x მონაცემები გავლენას ვერ უნდა ახდენდნენ უკვე განსაზღვრულ მოცემულობებზე. ამიტომ ცხადია, რომ რა ორობითი მნიშვნელობებიც არ უნდა მიიღოს x-მა, ანუ **x-ის ნებისმიერი მნიშვნელობებისათვის, უკვე მოცემული გამოსავალი მონაცემების საფუძველზე შედგენილი გამოსახულება ზეგავლენას ვერ მოახდენს კონკრეტულ მოცემულ გამოსავალ მნიშვნელობებზე.** ეს თვისება მნიშვნელოვნად ამარტივებს პრობლემის გადაჭრის მეთოდს, რადგან საშუალებას გვაძლევს უცნობი მონაცემები ჩვენი შეხედულებისამებრ შევცვალოთ „0“-ით ან „1“-ით. ცხადია, რომ x-ს შეცვლის დროს შეგვიძლია გავითვალისწინოთ გამოსახულების ოპტიმიზაციის სხვადასხვა მეთოდები, მათ შორის ალგებრული მეთოდი ან/და კარნოს ცხრილები.

პრაქტიკული თვალსაზრისით პრობლემის გადაწყვეტას მნიშვნელოვნად გაამარტივებს ვარიანტი, როდესაც ჯერ ცხრილით მოცემული მონაცემებისთვის შევადგენთ კარნოს ცხრილს, ხოლო შედეგ უცნობ მონაცემებს, x-ებს, შევცვლით „1“-ით ან „0“-ით იმისდა მიხედვით თუ რომელი უფრო მომგებიანი იქნება საბოლოო გამოსახულების დადგენისათვის, ანუ უნდა შევეცადოთ „0“-ბი და „1“-ბი ჩაიწეროს იქ სადაც „0“-ბის ან „1“-ბის უფრო დიდი დაჯგუფების შედგენა იქნება შესაძლებელი რათა კარნოს

ბარათის პრინციპმა მაქსიმალურად ეფექტიანად იმოქმედეს (იხილეთ პარაგრაფი 4.2). შევადგინოთ კარნოს ბარათი ჭეშმარიტების ცხრილი 4.6-ის მიხედვით (ნახ.4.10).

| | $\bar{D}$ $\bar{C}$ | D $\bar{C}$ | D C | $\bar{D}$ C |
|---|---|---|---|---|
| $\bar{A}\,\bar{B}$ | 1: 0 | 2: 0 | 4: X | 3: X |
| $\bar{A}\,B$ | 5: 0 | 6: X | 8: 0 | 7: 1 |
| $A\,B$ | 13: 1 | 14: X | 16: 1 | 15: 1 |
| $A\,\bar{B}$ | 9: X | 10: X | 12: 1 | 11: 1 |

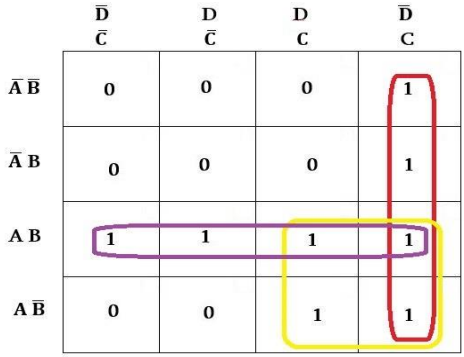
**ნახ.4.10. კარნოს ბარათი არასრული მონაცემებით**

ნახ.4.10-ზე შედგენილი კარნოს ბარათი მსჯელობის გამარტივების მიზნით დავნომროთ. უნდა შევარჩიოთ ორი მეთოდიდან ერთ-ერთი, სრულყოფილი დიზუნქციური ნორმალური ფორმა თუ სრულყოფილი კონიუნქციური ნორმალური ფორმა? ცხადია, რომ თუ მე-3 უჯრაში x-ს შევცვლით 0-ით, მე-4 - „1“, მე-6 – „0“,მე-10 – „1“, მე-13 – „0“ და მე-14 – „0“ მივიღებთ ნახ.4.11 ზე გამოსახულ კარნოს ბარათს, რომელის მიმართაც შესაძლებელი იქნება სრულყოფილი **დიზუნქციური** ნორმალური ფორმის მიყენება „1“-ბის მაქსიმალურად დიდი ჯგუფების შედგენის თვალსაზრისით (ნახ. 4.12). ცხადია x-ების ჩანაცვლება სხვაგვარადაც შეიძლება და დამოკიდებულია ჩვენს ხედვაზე კარნოს ბარათის მიმართ. ამ შემთხვევაში საბოლოო

გამოსახულება განსხვავებული იქნება, მაგრამ ჭეშმარიტების ცხრილში მოცემული უკვე განსაზღვრული ცვლადების მიმართ განსხვავებული გამოსახულება ზუსტად იგივე შედეგს მოგვცემს რასაც ჩვენს მიერ შერჩეული ჩანაცვლებების შემდეგ მიღებული საბოლოო გამოსახულება.

| | $\bar{D}\,\bar{C}$ | $D\,\bar{C}$ | $D\,C$ | $\bar{D}\,C$ |
|---|---|---|---|---|
| $\bar{A}\,\bar{B}$ | 0 | 0 | 0 | 1 |
| $\bar{A}\,B$ | 0 | 0 | 0 | 1 |
| $A\,B$ | 1 | 1 | 1 | 1 |
| $A\,\bar{B}$ | 0 | 0 | 1 | 1 |

ნახ.4.11. კარნოს ბარათი არასრული მონაცემებით უცნობი ინფორმაციის ჩანაცვლების შემდეგ

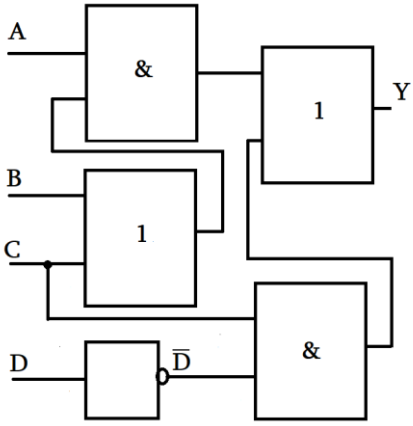



ნახ.4.12. კარნოს ბარათში „1“-ის დაჯგუფება

ნახ.4.12-ზე გამოსახულ კარნოს ბარათს მივუყენოთ სრულყოფილი დიზუნქციური ნორმალური ფორმის მეთოდი და მივიღებთ ლოგიკურ გამოსახულებას:

$$Y = A * B + C * \overline{D} + A*C = A\,(B + C) + C * \overline{D} \qquad (4.20)$$

ცხადია 4.20 გამოსახულება დამატებით გავამარტივეთ ალგებრული მეთოდით. ბევრ შემთხვევაში კარნოს ბარათის გამოყენების შემდეგ მინიმიზაციისათვის მიზანშეწონილია ალგებრული მეთოდის გამოყენებაც, რაც კიდევ უფრო ამარტივებს ასაგებ ლოგიკურ სქემას. ამოცანის დასასრულს ავაგოთ ლოგიკური სქემა 4.20 ფორმულის მიხედვით (იხ.ნახ.4.13). მოცემული სქემა შეასრულებს არასრული ჭეშმარიტების ცხრილი 4.6-ის მიხედვით მოცემული კონკრეტული განსაზღვრული მონაცემების რეალიზაციას.




**ნახ.4.13. არასრული ჭეშმარიტების ცხრილის 4.6 რეალიზაციის ლოგიკური სქემა.**

განვიხილოთ შემთხვევა, როდესაც მოცემული გვაქვს ჭეშმარიტების ცხრილის მხოლოდ **ფრაგმენტი** სამი ცვლადისათვის (ცხრილი 4.7), მაშინ ცხრილის დარჩენილი თუ გამოტოვებული ნაწილის უცნობი გამოსავალი მონაცემები ჩაითვლება როგორც უცნობი x მონაცემები და ცხრილის აღდგენის შემდეგ რეალურად მივიღებთ ჭეშმარიტების ცხრილს არასრული მონაცემებით.

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

**ცხრილი 4.7. ჭეშმარიტების ცხრილის ფრაგმენტი**

ჭეშმარიტების ცხრილის აღდგენის დროს ყურადღება უნდა მიექცეს იმას, რომ შესავალი ცვლადების კომბინაცია უნდა დალაგდეს დადგენილი წესის მიხედვით და ამის შემდეგ მიენიჭოს გამოსავალ ინფორმაციას უცნობის სტატუსი. ჩვენს შემთხვევაში ჭეშმარიტების ცხრილი შემდეგ სახეს მიიღებს (ცხრილი 4.8).

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | X |
| 0 | 1 | 0 | X |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | X |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | X |

**ცხრილი 4.8. ფარგმენტიდან აღდგენილი ჭეშმარიტების ცხრილი არასრული მონაცემებით**

ფრაგმენტიდან ჭეშმარიტების ცხრილის აღდგენის შემდეგ მივიღებთ ჭეშმარიტების ცხრილს არასრული მონაცემებით. ამ მომენტიდან საბოლოო ლოგიკური გამოსახულების გამოსაკვლევად და/ან ლოგიკური სქემის შესადგენად უნდა შესრულდეს პროცედურები, რომლებიც წინა განვიხილეთ მაგალითში.

# თავი 5. კომპიუტერის ციფრული ელემენტები

## 5.1. ნახევარამჯამავი

ნახევარამჯამვი წარმოადგენს მოწყობილობას, რომელიც აჯამებს (კრებს) მხოლოდ ერთი თანრიგის ორ ორობით რიცხვს. ორი ორობითი რიცხვის შეკრების ოპერაციის ჭეშმარიტების ცხრილი ცხრილ 5.1-შია მოყვანილი, სადაც S ორობითი რიცხვების ჯამია, ხოლო C – „1“-ის შემდეგ თანრიგში გადატანა.

| A | B | C | S |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

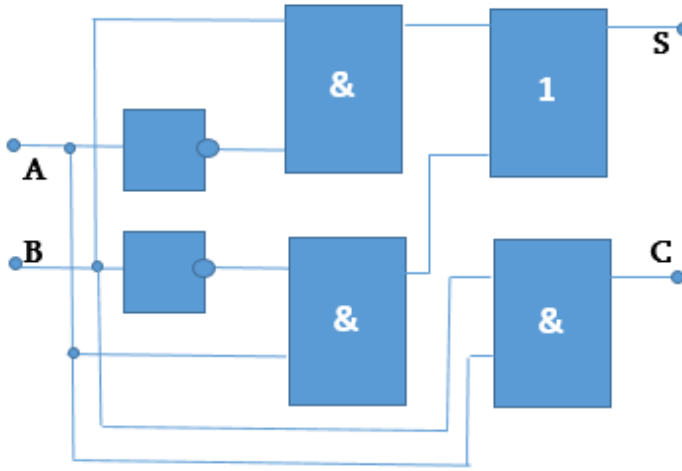
**ცხრილი 5.1. ნახევარამჯამავის ჭეშმარიტების ცხრილი**

აღნიშნული ჭეშმარიტების ცხრილიდან S და C-სთვის შეგვიძლია დავწეროთ შესაბამისი ლოგიკური გამოსახულებები:

$$S = \overline{A}\,B + \overline{B}\,A \qquad (5.1)$$

$$C = A\,B \qquad (5.2)$$

მიღებული ლოგიკური გამოსახულების საფუძველზე შეიძლება ავაგოთ ლოგიკური სქემა (ნახ.5.1)



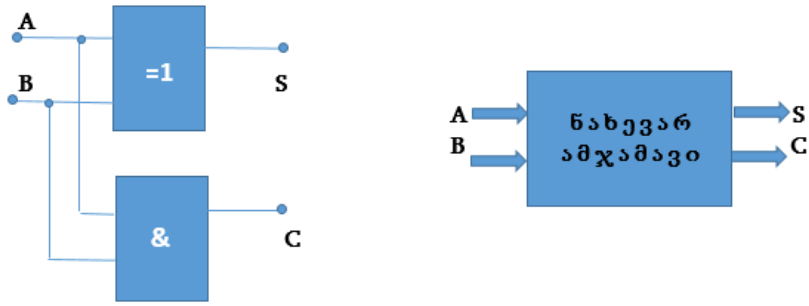

ნახ.5.1. ნახევარამჯამავის ლოგიკური სქემა

მეორეს მხრივ, თუ გამოვიყენებთ ბულის ალგებრას და ლოგიკური გამოსახულებების ოპტიმიზაციის მეთოდს დავინახავთ, რომ (5.1) გამოსახულება წარმოადგენს ფუნქციას „ჯამი 2-ის მოდულით“, ხოლო (2) გამოსახულება დარჩება უცვლელი, ანუ:

$$S = \bar{A}\,B + \bar{B}\,A = A \oplus B \qquad (5.3)$$

$$C = A\,B \qquad (5.4)$$

ამ ორი გამოსახულების საფუძველზე აგებული სქემა წარმოადგენს ნახევარამჯამავს და იგი მოყვანილია ნახაზზე (ნახ.5.1).




**ნახ.5.2. ნახევარამჯამავი. ლოგიკური სქემა ოპტიმიზაციის გამოყენებით და სიმბოლური აღნიშვნა.**

## 5.2 ამჯამავი

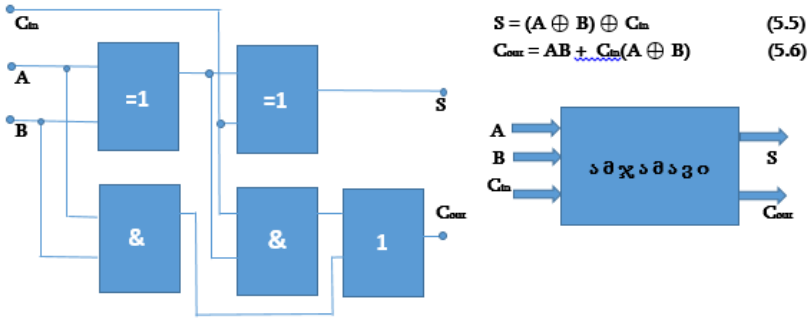
ამჯამავი (adder) წარმოადგენს მოწყობილობას რომელიც გამოიყენება ნებისმიერი თანრიგის ორობითი რიცხვების შეკრებისათვის (აჯამვისათვის). ასეთი მოწყობილობის მოქმედების პრინციპის გასარკვევად განვიხილოთ კონკრეტული მაგალითი: შევკრიბოთ ორი ოთხთანრიგიანი ორობითი რიცხვი $1011_2$ და $1101_2$

$$
\begin{array}{l}
\phantom{+}\ 1\ 1\ 1 \\
\phantom{+}\ 1\ 0\ 1\ 1_2 \\
+ \\
\phantom{+}\ 1\ 1\ 0\ 1_2 \\
\hline
1\ 1\ 0\ 0\ 0_2
\end{array}
$$

ამ მაგალითიდან ჩანს, რომ ნულოვან თანრიგში ორი ორობითი რიცხვის შეკრების შემდეგ ($1+1=10_2$) ჯამში უნდა ჩაიწეროს „0“ ხოლო „1“ გადადის შემდეგ თანრიგში (მომდევნო

თანრიგში გადატანილი ციფრები შესაკრები ციფრების თავზე გვაქვს ჩაწერილი). ესე იგი შემდეგ თანრიგში გვექნება არა ორი შესაკრები, არამედ სამი, ანუ ორი ძირითადი და ერთი წინა თანრიგიდან გადმოტანილი და ასე შემდეგ. სწორედ ეს განსხვავებაა ნახევარამჯამავსა და ამჯამავს შორის. ნახევარ ამჯამავს აქვს 2 შესასვლელი A და B, ხოლო ამჯამავში კი შესაკრებთა რიცხვი სამია A, B და C, სადაც A და B ორობითი ციფრებია, ხოლო C - წინა თანრიგიდან გადმოტანილი „0“ ან „1“.

