

UNIVERSITATEA „*TRANSILVANIA*” DIN BRASOV

Facultatea de *Inginerie Electrica* si *Stiinta Calculatoarelor*

Catedra de *Automatica* si *Informatica Aplicata*

2008

**Proiect ASCN**

Tema Proiect Nr. 25

Să se proiecteze un convertor de cod de 4 biţi pentru conversia codului binar zecimal ponderat 2421 în codul binar zecimal ponderat 4221 (logică combinaţională)

Cuprins

**Introducere in circuite logice combinationale2**

**Enuntul proiectului3**

**Tabel de adevar4**

**Formele canonice disjunctive si conjunctive ale functiilor5**

**Formele minime disjunctive si conjunctive ale functiilor prin metoda diagramelor Karnaugh7**

**Formele minime disjunctive pentru primele doua functii cu metoda Quine-McCluskey10**

**Implementarea funcţiilor folosind porţi logice ŞI-NU în tehnologie TTL 15**

**Implementarea ansamblului functiilor folosind porţi logice ŞI-NU în tehnologie TTL 18**

**Implementarea ansamblului folosind porţi SAU-NU în tehnologia TTL pentru primele două funcţii şi porţi SI-NU în tehnologia CMOS pentru următoarele două funcţii 20**

**Implementarea schemei cu MUX-uri în tehnologie TTL22**

**Implementarea funcţiei A cu 2xMUX de 8 căi conectate în paralel 22**

**Implementarea funcţiei B cu MUX de 8 căi şi o variabilă aplicată pe intrările de date 24**

**Implementarea funcţiei C cu MUX de 16 căi 26**

**Implementarea schemei cu DMUX-uri în tehnologie CMOS27**

**Implementarea lui D cu DMUX-uri de 8 căi conectate în paralel 27**

**Implementarea funcţiei B cu DMUX 8 căi şi reţea de porţi logice29**

**Implementare C cu DMUX 16 căi şi porţi logice ŞI-NU31**

**Implementarea funcţiei logice D cu DMUX de 16 căi şi porti ŞI 32**

**Date de catalog ale circuitelor integrate folosite34**

**Calcularea timpilor de propagare „intrare-ieşire”34**

**Calculul puterilor disipate35**

**Concluzii finale36**

**Bibliografie36**

**Introducere :**

Un circuit logic combinaţional este un circuit de comutare combinaţională ce se caracterizează prin aceea că la un moment dat starea ieşirilor circuitului depinde doar de starea intrărilor sale. Legătura dintre starea ieşirilor şi starea intrărilor este dată şi în acest caz de funcţia de transfer a circuitului.

Suportul fizic utilizat în CLC este destul de variat. Studiul CLC şi al celor secvenţiale se face folosind un model al acestora numit reţea de comutare sau schemă logică. În reţeaua de comutare sau schema logică se face abstracţie de caracteristicile constructive ale elementelor ce concep reţeaua sau schema logică avându-se în vedere doar proprietăţile lor funcţionale. Din acest motiv reţeaua are un înalt grad de generalitate, permiţând studiul unor clase largi de circuite logice.

Schema bloc generală (modelul general) al unui CLC reprezentat printr-o reţea de comutare sau schemă logică este următorul:

Retea de comutare

X1, X2, …,Xn – setul valorilor de intrare

Z1, Z2, …, Zm – setul valorilor de iesire

Relaţii generale între aceste valori:

f(X1, …,Xn) fi – funcţii logice

Z1 = ……………. Xi – variabile logice

Zm = f(X1, …,Xn)

Analiza circuitelor logice combinaţionale

Prin analiza unui CLC se înţelege obţinerea expresiilor mărimilor de ieşire cunoscându-se setul variabilelor de intrare **** si configuraţia reţelei. Înţelegem numărul şi tipul de elemente logice care intră în componentă ,modul de conectare, punctul în care se aplică variabilele de intrare, numărul de nivele logice etc.

Analiza reţelelor CLC realizate cu elemente logice de tip inversor (ŞI-NU, SAU-NU): numărul maxim de elemente logice aflate între intrarea şi ieşirea reţelei determinã numărul de nivele logice ale acestuia. Numerotarea lor se face de la ieşire către intrare. Într-o reţea de comutare realizate cu elemente ŞI-NU respectiv SAU-NU s-a observat cã o variabilã de intrare apare negată în expresia ieşirii în cazul în care a parcurs un număr impar de elemente de inversare si necomplementată dacă a parcurs un număr par de astfel de elemente.

