**Kombinacinės logikos schemos**

**Turinys**

1. [Tikslas](#_bookmark0) 1
2. [Teorija](#_bookmark1) 1
3. [Laboratorinio darbo užduotis](#_bookmark12) 11
4. [Pavyzdys](#_bookmark13) 11

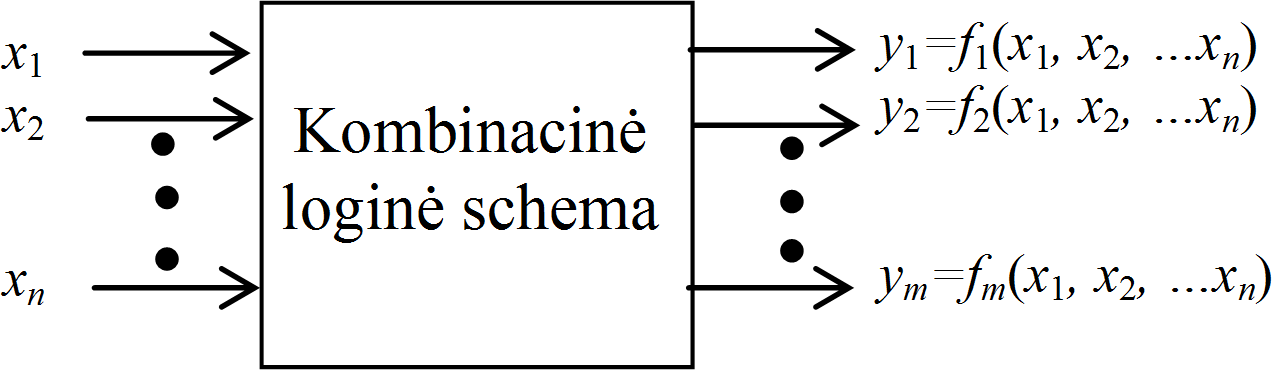
# Tikslas

Įsisavinti Būlio funkcijų minimizavimą ir kombinacinių loginių schemų projektavimą bei modeliavimą.

# Teorija

Loginės schemos skirstomos į kombinacines ir trigerines (literatūroje dar vadinamas nuosekliosiomis, sekvencinėmis) schemas.

**Kombinacinėmis** vadinamos loginės schemos, sudarytos iš sujungtų loginių elementų, neturinčios kilpų ir atminties elementų, o išvesties signalų vektoriaus *Y* reikšmes vienareikšmiškai nustato tuo metu veikiančių įvesčių signalų vektoriaus reikšmės *X*. Kombinacinės schemos dažnai vadinamos loginėmis schemomis be grįžtamųjų ryšių. Kombinacinė schema pavaizduota [1](#_bookmark2) pav.



**1 pav. Kombinacinė schema**

Skaitmeninių schemų funkcionavimui aprašyti plačiai taikoma Būlio algebra. Tai algebra (***B****, ●,* **∪***,* **¬***,* **0***,* **1**) , kurią sudaro baigtinė elementų aibė **B**, kurios elementai gali įgyti reikšmes 0 arba 1, ir trys apibrėžtos operacijos su elementais:

* **IR** operacija (loginė daugyba arba **konjunkcija**), žymima *●*, & arba ^; jei iš kontekso aišku, kad tai kintamųjų konjunkcija, šis operacijos ženklas dažnai praleidžiamas;
* **ARBA** operacija (loginė sudėtis arba **disjunkcija**), žymima +, v arba ∪;
* **NE** operacija (neigimas arba **inversija**), žymima ¬ arba brūkšniu virš kintamojo (išraiškos).

Taigi kiekviena loginė schema realizuoja loginę (Būlio ) funkciją (išvesties ***y***reikšmę), kurios argumentai yra Būlio kintamieji (įvesčių ***x***reikšmės). Būlio funkcijos vaizduojamos:

* teisingumo lentelėmis, kai Būlio funkcija turi ***n***kintamųjų ir aprašoma visomis gali- momis kintamųjų 2**n** kombinacijų reikšmėmis;
* analitinėmis išraiškomis, kai funkcijos išreiškiamos formulėmis;
* skaitmenine forma, kai užrašomos įvesčių reikšmės, su kuriomis funkcija įgauna reikšmę, lygią 1;
* Karno (Veičo) diagramomis, kai Būlio funkcijos vaizduojamos grafiniu būdu.

**Pavyzdys.** Sudarykime trijų kintamųjų funkcijos ***y***= ***f*** (***x*1*, x*2***,* ***x*3**), įgaunančios vienetui lygią reikšmę tik tuomet, kai nelyginis įėjimų skaičius įgyja reikšmę 1, teisingumo lentelę ([1](#_bookmark3) lentelė).

**1 lentelė. Funkcijos *y* = *f* (*x*1*, x*2*, x*3) = (1*,* 2*,* 4*,* 7)** **teisingumo lentelė**

|  |  |  |  |
| --- | --- | --- | --- |
| ***x*1** | ***x*2** | ***x*3** | ***y*** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

Lentelės apimčiai sumažinti dažnai joje pateikiamos tik tos kintamųjų reikšmių kombinacijos, kurioms esant funkcija lygi 1. Tai pavaizduota [2](#_bookmark4) lentelėje.