ამჯამავის ლოგიკური სქემის ასაგებად ჩავთვალოთ რომ A, B და C უშუალოდ პირდაპირ კი არ იკრიბებიან, არამედ ჯერ იკრიბება A და B ხოლო მიღებულ შედეგს შემდეგ ემატება C. ამ პროცედურით გავამარტივებთ ლოგიკურ სქემას იმ თვალსაზრისით, რომ თითოეული წყვილის შეკრების დროს გამოვიყენებთ ნახევარამჯამავს. ანუ ერთ ნახევარამჯამავს ვიყენებთ ჯერ A და B-ს შესაკრებად და მეორე ნახევარამჯამავით შევკრებთ მიღებულ შედეგს და C-ს. თუ გავითვალისწინებთ ამ მოსაზრებებს, მაშინ ერთთანრიგიანი ამჯამავის ლოგიკური სქემისათვის მივიღებთ სქემას რომელიც ნახ.5.3-ზეა მოყვანილია.



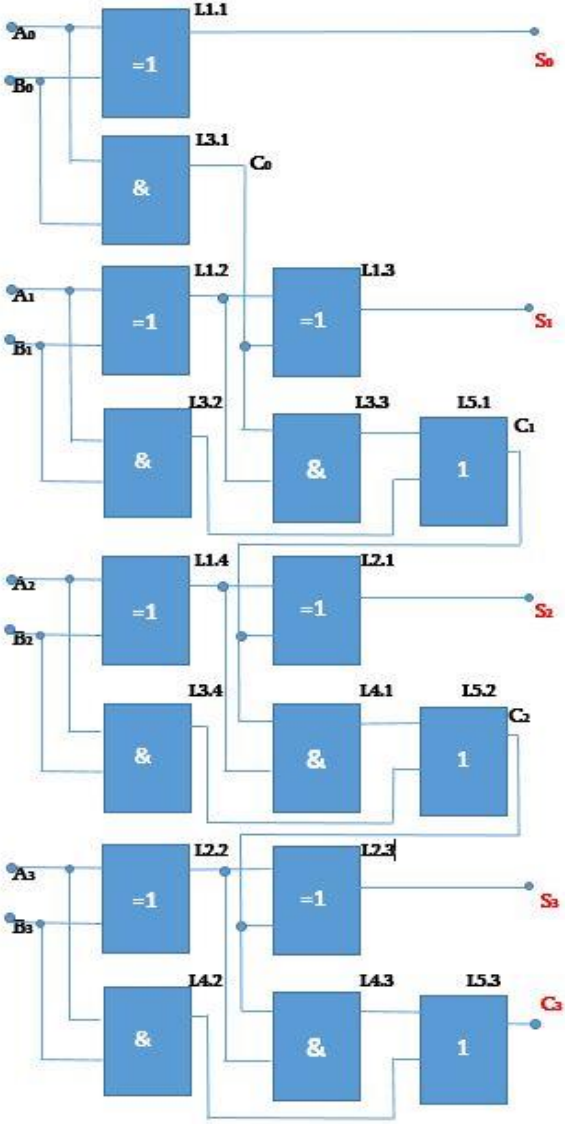

**ნახ.5.3. ერთთანრიგიანი ამჯამვის ლოგიკა, ლოგიკური სქემა და ბლოკსქემა**

| $C_{in}$ | A | B | S | $C_{out}$ |
|---|---|---|---|---|
| 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 |

**ცხრილი.5.2. ერთთანრიგიანი ამჯამავის ჭეშმარიტების ცხრილი**

სრული ოთხი თანრიგისათვის პარალელური ამჯამავის ლოგიკური სქემა ნახ.5.4-ზეა მოტანილი. აქვე აღვნიშნოთ, რომ ოთხთანრიგიანი ამჯამავის მიერ შეკრებილი საბოლოო შედეგის წაკითხვა უნდა მოხდეს შემდეგი სიმბოლოების მიხედვით: $C_3S_3S_2S_1S_0$.

ცხადია, რომ ნებისმიერო თანრიგის ორობითი რიცხვების შესაკრებად ამჯამავის საბოლოო სქემაში გვექნება იმ რაოდენობის ერთთანრიგიანი ამჯამავები რამდენ თანრიგიან რიცხვებთანაც ვმუშაობთ. აქვე მივაქციოთ ყურადღება იმას, რომ ნულოვან თანრიგში გვაქვს ნახევარამჯამავი, რადგან ნულოვან თანრიგში წინა პოზიციიდან გადმოტანა არ გვაქვს და იკრიბება მხოლოდ საწყისი ორი ორობითი რიცხვი. აქ შეიძლებოდა ამჯამავის გამოყენებაც შესავალი C გადმოტანის „0“ მნიშვნელობით.



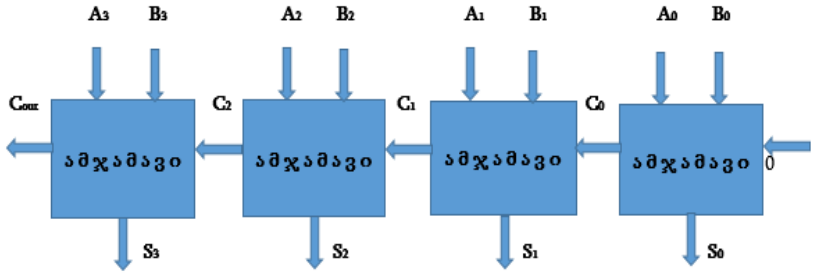

ნახ.5.4. 4-თანრიგიანი ორობითი რიცხვების ამჯამავის ლოგიკური სქემა

განვიხილოთ როგორ მუშაობს ოთხთანრიგიანი ამჯამავი კონკრეტულ მაგალითზე. განვიხილოთ მაგალითი რომელიც ზემოთ გვქონდა მოტანილი, ანუ უნდა შევკრიბოთ ორი ორობითი რიცხვი $1011_2$ და $1101_2$. ამჯამავის შესასვლელებს ორობითი ციფრები უნდა მივანიჭოთ უმცროსი თანრიგიდან დაწყებული მომდევნო მაღალი თანრიგისაკენ, ესე იგი: A0=1, B0=1, A1=1, B1=0, A2=0, B2=1, A3=1, B3=1. თუ გავითვალისწინებთ თითოეული ლოგიკური ელემენტის მოქმედების პრინციპს შესაბამისად მივიღებთ: L1.1 ლოგიკური ელემენტის გამოსასვლელზე გვექნება „0“, რაც იგივეა $S_0=0$, ხოლო L3.1 ის გამოსასვლელზე „1“ რაც იგივე, რომ გადატანა ნულოვანი პოზიციიდან პირველ პოზიციაზე $C_0=1$. ამასთან ეს მნიშვნელობა მოდებული იქნება L1.3 და L3.3-ის თითო შესასვლელზე. მეორეს მხრივ $A_1$ და $B_1$-ს მნიშვნელობებიდან გამომდინარე გამოსასვლელიბზე L1.2=1 და L3.2=0. ამ მნიშვნელობებისა და ადრე მიღებული შედეგის $C_0=1$ გათვალისწინებით შემდეგი ლოგიკური ელემენტებისათვის მივიღებთ: L1.3=$S_1=0$, L3.3=1 და L5.1=$C_1$=1. ასეთივე მსჯელობით დანარჩენი ელემენტებისათვის მივიღებთ: L2.1=$S_2=0$, L2.3=$S_3=1$, L5.3=$C_3=1$. წავიკითხოთ მიღებული შედეგები მაღალი თანრიგის მხრიდან გადატანის გათვალისწინებით - $11000_2$.

ნახ.5.5-ზე მოყვანილია ნახ.5.4-დან გამომდინარე 4-თანრიგიანი ამჯამავის ბლოკსქემა იმ განსხვავებით, რომ აქ ნულოვან თანრიგში ნაცვლად ნახევარამჯამავისა გამოყენებული სრული ამჯამავი C გადატანის „0“ მნიშვნელობით.

პრაქტიკული თვალსაზრისით ამჯამავის კონსტრუირება შესაძლებელია როგორც ცალკეული ლოგიკური ელემენტებისაგან, ასევე იგი მზდდება ერთი ინტეგრალური სქემის (IC) ჩიპის

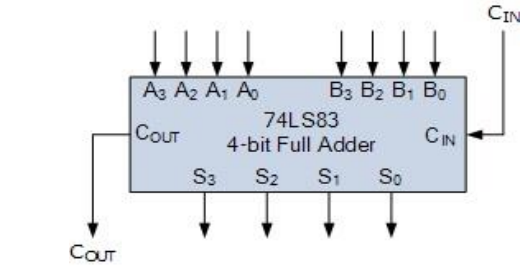
სახით, მაგალითად, ინტეგრალური სქემები **74LS83, 74LS283, CD4008.**

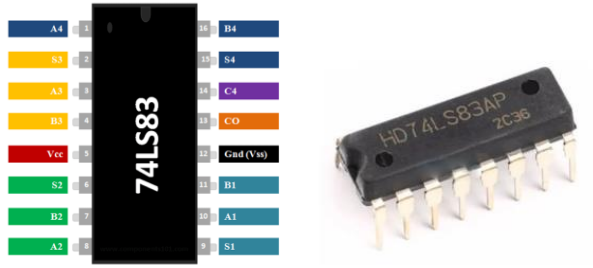



**ნახ.5.5. ოთხთანრიგიანი ამჯამავის ბლოკსქემა**

ეს ინტეგრალური სქემები წარმოადგენენ 4-ბიტიან ამჯამავებს. უფრო მაღალი ბიტიანი რიცხვების შემთხვევაში შეიძლება პარალელურად რამდენიმე მათგანის გამოყენება, რის შედეგადაც მიიღება 8, 12, 16 ბიტიანი ამჯამავები. ნახ.5.6-ზე მოცემულია ინტეგრალური მიკროსქემის სახით დამზადებული ამჯამავის ერთ-ერთი პოპულარული მოდელი **74LS83,** მისი ზოგადი სიმბოლური აღნიშვნა, პინების დანიშნულება და ფოტოსურათი. მიკროსქემის ტექნიკური მონაცემები, პინების დანიშნულება, კვებისა და სასიგნალო პინებზე მოდებული ძაბვების მნიშვნელობები და სხვა, ზოგადად მოცემულია მიკროსქემის ტექნიკურ დოკუმენტაციაში (DataSheet), რომელიც ან დაბეჭდილი ფიზიკურად ახლავს მიკროსქემას შეძენის დროს ან იძებნება მიკროსქემების ფართო სპექტრისათვის ინტერნეტში, მაგალითად საიტზე: https://www.alldatasheet.com.

კონკრეტულად **74LS83** მიკროსქემის პარამეტრები იძებნება საიტზე (ნახ.5.7):
https://pdf1.alldatasheet.com/datasheet-pdf/view/128899/FAIRCHILD/74LS83A.html



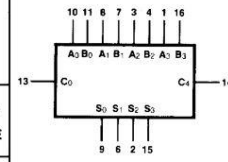

ნახ.5.6. ოთხთანრიგიანი ამჯამავი 74LS83

**DESCRIPTION** — The '83A high speed 4-bit binary full adders with internal carry lookahead accept two 4-bit binary words ($A_0$ — $A_3$, $B_0$ — $B_3$) and a Carry input ($C_0$). They generate the binary Sum outputs ($S_0$ — $S_3$) and the Carry output ($C_4$) from the most significant bit. They operate with either HIGH or active LOW operands (positive or negative logic). The '283 is recommended for new designs since it features standard corner power pins.

**ORDERING CODE:** See Section 9

| PKGS | PIN OUT | COMMERCIAL GRADE | MILITARY GRADE | PKG TYPE |
|---|---|---|---|---|
| | | $V_{CC}$ = +5.0 V ±5%, $T_A$ = 0°C to +70°C | $V_{CC}$ = +5.0 V ±10%, $T_A$ = -55°C to +125°C | |
| Plastic DIP (P) | A | 7483APC, 74LS83APC | | 9B |
| Ceramic DIP (D) | A | 7483ADC, 74LS83ADC | 5483ADM, 54LS83ADM | 6B |
| Flatpak (F) | A | 7483AFC, 74LS83AFC | 5483AFM, 54LS83AFM | 4L |

**LOGIC SYMBOL**



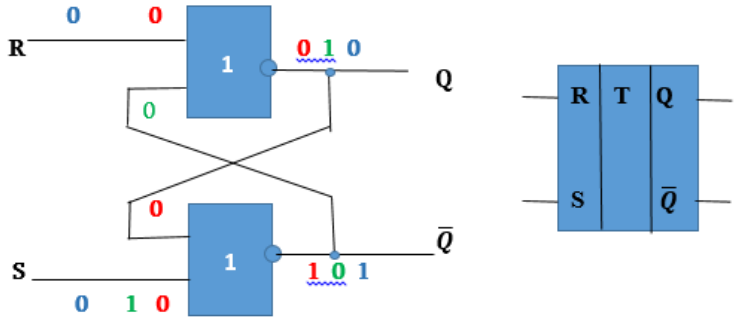

$V_{CC}$ = Pin 5
GND = Pin 12

ნახ.5.7. ფრაგმენტი 74LS83 მიკროსქემის ტექნიკური მონაცემებიდან

## 5.3. ტრიგერები

### ასინქრონული RS-ტრიგერი

ტრიგერი წარმოადგენს მოწყობილობას, რომელის მთავარი დანიშნულებაა ორობითი ციფრული სიგნალის დამახსოვრება. ისევე, როგორც უმრავლესი ციფრული მოწყობილობებისა, ტრიგერის რეალიზება შესაძლებელი საბაზო ლოგიკური ელემენტების საფუძველზე. განვიხილოთ ნახ.5.8-ზე მოტანილი სქემა, სადაც გამოყენებულია ორი ორშესასვლელიანი ლოგიკური ელემენტი 2ან-არა (NOR).




**ნახ.5.8. RS-ტრიგერის რეალიზაცია საბაზო ლოგიკურ ელემენტებზე 2ან-არა და RS-ტრიგერის პირობითი სქემატური აღნიშვნა**

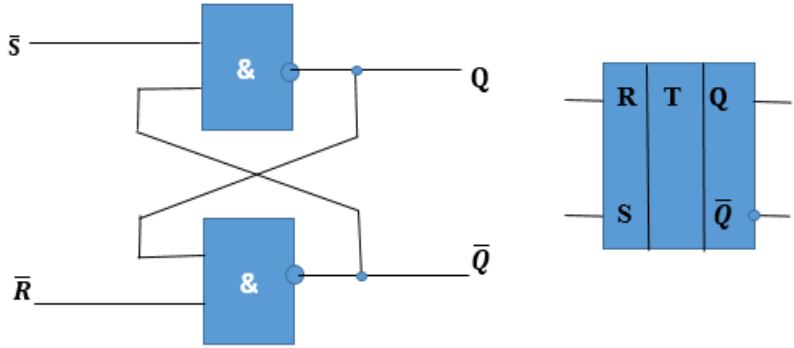
ამ მოწყობილობას RS-ტრიგერს ან ასინქრონულ RS-ტრიგერს უწოდებენ. გავაანალიზოთ როგორ იმუშავებს აღნიშნული სქემა შესასვლელებზე სხვადასხვა ორობითი სიგნალის მიწოდებისას. დავუშვათ საწყის მომენტში R (Reset) და S (Set) შესავლელებზე

სიგნალი არ მიეწოდება ანუ ორივე შესასვლელზე ლოგიკური ნულია, R=0 და S=0. თუ გავითვალისწინებთ 2ან-არა ელემენტის მოქმედების პრინციპს მივიღებთ, რომ პირდაპირი გამოსასვლელი Q=0, ამ შემთხვევაში მეორე ელემენტის ორივე შესასვლელზე გვექნება „0“ და ამიტომ $\bar{Q}$=1. დავუშვათ S შესასვლელზე მიეწოდება „1“, S=1, მაშინ, იგივე ლოგიკური ელემენტის მუშაობის პრინციპიდან გამომდინარე, ინვერსიულ გამოსასვლელზე მნიშვნელობა შეიცვლება $\bar{Q}$=0. ამიტომ პირველი ელემენტის ორივე შესასვლელზე გვექნება „0“ და ძირითად გამოსასვლელზე მივიღებთ „1“-ს, Q=1. ეს პროცესი არის „1“-ის ჩაწერის მომენტი, ანუ თუ S შესასვლელზე არსებულ „1“-ს შევცვლით“0“-ით გამოსავალი სიგნალები არ შეიცვლება, ესეიგი სიგნალი „1“ დაფიქსირდა ძირითად Q გამოსასვლელზე. ამგვარად მოხდა ინფორმაციის დამახსოვრება. ინფორმაციის წაშლისათვის საკმარისია მოვხსნათ S შესასვლელზე „1“ ანუ S =0 და R შესასვლელზე კვლავ მოვდოთ „0“, ანუ R=0, მაშინ ძირითადი გამოსასვლელი განულდება Q=0, ამ მომენტს ტრიგერის განულებას უწოდებენ. ტრიგერის ამ ვარიანტში არის ერთი აკრძალვა, რომელიც ეხება მის ორივე შესასვლელი ლოგიკური „1“ -ის მოდებას. არ შეიძლება RS-ტრიგერის ორივე შესასვლელზე ერთდროულად ლოგიკური „1“-ის მოდება. ამ შემთხვევაში ტრიგერი ნორმალურად ვერ იმუშავებს.