**Temã Proiect Nr.25**

Să se proiecteze un convertor de cod de 4 biţi pentru conversia codului binar zecimal ponderat 2421 în codul binar zecimal ponderat 4221 (logică combinaţională).Proiectul va cuprinde următoarele puncte:

a) Să se exprime funcţiile logice asociate circuitului combinaţional cu FCD (forma canonică disjunctivă), FCC (forma canonică conjunctivă) tabel de adevăr şi di-agrame Karnaugh.

b) Să se obţină formele minime disjunctive şi conjunctive pentru funcţiile logice asociate convertorului de cod, (utilizând combinaţiile indiferente) prin metoda diagramelor Karnaugh; de asemenea se vor obţine formele minime disjunctive pentru **ultimele** două funcţii logice de ieşire şi prin metoda Quine-McCluskey.

c) Să se implementeze fiecare funcţie, independent, numai cu porţi logice ŞI-NU (porţile logice sunt realizate în tehnologia TTL).

d) Să se implementeze ansamblul funcţiilor logice numai cu porţi logice ŞI-NU (porţile logice sunt realizate în tehnologia TTL).

e) Să se implementeze ansamblul funcţiilor logice în următoarea variantă: primele două funcţii logice de ieşire cu porţi logice **SAU-NU**, realizate în tehnologia **CMOS**, iar următoarele două cu porţi logice **ŞI-NU**, realizate în tehnologia **TTL**.

f) Să se implementeze ansamblul funcţiilor logice cu MUX-uri de 8 respectiv 16 căi (circuitele sunt realizate în tehnologia **TTL**).

g) Să se implementeze ansamblul funcţiilor logice cu DMUX-uri de 8 respectiv 16 căi şi porţi logice ŞI-NU în prima variantă, respectiv ŞI în a doua variantă (toate circuitele sunt realizate în tehnologia **CMOS**).

h)Să se calculeze timpii de propagare „intrare-ieşire”, pentru toate schemele logice obţinute.

i) Să se calculeze puterile disipate pentru toate schemele logice obţinute.

j) Să se compare soluţiile de implementare obţinute.

k) Se va face analiza, prin simulare, a tuturor schemelor logice obţinute utilizându-se pachetul de programe OrCAD.

Pe schemele logice obţinute se vor specifica tipul şi gradul de utilizare al fiecărui circuit integrat.

TEMĂ PROIECT Nr.25

Pentru a putea scrie o functie sub una din formele ei canonice, trebuie cunoscut *tabelul de adevar* unde se trec toate combinatiile liniare ale variabilelor de intrare, adica 24 = 16 combinatii.

**Tabel de adevar**:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Echiv.  Binar | Nr. Crt. | Codul 2421 | | | | Codul 4221 | | | |
| X1 | X2 | X3 | X4 | a | b | c | d |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 2 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 3 | 3 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 4 | 4 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 11 | 5 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 12 | 6 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 13 | 7 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 14 | 8 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 15 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

a) Pentru a obtine din tabelul de adevar FCC (forma canonica conjunctiva) – se iau in considerare combinatiile pentru care o functie are valoarea 0, iar pentru FCD (forma canonica disjunctiva) – se iau in considerare combinatiile pentru care functia are valoarea 1.

Combinatiile indiferente:

\*P5= = 0 1 0 1 \*P8= = 1 0 0 0

\*P6= = 0 1 1 0 \*P9= = 1 0 0 1

\*P7= = 0 1 1 1 \*P10= = 1 0 1 0

**aFCD** =

= P11 + P12 + P13 + P14 + P15 =**Σ** (11, 12, 13, 14, 15);

**aFCC** =

= S0 \* S1 \* S2 \* S3 \* S4 = **Π** (0, 1, 2, 3, 4);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 Diagrama Karnaugh ***Pentru functia “a”.*** |
| 00 | **0** | **0** | **1** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **0** | **\*** | **1** | **1** |
| 10 | **0** | **\*** | **1** | **\*** |

**bFCD** =

= P4 + P12 +P13 +P14 +P15 = **Σ** (4, 12, 13, 14, 15);

**bFCC** =

= S0 \* S1 \* S2 \* S3 \* S11= **Π** (0, 1, 2, 3, 11);

Diagrama Karnaugh pentru functia “b”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **1** | **1** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **0** | **\*** | **1** | **0** |
| 10 | **0** | **\*** | **1** | **\*** |

**cFCD** =

= P2 + P3 + P4 + P14 + P15 = **Σ** (2, 3, 4, 14,15);

**cFCC** =

= S0 \* S1 \* S11 \* S12 \* S13 = **Π** (0, 1, 11, 12, 13);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10  Diagrama Karnaugh pentru functia “c”. |
| 00 | **0** | **1** | **0** | **\*** |
| 01 | **0** | **\*** | **0** | **\*** |
| 11 | **1** | **\*** | **1** | **0** |
| 10 | **1** | **\*** | **1** | **\*** |

**dFCD** = = P1 + P3 + P11 + P13 + P15 =**Σ** (1, 3, 11, 13, 15);

**dFCC** =

= S0 \* S2 \* S4 \* S12 \* S14 = **Π** (0, 2, 4, 12, 14);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10  Diagrama Karnaugh pentru functia “d”. |
| 00 | **1** | **0** | **0** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **1** | **\*** | **1** | **1** |
| 10 | **0** | **\*** | **0** | **\*** |

b) Metoda diagramelor „K” este o metoda grafo-analitica foarte utila la minimizarea functiilor cu un numar relativ mic de variabile. Metoda porneste de la una din formele canonice ale functiei.

Diagramele Karnaugh se prezinta sub forma unui patrat (sau dreptunghi) cu 2n locatii, in cazul nostru cu 16 locatii. In fiecare locatie va apare un termen canonic al functiei.

Diagrama este astfel organizata, incat doua componente vecine pe linie sau pe coloana, sa difere printr-o aceeasi variabila, variabila care intr-o combinatie sa apara negata, si in alta adevarata (proprietatea de adiacenta).