**2 lentelė. Funkcijos *y* = *f* (*x*1*, x*2*, x*3) = (1*,* 2*,* 4*,* 7) sutrumpinta teisingumo lentelė**

|  |  |  |  |
| --- | --- | --- | --- |
| ***x*1** | ***x*2** | ***x*3** | ***y*** |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Pasinaudoję teisingumo lentele, užrašykime pateiktos funkcijos analitinę išraišką – formulę. Paprasčiausiu atveju funkciją galima užrašyti kaip konjunkcijų, atitinkančių tas kintamųjų reikšmes, su kuriomis funkcija lygi vienetui, disjunkciją:

***y*** = ***f*** (***x*1*, x*2*, x*3**)= ***x*1 *x*2 *x*3 + *x*1*x*2*x*3 + *x*1*x*2*x*3 + *x*1 *x*2 *x*3 *.***

Kiekviena konjunkcijų grupė vadinama mintermu. Gautąją funkciją galima dar supaprastinti, iškeliant bendrus mintermų narius prieš skliaustus:

***y*** = ***f*** (***x*1*, x*2*, x*3**) = ***x*1 ( *x*2 *x*3 + *x*2*x*3** ) + ***x*1 ( *x*2 *x*3 + *x*2*x*3** ) *.*

arba, naudojantis Būlio algebros taisyklėms, pakeisti ją kita išraiška:

***y*** *=* ***f (x1, x2, x3****)**=* ***x1****(****x2 ⊕ x3****)* + ***x1****(****x2 ⊕ x3****) =* ***x1 ⊕ x2 ⊕ x3 .***

Užrašykime pateiktą funkciją skaitmenine forma (skaičiai – teisingumo lentelėje mintermą atitinkantys numeriai):

***y*** = ***f*** (***x*1*, x*2*, x*3**) = (**1*,* 2*,* 4*,* 7**) *.*

**2.1. Būlio funkcijos minimizavimas**

Tą pačią Būlio funkciją galima išreikšti įvairiai. Paprastai Būlio funkcija yra minimizuojama, tai yra gaunama funkcijos analitinė išraiška, turinti mažiau operacijų, konjunkcijų ar konjunkcijos raidžių. Tai leidžia schemą realizuoti mažesniu loginių elementų kiekiu.

Nedidelėms funkcijoms minimizuoti gerai tinka vadinamasis bandymų ir klaidų metodas, kai minimizuojama remiantis projektuotojo žiniomis ir patirtimi. Minimizuoti naudojama ši procedūra:

* naudojantis Būlio algebros aksiomomis ir teoremomis, išplėstinę disjunkcinę normaliąją formą galima pertvarkyti ir gauti paprastesnę disjunkcinę normaliąją formą, turinčią mažiau konjunkcijų ar mažiau raidžių konjunkcijose;
* grupuojant konjunkcijas ir iškeliant bendras raides prieš skliaustus, galima gauti trumpesnę funkcijos išraišką;
* greta pagrindinių operacijų (disjunkcijos, konjunkcijos ir inversijos) naudojant sudė- tingesnes operacijas (sudėties moduliu du ir pan.), funkcijos išraišką taip pat galima sutrumpinti.

Norint minimizuoti funkciją, gali būti naudojamos **Karno diagramos**. Jos sudaromos teisingumo lentelės rezultatus perrašant į dvimatę lentelę, kurios eilutės ir stulpeliai numeruojami Grėjaus kodu. Kiekvienas lentelės langelis atitinka vieną įvesčių kombinaciją. Taikant šį metodą, efektyviai naudojami projektuotojo gebėjimai vizualiai atpažinti būdingą vienetų išdėstymą diagramoje. Randamos optimalios ’1’ ar ’0’ grupės, žyminčios kanoninės formos termus. Šie termai naudojami užrašant minimalią Būlio išraišką, kuri aprašo reikiamus loginius elementus. Funkcijai ***y* = *f* (*x*1*, x*2*, x*3) = (1*,* 2*,* 4*,* 7)** Karno diagrama pavaizduota [3](#_bookmark5) lentelėje (panaudojus teisingumo lentelę, į atitinkamus langelius surašomos vienetinės funkcijos reikšmės).

**3 lentelė. Funkcijos *y*** = ***f*** (***x*1*, x*2*, x*3**) = (**1*,* 2*,* 4*,* 7**) **Karno diagrama**

|  |  |  |
| --- | --- | --- |
| **x1x2 x3** | 0 | 1 |
| 00 |  | **1** |
| 01 | **1** |  |
| 11 |  | **1** |
| 10 | **1** |  |

Kombinacinės loginės schemos yra sudarytos iš elementariųjų loginių elementų. Jos projektuojamos tokia tvarka:

1. Nustatomos loginės funkcijos ir jų argumentai (loginiai kintamieji).
2. Sudaroma loginė funkcija, kurią turi vykdyti projektuojamoji schema; loginė funkcija gali būti pateikta teisingumo lentele arba loginės funkcijos kanoninės disjunkcinės formos išraiška.
3. Minimizuojant loginės funkcijas vienu iš žinomų metodų (Karno diagramų, Kvaino ir Maklaskio ir t. t.), gaunama šios loginės funkcijos minimali išraiška. Šią išraišką galima pertvarkyti taip, kad ją būtų patogiau realizuoti loginiais elementais.
4. Atliekama sintezė: paruošta loginė funkcija realizuojama norimais loginiais elemen- tais. Gaunama kombinacinė loginė schema. Atsižvelgiant į greitaveikos ir galios suvartojimo reikalavimus gali būti taikomi specialūs automatizuoti kombinacinių schemų sintezės metodai, leidžiantys gauti schemas su optimaliu integrinių grandynų kiekiu.
5. Susintezavus kombinacinę schemą sudaromas kontrolinis testas, tai yra kintamųjų seka, kurią perdavus į suprojektuotos schemos įvestis galima patikrinti, ar ji realizuoja reikiamą funkciją.