RS-ტრიგერის მოდელირება შესაძლებელია საბაზო ლოგიკური ელემენტების 2და-არა (NAND) ბაზაზეც. განვიხილოთ შესაბამისი სქემა (ნახ.5.9).

ამ სახით მოდელირებული ტრიგერის მართვა ხდება ლოგიკური „0“-ის საშუალებით განსხვავებით ზემოთ მოყვანილი „2ან-არა“ (NOR) ლოგიკური ელემენტების ბაზაზე მოდელირებული ტრიგერისა, სადაც მართვა ხდებოდა ლოგიკური

„1“-ის საშუალებით. სხვა შემთხვევაში ორივე ტრიგერის მუშაობის პრინციპები ერთნაირია.




**ნახ.5.9. RS-ტრიგერის რეალიზაცია საბაზო ლოგიკურ ელემენტებზე 2და-არა და RS-ტრიგერის პირობითი სქემატური აღნიშვნა.**

ტრიგერის მუშაობის პრინციპიდან გამომდინარე შეგვიძლია შევადგინოთ მისი ჭეშმარიტების ცხრილი (ცხრილი 5.3)

| R | S | Q(წინა) | Q | რეჟიმი |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | შენახვა |
| 0 | 0 | 1 | 1 | |
| 0 | 1 | 0 | 1 | 1-ის დაყენება |
| 0 | 1 | 1 | 1 | |
| 1 | 0 | 0 | 0 | 0-ის დაყენება |
| 1 | 0 | 1 | 0 | |
| 1 | 1 | 0 | X | აკრძალვა |
| 1 | 1 | 1 | X | |

**ცხრილი.5.3. RS-ტრიგერის ჭეშმარიტების ცხრილი**

ცხრილიდან ჩანს, რომ შესავალი R და S ნულოვანი მნიშვნელობის დროს Q გამოსასვლელზე რჩება იგივე მნიშვნელობა რაც მას წინა შემთხვევაში ჰქონდა. ანუ წარმოებს ინფორმაციის შენახვის რეჟიმი.

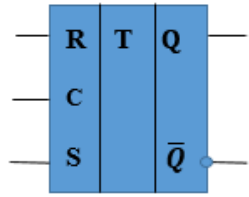
S შესასვლელზე ლოგიკურ „1“-ს მოდების შემთხვევაში, დამოუკიდებლად იმისა თუ რა მნიშვნელობა ქონდა Q-ს ადრე, გამოსასვლელზე დაფიქსირდება ლოგიკური „1“. ანუ მოხდება „1“-ის დაყენება, ან „1“-ის ჩაწერა.

R შესასვლელზე ლოგიკურ „1“-ს მოდების შემთხვევაში, დამოუკიდებლად იმისა თუ რა მნიშვნელობა ქონდა Q-ს ადრე, გამოსასვლელზე დაფიქსირდება ლოგიკური „0“. ანუ მოხდება „0“-ის დაყენება, ან „0“-ის ჩაწერა.

R და S შესასვლელებზე ერთდროულად ლოგიკურ „1“-ს მოდების შემთხვევაში, დამოუკიდებლად იმისა თუ რა მნიშვნელობა ქონდა Q-ს ადრე, გამოსასვლელზე გვექნება აკრძალული, ანუ დაუშვებელი შემთხვევა.

### სინქრონული RS-ტრიგერი

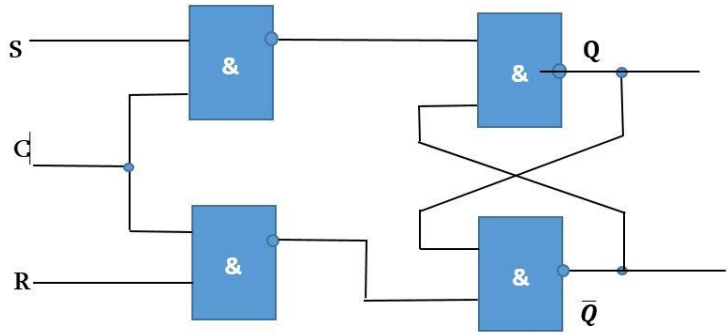
სინქრონული RS-ტრიგერი გამოირჩევა RS-ტრიგერისაგან მაღალი სტაბილურობით. ნახ.5.10-ზე მოყვანილია მისი სიმბოლური აღნიშვნა.




**ნახ.5.10. სინქრონული RS-ტრიგერის პირობითი სქემატური აღნიშვნა.**

სინქრონული RS-ტრიგერი რეაგირებს R და S შესასვლელებზე მოდებულ სიგნალებზე მხოლოდ და მხოლოდ იმ შემთხვევაში როდესაც C შესასვლელზე მოდებულია ლოგიკური „1“. ეს მომენტი მნიშნელოვანია იმ თვალსაზრისით, რომ ზოგადად R და S შესასვლელებზე სიგნალები შეიძლება მოხვდეს არაერთდროულად და გამოიწვიოს სქემის არასტაბილური მუშაობა. C სინქრონიზაციის შემთხვევაში კი ეს შემთხვევა არ გვექნება რადგან მხოლოდ C-ზე “1“-ის არსებობის შემთხვევაში მოხდება ტრიგერის მიერ R და S შესასვლელებზე სიგნალების აღქმა.

სინქრონული RS-ტრიგერის რეალიზაცია შესაძლებელია ასინქრონული RS-ტრიგერისა და ორი ლოგიკური 2და-არა ელემენტის გამოყენებით (ნახ.5.11.)




**ნახ.5.11. სინქრონული RS-ტრიგერის საბაზო ლოგიკური ელემენტების ბაზაზე რეალიზაციის სქემა**

სინქრონული RS-ტრიგერისათვის შევადგინოთ ჭეშმარიტების ცხრილი.

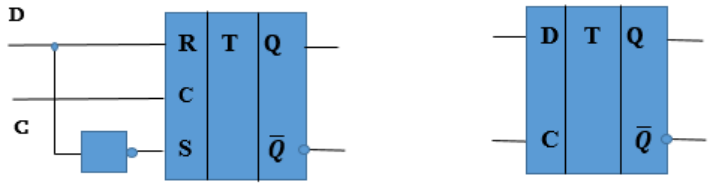
| C | R | S | Q (წინა) | Q | რეჟიმი |
|---|---|---|---|---|---|
| 0 | X | X | 0 | 0 | შენახვა |
| 0 | X | X | 1 | 1 | |
| 1 | 0 | 0 | 0 | 0 | შენახვა |
| 1 | 0 | 0 | 1 | 1 | |
| 1 | 0 | 1 | 0 | 1 | „1“-ის ჩაწერა |
| 1 | 0 | 1 | 1 | 1 | |
| 1 | 1 | 0 | 0 | 0 | „0“-ის ჩაწერა |
| 1 | 1 | 0 | 1 | 0 | |
| 1 | 1 | 1 | 0 | X | აკრძალვა |
| 1 | 1 | 1 | 1 | X | |

**ცხრილი 5.4. სინქრონული RS-ტრიგერის ჭეშმარიტების ცხრილი.**

ცხრილიდან ჩანს, რომ სინქრონული RS-ტრიგერის C სინქრონიზაციის შესასვლელზე ლოგიკური „0“-ის არსებობის შემთხვევაში დამოუკიდებლად იმისა თუ რა მნიშვნელობა არის R და S შესასვლელებზე ან იცვლება თუ არა ეს მნიშვნელობები, Q გამოსასვლელზე ინფორმაცია არ იცვლება, რაც ნიშნავს იმას, რომ ამ შემთხვევაში ხდება ინფორმაციის შენახვა. ყველა დანარჩენ შემთხვევაში სინქრონიზაციის სიგნალი „1“-ია და სინქრონული ტრიგერი მუშაობს ისევე როგორც ზემოთ განხილული ასინქრონული ტრიგერი.

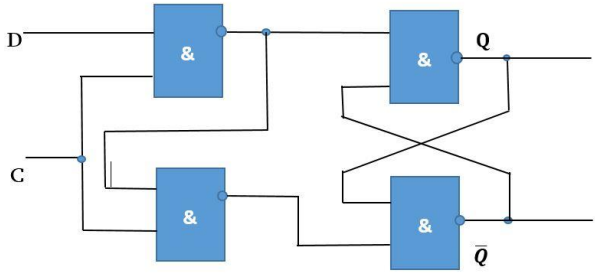
### D-ტრიგერი

D-ტრიგერს სახელი ეწოდა ინგლისური სიტყვა delay (შეყოვნება) საფუძველზე. განვიხილოთ D-ტრიგერის სქემა სინქრონული RS-ტრიგერის საფუძველზე და მისი პირობითი სქემატური აღნიშვნა (ნახ.5.12)




**ნახ.5.12 D-ტრიგერის სქემა სინქრონული RS-ტრიგერის საფუძველზე და მისი პირობითი სქემატური აღნიშვნა**

D-ტრიგერს აქვს ორი შესასვლელი: საინფორმაციო D (Data) და შესასვლელი სინქრონიზაციის სიგნალისათვის C. D-ტრიგერს ზოგ შემთხვევაში ინფორმაციის შენახვის ტრიგერსაც უწოდებენ. მისი აგება საბაზო ლოგიკური ელემენტების ბაზაზეცაა შესაძლებელი და D-ტრიგერის ეს თავისებურება მდგომარეობს იმაში, რომ საინფორმაციო D შესასვლელიდან ტრიგერში ინფორმაციის ჩაწერა ხდება მხოლოდ მაშინ, როცა სინქრონიზაციის C შესასვლელზე მოდებულია შესაბამისი სიგნალი.




**ნახ.5.13 D-ტრიგერი საბაზო ლოგიკური ელემენტების ბაზაზე.**

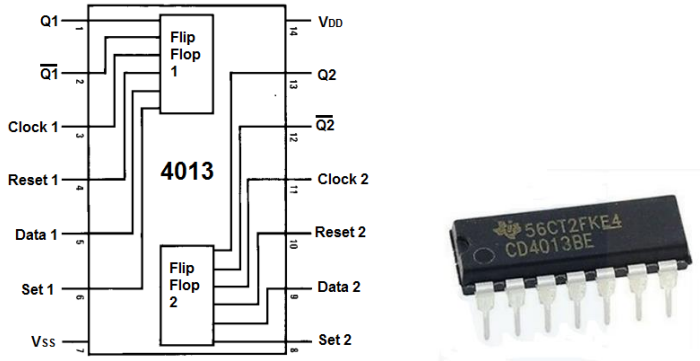
ანუ როცა C შესასვლელზე სინქროსიგნალი არ არის D შესასვლელზე სიგნალების ცვლილება ტრიგერის მდგომარეობაზე ზეგავლენას ვერ მოახდენს. D-ტრიგერი ყოველთვის

სინქრონულია. ასინქრონული D-ტრიგერის არ არსებობს. მოვიყვანოთ D-ტრიგერის ჭეშმარიტების ცხრილი (ცხრილი 5.5).

| C | D | Q (წინა) | Q | რეჟიმი |
|---|---|---|---|---|
| 0 | - | Q (წინა) | Q (წინა) | შენახვა |
| 1 | 1 | - | 1 | “1“-ის ჩაწერა |
| 1 | 0 | - | 0 | „0“-ის ჩაწერა |

**ცხრილი 5.5. D-ტრიგერის ჭეშმარიტების ცხრილი**

ცხრილიდან ვხედავთ, რომ თუ სინქროსიგნალი არ არის C=0, მაშინ Q გამოსასვლელზე დარჩება იგივე სიგნალი რაც მასზე არის მოცემულ მომენტში, მიუხედავად იმისა იცვლება თუ არა სიგნალი D შესასვლელზე. როდესაც სინქროსიგნალი C=1 და მონაცემთა D=1, მაშინ Q=1, ე.ი. ტრიგერში ჩაიწერება „1“, ხოლო თუ ამ მომენტში D=0, მაშინ Q=0, ე.ი. ტრიგერში ჩაიწერება „0“.



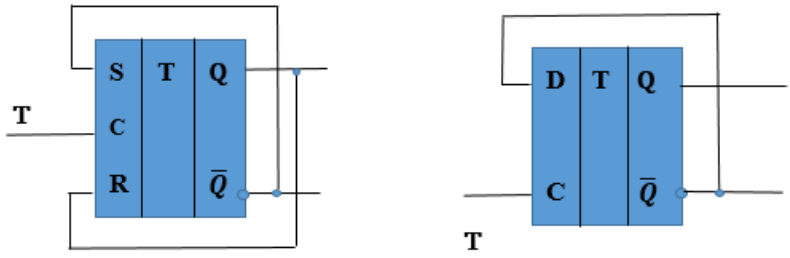

**ნახ. 5.14. ორი ერთეული D-ტრიგერი მიკროსქემაში CD4013.**

D-ტრიგერის ძირითადი უპირატესობა RS-ტრიგერთან შედარებით მდგომარეობს იმაში, რომ ამ მოწყობილობაში მონაცემების ჩაწერა უფრო მოსახერხებელია.

### T-ტრიგერი

T-ტრიგერები განკუთვნილნი არიან თვლისათვის და მათ გააჩნიათ მხოლოდ ერთი შესავალი. მისი რეალიზება შესაძლებელია როგორც D-ტრიგერის ასევე RS-ტრიგერის საფუძველზე (ნახ.5.15).

T-ტრიგერის მოქმედების პრინციპი შემდეგია. მის T შესასვლელზე სიგნალის მოსვლის შემთხვევაში ტრიგერის შიდა მდგომარეობა იცვლება საწინააღმდეგოთი, ე.ი. თუ იყო „0“ გახდება „1“ და პირიქით, თუ იყო „1“ გახდება „0“. ამ პროცესის გამო მას მთვლელ ტირგერს უწოდებენ, ანუ ყოველი იმპულსის მოსვლის შემდეგ ტრიგერი იცვლის შიდა მდგომარეობას იმპულსების რაოდენობის შესაბამისად.