Formele minime disjunctive (FMD) prin diagrame Karnaugh:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **0** | **1** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **0** | **\*** | **1** | **1** |
| 10 | **0** | **\*** | **1** | **\*** |

**aFMD =**;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **1** | **1** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **0** | **\*** | **1** | **0** |
| 10 | **0** | **\*** | **1** | **\*** |

**bFMD =**;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **1** | **0** | **\*** |
| 01 | **0** | **\*** | **0** | **\*** |
| 11 | **1** | **\*** | **1** | **0** |
| 10 | **1** | **\*** | **1** | **\*** |

**cFMD =**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **0** | **0** | **\*** |
| 01 | **1** | **\*** | **1** | **\*** |
| 11 | **1** | **\*** | **1** | **1** |
| 10 | **0** | **\*** | **0** | **\*** |

**dFMD =**;

Formele minime connjunctive (FMC) prin diagrame Karnaugh:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **0** | **1** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **0** | **\*** | **1** | **1** |
| 10 | **0** | **\*** | **1** | **\*** |

**aFMC =** ;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **1** | **1** | **\*** |
| 01 | **0** | **\*** | **1** | **\*** |
| 11 | **0** | **\*** | **1** | **0** |
| 10 | **0** | **\*** | **1** | **\*** |

**bFMC =**;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **1** | **0** | **\*** |
| 01 | **0** | **\*** | **0** | **\*** |
| 11 | **1** | **\*** | **1** | **0** |
| 10 | **1** | **\*** | **1** | **\*** |

**cFMC =**;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X1,X2X3,X4 | 00 | 01 | 11 | 10 |
| 00 | **0** | **0** | **0** | **\*** |
| 01 | **1** | **\*** | **1** | **\*** |
| 11 | **1** | **\*** | **1** | **1** |
| 10 | **0** | **\*** | **0** | **\*** |

**dFMC =** .

**Formele minime disjunctive pentru primele două funcţii:**

**- cu metoda Quine-McCluskey -**

Această metodă porneşte de la forma canonică a funcţiei de minimizat. Metoda are două etape:

* + se determină implicanţii primi;
  + se selectează dintre implicanţii primi obţinuţi doar aceia care acoperă toţi termenii canonici ai funcţiei date şi asigură realizarea acesteia la un cost minim.

Termenii canonici se compară în felul următor:

* + se compară fiecare termen canonic cu toţi ceilalţi;
  + când se găsesc doi termeni care au proprietate de adicenţă, variabila redundantă se elimină, obţinându-se un termen elementar;
  + primul ciclu de comparaţii se consideră încheiat în momentul în care s-au comparat între ei toţi termenii canonici, obţinându-se toţi implicanţii primi posibili;
  + se compară între ei pe acelaşi criteriu termenii elementari obţinuţi;
  + se vor face atâtea cicluri de comparaţie câte sunt necesare, pentru a nu mai exista termeni elementari cu proprietatea de adiacenţă.

Combinaţii indiferente se vor compara cu celelalte, dar nu se vor compara între ele.

Dacă nu se compară o combinaţie, ea ar putea fi conţinută de forma minimă a funcţiei. În primul ciclu de comparaţii s-au comparat toate combinaţiile posibile. Se trece la al doilea ciclu de comparaţii.

**1. cFCD** =

= P2 + P3 + P4 + P14 + P15 = **Σ** (2, 3, 4, 14,15)

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 2,3 | 0 0 1 - | A |
| 2,6\* | 0 - 1 0 | + |
| 2,10\* | - 0 1 0 | + |
| 4,5\* | 0 1 0 - | B |
| 4,6\* | 0 1 - 0 | C |
| 2 | 3,7\* | 0 - 1 1 | + |
| 6\*,14 | - 1 1 0 | + |
| 10\*,14 | 1 - 1 0 | + |
| 3 | 14,15 | 1 1 1 - | D |
| 7\*,15 | - 1 1 1 | + |

P2 = = 0 0 1 0 \*P5= = 0 1 0 1

P3= = 0 0 1 1 \*P6= = 0 1 1 0

P4= = 0 1 0 0 \*P7= = 0 1 1 1

P14= = 1 1 1 0 \*P8= = 1 0 0 0

P15= = 1 1 1 1 \*P9= = 1 0 0 1

\*P10= = 1 0 1 0

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 2 | 0 0 1 0 | + |
| 4 | 0 1 0 0 | + |
| 8\* | 1 0 0 0 |  |
| 2 | 3 | 0 0 1 1 | + |
| 5\* | 0 1 0 1 | + |
| 6\* | 0 1 1 0 | + |
| 9\* | 1 0 0 1 |  |
| 10\* | 1 0 1 0 | + |
| 3 | 14 | 1 1 1 0 | + |
| 7\* | 0 1 1 1 | + |
| 4 | 15 | 1 1 1 1 | + |