**Pavyzdys.** Turime funkciją ***y*** = ***f***(***a, b, c, d***), kurios sutruminta teisingumo lentelė pavaizduota 4 lentelėje.

Pasinaudoję teisingumo lentele, sudarome Karno diagramą. Kadangi yra 4 įvesties signalai, jie gali įgyti 16 skirtingų reikšmių, todėl pilnoji teisingumo lentelė turi 16 eilučių, ir Karno diagrama turi 16 langelių. Dėl šios priežasties diagrama vaizduojama 4 × 4 lentele. Eilučių ir stulpelių reikšmės sužymimos Grėjaus kodu. Grėjaus kodas užtikrina, kad šalia esantys langeliai skiriasi tik per vieną kintamąjį. Jei iš konkrečios įvesčių signalų kombinacijos išvestyje gaunamas 1, tame Karno lentelės langelyje pažymimas 1.

**4 lentelė. Funkcijos *y*** = ***f***(***a, b, c, d***) **teisingumo lentelė**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***a*** | ***b*** | ***c*** | ***d*** | ***y*** |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Sudarius Karno lentelę, ieškoma supaprastinta kanoninė užrašytos informacijos išreiškimo forma. Greta esantys vienetai lentelėje sugrupuojami laikantis tam tikrų taisyklių:

* Mintermų grupės Karno diagramoje turi būti stačiakampio formos. Stačiakampio plotas (grupės 1 skaičius) gali būti tik dvejeto laipsnis (2,4,8...).
* Mintermų grupės turi būti kiek įmanoma didesnės.
* Grupės gali dengti viena kitą, kad kiekviena jų būtų kiek įmanoma didesnė.
* Lentelė yra toroidiškai sujungta. Tai reiškia, kad grupių stačiakampiai gali pereiti lentelės kraštus. Kairiojo krašto langeliai yra gretimi dešiniojo krašto langeliams, viršutiniai langeliai – apatiniams. Pvz., visi keturi lentelės kampai laikomi gretimais.

Sugrupuokime Karno lentelės elementus į grupes. Optimalios grupės šioje lentelėje pa vaizduotos skirtingomis spalvomis. Kai kuriose vietose jos dengia viena kitą. Elementai, priklausantys kelioms grupėms, pažymėti tarpinėmis spalvomis. Raudona linija apibrėžtą (raudonąją) grupę sudaro 4 × 1 dydžio stačiakampis, mėlyna linija apibrėžtą (mėlynąją) – 1 × 4 dydžio stačiakampis, žalia linija apibrėžtą (žaliąją) – 2 × 2 dydžio stačiakampis (nes lentelės kraštai – gretimi).

**5 lentelė. Funkcijos *y*** = ***f*** (***a, b, c, d***) **Karno diagrama**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cd ab | 00 | 01 | 11 | 10 |
| 00 |  | **1** | **1** |  |
| 01 |  | **1** |  |  |
| 11 | **1** | **1** | **1** | **1** |
| 10 |  | **1** | **1** |  |

Grupių mintermai randami nustatant, analizuojant grupės stulpelių ir/ar eilučių kodus.

Pavyzdžiui, žaliojoje grupėje:

* stulpelių kodai yra 01 ir 11; vadinasi kintamasis ***a*** grupėje neišlaiko pastovios reikšmės (kaitaliojasi iš 0 į 1), todėl į mintermą neįrašomas; tuo tarpu kintamasis ***b***visai grupei vienodas ir lygus 1, todėl į mintermą, atitinkantį žaliąją grupę, įtraukiamas ***b***;
* eilučių kodai yra 00 ir 10; vadinasi kintamasis ***d***visai grupei vienodas ir lygus 0, todėl į mintermą, atitinkantį žaliąją grupę, įtraukiamas ***d***; kintamasis ***c*** taip pat grupėje kaitaliojasi, todėl nerašomas.

Taigi, žaliąją grupę atitinka mintermas yra ***b d***.

Raudonojoje grupėje ***a*** ir ***b*** nesikeičia, o ***c*** ir ***d*** kaitaliojasi. ***a*** visuomet lygus 0, todėl jį rašome su inversijos ženklu, o ***b*** visuomet lygus 1. Taigi šią grupę atitinka mintermas ***a b.***

Mėlynojoje grupėje ***c*** ir ***d*** nesikeičia ir yra lygūs 1. ***a*** ir ***b*** kaitaliojasi, todėl gauname ***c d***.

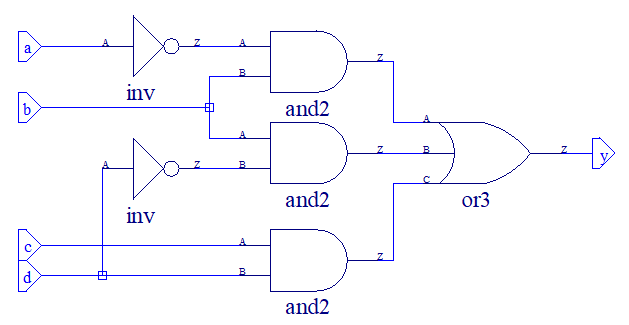
Taigi, pasinaudoję Karno diagrama, funkciją minimizavome ir gavome:

***y* = *bd*+*ab*+*cd.***

Šią minimizuotą išraišką galima gauti ir nenaudojant Karno diagramos, bet taikant Būlio algebros aksiomas, tačiau, jei konkunkcijų skaičius didelis, tai tampa labai sudėtinga.