**ნახ.5.15. T-ტრიგერების ტეალიზება RS-ტრიგერისა და D-ტრიგერების ბაზაზე.**

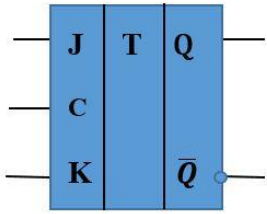
განვიხილოთ T-ტრიგერის ჭეშმარიტების ცხრილი (ცხრილი 5.6). როდესაც სასიგნალო T შესასვლელზე სიგნალი არ არის T=0, მაშინ Q გამოსასვლელზე მნიშვნელობა არ იცვლება და რჩება უცვლელად, ხოლო T შესასვლელზე ყოველი შემდეგი „1“-ის მოდების შემთხვევაში გამოსავალი შეიცვლის მნიშვნელობას საწინააღმდეგოთი.

| T | Q (წინა) | Q | რეჟიმი |
|---|---|---|---|
| 0 | 1 | 1 | შენახვა |
| 0 | 0 | 0 | |
| 1 | 1 | 0 | გადართვა |
| 1 | 0 | 1 | |

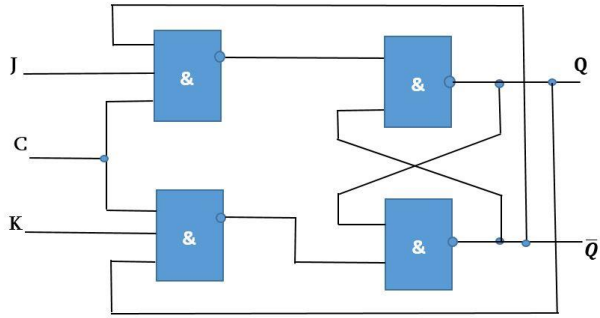
ცხრილი 5.6. T-ტრიგერის ჭეშმარიტების ცხრილი

### JK-ტრიგერი

ეგრეთ წოდებული JK-ტრიგერი უნივერსალურ ტრიგერად ითვლება. ფაქტიურად იგი წარმოადგენს RS-ტრიგერიგერის მოდიფიკაციას აკრძალული რეჟიმის პრობლემის გადასაწყვეტად. იმისათვის, რომ აღნიშნული აკრძალვა მოიხსნას RS-ტრიგერიგერის სქემა გადაკეთებულია ნახაზ 5.16-ზე მოყვანილი სახით.




ნახ.5.16 JK-ტრიგერის პირობითი სქემატური აღნიშვნა.




ნახ.5.17. JK-ტრიგერის რეალიზება საბაზო ლოგიკური ელემენტების ბაზაზე.

| C | J | K | Q(წინა) | Q | რეჟიმი |
|---|---|---|---|---|---|
| 0 | - | - | 0 | 0 | შენახვა |
| 0 | - | - | 1 | 1 | |
| 1 | 0 | 0 | 0 | 0 | |
| 1 | 0 | 0 | 1 | 1 | |
| 1 | 0 | 1 | - | 1 | „1“-ის ჩაწერა |
| 1 | 1 | 0 | - | 0 | „0“-ის ჩაწერვა |
| 1 | 1 | 1 | 0 | 1 | თვლის რეჟიმი |
| 1 | 1 | 1 | 1 | 0 | |

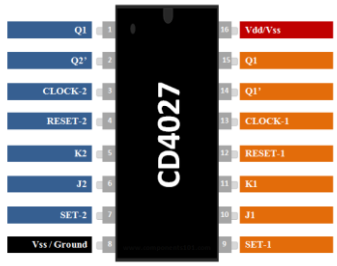
**ცხრილი 5.7. JK-ტრიგერის ჭეშმარიტების ცხრილი**

ცხრილიდან ჩანს, რომ ტრიგერის C სინქრონიზაციის შესასვლელზე ლოგიკური „0“-ის მოდების შემთხვევაში დამოუკიდებლად იმისა თუ რა მნიშვნელობა არის J და K შესასვლელებზე ან იცვლება თუ არა ეს მნიშვნელობები, Q გამოსასვლელზე ინფორმაცია არ იცვლება, რაც ნიშნავს იმას, რომ ამ შემთხვევაში ხდება ინფორმაციის შენახვა.

ტრიგერის C სინქრონიზაციის შესასვლელზე ლოგიკური „1“-ის მოდების შემთხვევაში, როცა J და K მნიშვნელობები „0“-ია, Q გამოსასვლელზე ინფორმაცია კვლავ არ იცვლება, რაც ნიშნავს იმას, რომ ამ შემთხვევაშიც ხდება ინფორმაციის შენახვა.

ყველა დარჩენილ შემთხვევაში სინქრონიზაციის სიგანალი“1“-ის ტოლია C=1. როდესაც J=0 და K=1 გამოსავალი Q მიიღებს „1“-ის ტოლ მნიშვნელობას იმის მიუხედავად თუ რა მნიშვნელობა ჰქონდა მას წინა სიტუაციაში, ანუ მოხდება „1“-ისჩაწერა. როდესაც J=1 და K=0 გამოსავალი Q მიიღებს „0“-ის ტოლ მნიშვნელობას იმის მიუხედავად თუ რა მნიშვნელობა ჰქონდა მას წინა სიტუაციაში. ანუ მოხდება „0“-ის ჩაწერა.

როდესაც სინქრონიზაციის სიგნალი „1“-ია და ორივე შესასვლელზე „1“-ის ტოლი სიგნალია, ტრიგერი თვლის რეჟიმშია და მისი გამოსასვლელის მნიშვნელობა იცვლება რიგრიგობით შესასვლელზე მიწოდებული „1“-ის ტაქტში.




### Features

- High Voltage Type (20V Rating)
- Set - Reset Capability
- Static Flip-Flop Operation - Retains State Indefinitely with Clock Level Either "High" or "Low"
- Medium Speed Operation - 16MHz (typ.) Clock Toggle Rate at 10V
- Standardized Symmetrical Output Characteristics
- 100% Tested For Quiescent Current at 20V
- Maximum Input Current of 1μA at 18V Over Full Package-Temperature Range;
  - 100nA at 18V and +25°C
- Noise Margin (Over Full Package Temperature Range):
  - 1V at VDD = 5V
  - 2V at VDD = 10V
  - 2.5V at VDD = 15V
- 5V, 10V and 15V Parametric Ratings
- Meets All Requirements of JEDEC Tentative Standard No. 13B, "Standard Specifications for Description of 'B' Series CMOS Devices"

### Applications

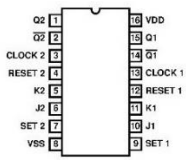
- Registers, Counters, Control Circuits

### Description

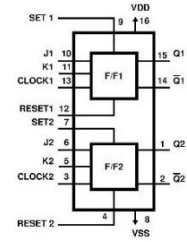
CD4027BMS is a single monolithic chip integrated circuit containing two identical complementary-symmetry J-K master-slave flip-flops. Each flip-flop has provisions for individual J, K, Set Reset, and Clock input signals. Buffered Q and $\overline{Q}$ signals are provided as outputs. This input-output arrangement provides for compatible operation with the Intersil CD4013B dual D type flip-flop.

### Pinout

CD4027BMS
TOP VIEW




### Functional Diagram




ნახ.5.18. JK-ტრიგერი CD4027-ის ტექნიკური პარამეტრები

JK-ტრიგერის გამოყენებით შეიძლება D-ტრიგერისა და T-ტრიგერის რეალიზაცია იმის მიხედვით თუ რა ამოცანასთან გვაქვს საქმე. მაგალითად, როგორც ჭეშმარიტების ცხრილიდან ჩანს, რადგან თვლის რეჟიმში J=1 და K=1, ამიტომ თუ JK-ტრიგერის J და K შესასვლელებს გავაერთიანებთ მივიღებთ T-ტრიგერს.

ნახ.5.18-ზე მოცემულია ინფორმაცია ერთ-ერთი გავრცელებული JK-ტრიგერის CD4027 შესახებ. მოცემულია მასში განლაგებული 2 ტრიგერის გამოსავალი და შესავალი კონტაქტების (პინების) შესახებ, მისი ფოტო და ინფორმაციის ნაწილი ტექნიკური პაარამეტრების შესახებ. CD4027 მიკროსქემის სრული ტექნიკური მონაცემები (datasheet) შეიძლება მოპოვებული იქნეს ინტერნეტ ვებ-გვერდზე:
www.radioradar.net/en/datasheets_search/C/D/4/CD4027_IntersilCorporation.pdf.html

განხილული ტრიგერების ყველა სახეობის დანიშნულებაა შეინახოს ერთი ბიტი ინფორმაცია. ტრიგერების გამოყენების ძირითადი დანიშნულებაა სხვა და სხვა დანიშნულების მახსოვრობის მოწყობილობების, მთვლელების, რეგისტრებისა და სხვა მოდელირება და კონსტრუირება.

## 5.4. რეგისტრი

რეგისტრი წარმოადგენს მოწყობილობას, რომელიც განკუთვნილია ორობითი რიცხვების მიღება, შენახვისა და გადაგზავნისათვის. ტრიგერისაგან განსხვავებით რეგისტრი მიიღებს, შეინახავს და გადააგზავნის არა მხოლოდ 0-ს და 1-ს არამედ ერთიანდ მათ გარკვეულ რაოდენობას (ორობითი სიტყვა, კოდი),

რომლებიც შეიძლება წარმოდგენილი იყოს როგორც მიმდევრობით (ერთი გამტარით) ასევე პარალელურად (რამდენიმე პარალელური გამტარით). შესაბამისად მათ უწოდებენ მიმდევრობით რეგისტრს და პარალელურ რეგისტრს. ანუ თუ ტრიგერმა შეიძლება შეინახოს 1 ბიტი ინფორმაცია, რეგისტრს შეუძლი შეინახოს ნახევარი ბაიტი, ერთი ბაიტი ან მეტი. რეგისტრები კომპიუტერის ძირითადი ბლოკების (მიკროპროცესორი, მახსოვრობები და სხვა) განუყოფელი ნაწილია. მათი წარმოება ხდება როგორც ცალკე მიკროსქემის სახით სხვადასხვა დანიშნულებისათვის, ასევე რეგისტრები ჩაშენებულია მიკროპროცესორებში, მახსოვრობებში და სხვა.

დანიშნულების მიხედვით რეგისტრების კლასიფიკაცია შემდეგია:

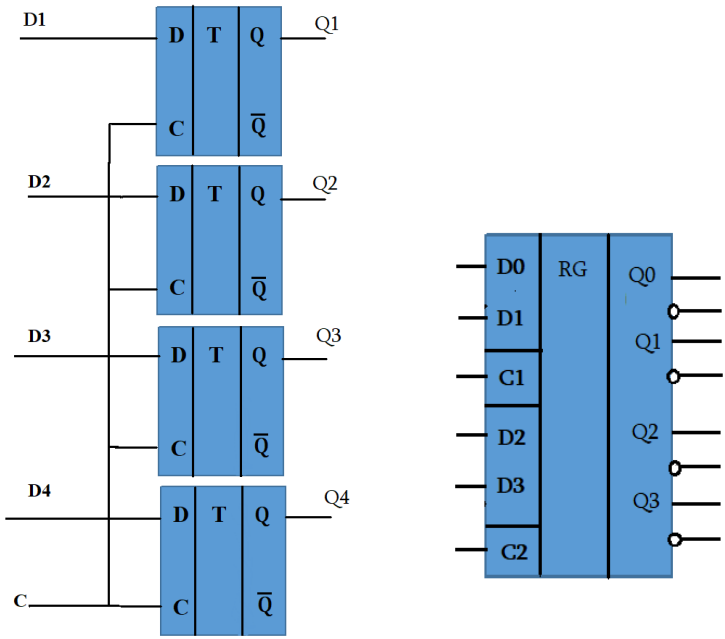
- **პარალელური რეგისტრი** - ანუ ინფორმაციის შენახვის რეგისტრი, ინფორმაციის შეტანა და გამოტანა წარმოებს ერთდროულად ყველა თანრიგის მიხედვით;
- **მიმდევრობითი რეგისტრი** - წანაცვლების რეგისტრი, ინფორმაცია თითოეული ბიტის ბიჯით გადაადგილდება რეგისტრში მიმდევრობით და გამოიტანება ასევე მიმდევრობით;
- **კომბინირებული რეგისტრი** - ინფორმაცია შედის პარალელურად და გამოდის მიმდევრობით, ან პირიქით.

### პარალელური რეგისტრი

პარალელური რეგისტრის დანიშნულებაა პარალელური კოდის (პარალელური ორობითი ინფორმაციის) მიღება, მიმდევრობით კოდში (მიმდევრობით ორობით ინფორმაციაში)

გადაყვანა და შენახვა. იგი ძირითადად გამოიყენება მოწყობილობებში რომლებიც ინფორმაციის გადაგზავნისათვის ერთ სასიგნალო ხაზს (გამტარს) იყენებენ. ასეთი ტრიგერის იყენებენ იმ შემთხვევაშიც, როდესაც შედარებით სწრაფი მოწყობილობიდან ინფორმაცია უნდა გადაეცეს შედარებით ნელ მოწყობილობას ერთი სასიგნალო ხაზის საშუალები, მაგალითად კომპიუტერიდან პრინტერს. როგორც მახსოვრობის უმეტესი მოწყობილობები ასევე პარალელური რეგისტრი ტრიგერების საფუძველზე იქმნება. განვიხილოთ ნახ.5.19-ზე წარმოდგენილი პარალელური რეგისტრის გამარტივებული სქემა, რომელიც სინქრონული D-ტრიგერის ბაზაზეა კონსტრუირებული. აქვე მოცემულია მისი სიმბოლური აღნიშვნაც.

D - ტრიგერზე აგებულ პარალელურ რეგისტრში შენახული რიცხვის კოდი მიეწოდება ოთხივე ტრიგერის D ინფორმაციულ შესასვლელებს და ჩაიწერება რეგისტრის C შესასვლელზე ტაქტური იმპულსის მოსვლის მომენტში. გამოსასვლელი ინფორმაცია იცვლება ახალი შესასვლელი კოდის მიწოდებით და ჩაწერისათვის განკუთვნილი შემდეგი C იმპულსის მოსვლის შემდეგ. მათში ტრიგერთა რაოდენობა ტოლია შესანახი სიტყვის მაქსიმალური თანრიგიანობის. ნახ.5.19.-ზე მოყვანილია ოთხთანრიგიანი პარალელური რეგისტრი. დამატებითი ტრიგერების გამოყენებით, რომელთა C შესასვლელები გაერთიანდება და სქემატური ჩართვა ნახაზზე მოცემული სქემის ანალოგურინიქნება, მივიღებთ უფრო მეტ თანრიგიან რეგისტრებს (8, 12,16 და ა. შ.). ამ ტიპის რეგისტრებს იყენებენ ოპერაციული მეხსიერების სისტემებში.

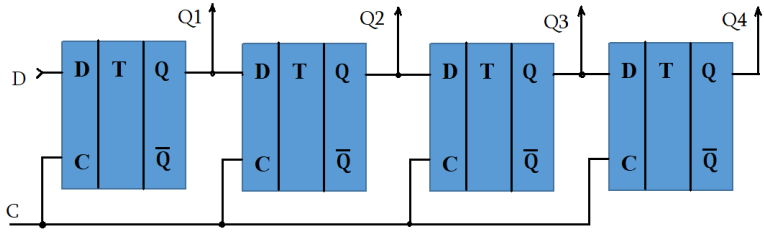



**ნახ.5.19. პარალელური რეგისტრის გამარტივებული სქემა და სიმბოლური აღნიშვნა.**

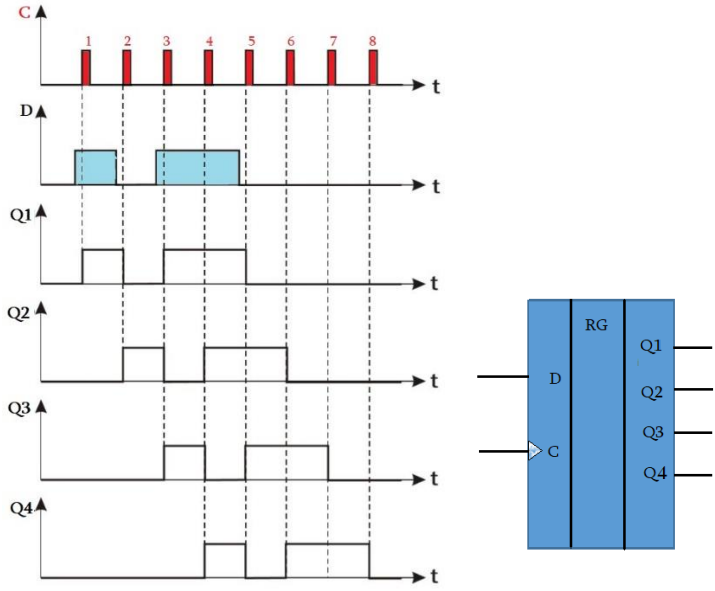
## მიმდევრობითი რეგისტრი

D-ტრიგერებზე აგებული მიმდევრობითი რეგისტრის სქემა, რომელსაც სხვაგვარად წანაცვლების რეგისტრსაც (shift register) უწოდებენ, აღნიშვნა და მისი მუშაობის დროითი დიაგრამა მოყვანილია ნახ.5.20, 5.21-ზე. მოწყობილობა იმართება ტაქტური C იმპულსებით (სიგნალებით). ტაქტური იმპულსის მოსვლასთან ერთად პირველი ტრიგერი იწერს D ინფორმაციას (0-ს ან 1-ს), რომელიც ამ მომენტში იმყოფება მის შესასვლელზე, ხოლო

ყოველი შემდეგი ტრიგერი გადაირთვება იმ მდგომარეობაში, რომელშიც მანამდე იმყოფებოდა წინამდებარე ტრიგერი.