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 2,6\*,3,7\* | 0 - 1 - | E |
| 2,6\*,10\*,14 | - - 1 0 | F |
| 2,10\*,6\*,14 | - - 1 0 | F |
| 2 | 6\*,14,7\*,15 | - 1 1 - | G |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **P2** | **P3** | **P4** | **P14** | **P15** |
| **A** | **\*** | **\*** |  |  |  |
| **B** |  |  | **\*** |  |  |
| **C** |  |  | **\*** |  |  |
| **D** |  |  |  | **\*** | **\*** |
| **E** | **\*** | **\*** |  |  |  |
| **F** | **\*** |  |  | **\*** |  |
| **G** |  |  |  | **\*** | **\*** |

**cFMD =**

**dFCD** =

= P1 + P3 + P11 + P13 + P15 = **Σ** (1, 3, 11, 13, 15);

P1== 0 0 0 1 \*P5== 0 1 0 1

P3== 0 0 1 1 \*P6== 0 1 1 0

P11== 1 0 1 1 \*P7== 0 1 1 1

P13== 1 1 0 1 \*P8== 1 0 0 0

P15== 1 1 1 1 \*P9== 1 0 0 1

\*P10==1 0 1 0

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 1,3 | 0 0 - 1 | + |
| 1,5\* | 0 - 0 1 | + |
| 1,9\* | - 0 0 1 | + |
| 2 | 3,7\* | 0 - 1 1 | + |
| 3,11 | - 0 1 1 | + |
| 5\*,13 | - 1 0 1 | + |
| 9\*,11 | 1 0 - 1 | + |
| 9\*,13 | 1 - 0 1 | + |
| 10\*,11 | 1 0 1 - | A |
| 3 | 7\*,15 | - 1 1 1 | + |
| 11,15 | 1 - 1 1 | + |
| 13,15 | 1 1 - 1 | + |

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 1 | 0 0 0 1 | + |
| 8\* | 1 0 0 0 |  |
| 2 | 3 | 0 0 1 1 | + |
| 5\* | 0 1 0 1 | + |
| 6\* | 0 1 1 0 |  |
| 9\* | 1 0 0 1 | + |
| 10\* | 1 0 1 0 | + |
| 3 | 7\* | 0 1 1 1 | + |
| 11 | 1 0 1 1 | + |
| 13 | 1 1 0 1 | + |
| 4 | 15 | 1 1 1 1 | + |

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 1,3,9\*,11 | - 0 - 1 |  |
| 1,5\*,3,11 | 0 - - 1 |  |
| 1,5\*,9\*,13 | - - 0 1 |  |
| 1,9\*,3,11 | - 0 - 1 |  |
| 1,9\*,5\*,13 | - - 0 1 |  |
| 2 | 3,7\*,11,15 | - - 1 1 |  |
| 3,11,7\*,15 | - - 1 1 |  |
| 5\*,13,7\*,15 | - 1 - 1 |  |
| 9\*,11,13,15 | 1 - - 1 |  |
| 9\*,13,11,15 | 1 - - 1 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 1,3,9\*,11 | - 0 - 1 | + |
| 1,5\*,3,11 | 0 - - 1 | + |
| 1,5\*,9,13 | - - 0 1 | + |
| 2 | 3,7\*,11,15 | - - 1 1 | + |
| 5\*,13,7\*,15 | - 1 - 1 | + |
| 9\*,13,11,15 | 1 - - 1 | + |

|  |  |  |  |
| --- | --- | --- | --- |
| Grupa | Indicii  Term. |  | Comp. |
| 1 | 1,3,9\*,11,5\*,13,7\*,15 | - - - 1 | B |
| 1,5\*,3,11,9\*,13,11,15 | - - - 1 | B |
| 1,5\*,9\*,13,3,7\*,11,15 | - - - 1 | B |

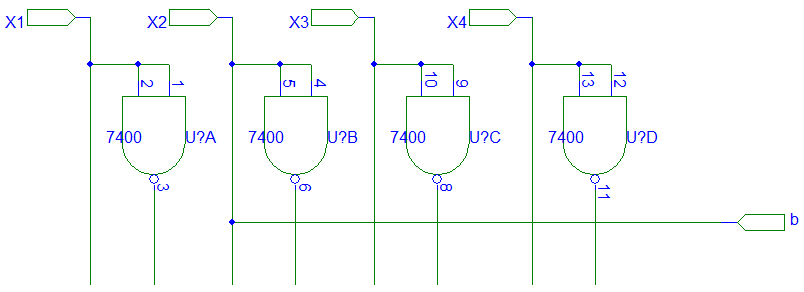
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **P1** | **P3** | **P11** | **P13** | **P15** |
| **A** |  |  | **\*** |  |  |
| **B** | **\*** | **\*** | **\*** | **\*** | **\*** |