**2.2. Schemos realizacija, naudojant IR, ARBA, NE elementus**

Šią išraišką (kaip ir kiekvieną išraišką, užrašytą normaliąja disjunkcine forma) galima realizuoti naudojantis trijų tipų elementais: IR, ARBA ir NE. Kombinacinė schema, realizuojanti funkciją, pavaizduota [2](#_bookmark9) pav.



1. **pav. Kombinacinė schema, gauta naudojant IR, ARBA, NE elementus**

Gautąją funkciją užrašome iškeldami prieš skliaustus ***b***:

***y* = *b(d*+*a)*+*cd.***

**2.3. Schemos realizacija, naudojant IR-NE elementus**

De Morgano dėsnis leidžia bet kokią funkciją išreikšti tik neigimo ir konjunkcijos operacijomis, t. y. minimaliojoje (dar vadinamoje universaliąja) elementų bazėje. De Morgano dėsnis užrašomas taip:

***a* + *b* = *a* · *b***  arba ***a* ·*b* = *a* + *b.***

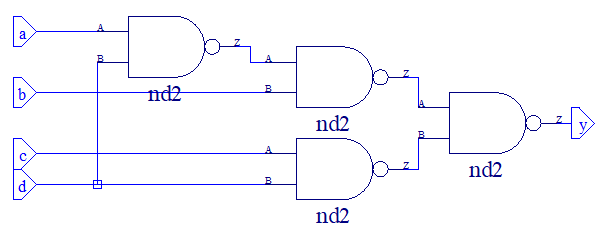
Pritaikę De Morgano dėsnį aukščiau pateiktai išraiškai, pertvarkykime ją taip, pradžioje kiekvieną išraiškos pusę invertuodami dukart (žinome, kad ***x*** = ***x*** ):

***y* = *b(d*+*a)*+*cd.***

Tuomet

***y* = *b(d*+*a)* · *cd = b*· *d*·*a* · *c* *d.***

Taip pertvarkytą funkciją lengva realizuoti naudojant tik IR-NE elementus. Kombinacinė schema, gauta naudojant IR-NE loginius elementus, parodyta [3](#_bookmark11) pav.



1. **pav. Kombinacinė schema, gauta naudojant IR-NE loginius elementus**

Matome, kad [2](#_bookmark9) paveiksle pateiktoje schemoje naudojami trijų tipų elementai, o [3](#_bookmark11) paveiksle pateiktai schemai reikia tik vieno tipo elementų.

Tokiu būdu pertvarkyti sudėtingesnę išraišką gan sudėtinga, nes gaunamos „daugiaaukštės“ inversijos. Tuo tarpu yra paprastas būdas schemai iš IR, ARBA, NE elementų pertvarkyti į schemą iš IR-NE elementų ir invertorių.

Dviejų pakopų schemą, kurios pirmojoje pakopoje yra elementai IR, o antrojoje – juos apjungiantis elementas ARBA, atitinka tokios pačios konfigūracijos schema, kurioje visi elementai yra IR-NE (žr. 4 pav. a ir b schemas).

Iš tiesų, 4 paveiksle a schema iš IR, ARBA, NE elementų realizuoja funkciją

***y***= ***ab + bcd.***

4 paveiksle b schema realizuojama funkcija yra

***z***= ***ab*  ·  *bcd*** = ***ab + bcd.***

Akivaizdu, kad ***y***= ***z***. Matome, kad pakako kiekvieną elementą IR ir elementą ARBA pakeisti elementais IR-NE, o schemos konfigūracija išliko tokia pat.

Tačiau jei į apjungiantį elementą ARBA paduodami kintamieji, tai į jį atitinkantį apjungiantį elementą IR-NE šie kintamieji turi būti paduoti invertuoti.

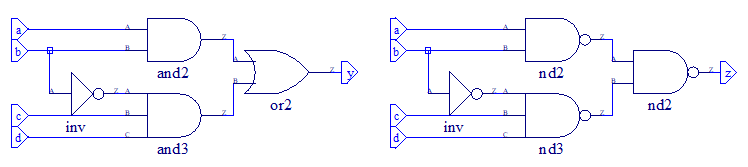
4 paveiksle c schema realizuoja funkciją

***v***= ***ab + bcd + e + f.***

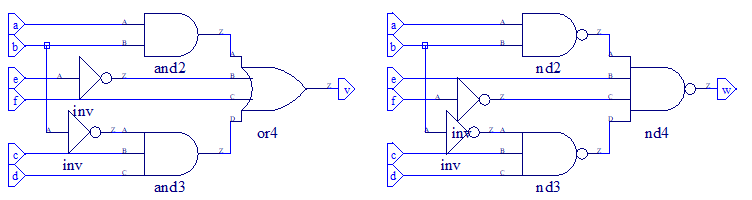
4 paveiksle d schema realizuojama funkcija yra

***w*** = ***ab*  ·  *bcd* · *e* · *f*** = ***ab + bcd + e + f*** = ***ab + bcd + e + f.***

Matome, kad ***v***= ***w***.



a) b)



c) d)

1. **pav. Funkcijos realizacijų IR, ARBA, NE ir IR-NE elementais palyginimas**

**2.4. Schemos realizacija, naudojant multiplekserius**

Multiplekserio panaudojimas funkcijai realizuoti remiasi **skleidimo teorema**

**f**(**x1**, **x**2, ... , **xn**) = **x1 ·** **f**(**1**, **x2**, ... , **xn** ) + **x1 ·** **f**(**0**, **x2**, ... , **xn** )