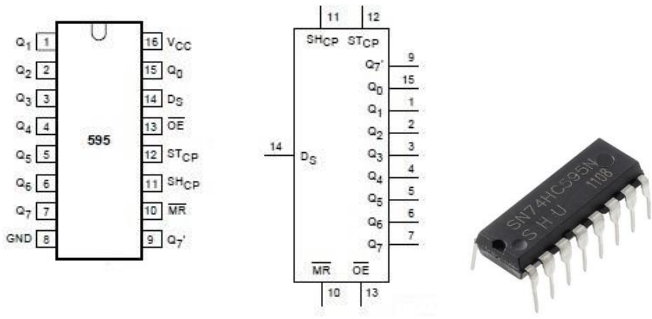

ნახ.5.20. მიმდევრობითი რეგისტრის ბლოკ-სქემა



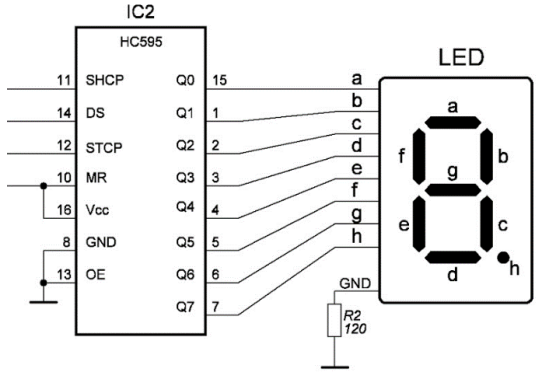

ნახ.5.21. მიმდევრობითი რეგისტრის მუშაობის დროითი დიაგრამა და სიმბოლური აღნიშვნა

პროცესის ასეთ მიმდინარეობას ის განაპირობებს, რომ ჩასაწერი სიგნალი გაივლის ტრიგერის შესასვლელიდან მის გამოსასვლელამდე გარკვეული დაყოვნებით, რომელიც მეტია ტაქტური იმპულსის წინა ფრონტის დაყოვნებაზე (რომლის განმავლობაშიც ხდება ჩაწერა). თითოეული C ტაქტური იმპულსი მიმდევრობით ერთი თანრიგით წაანაცვლებს რიცხვის კოდს რეგისტრში. ესე იგი ტრიგერში n თანრიგიანი კოდის ჩასაწერად საჭიროა n რაოდენობის ტაქტური იმპულსი.

როგორც დროით დიაგრამაზე ვხედავთ (ნახ.5.21) რეგისტრის C შესასვლელზე მოდებული ტაქტური იმპულსები განსაზღვრავენ ინფორმაციის ჩაწერასა და წანაცვლებას მასში. C შესასვლელზე ოთხი იმპულსის მოსვლის შემდეგ D შესასვლელზე მოდებული ორობითი რიცხვი 1011 მთლიანად ჩაიწერება რეგისტრში, ხოლო ყოველი შემდგომი C იმპულსი ერთი ნაბიჯით წაანაცვლებს ჩაწერილ ინფორმაციას. ესე იგი ოთხთანრიგიანი ინფორმაციის ჩასაწერად საჭიროა 4 ტაქტური იმპულსი, ამ მომენტში რეგისტრის გამოსასვლელებზე დაფიქსირდება შესასვლელზე მოდებული ინფორმაცია: Q4=1, Q3=0, Q2=1, Q1=1. გამოსასვლელებზე ეს ინფორმაცია დარჩება ჩაწერილი შემდეგი ტაქტური იმპულსის მოსვლამდე. ზოგადად შეიძლება ითქვას, რომ n-თანრიგიანი ორობითი რიცხვის რეგისტრში ჩასაწერად საჭიროა n რაოდენობა ტაქტური იმპულსებისა. აქვე ცხადია, რომ ჩაწერილი ინფორმაციის მიმდევრობითი წაკითხვის რეჟიმისთვის საჭირო იქნება შემდეგი 4 ტაქტური იმპულსები 5, 6, 7, 8 და ამ დროს Q4 გამოსასვლელიდან მიიღება მიმდევრობით ორობითი რიცხვს 1011.




ნახ.5.22. მიმდევრობითი რეგისტრი SN74HC595



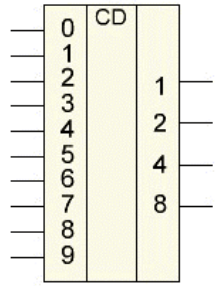

ნახ.5.23. მიმდევრობითი რეგისტრის SN74HC595-ის გამოყენების მაგალითი 8-სეგმენტისნი LED-ინდიკატორითვის

ნახ.5.22 და ნახ.5.23-ზე მოცემულია მიმდევრობითი რეგისტრის მიკროსქემა **SN74HC595**-ის შესავალი და გამოსავალი კონტაქტების დანიშნულება, ფოტო გამოსახულება და მისი გამოყენების ერთ-ერთი მაგალითი 8-სეგმენტიანი LED-ინდიკატორთან გამოყენებისათვის. აგრეთვე ნახ.5.23 ჩართვის სქემა

ფართოდ გამოიყენება აღწერილი რეგისტრის მიკროკონტროლერებიდან მართვისათვის SPI მიმდევრობითი პორტიდან.

## 5.5. შიფრატორი

შიფრატორი ანუ კოდერი არის ელექტრონული მოწყობილობა, რომელიც ერთი კონკრეტული თვლის სიტემის კოდს სხვა თვლის სისტემის კოდში გადაიყვანს. ელექტრონიკაში ყველაზე ფართოდ გავრცელებულია შიფრატორები, რომლებიც პოზიციურ ათობით კოდს პარალელურ ბინარულ (ორობით) კოდად აქცევს. შიფრატორის  სქემატური დიაგრამაზე შემდეგნაირად გამოისახება (ნახ.5.24.)




**ნახ.5.24. შიფრატორის სქემატური აღნიშვნა.**

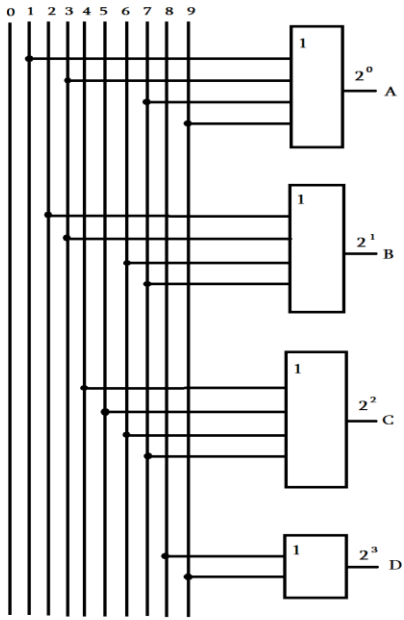
შიფრატორის ფუნქციის მარტივად გააზრებისათვის განვიხილოთ როგორ წარმოებს ციფრების გამოსახვა სტანდარტულ კალკულატორში. ცხადია, რომ კალკულატორში ყველა მოქმედება ხორციელდება ორობითი რიცხვების მანიპულირებით, ამიტომ კლავიატურის ნებისმიერ კლავიშზე ზემოქმედებით

უნდა გენერირდებოდეს შესაბამისი ორობითი კოდი რასაც კლავიატურასთან დაკავშირებული შიფრატორი ახორციელებს. იგი კლავიატურიდან შეყვანილ რიცხვებს ორობით ფორმად აქცევს. კალკულატორის ყველა ღილაკი უკავშირდება საერთო გამტარს და მაგალითად, შიფრატორის შესასვლელზე ღილაკზე 3 დაჭერით მის გამოსავალში დაუყოვნებლივ დაგენერირდება ამ რიცხვის ორობით ფორმა, ანუ „0011“ ოთხთანრიგიანი ორობითი სახით. განვიხილოთ შიფრატორის ჭეშმარიტების ცხრილი:

| შ ი ფ რ ა ტ ო რ ი | | | | |
|---|---|---|---|---|
| ათობითი რიცხვი | ორობითი კოდო | | | |
| | A | B | C | D |
| 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 |
| | D(3) | D(2) | D(1) | D(0) |
| დ ე შ ი ფ რ ა ტ ო რ ი | | | | |

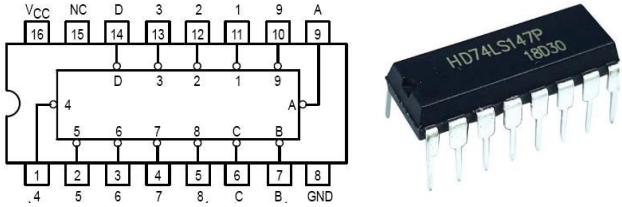
**ცხრილი 5.8. შიფრატორის და დეშიფრირების ჭეშმარიტების ცხრილი**

თუ გავითვალისწინებთ შიფრატორის სტრუქტურას, მაშინ მარტივია დავრწმუნდეთ, რომ იგი შესრულებულია საბაზო ლოგიკურ ელემენტებზე (ნახ.5.25) და ასახვას ჭეშმარიტების ცხრილ 5.8-ში მოცემულ მონაცემებს.



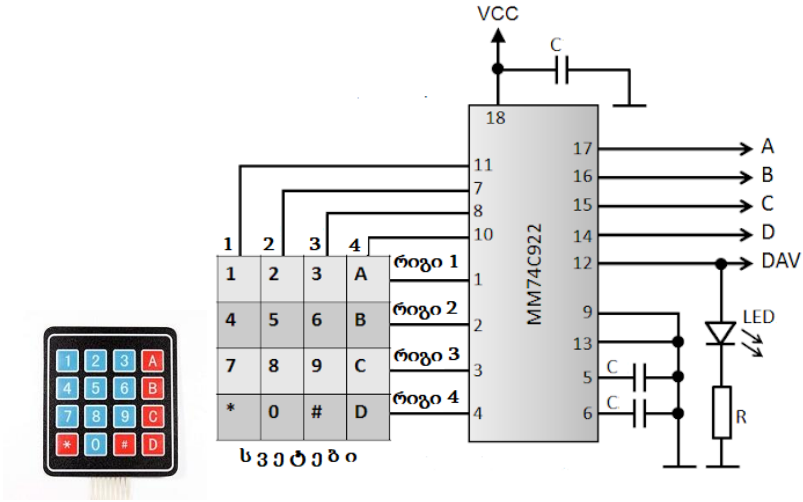

ნახ.5.25. შიფრატორის სტრუქტურა ლოგიკურ სქემებზე

მოცემული შიფრატორის რეალიზაცია ხორციელდება 74LS147 მიკროსქემის საშუალებით, რომლის სქემატური და ფოტო გამოსახულებები ნახ.5.26-ზეა მოცემული.




ნახ.5.26. 74LS147 შიფრატორის მიკროსქემის სქემატური და ფოტო გამოსახულებები

სხვადასხვა მოწყობილობებს და კონკრეტულად კალკულატორის შიფრატორს აქვს შესასვლელი პინების უფრო მეტი რაოდენობა, ვინაიდან ციფრების გარდა მასში რამდენიმე არითმეტიკული სიმბოლოების (ოპერაციების) შეტანაა კიდევ საჭირო. მაშასადამე, საჭიროა არა მხოლოდ ოროობითი ფორმით მოცემული რიცხვი, არამედ საჭიროა კოდირების გამოსასვლელიდან შესაბამისი ბრძანებებიც. ასეთი და მსგავსი შემთხვევებისათვის გამოიყენებენ სხვადასხვა სახის შიფრატორებს, რომლის ერთი მაგალითი ნახაზზეა მოცემული მატრიცული კლავიატურის რეალიზაციისათვის მიკროსქემა MM74C922 ბაზაზე .



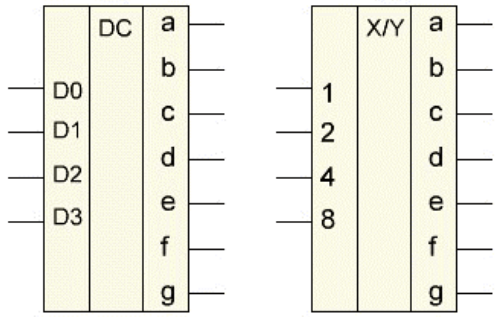

**ნახ.5.27. მატრიცული კლავიატურა და მისი დეშიფრატორით MM74C922რეალიზაციის სქემა**

შიფრატორები, გარდა კომპიუტერული ტექნიკისა, პრაქტიკულად ფართოდ გამოიყენება სხვადასხვა სახის ელექტრონულ

საკონტროლო მოწყობილობაში, როგორიცაა, მაგალითად კონდენციონერის მართვის პულტები, სარეცხი მანქნის დისპლეი და სხვა, რომლებიც მუშაობენ ორობითი ლოგიკით, მაგრამ ოპერაციების მოხერხებულობისთვის აქვთ ათობითი ან მზგავსი კლავიატურა.

## 5.6. დეშიფრატორი

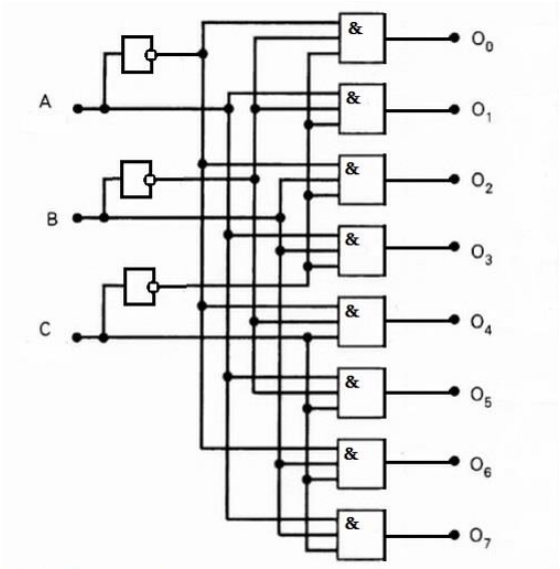
დეშიფრატორები ანუ დეკოდერები იმავე ჯგუფს მიეკუთვნებიან რომელსაც შიფრატორები, მხოლოდ ისინი მუშაობენ საპირისპიროდ, ანუ ისინი გარდაქმნიან პარალელურ ორობით კოდს პოზიციურ ათობითიში. ნახაზზე მოცემულია დეშიფრატორის გრაფიკული აღნიშვნის ორი ვარიანტი.



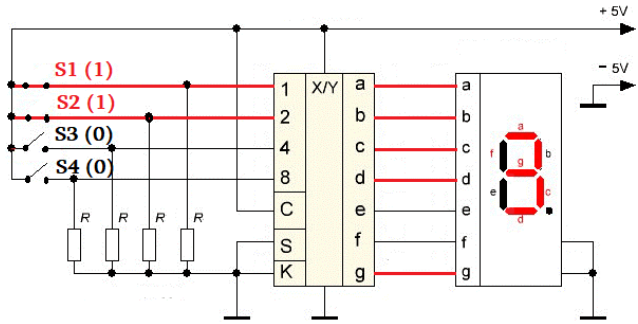

**ნახ.5.28. დეშიფრატორის სქემატური აღნიშვნა**

ზოგადად დეშიფრატორების დანიშნულებაა ორობითი კოდის გადაყვანა სხვადასხვა რიცხვით სისტემაში (ათობითი, თექვსმეტობითი და ა.შ.), რომელიც დამოკიდებულია მიკროსქემის სპეციფიკაზე და დანიშნულებაზე.