**dFMD=**

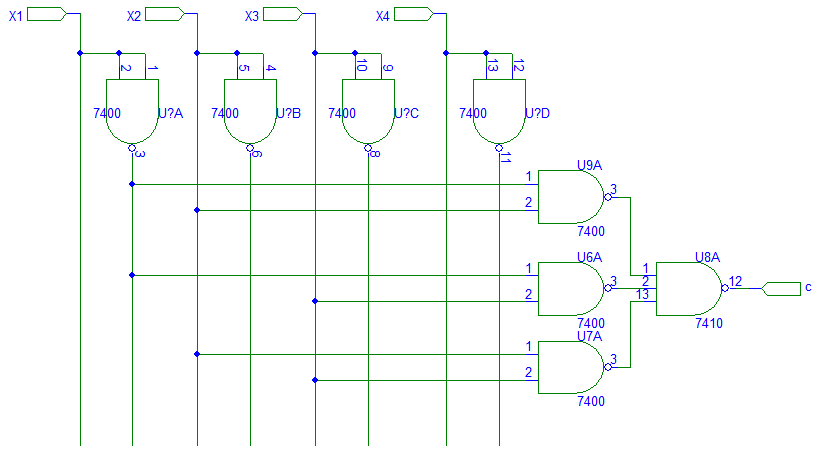
**Implementarea functiilor (individual) numai cu porti logice SI-NU (realizate in tehnologia TTL).**

**aFMD =**;



**bFMD =**;

**cFMD =** ;

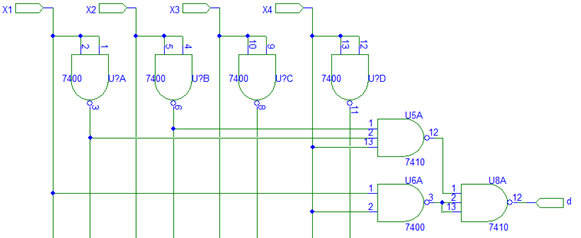


Incapsularea:



S-au folosit următoarele circuite integrate: 1x74LS00 şi 1x74LS10

**dFMD =**;

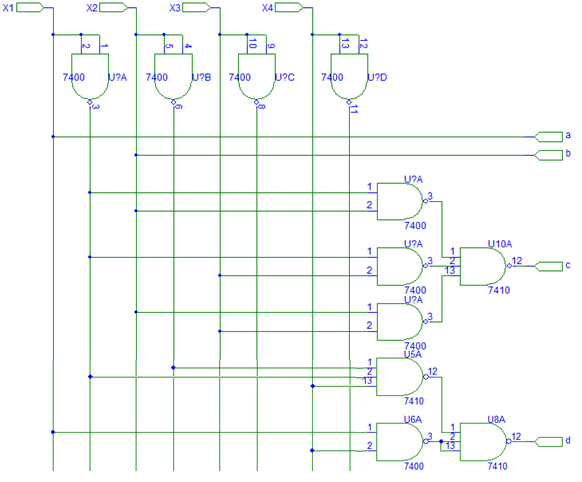


Incapsularea:



S-au folosit următoarele circuite integrate: 1x74LS00 şi 1x74LS10

**Ansamblul functiilor logice realizat numai cu porti logice SI-NU (porti logice realizate in tehnologia TTL).**



Incapsularea:

** **



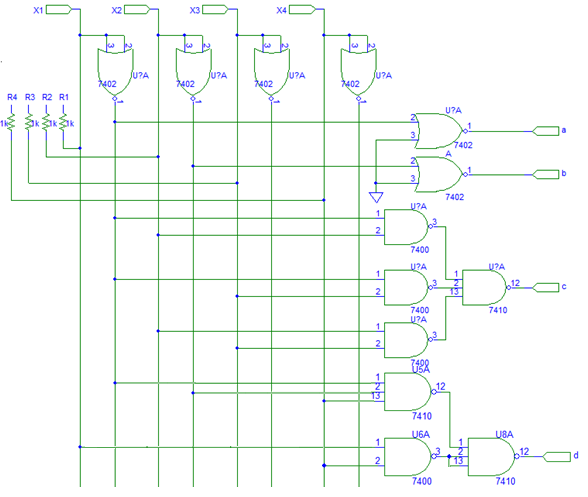
S-au folosit următoarele circuite integrate: 2x74LS00 şi 1x74LS10

**Implementarea ansamblului functiilor logice:**

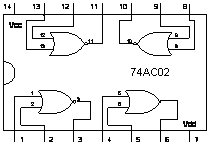
* primele doua porti prin SAU-NU (tehnologia CMOS);
* ultimele doua porti prin SI-NU (tehnologia TTL).

Pentru implementare cu funcţii SAU-NU se pleacă de la forma minimă obţinută pentru realizarea implementării cu funcţii ŞI-NU aplicând formulele lui De-Morgan.

Am negat primele doua functii prin 2 porti SAU-NU, chiar daca nu era nevoie, pentru a ma folosi de portile SAU-NU ( ) .



Incapsularea:





S-au folosit următoarele circuite integrate: 2x74AC02, 1x74LS00 şi 1x74LS10

**Implementarea schemei cu MUX-uri în tehnologie TTL**

Un circuit de multiplexare este un circuit logic combinaţional care, în cazul general, are 2n intrări de date (I2n-1 … I2 I1 I0), n intrări de selecţie (S0 S1 … Sn+1) şi o ieşire (Z).

Expresia ieşirii Z la un moment dat este dată de intrare Ik, k=0, … ,2n-1 unde k reprezintă echivalentul zecimal al numărului binar dat de stările 1 şi 0 ale intrărilor de selecţie: k=Sn-1, Sn-2, … , S1, S0.