4 kintamųjų atveju, panaudoję skleidimą pagal **x1** ir **x2**, gausime:

**f**(**x1**, **x2**, **x3**, **x4**) = **x1 ·** **f**(**1**, **x2**, **x3**, **x4** ) + **x1 ·** **f**(**0**, **x2**, **x3**, **x4**) =

= **x1x2 ·** **f**(**1, 1, x3**, **x4**)

+ **x1x2 ·** **f**(**1, 0, x3**, **x4**)

+ **x1x2 ·** **f**(**0, 1, x3**, **x4**)

+ **x1x2 ·** **f**(**0, 0, x3**, **x4**)

Projektuojant schemą, naudojančią multiplekserį 4-1 (žr. 5 pav.), reikia:

* iš normaliosios disjunkcinės išraiškos išskirti adresinius kintamuosius, kurių skaičius priklauso nuo to, kiek adreso linijų turi pasirinktas multiplekseris. Šiame pavyzdyje kaip adresinius kintamuosius panaudosime ***a*** ir ***b***;
* pertvarkyti išraišką, šiuos kintamuosius iškeliant prieš skliaustus.

**MUX41**

**D0**

**D1**

**D2**

**D3**

**SD1**

**SD2**

**Z**

1. **pav. Multiplekseris 4-1**

Patogumo dėlei mūsų funkciją (žr. 4 lentelę) ***f*** = (**3*, 4, 5, 6, 7, 11, 1*2*, 14, 15***) užrašykime kubų pavidalu, paryškindami adresinius kintamuosius ***a*** ir ***b***:

**6 lentelė. Funkcijos kubai**

|  |  |  |  |
| --- | --- | --- | --- |
| Mintermo numeris | Kubas | Mintermo numeris | Kubas |
| 3 | **00**11 | 11 | **10**11 |
| 4 | **01**00 | 12 | **11**00 |
| 5 | **01**01 | 14 | **11**10 |
| 6 | **01**10 | 15 | **11**11 |
| 7 | **01**11 |  |  |

Iškėlus šiuos kintamuosius prieš skliaustus, išraiška atrodys taip:

***f***= ***a b***(***c d***) +

***a b***(***c d*** + ***c d*** + ***c d*** + ***c d***) +

***a b***(***c d*** ) +

***a b***(***c d*** + ***c d*** + ***c d***)***.***

Skliaustuose esančias išraiškas galima suprastinti (minimizuoti). Tuomet turėsime:

***f***= ***a b***(***c d*** ) +

***a b***( ***1*** ) +

***a b***(***c d*** ) +

***a b***(***d*** + ***c***)***.***

Kitas būdas išraiškai pertvarkyti, kad panaudotume multiplekserį, remiasi funkcijos Karno diagramos analize. Kaip adresinius kintamuosius naudojant ***a*** ir ***b***, diagramą galima suskaidyti į keturias dalis (jos čia skirtingai nuspalvintos, o apačioje nurodytas multiplekserio įvesties numeris):

**5 lentelė. Funkcijos *y*** = ***f*** (***a, b, c, d***) **Karno diagrama**

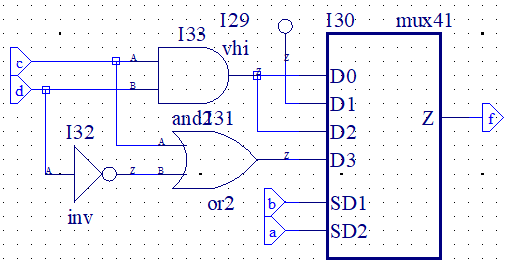
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cd ab | 00 | 01 | 11 | 10 |
| 00 |  | **1** | **1** |  |
| 01 |  | **1** |  |  |
| 11 | **1** | **1** | **1** | **1** |
| 10 |  | **1** | **1** |  |
|  | **D0** | **D1** | **D3** | **D2** |

Minimizavę kiekvienos dalies funkcijas, gausimę tą patį (aukščiau užrašytą) rezultatą. Šias funkcijos realizuojančios schemos prijungiamos prie atitinkamų multiplekserio įvesčių (D0-D3).

Pastabos:

* Liekamųjų funkcijų sudėtingumas priklauso nuo pasirinktų adresinių kintamųjų.
* Jeigu kuri nors liekamoji funkcija gaunama lygi nuliui arba vienetui, atitinkama multiplekserio įvestis prijungiama prie loginio nulio (vlo) arba loginio vieneto (vhi).

Funkcijos realizacija panaudojant multiplekserį pateikta [6](#_bookmark16) pav.



1. **pav. Schema su multiplekseriu 4-1**

Ar kombinacinė schema veikia gerai, galima patikrinti atlikus kontrolinį testą: schemos įvestims pateikus argumentų kombinacijų sekas ir patikrinus, ar schemos išvestys įgyja teisingumo lentelėje nurodytas reikšmes.

# Laboratorinio darbo užduotis

Užduočių variantų lentelėje duotos funkcijos, kurių argumentų konjunkcijos pateiktos skaičiais. Kiekvienas studentas gauna jam priklausančio varianto numerį. Atlikti užduočiai reikia:

* 1. Užrašyti pateiktą funkciją normaliąja disjunkcine forma.
  2. Minimizuoti pateiktą funkciją.
  3. Realizuoti šią funkciją trimis būdais:
     1. naudojant IR, ARBA, NE elementus;
     2. naudojant tik IR-NE arba ARBA-NE ir NE elementus;
     3. naudojant multiplekserį ir reikiamus IR, ARBA, NE, IR-NE, ARBA-NE elementus.
  4. Patikrinti suprojektuotų schemų funkcionavimą.
  5. Paruošti laboratorinio darbo ataskaitą. Ataskaitoje pateikti funkcijos minimizavimo rezultatus, realizuotas schemas bei šių schemų modeliavimo rezultatus.