ისევე როგორც კომპიუტერული და ციფრული ელექტრონიკის ძირითადი მოდულები ასევე დეშიფრატორებიც აიგება საბაზო ლოგიკურ ელემენტებზე. ნახ.5.29-ზე წარმოდგენილია დეშიფრატორის სტრუქტურული პრინციპული სქემა. ასეთი სტრუქტურული წარმოდგენა უფრო გასაგებს ხდის მისი ფუნქციონირების პრინციპებს. დეშიფრატორის მუშაობის პრინციპის უფრო მარტივად გასაგებად ნახაზზე წარმოდგენილია ვარიანტი 3 შესავალითა და 8 გამოსასვლელით, პრაქტიკაში კი გარდა აღნიშნულისა გამოიყენება სხვადასხვა მწარმოებლების მიერ წარმოებული სხვადასხვა ფუნქციონალური დანიშნულების დეშიფრატორები გამომდინარე გადასაწყვეტი პრობლემისა. დეშიფრატორები შეიძლება გარდაქმნიდნენ ორობით კოდს, რვაობით, ათობით, თექვსმეტობით თვლის სიტემებში, ან ასრულებდნენ სხვა სპეციფიკურ ფუნქციას.




**ნახ.5.29. დეშიფრატორის სტრუქტურა ლოგიკურ სქემებზე**




ნახ.5.30. დეშიფრატორის პრინციპული სქემა 7-სეგმენტიანი მატრიცული დისპლეისათვის

მაგალითისათვის, 7-სეგმენტიანი მატრიცული დისპლეისათვის განკუთვნილია დეშიფრატორი 4511 სერიის მიკროსქემების სახით. ამ მიკროსქემებს შეუძლიათ ორობითი კოდი 0000–დან 1001–მდე გადაიყვანონ ათობით ციფრებში 0–დან 9–მდე. C, S, K გამოსასვლელები წარმოადგენენ დამხმარე პინებს და გამოიყენებიან ამოცანის სპეციფიკის მიხედვით. მიკროსქემას აქვს ოთხი შესავალი პინი (1, 2, 4, 8). ისინი ზოგჯერ ასევე მოიხსენიებიან როგორც D0 - D3 შესასვლელებად. ამ შესასვლელებზე მოედება პარალელური ორობითი კოდი (მაგალითად, 0011). ამ შემთხვევაში, ორობითი რიცხვი წარმოდგენილია 4 ბიტით. მიკროსქემა გარდაქმნის კოდს ისე, რომ გამომავალ პინებზე (a - g) გამოვიდეს შესაბამისი სიგნალები, რომლებიც ქმნიან ათობითი ციფრებს და ეს ციფრები აისახება შვიდი სეგმენტის დახმარებით მატრიცულ ეკრანზე, რომლებიც მართალია მოძველებული სისტემაა, მაგრამ ჯერ კიდევ ფართოდ გამოიყენება და ჩვენს შემთხვევაში კარგად ხსნის დეშიფრატორის დანიშნულებას. 4 ჩამრთველი (S1 - S4) უკავშირდება დეშიფრატორის შესასვლელებს, რომელთა დახმარებით პარალელური

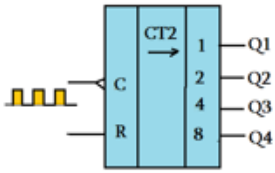
ორობითი კოდი მოედება დეშიფრატორის შესასვლელ პინებს. ჩართული S ჩამრთველები შეესაბამება ლოგიკურ „1“-ს, ხოლო გამორთული - ლოგიკურ „0“-ს. ასეთ შემთხვევაში მიკროსქემის შესასვლელებზე ხელით შეიძლება დაყენდეს ლოგიკა „1“ ან „0“. ნახ.7-ზე ნაჩვენებია თუ როგორ მოიდება კოდი 0011 დეშიფრატორის შესასვლელებზე. მატრიცულ დისპლეიზე გამოჩნდება ციფრი 3. თუ დამატებით ჩავრთავთ S3 ჩამრთველს, მაშინ ეკრანზე გამოჩნდება ციფრი 7 და ა.შ. ათობითი 0-დან 9-მდე ციფრების წარმოსადგენად ორობით კოდში საკმარისია ოთხი თანრიგი: a3 * 8 + a2 * 4 + a1 * 2 + a0 * 1, სადაც a0 - a3 ორობითი თვლის სისტემის რიცხვებია (0 ან 1). წარმოვადგინოთ რიცხვი 0011 ათობითი ფორმით 0011 = 0 * 8 + 0 * 4 + 1 * 2 + 1 * 1 = 2 + 1 = 3 რაც შესაბამისობაში იქნება 7-სეგმენტიანი მატრიცული დისპლეის ჩვენაბასთან.

ნახ.5.31-ზე მოცემულია 74XXYY სერიის 74LS42 დეშიფრატორის მიკროსქემის შესავალ-გამოსასვლელების პინების დანიშნულება და მისი ფოტო გამოსახულება. სრული ინფორმაცია აღნიშნული და სხვა მიკროსქემების შესახებ შესაძლებელია მოპოვებული იქნეს ვებ-გვერდიდან:
https://circuits-diy.com/amp/74ls42-4-line-bcd-to-10-line-decimal-decoder-ic-datasheet/


**ნახ.5.31. 74LS42 დეშიფრატორის მიკროსქემის შესავალ-გამოსასვლელების პინების დანიშნულება და მისი ფოტო გამოსახულება**

## 5.7. მთვლელი

მთვლელი წარმოადგენს ელექტრონულ მოწყობილობას, რომელის დანიშნულებაა იმპულსების რაოდენობის დათვლა. იმპულსების თვლის შედეგად მიღებული იმპულსების რაოდენობა გამოისახება ორობით სისტემაში.




| იმპულსების რაოდენობა | Q4 | Q3 | Q2 | Q1 |
|---|---|---|---|---|
| 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 |
| 10 | 1 | 0 | 1 | 0 |
| 11 | 1 | 0 | 1 | 1 |
| 12 | 1 | 1 | 0 | 0 |
| 13 | 1 | 1 | 0 | 1 |
| 14 | 1 | 1 | 1 | 0 |
| 15 | 1 | 1 | 1 | 1 |

**ნახ.5.32. ოთხთანრიგიანი მთვლელი და მისი ჭეშმარიტების ცხრილი**

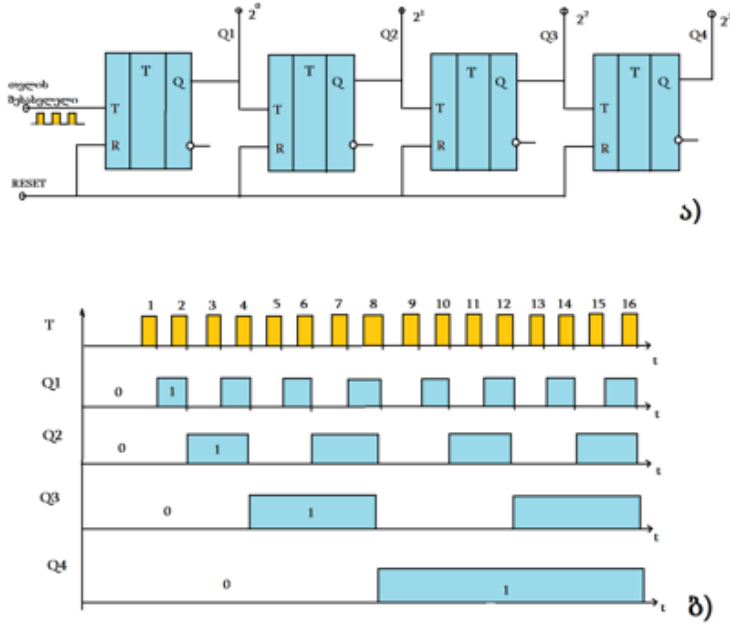
დანიშნულების შესაბამისად მთვლელების სხვადასხვა სახეობები არსებობს. ჩვენ მაგალითისათვის ნახ. 5.32-ზე მოვიტანეთ ორობითი იმპულსების ოთხთანრიგიანი მთვლელის გრაფიკულ გამოსახულებასა და ჭეშმარიტების ცხრილს.

იმპულსების მთვლელები ერთგვარი რეგისტრებია (მთვლელი რეგისტრები) და აგებულია, შესაბამისად, ტრიგერებისა და ლოგიკური ელემენტების საფუძველზე. მთვლელების ერთერთი მთავარი პარამეტრია თვლის კოეფიციენტი, რომელიც გამოითვლება ფორმულით K=2n, სადაც n მთვლელის თანრიგია, ან ტრიგერების რაოდენობაა მთვლელში.

მაგალითად, მთვლელს, რომელიც შედგება ოთხი ტრიგერისაგან შეიძლება ჰქონდეს თვლის მაქსიმალური კოეფიციენტი 16. ოთხი ტრიგერისგან შემდგარი მთვლელისათვის მინიმალური გამომავალი ორობითი კოდია 0000, მაქსიმუმი -1111, ხოლო როდესაც თვლის კოეფიციენტი K = 10 გამომავალი თვლა შეჩერდება იმ შემთხვევაში როდესაც კოდი მიაღწევს 1001 = 9.

ნახაზი 5.33-ზე მოცემულია ოთხთანრიგიანი მთვლელის სქემა, რომელიც შედგება თანმიმდევრულად შეერთებული T-ტიგერებისაგან. დასათვლელი იმპულსები მოედება პირველი ტრიგერის თვლისთვის განკუთვნილ შესავალს. მომდევნო ტრიგერების თვლის შესასვლელი პინები შესაბამისად უკავშირდებიან წინა ტრიგერის გამოსავლებს. აღნიშნული სქემის მუშაობის პრინციპს ნათლად ასახავს ნახ.5.33.ბ-ზე მოცემული დროთი დიაგრამა. დასათვლელი იმპულსების პირველი იმპულსის უკანა ფრონტის ტრიგერის შესასვლელზე მოსვლის შემდეგ ტიგერი გადადის Q1=1 მდგომარეობაში, ესე იგი ტრიგერში ჩაიწერა ციფრული კოდი 0001. მეორე იმპულსის უკანა ფრონტის მოსვლის შემდეგ პირველი ტრიგერი გადავა „0“ მდგომარეობაში, ხოლო მეორე „1“ მდგომარეობაში და შესაბამისად მთვლელში ჩაიწერება რიცხვი 2 კოდით 0010 და ასე შემდეგ.

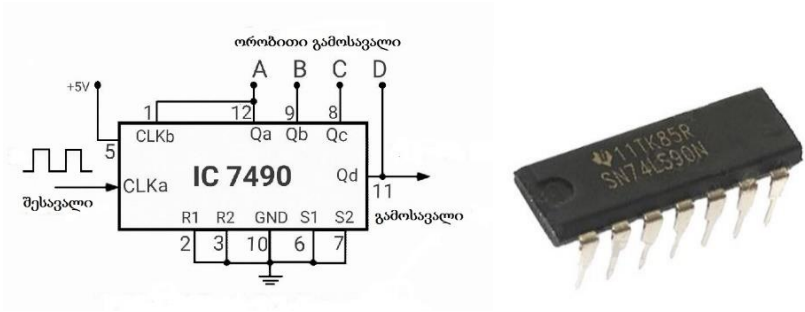
დროითი დიაგრამის მიხედვით ჩანს, რომ, მაგალითად მე-6 იმპულსის მოსვლის შემდეგ მთვლელში ჩაიწერება კოდი 0110, მე-11 იმპულსისითვის - 1101, ხოლო მე-15 იმპულსის დასრულების დროს (იმპულსის უკანა ფრონტი) მთვლელის ყველა თანრიგი შეივსება „1“-ბით და მე-16 იმპულსის დასრულების შემდეგ მთვლელის ყველა რეგისტრი განულდება „0“, ანუ მთვლელი გადავა საწყის მდგომარეობაში. ასევე შესაძლებელია მთვლელის ნებისმიერ დროს განულება მის „RESET” შესასვლელზე შესაბამისი იმპულსის მოდებით.




ნახ.5.33. მთვლელის პრინციპული სქემა (ა) და დროითი დიაგრამა (ბ)

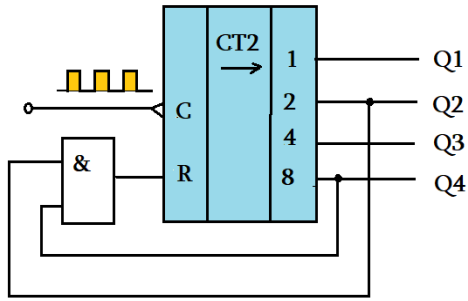
ორობითი მთვლელების მუშაობის პროცესში იმპულსების სიხშირე ყოველი ტრიგერის გამოსასვლელზე ორჯერ ნაკლებია მის შესასვლელზე მოდებულ იმპულსებთან შედარებით (ნახ.5.33.ბ), ამიტომ გარდა თვლისა მთვლელებს გარკვეულ შემთხვევებში იყენებენ როგორც სიხშირის გამყოფად.

პრაქტიკული რეალიზაციის თვალსაზრისით სხვადასხვა მწარმოებლები ამზადებენ მთვლელების სხვადასხვა ტიპებს მიკროსქემების სახით. მაგალითისათვის მოგვაქვს 74LS90 მთვლელის მიკროსქემის ფოტო გამოსახულება და ჩართვის პრინციპული სქემა თვლის რეჟიმისათვის (ნახ.5.34)




**ნახ.5.34. 74LS90 მთვლელის მიკროსქემის ფოტო გამოსახულება და ჩართვის პრინციპული სქემა**

სიგნალები მთვლელის გამოსასვლელებზე, გარდა ძირითადი დანიშნულებისა, შეიძლება გამოყენებული იქნეს როგორც უკუკავშირის სიგნალები. ეს საშუალებას იძლევა თვლის პროცესის ნებისმიერ საჭირო მომენტში გამოსასვლელი სიგნალები მოვდოთ მთვლელის „განულების“ RESET შესასვლელს. ამ დროს მიმდინარე თვლა შეწყდება და თვლა დაიწყება თავიდან. ასეთი აუცილებლობის ნათელი მაგალითია ელექტრონული საათები, სადაც საჭიროა მთვლელები თვლის კოეფიციენტით 6, 10, 7 და 24, შესაბამისად წუთის მეათედების, წუთის ერთეულების, კვირის დღეებისა და საათების დასათვლელად. ნახ.5.35-ზე მოცემულია ჩამოთვლილთაგან ერთი მაგალითი, პრინციპული სქემა სტანდარტული მთვლელის 10-ობითი რეჟიმში გამოყენებისათვის. ხოლო ჩამოთვლილი სხვა თვლის კოეფიციენტების რეალიზაცია იწარმოება მოცემული მაგალითის ანალოგიურად.



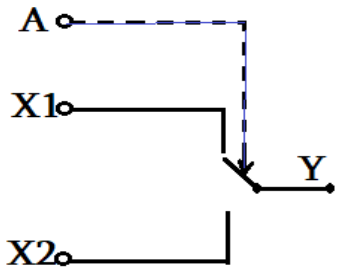

**ნახ.5.35. პრინციპული სქემა სტანდარტული მთვლელის 10-ობით რეჟიმში გამოყენებისათვის.**

ნახ.5.35-დან ცხადია, რომ როდესაც თვლა მიაღწევს ათობით 10-ს, გამოსასვლელზე მივიღებთ ორობით კოდს 1010-ს, ანუ Q2=1 და Q4=1, რომელიც, თავის მხრივ მოედება ლოგიკურ კონიუნქტორს. ამ შემთხვევაში ლოგიკური ელემენტის მუშაობის პრინციპიდან გამომდინარე მის გამოსასვლელზე გვექნება „1“, რომელიც მთვლელის R შესავალს მოედება და მთვლელის მიმდინარე შედეგი განულდება. ანუ თვლა იწარმოებს 0-დან 10-მდე. ასეთივე პრინციპით შეიძლება ყველა სხვა თვლის კოეფიციენტის რეალიზება. მაგალითად 6-მდე თვლისათვის კონიუნქტორის შესავლელებზე სიგნალები უნდა მოედოს მთვლელის Q2 და Q3 შესაბამისი გამოსასვლელებიდან. 7-მდე თვლისათვის კი საჭიროა 3 შესასვლელიანი კონიუნქტორი (ელემენტი „3და“) და მის შესასვლელებზე სიგნალები უნდა მოედოს მთვლელის Q1, Q2 და Q3 შესაბამისი გამოსასვლელებიდან.