**Implementarea funcţiei logice a cu 2xMUX de 8 căi conectate în paralel**

La implementarea unei funcţii logice cu 2 multiplexoare de 8 căi legate în paralel variabilele funcţiei de implementate se aplică astfel : prima variabilă (cea mai semnificativă ) se aplică pe intrarea de strobare la primul multiplexor aşa cum este ea iar la al multiplexor doilea negată, celelalte variabile aplicându-se în ordinea semnificativitaţii lor pe intrările de date. Pe intrările de selecţie se vor aplica semnale corespunzătoare echivalenţilor zecimali prezenţi în expresia funcţiei urmând ca ieşirile celor două multiplexoare să fie legate într-o poartă SAU.





**74LS151 74LS04 74LS32**

**Incapsulare:**

****

S-au folosit următoarele circuite integrate: 2x74LS151, 1x74LS32 şi 1x74LS04

**Implementarea funcţiei logice b cu MUX de 8 căi şi o variabilă aplicată pe intrările de date**

La implementarea unei functii cu un multiplexor de 8 căi şi o variabilă aplicată pe intrările de date se pleaca de la tabelul de adevăr al funcţiei de implementat separându-se variabila cea mai puţin semnificativă. Astfel din cele *n* variabile se separă *n-1* având ponderile cele mai mari si se plică pe intrările de selecţie ale multiplexorului. Astfel se obţin termeni canonici de *n-1* variabile. Pentru a obţine însă termenii canonici de *n* variabile prezenţi în expresia funcţiei de implementat trebuie adăugată variabila separată. Acest lucru se realizează prin aplicarea variabilei pe intrările de date ale MUX-ului

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Echiv.  zec. | Variabilele de intrare | | | | Funcţia |
| **X1** | **X2** | **X3** | **X4** | **b** |
| **0** | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 0 | 1 | 0 |
| **2** | 0 | 0 | 1 | 0 | 0 |
| **3** | 0 | 0 | 1 | 1 | 0 |
| **4** | 0 | 1 | 0 | 0 | 1 |
| **11** | 1 | 0 | 1 | 1 | 0 |
| **12** | 1 | 1 | 0 | 0 | 1 |
| **13** | 1 | 1 | 0 | 1 | 1 |
| **14** | 1 | 1 | 1 | 0 | 1 |
| **15** | 1 | 1 | 1 | 1 | 1 |

Astfel pentru : X1X2X3=000 b=0 oricare ar fi X4 =>D0=0

X1X2X3=001 b=0 oricare ar fi X4 =>D1=0

X1X2X3=010 b=1 pentru X4 =0 =>D2= 

X1X2X3=101 b=0 pentru X4 =1=>D3= *X*4

X1X2X3=110 b=1 oricare ar fi X4 =>D4=1

X1X2X3=111 b=1 oricare ar fi X4 =>D5=1

****

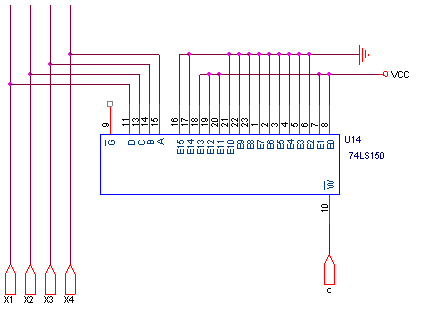
****

Incapsulare:

S-au folosit următoarele circuite integrate: 1x74LS151 şi 2x74LS04

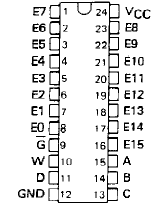
**Implementarea funcţiei logice c cu MUX de 16 căi**

La implementarea cu multiplexoare de 16 căi variabilele funcţiei se vor aplica pe intrările de date. Deoarece circuitul are ieşiri active în 0 logic rezultă că pe intrările de selecţie se vor lega la 1 logic termenii canonici care nu sunt prezenţi în expresia funcţiei iar termenii canonici prezenţi în expresia funcţiei se vor lega la 0 logic.



**Incapsulare:**





**74LS150**

S-au folosit următoarele circuite integrate: 1x74LS150

**Implementarea schemei cu DMUX-uri în tehnologie CMOS**

Demultiplexoarele sunt circuite combinaţionale care, în cazul general au o intrare de date I, n intrări de selecţie S0, S1, …, Sn-1 si 2ieşiri Z0, Z1, …, Z2.

Pentru implementarea funcţiilor cu DMUX variabilele funcţiei se vor aplica pe intrările de selecţie în raport cu ponderile acestora.

Pentru realizarea nivelului logic SAU se pot folosi porti:

1. ŞI-NU, în acest caz se leagă la intrările porţilor ŞI-NU ieşirile DMUX corespunzătoare termenilor canonici prezentaţi în expresia funcţiei de implementat.
2. ŞI, în acest caz se leagă la intrările porţilor ŞI ieşirile DMUX corespunzătoare termenilor canonici care nu apar în expresia funcţiei (se vor considera funcţiile negate).

În cazul în care funcţia are, în caz general, n variabile şi se impune să se implementeze un DMUX 1:2 se va separa variabila cu ponderea cea mai mare, iar cele n-1 variabile de stare se vor aplica pe intrarile DMUX în raport cu ponderile lor. Deoarece la ieşirile acestor circuite se obţin termeni canonici de n-1 variabile, iar în forma în care a fost funcţia de implementat sunt termeni canonici de n variabile, este necesar sa se adauge si variabila lipsă. Acest lucru se realizeaza prin intermediul unei reţele cu porţi logice. La intrările unei porţi ŞI se vor aplica ieşirile DMUX-ului corespunzătoare termenilor canonici care nu apar în expresia funcţiei (se ia în considerare negata funcţiei).