# Pavyzdys

Funkcijos argumentų konjunkcijos pateiktos šiais skaičiais:

***f*** = (**1*,* 2*,* 3*,* 9*,* 17*,* 20*,* 21*,* 22*,* 23*,* 25*,* 33*,* 41*,* 49*,* 57*,* 61*,* 63**)*.*

* 1. Šešių kintamųjų funkcijos Karno diagramos šablonas atrodo taip (langeliuose įrašyti juos atitinkančių mintermų numeriai):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x*4*x*5*x*6 *x*1*x*2*x*3 | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
| 000 | **0** | **8** | **24** | **16** | **48** | **56** | **40** | **32** |
| 001 | **1** | **9** | **25** | **17** | **49** | **57** | **41** | **33** |
| 011 | **3** | **11** | **27** | **19** | **51** | **59** | **43** | **35** |
| 010 | **2** | **10** | **26** | **18** | **50** | **58** | **42** | **34** |
| 110 | **6** | **14** | **30** | **22** | **54** | **62** | **46** | **38** |
| 111 | **7** | **15** | **31** | **23** | **55** | **63** | **47** | **39** |
| 101 | **5** | **13** | **29** | **21** | **53** | **61** | **45** | **37** |
| 100 | **4** | **12** | **28** | **20** | **52** | **60** | **44** | **36** |

* 1. Panaudodami šį šabloną, sudarome Karno diagramą ir ją užpildome funkcijos reikšmėmis:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x*4*x*5*x*6 *x*1*x*2*x*3 | 000 | 001 | 011 | 010 | 110 | 111 | 101 | 100 |
| 000 |  |  |  |  |  |  |  |  |
| 001 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 011 | 1 |  |  |  |  |  |  |  |
| 010 | 1 |  |  |  |  |  |  |  |
| 110 |  |  |  | 1 |  |  |  |  |
| 111 |  |  |  | 1 |  | 1 |  |  |
| 101 |  |  |  | 1 |  | 1 |  |  |
| 100 |  |  |  | 1 |  |  |  |  |

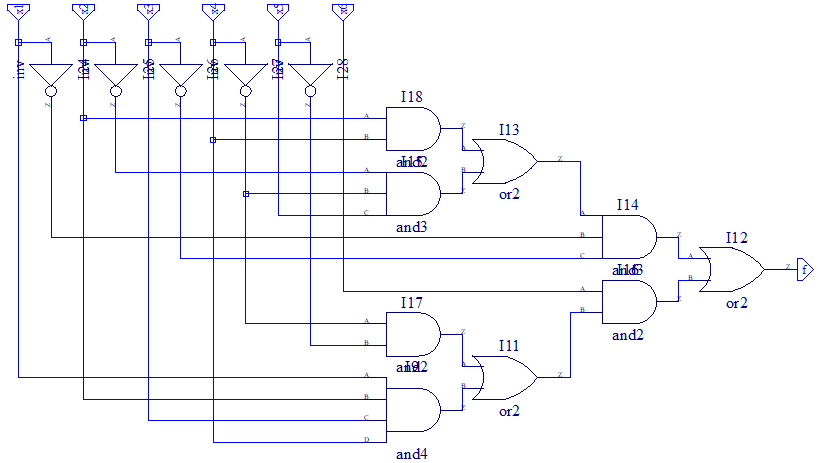
Pasinaudoję Karno diagrama, gausime minimizuotą funkcijos išraišką (į grupes jungiami mintermai nuspalvinti atitinkama spalva):

***f*** = ***x*1*x*2*x*3*x*4*x*5** + ***x*4*x*5*x*6** + ***x*1*x*2*x*3*x*4** + ***x*1*x*2*x*3*x*4*x*6.**

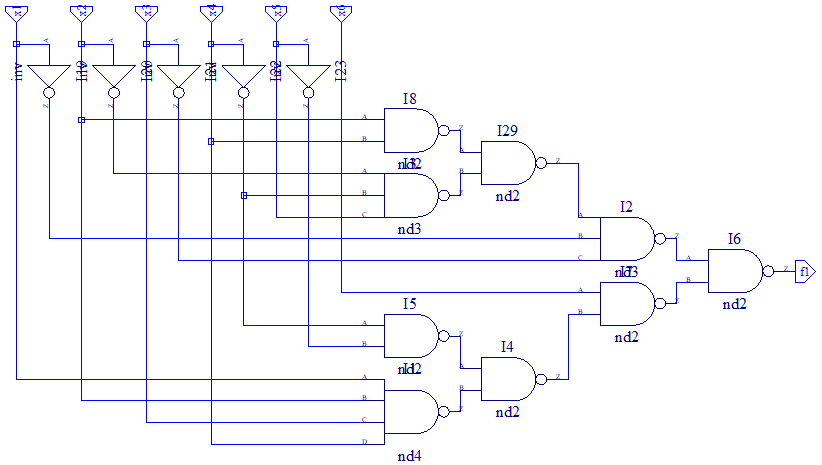
Funkcijos išraišką galima pertvarkyti, iškeliant bendrus konjunkcijų kintamuosius prieš skliaustus:

***f***= ***x*1*x*3**(***x*2*x*4*x*5** + ***x*2*x*4**) + ***x*6**(***x*4*x*5** + ***x*1*x*2*x*3*x*4**)*.*

* 1. Schemos realizacija.
     1. Šios funkcijos realizacija naudojant IR, ARBA, NE elementus parodyta [7](#_bookmark14) pav.