## 5.8. მულტიპლექსორი და დემულტიპლექსორი

მულტიპლექსორები და დემულტიპლექსორები წარმოადგენენ კომბინაციურ მოწყობილობებს რომლებიც განკუთვნილია ციფრულ სიგნალთა და მონაცემთა ნაკადის გადართვისათვის (კომუტაციისათვის) სასიგნალო არხებში მოცემული მისამართებისა და მარშრუტების შესაბამისად.

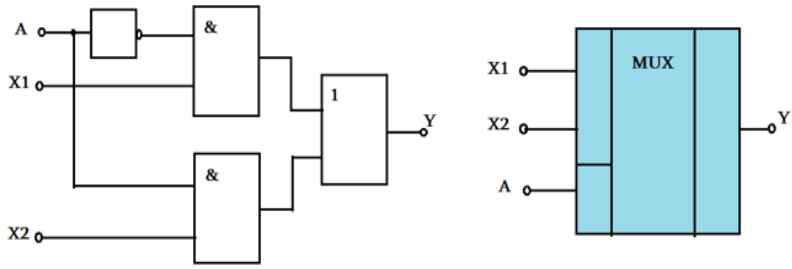
**მულტიპლექსორი.** თუ გვაქვს რამდენიმე სასიგნალო არხი და საჭიროა მათზე მოწოდებული ციფრული სიგნალების გარკვეული თანმიმდევრობით ერთ არხში გადართვა იყენებენ მულტიპლექსორს. პროცესის მარტივად წარმოდგენისათვის განვიხილოთ ნახ.5.36. აქ X1 და X2 არის სასიგნალო არხები რომლებშიც გარკვეული ციფრული ინფორმაციები გადაიცემა. თუ რომელი არხიდან გადავა ინფორმაცია Y გამოსასვლელზე დამოკიდებულია A-ს მიერ განსაზღვრული მისამართით. ანუ თუ მისამართი 0-ია Y გამოსასვლელზე ინფორმაცია ჩაირთვება X1 არხიდან და თუ მისამართი 1-ია -X2 არხიდან.




| შესასვლ. მისამართი | გამოსასვ. Y |
|---|---|
| 0 | X1 |
| 1 | X2 |

ნახ.5.36. მულტიპლექსორის მოქმედების პრინციპი და ჭეშმარიტების ცხრილი

მულტიპლექსორებს აღნიშნავენ MUX ან MS სიმბოლოებით. სიმბოლური სქემატური გამოსახულება და მოქმედების პრინციპი ორშესასვლელიანი და ერთგამოსასვლელიანი მულტიპლექსორის სქემისათვის ნახ.5.37-ზეა მოცემული.




**ნახ.5.37. ორშესასვლიანი მულტიპლექსორის სქემატური აღნიშვნდა პრინციპული სქემა.**

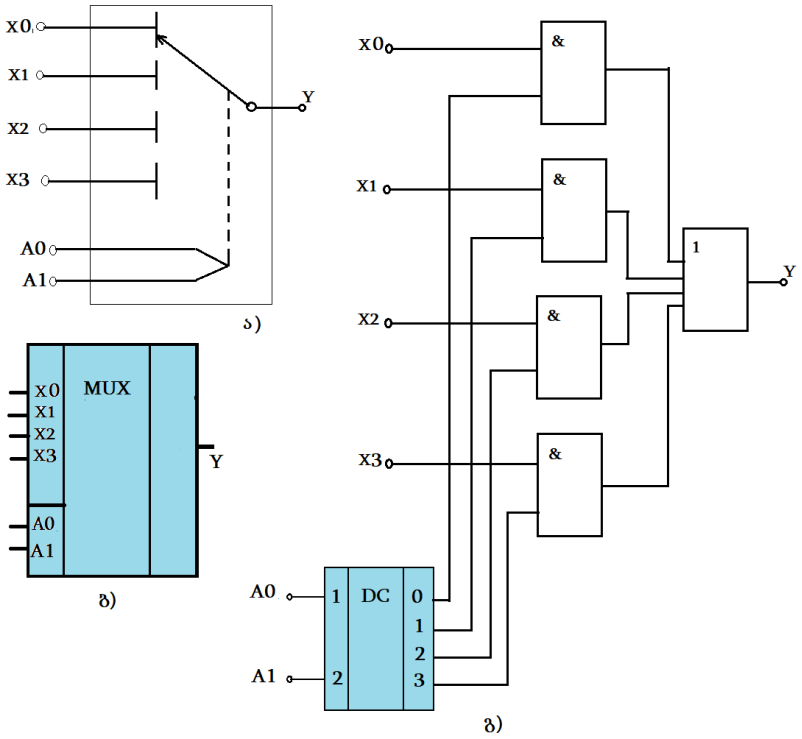
ნახაზზე მოცემული ორშესასვლელიანი (2-1) მულტიპლექსორის შესასვლელების და გამოსასვლელის მდგომარეობა ნახ 5.35-ზე მოცემული ცხრილის მიხედვითაა განსაზღვრული. ამ ცხრილიდან გამომდინარეობს:

$$Y = X1*A + X2*A$$

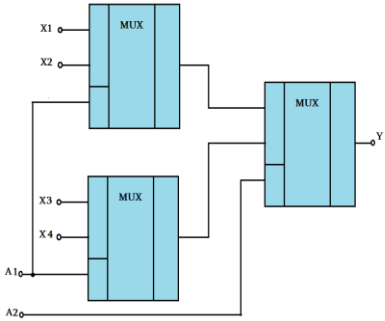
ამ ლოგიკური გამოსახულების რეალიზაცია მოცემულია ნახ.5.36-ზე. სქემა შედგება ორი თანხვედრის სქემისაგან და A შესასვლელზე ლოგიკური 0-ის ან 1-ის შემთხვევაში Y გამოსასვლელზე გაიმეორებს ერთ-ერთ შესასვლელზე არსებულ სიგნალს.

პრაქტიკაში გამოიყენება 2-ზე მეტ შესასვლელიანი მულტიპლექსორები. ნახ.5.38-ზე მოცემულია 4-შესასვლელიანი მულტიპლექსორის მოქმედების პრინციპი, სქემატური აღნიშვნა და პრინციპული სქემა.

იმ შემთხვევაში თუ საჭიროა 2-ზე მეტი შესასვლელიანი მულტიპლექსორი, იყენებენ მულტიპლექსორების ე.წ. კასკადურ ჩართვას (ნახ.5.39.) აქვე მოცემულია შესაბამისი ჭეშმარიტების ცხრილიც.




ნახ.5.38. ოთხშესასვლელიანი მულტიპლექსორის მოქმედების პრინციპი (ა), პირობითი გრაფიკული აღნიშვნა (ბ) და პრინციპული სქემა (გ)




| სამისამართო შესასვლელები | | გამოსასვლელი |
|---|---|---|
| A1 | A2 | Y |
| 0 | 0 | X1 |
| 0 | 1 | X2 |
| 1 | 0 | X3 |
| 1 | 1 | X4 |

ნახ.5.39. ოთხშესასვლელიანი მულტიპლექსორის ბლოკსქემა და ჭეშმარიტების ცხრილი

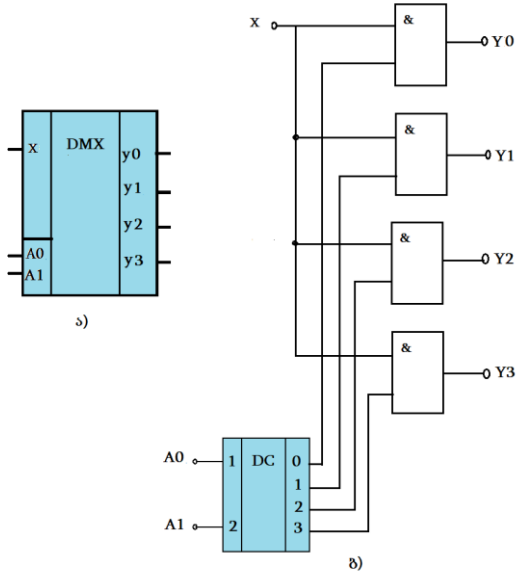
**დემულტიპლექსორი** წარმოადგენს ციფრულ მოწყობილობას, რომელიც სასურველი თანმიმდევრობით გადართავს ერთი არხით შემომავალ ინფორმაციის ნაკადს გამომავალი რამდენიმე არხიდან ერთ-ერთზე. ცხადია, რომ დემულტიპლექსორი ასრულებს მულტიპლექსორის შებრუნებულ ფუნქციას.

| გამოსასვლელის მისამართი | | შესასვლელი | გამოსასვლელები | | | |
|---|---|---|---|---|---|---|
| A0 | A1 | X | Y0 | Y1 | Y2 | Y3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |

ცხრილი 5.9. ოთხგამოსასვლელიანი დემულტიპლექსორის ჭეშმარიტების ცხრილი

დემულტიპლექსორში გამოსავალი არხის შერჩევა ხდება იმ სიგნალების შესაბამისად, რომლებიც მოედება სამისამართო შესასვლელებს. ოთხგამოსასვლელიანი დემულტიპლექსორის ჭეშმარიტების ცხრილი შემდეგი სახით წარმოდგება:

ოთხგამოსასვლელიანი დემულტიპლექსორის მოქმედების პრინციპული სქემა და სქემატური აღნიშვნა ნახ.5.40-ზეა მოცემული. ისევე როგორც მულტიპლექსორის პრინციპულ სქემაზე, აქაც მისამართის მართვა ხდება დეშიფრატორის დახმარებით.




**ნახ.5.40 დემულტიპლექსორის პირობითი გრაფიკული აღნიშვნა (ა) და პრინციპული სქემა (ბ)**

პრაქტიკული თვალსაზრისით მულტიპლექსორებსა და დემულტიპლექსორებს ძირითადად იყენებენ კომპიუტერული

ტექნიკის სხვადსხვა კომპონენტების დამზადების დროს, მაგრამ ასევე ფართოდ იყენებენ მათ ციფრული ელექტრონიკის სქემების შემუშავების დროსაც. მაგალითად, ისინი ფართოდ გამოიყენებიან თანამედროვე ტელეფონებში ხმოვანი სიგნალების გარდაქმნისათვის, ან როდესაც მცირე რაოდენობის სასიგნალო გამტარებით (კაბელების) საჭიროა გარკვეულ მანძილზე რამდენიმე ციფრულ მოწყობილობას შორის ინფორმაციის გაცვლა და სხვა.

დანიშნულების მიხედვით სხვადასხვა მწარმოებელი ამზადებს მულტიპლექსორებისა და დემულტიპლექსორების სხვადასახა სახეობებს, რომელთა შესახებაც ინფორმაციები ფართოდ არის წარმოდგენილი ინტერნეტში სხვადასხვა ვებ-გვერდებზე. ნახაზზე 5.41(ა) მოცემულია 74XXYY სერიის 74HC157 2x1 ოთხმაგი მულტიპლექსორის მიკროსქემის პინების დანიშნულება, ხოლო ნახაზზე 5.41(ბ) მოცემულია 74XXYY სერიის 74LS138 1-8 დემულტიპლექსორ/დეკოდერის მიკროსქემის პინების დანიშნულება. უფრო დაწვრილებითი ინფორმაციები მოცემული მიკროსქემების შესახებ შეიძლება მიღებული იქნეს ვებ-გვერდებიდან:

[74LS157 Quad 2 To 1 Line Multiplexer IC - Datasheet (circuits-diy.com)]

[74LS138 1-To-8 Decoder/Demultiplexer IC - Datasheet (circuits-diy.com)]


ა) ბ)

**ნახ.5.41. 74 სერიის 74HC157 მულტიპლექსორი (ა) და 74LS138 დემულტიპლექსორი (ბ).**

## 5.9. კომპარატორი

კომპარატორის განმარტება მისი სახელოდან გამომდინარეობს (comparison - შედარება), ანუ ამ ტიპის მოწყობილობები ახდენენ ზოგადად ორი სიდიდის შედარებას. ციფრული კომპარატორები ახდენენენ ორი ორობით ფორმატში წარმოდგენილი რიცხვების შედარებას. ციფრულ კომპარატორებს ორი თანაბართანრიგიანი A და B ორობითი რიცხვებისათვის შეუძლიათ დაადგინონ მათი ტოლობა, მეტობა და ნაკლებობა. აქედან გამომდინარე ციფრულ კომპარატორებს უნდა ჰქონდეთ ორი შესასვლელი A და B ორობითი რიცხვებისთვის და სამ გამოსასვლელი მოცემული რიცხვების ტოლობის, მეტობის და ნაკლებობის აღსანიშნავად.

განვიხილოთ კომპარატორის ჭეშმარიტების ცხრილი:

| A | B | C(A > B) | F(A = B) | D(A < B) |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |

**ცხრილი 5.9. ერთთანრიგიანი კომპარატორის ჭეშმარიტების ცხრილი**

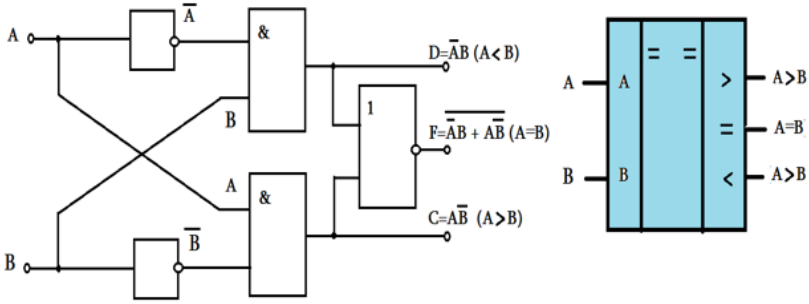
ჭეშმარიტების ცხრილის ანალიზიდან გამომდინარეობს კომპარატორის მოქმედების ლოგიკური პრინციპი და ხსნის იმ ლოგიკურ სქემას, რომელიც 5.42-ზეა მოცემული:

$$C\,(A > B) = A\overline{B};$$

$$D\ (A < B) = \overline{A}\ B;$$

$$F\ (A = B) = A \oplus B$$

მიღებული ლოგიკური გამოსახულებების გამოყენებით განვიხილოთ კომპარატორის მუშაობის პრინციპი (ნახ.5.42).

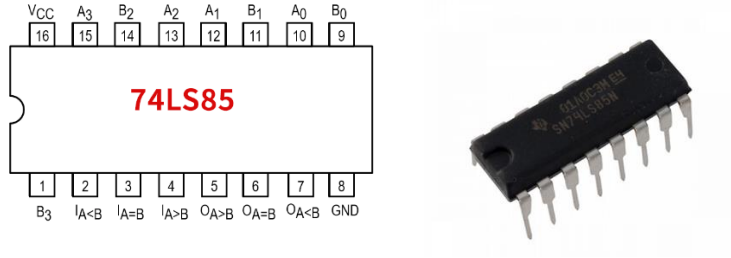



**ნახ.5.42. კომპარატორის მოქმედების პრინციპი და გრაფიკული აღნიშვნა**

ლოგიკური სქემის ანალიზი გვაჩვენებს, რომ როდესაც A=B მაშინ F=1, როდესაც A>B-ზე მაშინ მხოლოდ C=1 და დანარჩენი გამოსასვლელები ნულია, როცა A<B-ზე მხოლოდ D=1 და დანარჩენი გამოსასვლელები ნულია. ასევე ცხადია, რომ ორი მრავალთანრიგიანი ორობითი რიცხვები ტოლია იმ შემთხვევაში თუ მათი ციფრები წყვილად ტოლია თანრიგების შესაბამისად.