**Implementarea funcţiei logice d cu DMUX-uri de 8 căi conectate în paralel:**

La implementarea du demultipelxoare de 8 căi cele două demultiplexoare se conectează în paralel pe intrările de selecţie respectiv de strobare cu variabila cea mai semnificativă aplicată pe intrările de strobare pentru primul demultiplexor aşa cum este ea si pentru al doilea negată. Deoarece demultiplexorul are ieşiri active în 0 logic rezultă că pentru a realiza însumarea termenilor canonici prezenţi în expresia funcţiei se pleacă de la negata acestora aplicată prin porţi ŞI-NU, conform regulilor lui De-Morgan.





**74HC10 74HC04 74HC155**



**Incapsularea:**

S-au folosit următoarele circuite integrate: 2x74HC155, 1x74HC10

şi 1x74HC04

**Implementarea funcţiei logice b cu DMUX 8 căi şi reţea de porţi logice**

La implementarea unei funcţii cu DMUX-uri de 8 căi şi reţea de porţi logice dintre cele *n* variabile ale funcţiei se separă *n-1* având ponderile cele mai mici şi se aplică pe intrările DMUX-ului. Astfel la ieşirile acestuia se obţin termeni canonici de *n-1* variabile. Pentru a obţine termeni canonici de *n* variabile trebuie adaugată variabila separată . Acest lucru se face în exteriorul DMUX-ului de obicei printr-o reţea de porţi logice. Deoarece ieşirile DMUX-ului sunt active în 0 logic rezultă că pentru realizarea nivelului SAU se folosesc porţi logice ŞI-NU ale funcţiei negate.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Echiv.  zec. | Variabilele de intrare | | | | Funcţia |
| **X1** | **X2** | **X3** | **X4** | **b** |
| **0** | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 0 | 1 | 0 |
| **2** | 0 | 0 | 1 | 0 | 0 |
| **3** | 0 | 0 | 1 | 1 | 0 |
| **4** | 0 | 1 | 0 | 0 | 1 |
| **11** | 1 | 0 | 1 | 1 | 0 |
| **12** | 1 | 1 | 0 | 0 | 1 |
| **13** | 1 | 1 | 0 | 1 | 1 |
| **14** | 1 | 1 | 1 | 0 | 1 |
| **15** | 1 | 1 | 1 | 1 | 1 |

b =



**Incapsularea:**

****

S-au folosit următoarele circuite integrate: 1x74HC155 şi 2x74HC00

**Implementarea funcţiei logice c cu DMUX 16 căi şi porţi logice ŞI-NU :**

****

**Incapsularea:**

****

S-au folosit următoarele circuite integrate: 1x74HC154 şi 1x74HC10

**Implementarea funcţiei logice d cu DMUX de 16 căi şi porti ŞI**



**74HC11**



**Incapsularea:**

****

S-au folosit următoarele circuite integrate: 1x74HC154 şi 1x74HC11

**Date de catalog ale circuitelor integrate folosite**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **C.I.** | **Tehnologie** | **tPLH**  **[ns]** | **tPHL**  **[ns]** | **ICCH(TTL)**  **IOH(CMOS)**  **[mA]** | **ICCL(TTL)**  **IOL(CMOS)**  **[mA]** | **tp**  **[ns]** | **Pd**  **[mW]** |
| 74LS00 | TTL | 10 | 10 | 0.8 | 2.4 | 10 | 8 |
| 74LS10 | TTL | 10 | 10 | 0.6 | 1.8 | 10 | 6 |
| 74AC02 | CMOS | 4.5 | 4.5 |  |  | 4.5 | 115 |
| 74LS151 | TTL | 27 | 18 |  |  | 22.5 | 145 |
| 74LS04 | TTL | 10 | 10 |  |  | 10 | 60 |
| 74LS32 | TTL | 14 | 14 |  |  | 14 | 140 |
| 74LS150 | TTL | 27 | 18 |  |  | 22.5 | 200 |
| 74HC10 | CMOS | 9 | 9 |  |  | 9 | 187.5 |
| 74HC04 | CMOS | 7 | 7 |  |  | 7 | 185 |
| 74HC155 | CMOS | 13 | 13 |  |  | 13 | 185 |
| 74HC00 | CMOS | 7 | 7 |  |  | 7 | 175 |
| 74HC154 | CMOS | 14 | 14 |  |  | 14 | 185 |
| 74HC11 | CMOS | 10 | 10 |  |  | 10 | 187.5 |

**\***Pentru aflarea puterii disipate a integratelor in tehnologie CMOS am folosit formula:

, unde Vcc=5 V, fi=105 , CL=50 pF, iar Cp reprezinta puterea disipata si se gaseste in datele de catalog ale integratelor.