1. **pav. Funkcijos realizacija IR, ARBA, NE elementais**
   * 1. Remdamiesi anksčiau pastebėtu ryšiu tarp schemos iš IR, ARBA, NE elementų ir schemos iš IR-NE elementų ir invertorių, 7 paveiksle parodytą schemą, kurioje dviejų pakopų schemų fragmentai, kur pirmojoje pakopoje yra elementai IR, o antrojoje – juos apjungiantis elementas ARBA, išskirti rėmeliais, pertvarkome į 8 paveiksle parodytą schemą, kiekvieną IR ir ARBA elementą pakeisdami atitinkamą įvesčių skaičių turinčiu IR-NE elementu.



1. **pav. Funkcijos realizacija IR-NE elementais**
   * 1. Projektuojant schemą, naudojančią multiplekserį, reikia:

* iš normaliosios disjunkcinės išraiškos išskirti adresinius kintamuosius, kurių skaičius priklauso nuo to, kiek adreso linijų turi pasirinktas multiplekseris. Šiame pavyzdyje kaip adresiniai kintamieji panaudoti *x*1 ir *x*2;
* pertvarkyti išraišką, šiuos kintamuosius iškeliant prieš skliaustus.

Patogumo dėlei mūsų funkciją ***f*** = (**1*,* 2*,* 3*,* 9*,* 17*,* 20*,* 21*,* 22*,* 23*,* 25*,* 33*,* 41*,* 49*,* 57*,* 61*,* 63**) užrašykime kubų pavidalu, kur kubas – mintermo numerio dvejetainis atitikmuo:

|  |  |  |  |
| --- | --- | --- | --- |
| Mintermo numeris | Kubas | Mintermo numeris | Kubas |
| 1 | **00**0001 | 33 | **10**0001 |
| 2 | **00**0010 | 41 | **10**1001 |
| 3 | **00**0011 | 49 | **11**0001 |
| 9 | **00**1001 | 57 | **11**1001 |
| 17 | **01**0001 | 61 | **11**1101 |
| 20 | **01**0100 | 63 | **11**1111 |
| 21 | **01**0101 |  |  |
| 22 | **01**0110 |  |  |
| 23 | **01**0111 |  |  |
| 25 | **01**1001 |  |  |

Adresinius kintamuosius *x*1 ir *x*2 atitinkantys kubo elementai paryškinti. Iškėlus šiuos kintamuosius prieš skliaustus, išraiška atrodys taip:

***f***= ***x*1*x*2** (***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6**) +

***x*1*x*2**(***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6**) +

***x*1*x*2** (***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6**) +

***x*1*x*2**(***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6** + ***x*3*x*4*x*5*x*6**)***.***

Skliaustuose esančias išraiškas galima suprastinti (minimizuoti). Tuomet turėsime:

***f***= ***x*1*x*2**(***x*4*x*5*x*6** + ***x*3*x*4*x*5**) +

***x*1*x*2** (***x*4*x*5*x*6** + ***x*3*x*4**) +

***x*1*x*2** (***x*4*x*5*x*6**)+

***x*1*x*2**(***x*4*x*5*x*6** + ***x*3*x*4*x*6**)*.*

Kitas būdas išraiškai pertvarkyti, kad panaudotume multiplekserį, remiasi funkcijos Karno diagramos analize. Kaip adresinius kintamuosius naudojant *x*1 ir *x*2, diagramą galima suskaidyti į keturias dalis (jos čia skirtingai nuspalvintos, o kairėje nurodytas multiplekserio įvesties numeris):

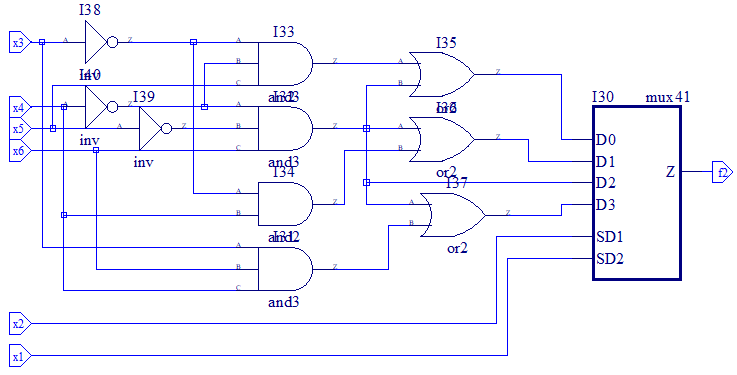
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **D0** | | **D1** | | **D3** | | **D2** | |
|  | *x*4*x*5*x*6 *x*1*x*2*x*3 | **00**0 | **00**1 | **01**1 | **01**0 | **11**0 | **11**1 | **10**1 | **10**0 |
|  | **000** |  |  |  |  |  |  |  |  |
| **001** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** |
|  | **011** | **1** |  |  |  |  |  |  |  |
| **010** | **1** |  |  |  |  |  |  |  |
|  | **110** |  |  |  | **1** |  |  |  |  |
| **111** |  |  |  | **1** |  | **1** |  |  |
|  | **101** |  |  |  | **1** |  | **1** |  |  |
| **100** |  |  |  | **1** |  |  |  |  |

Minimizavę kiekvienos dalies funkcijas, gausimę tą patį (aukščiau užrašytą) rezultatą. Šias funkcijos realizuojančios liekamosios prijungiamos prie atitinkamų multiplekserio įvesčių (D0-D3).