პრაქტიკული თვალსაზრისით ამზადებენ კოდების კომპარატორებს მიკროსქემების სახით. 74XXYY სერიიდან კომპარატორი 74LS85 განკუთვნილია ოთხთანრიგიანი ორობითი რიცხვების შედარებისათვის. თანრიგების რაოდენობის გაზრდისათვის იყენებენ ორ ან მეტ 74LS85 მიკროსქემას, რომლებიც გარკვეული წესით უკავშირდებიან ერთმანეთს. აღნიშნული მიკროსქემის შესავალი და გამოსავალი პინების განლაგება

მიკროსქემაზე და ფოტო გამოსახულება ნახ.5.43-ზეა მოცემული, ხოლო დამატებითი ინფორმაციები მის ტექნიკურ პარამეტრებზე შეიძლება მოპოვებული იქნეს ვებგვერდზე:
https://circuits-diy.com/74ls85-4-bit-magnitude-comparator-ic-datasheet




**ნახ.5.43. კომპარატორი 74LS85 მიკროსქემის შესავალი და გამოსავალი პინების დანიშნულება და მისი ფოტო გამოსახულება**

კომპარატორი 74LS85-ს გააჩნია I (Input-შესავალი) და O (output-გამოსავალი) პინები რომელთაგან შესავალი პინები განკუთვნილია იმ შემთხვევისთვის, როცა შესადარებელი რიცხვების თანრიგი ოთხზე მეტია. ამ შემთხვევაში მიკროს-ქემებს რთავენ მიმდევრობით (კასკადურად) და პირველი მიკროსქემის გამოსავალ პინებს უერთებენ მეორე მიკროსქემის შესაბამის შესავალ პინებს, ხოლო პირველი მიკროსქემის მეტობისა და ნაკლებობის შესავლებს უერთებენ „მიწას“, ხოლო ტოლობას უერთებენ ლოგიკურ „1“-ს“. ამ შემთხვევაში შესადარებელი რიცხვების თანრიგები ორჯერ გაიზრდება.

## მაგალითები და სავარჯიშოები

**სავარჯიშო 1.** მოცემული ლოგიკური გამოსახულებების მიხედვით:

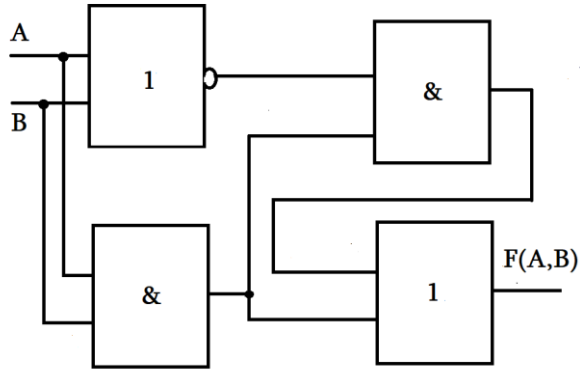
ა. ავაგოთ ლოგიკური სქემები;

ბ. შევადგინოთ ჭეშმარიტების ცხრილები:

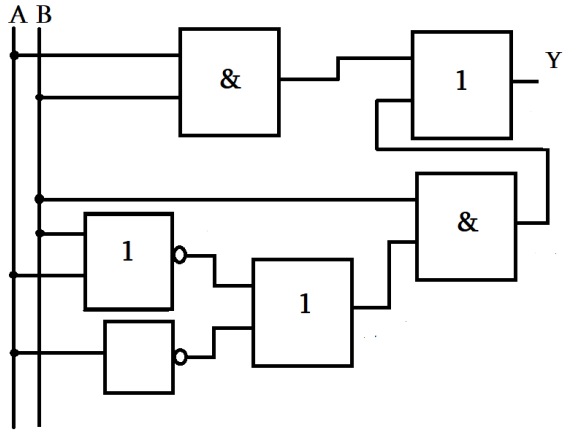
1. $Y= A+\overline{B}$
2. $Y= A*\overline{B} + B$
3. $Y= A + \overline{A} * B$
4. $F(A,B)= A*B + \overline{A * B}$
5. $Y= (A + B) *\overline{A}$
6. $F(A,B,C)= A+ \overline{A}*B + C(A*B+\overline{A}*\overline{B})$
7. $Y= A*B*C*D + \overline{A * B * C * D} + \overline{A}(B*\overline{C} + \overline{B}*D)$
8. $Y= \overline{A * B}*C*D + \overline{A * B * C * D} + \overline{A}(B*\overline{C} + \overline{B * D})$
9. $F(A,B,C,D) = A*(B*\overline{C}*D + \overline{B} *(C + \overline{A * D}))$
10. $Y= A*B*C*\overline{D} + A*\overline{B}(B*C*\overline{D} + \overline{A} * \overline{C} * D + \overline{B * C})$

**სავარჯიშო 2.** მოცემული სქემის მიხედვით დავადგინოთ: ა. ლოგიკური გამოსახულება; ბ. ჭეშმარიტების ცხრილი.

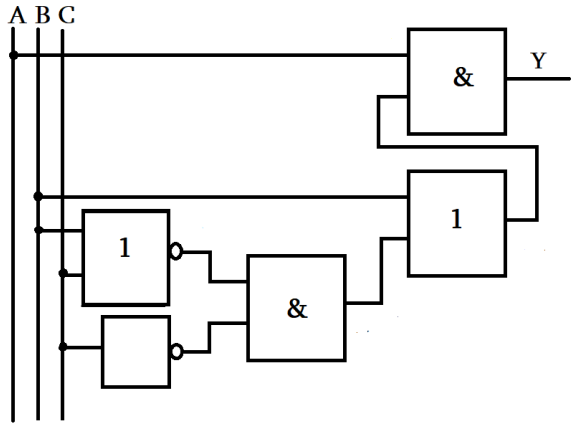
მაგალითი 1.



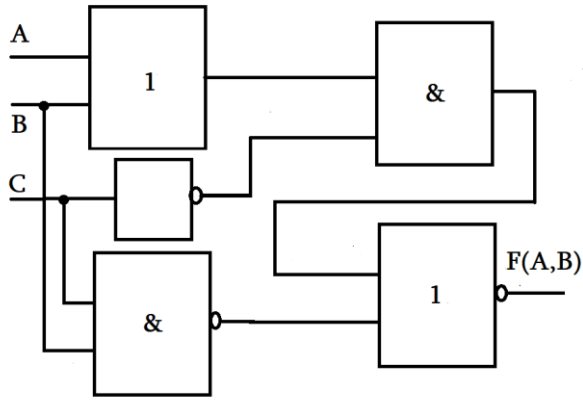
მაგალითი 2.



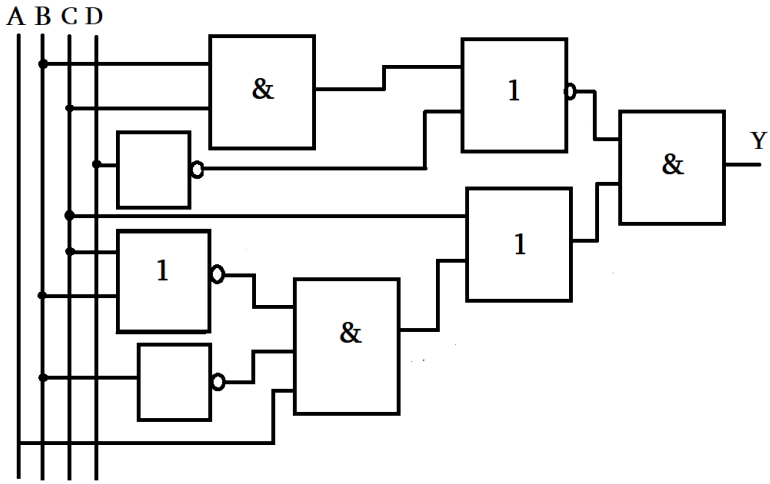

მაგალითი 3.



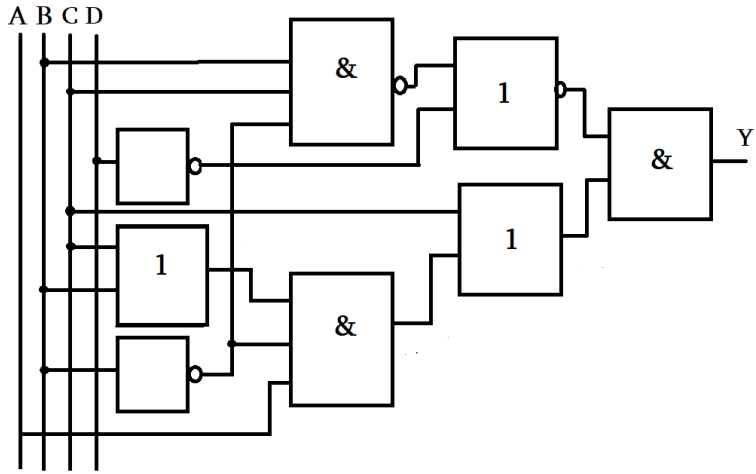
მაგალითი 4.



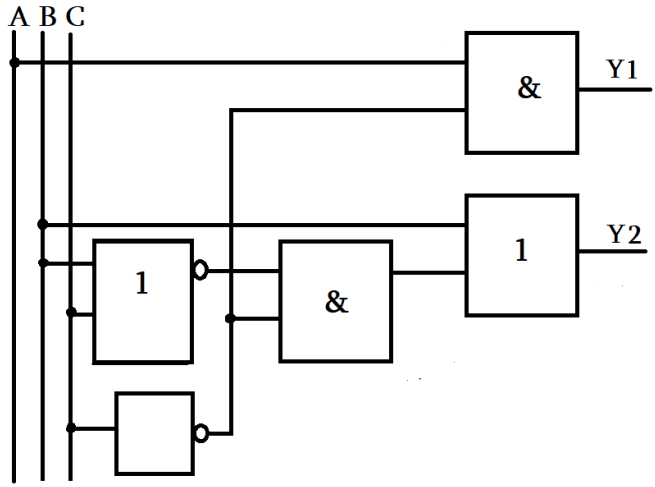

მაგალითი 5.



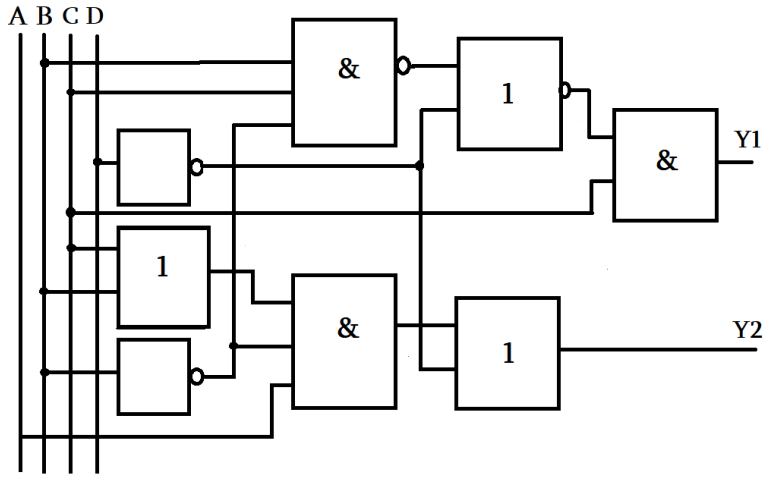
მაგალითი 6.




მაგალითი 7.



მაგალითი 8.




**სავარჯიშო 3.** მოცემული ჭეშმარიტების ცხრილის მიხედვით:

1. ა. დავადგინოთ ლოგიკური გამოსახულება;
   ბ. ავაგოთ ლოგიკური სქემა;
2. ა. მოვახდინოთ მინიმიზაცია კარნოს ბარათების მიხედვით;
   ბ. ავაგოთ ლოგიკური სქემა;
   გ. შევადაროთ მინიმიზაციამდე და მინიმიზაციის შემდეგ ლოგიკური ელემენტების რაოდენობა.

მაგალითი 1

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

მაგალითი 2

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

მაგალითი 3

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

მაგალითი 4

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

მაგალითი 5

| A | B | C | D | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

მაგალითი 6

| A | B | C | D | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

მაგალითი 7

| A | B | C | Y1 | Y2 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**სავარჯიშო 4.** მოვახდინოთ ლოგიკური ფუნქციების მინიმიზაცია:

1. $Y = A\overline{B} + B(\overline{A} + B)$
2. $F(A,B) = A(\overline{A} + B) + \overline{A}\ B$
3. $Y = \overline{A}\overline{B}\overline{C} + \overline{A}BC + \overline{A}BC + BA\overline{C}$
4. $Y = A\overline{B}\overline{C} + \overline{A}C + \overline{A}BC + A\overline{B}\overline{C} + C$
5. $Y = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}BC + A\overline{B}\overline{C}$
6. $F(A,B,C) = \overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{C}$
7. $F(A,B,C,D) = ABDC + D\overline{A}B\overline{C} + \overline{A}BCD + A\overline{C}DA$
8. $Y = \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + A\overline{C}DB$
9. $Y = ABCD + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + B\overline{C}DA$
10. $Y = ACD + \overline{A}BD + \overline{A}BC\overline{D} + \overline{C}DA$

**სავარჯიშო 5.** მოცემული გვაქვს ჭეშმარიტების ცხრილი არასრული მონაცემებით.

ა. დავადგინოთ ლოგიკური გამოსახულება;

ბ. ავაგოთ ლოგიკური სქემა:

მაგალითი 1

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | X |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | X |
| 1 | 0 | 0 | X |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | X |

მაგალითი 2

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | X |
| 0 | 0 | 1 | X |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | X |
| 1 | 1 | 0 | X |
| 1 | 1 | 1 | 0 |

მაგალითი 3

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | X |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | X |
| 1 | 0 | 0 | X |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | X |
| 1 | 1 | 1 | X |

მაგალითი 4

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | X |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | X |
| 1 | 0 | 0 | X |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

მაგალითი 5

| A | B | C | D | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | X |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | X |
| 0 | 1 | 0 | 1 | X |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | X |
| 1 | 0 | 0 | 1 | X |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | X |
| 1 | 1 | 1 | 0 | X |
| 1 | 1 | 1 | 1 | X |

მაგალითი 6

| A | B | C | D | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | X |
| 0 | 0 | 0 | 1 | X |
| 0 | 0 | 1 | 0 | X |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | X |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | X |
| 1 | 0 | 1 | 0 | X |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | X |
| 1 | 1 | 1 | 1 | X |

**სავარჯიშო 6.** მოცემული გვაქვს ჭეშმარიტების ცხრილის ფრაგმენტი:

ა. აღვადგინოთ ჭეშმარიტების ცხრილი;
ბ. დავადგინოთ ლოგიკური გამოსახულება;
გ. ავაგოთ ლოგიკური სქემა:

მაგალითი 1

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 |

მაგალითი 2

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

მაგალითი 3

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

მაგალითი 4

| A | B | C | D | y |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

მაგალითი 5

| A | B | C | D | y |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

## ლიტერატურა და გამოყენებული მასალები


1. Ata Elahi. “Computer Systems. Digital Design, Fundamentals of Computer Architecture and Assembly Language”. Book. Springer International Publishing AG 2018.
2. Linda Null, Julia Lobur. “The essentials of Computer organization and architecture”. Book. Jones and Bartlett Publishers. Massachusetts. 2003.
3. M. Morris Mano, Michael Ciletti. “Digital Designe”. Book. Publishing as Prentice Hall. New Jersy. 2013.
4. К. Бойт. «Цифровая электроника». Техносфера, 472 стр., 2007.
5. Кириченко П.Г. “Цифровая электроника для начинающих”. 2019.
6. Дэвид М. Харрис и Сара Л. Харрис. “Цифровая схемотехника и архитектура компьютера”. Издательство Morgan Kaufman © English Edition 2013.
7. ვებ რესურსი: https://circuits-diy.com/
8. ვებ რესურსი: https://www.alldatasheet.com/