**Calcularea timpilor de propagare „intrare-ieşire”**

Timpii de propagare se vor calcula după formula:

Calculul timpilor de propagare pentru fiecare din schemele următoare:

Implementarea funcţiei logice **C** cu porţi logice ŞI-NU:

tp=0.5(10+10)\*2+0.5(10+10)=**30ns**

Implementarea funcţiei logice **D** cu porţi logice ŞI-NU:

tp=0.5(10+10)\*2+0.5(10+10)=**30ns**

Implementarea **ansamblului** folosind porţi ŞI-NU:

tp=0.5(10+10)\*2+0.5(10+10)=**30ns**

Implementarea **ansamblului** folosind atât porţi SAU-NU cât şi porţi ŞI-NU, vom lua timpul cel mai lung prin circuit:

tp=0.5(10+10)\*2+0.5(10+10)=**30ns**

Implementarea lui **A** cu un 2xMUX de 8 căi conectate în paralel:

tp=0.5(10+10)+0.5(27+18)+0.5(14+14)=10+22.5+14=**46.5ns**

Implementarea lui **B** cu MUX de 8 căi:

tp=0.5(10+10)+0.5(27+18)=**32.5ns**

Implementarea lui **C** cu un MUX de 16 căi:

tp=0.5(27+18)=**22.5ns**

Implementarea lui **D** cu 2xDMUX-uri de 8 căi conectate în paralel:

tp=0.5(7+7)+0.5(13+13)+0.5(9+9)\*2=**38ns**

Implementarea lui **B** cu un DMUX de 8 căi şi reţea de porţi logice:

tp=0.5(13+13)+0.5(10+10)\*3=**43ns**

Implementarea lui **C** cu DMUX 16 căi şi porţi logice ŞI-NU:

tp=0.5(14+14)+0.5(10+10)\*2=**34ns**

Implementarea lui **D** cu DMUX 16 căi şi porţi logice ŞI:

tp=0.5(14+14)+0.5(10+10)\*2=**34ns**

**Calculul puterilor disipate**

Se vor calcula puterile disipate pe fiecare circuit integrat cu formula:

Puterile disipate la:

Implementarea funcţiei logice **C** cu porţi logice ŞI-NU:

Pd=P74LS00+P74LS10=8+2.5(0.6+1.8)=**14mW**

Implementarea funcţiei logice **D** cu porţi logice ŞI-NU:

Pd=P74LS00+P74LS10=8+2.5(0.6+1.8)=**14mW**

Implementarea **ansamblului** folosind porţi ŞI-NU:

Pd=P74LS00\*2+P74LS10=16+2.5(0.6+1.8)=**22mW**

Implementarea **ansamblului** folosind atât porţi SAU-NU cât şi porţi ŞI-NU:

Pd= P74AC02 \*2+ P74LS00+P74LS10=230+8+6=**244mW**

Implementarea lui **A** cu un 2xMUX de 8 căi conectate în paralel:

Pd=P74LS151\*2+P74LS04+P74LS32=290+60+140=**490mW**

Implementarea lui **B** cu MUX de 8 căi:

Pd=P74LS151+P74LS04= 145+60=**205mW**

Implementarea lui **C** cu un MUX de 16 căi:

Pd=P74LS150=**200mW**

Implementarea lui **D** cu 2xDMUX-uri de 8 căi conectate în paralel:

Pd=P74HC155\*2+ P74HC04+ P74HC10=185\*2+185+187.5=**742.5mW**

Implementarea lui **B** cu un DMUX de 8 căi şi reţea de porţi logice:

Pd=P74HC155+ P74HC00\*2=185+350=**535mW**

Implementarea lui **C** cu DMUX 16 căi şi porţi logice ŞI-NU:

Pd=P74HC154+ P74HC10=185+187.5=**372.5mW**

Implementarea lui **D** cu DMUX 16 căi şi porţi logice ŞI:

Pd=P74HC154+ P74HC11=185+187.5=**372.5mW**

**Concluzii finale**

Putem compara implementările obţinute din mai multe puncte de vedere.

Din punctul de vedere al complexităţii implementării schemei logice cea mai puţin complexă este schema cu un MUX de 16 căi în tehnologie TTL (pag. 26) sau schema cu un DMUX de 16 căi (pag. 31/32) şi probabil şi cele mai ieftine. La capitolul timp de propagare, schema cu MUX-uri de 16 căi TTL este cea mai rapidă.(pag. 26). Din punctul de vedere al consumul de energie, schema cu 2xDMUX-uri de 8 căi este cea mai mare consumatoare de putere.

**Bibliografie**

1. CURS „Analiza şi Sinteza Circuitelor Numerice”

Prof. dr. ing. Florin MOLDOVEANU

1. <http://www.ti.com> – pagina oficiala a Texas Instruments, Dallas SUA;

site utilizat pentru obţinerea caracteristicilor circuitelor integrate folosite în cadrul prezentului Proiect.

1. <http://www.google.ro> – motor de căutare internaţional

4. <http://www.datasheetcatalog.com/datasheets_pdf/> -pagină de căutare a diferitelor date de catalog ale produselor diferitelor firme producătoare de circute integrate

Schemele logice obţinute au fost desenate in pachetul de programe Orcad Famuly Release 9.2, de unde au fost incluse in prezentul proiect şisimulate în mediul de lucru *OrCAD* versiunea mai veche de DOS.