Pastabos:

* Liekamųjų funkcijų sudėtingumas priklauso nuo pasirinktų adresinių kintamųjų.
* Jeigu kuri nors liekamoji funkcija gaunama lygi nuliui arba vienetui, atitinkama multiplekserio įvestis prijungiama prie loginio nulio arba loginio vieneto.

Funkcijos realizacija panaudojant multiplekserį pateikta [9](#_bookmark16) pav.



1. **pav. Funkcijos realizacija panaudojant multiplekserį**

Visos trys schemos buvo nubraižytos tame pačiame Diamond lange, o jų išvestys pavadintos skirtingai: pirmosios (panaudojant IR, ARBA, NE elementus) – **f**, antrosios (panaudojant IR-NE elementus) – **f1**, trečiosios (panaudojant multiplekserį kartus su IR, ARBA, NE elementais) – **f2**. Tokiu atveju jos modeliuojamos kartu, ir galime palyginti visų trijų schemų modeliavimo rezultatus.

* 1. Testams generuoti panaudosime keletą VHDL kalbos procesų. Procesų VHDL kalbos teste kiekis neribotas, t. y., jų galima sukurti tiek, kiek reikia. Visi procesai vykdomi lygiagrečiai (vienu metu). Procesas yra begalinis: įvykdžius paskutinį priskyrimą vėl vykdomas pirmasis. Pavyzdyje testas sudarytas taip, kad būtų perrenkamos visos galimos įvesčių kombinacijos. Tam naudojami 6 procesai, atskirai valdantys kiekvieną įvestį. Pabaigoje testavimas stabdomas specialia komanda *assert*.

x6p: process

begin

x6 <= '0'; wait for 10 ns;

x6 <= '1'; wait for 10 ns;

end process ;

x5p: process

begin

x5 <= '0'; wait for 20 ns;

x5 <= '1'; wait for 20 ns;

end process ;

x4p: process

begin

x4 <= '0'; wait for 40 ns;

x4 <= '1'; wait for 40 ns;

end process ;

x3p: process

begin

x3 <= '0'; wait for 80 ns;

x3 <= '1'; wait for 80 ns;

end process ;

x2p: process

begin

x2 <= '0'; wait for 160 ns;

x2 <= '1'; wait for 160 ns;

end process ;

x1p: process

begin

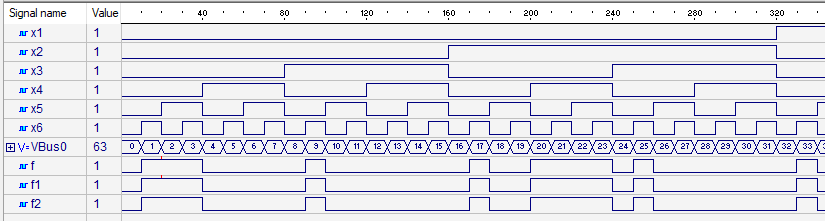
x1 <= '0'; wait for 320 ns;

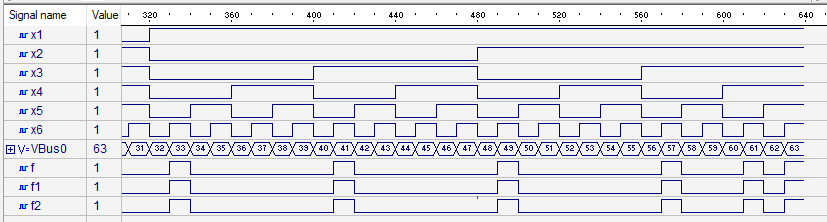
x1 <= '1'; wait for 320 ns;

assert false report " Pabaiga " severity failure ;

end process ;

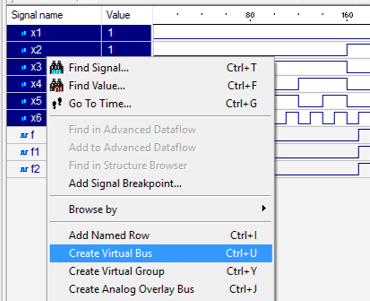
Suredagavus *\*.vhd* failą ir sukompiliavus testinius vektorius, modeliavimas davė rezultatus, parodytus [10](#_bookmark17) paveiksle.

Rezultatų analizės patogumui kintamieji ***x*1**, . . . , ***x*6** buvo apjungti į vadinamąją virtualią magistralę (angl. *Virtual bus*). Tam pažymime kintamuosius (jie būtinai turi būti surikiuoti, pradedant ***x*1** ir baigiant ***x*6**), spragtelime juos dešiniuoju pelės klavišu ir išskleistame kontekstiniame meniu pasirenkame *Create* *Virtual Bus* (11 pav.). Po to spragtelime dešiniuoju pelės klavišu pavadinimą **VBus0** ir kontekstiniame meniu pasirenkame **Radix** 🡪 **Decimal Radix**, kad magistralės truinys būtų rodomas dešimtainėje sistemoje. Tai ir bus mintermų numeriai.



1. **pav. Schemos modeliavimo laiko diagramos**

Matome, kad kai įvesčių reikšmės įgauna vieną iš užduotyje nurodytų reikšmių, iš- vesčių ***f***, ***f1*** ir ***f2***reikšmės įgyja vienetinę reikšmę. Visais kitais atvejais išvesčiųreikšmės lygios 0. Iš laikinės diagramos matome, kad suprojektuotos schemos veikia tinkamai.



1. **pav. Virtualios magistralės formavimas**