

UNIVERSITATEA TEHNICĂ A MOLDOVEI

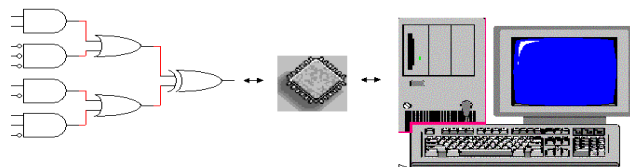
VASILE GÎSCĂ

VIORICA SUDACEVSCHI

SERGIU ZAPOROJAN

DISPOZITIVE NUMERICE

Prezentare teoretică și aplicații



**Chișinău
2005**

UNIVERSITATEA TEHNICĂ A MOLDOVEI
FACULTATEA CALCULATOARE, INFORMATICĂ ȘI
MICROELECTRONICĂ

CATEDRA CALCULATOARE

DISPOZITIVE NUMERICE

Prezentare teoretică și aplicații

Chișinău
U.T.M.
2005

Lucrarea este destinată studenților specializărilor: 2151, **Calculatoare**; 2152, **Tehnologii Informaționale** și 2153, **Automatica și Informatica** de la **Facultatea Calculatoare, Informatică și Microelectronică** pentru a fi utilizat la studierea cursului **ANALIZA ȘI SINTEZA DISPOZITIVELOR NUMERICE**.

Lucrarea conține descrierea sistemului **Logic Works**, destinat proiectării și simulării circuitelor numerice, o parte din materialul teoretic, predat la disciplina menționată mai sus și informația necesară îndeplinirii a șapte lucrări de laborator.

Prin problematica pe care o tratează lucrarea poate fi utilizată și de studenții altor specializări, care studiază structura și metodele de sinteză a dispozitivelor numerice.

Redactor responsabil: conf. univ., dr. Vasile Gîscă

Recenzent: conf. univ., dr. Valentin Negură

©U.T.M., 2005

DISPOZITIVE NUMERICE

Prezentare teoretică și aplicații

Elaborare: Vasile Gîscă
Viorica Sudacevski
Sergiu Zaporojan

Bun de tipar 12.04.05.	Formatul 60x84 1/16.
Hîrtie ofset. Tipar ofset	Tirajul 300 ex.
Coli de tipar 5,25.	Comanda nr.

U.T.M., 2004, Chișinău, bd. Ștefan cel Mare, 168.
Secția Redactare și Editare a U.T.M.
2068, Chișinău, str. Studenților, 11.

PREFAȚĂ

Dispozitivele numerice (digitale) sunt componentele de bază ale calculatoarelor electronice și ale altor sisteme și aparate destinate procesării informației. Aceste structuri numerice, pe care se bazează o foarte mare parte a tehnologiilor moderne din domeniul electronicii, asigură precizia, fiabilitatea, viteza și complexitatea, necesare prelucrării datelor. Începând cu structurile logice simple și pînă la circuitele secvențiale complexe sau logica programată se utilizează pe scară largă diferite metode și tehnici de proiectare (sinteză) și analiză a acestora. Pentru a utiliza cu succes aceste metode și tehnici proiectantul dispozitivelor numerice trebuie să aibă pregătirea teoretică respectivă și să dispună de deprinderile practice necesare.

Lucrarea de față conține materialul referitor la mijloacele și principalele aspecte ale proiectării digitale a diferitor structuri numerice. În capitolul 1 este prezentată descrierea, posibilitățile și principiile de utilizare a sistemului de proiectare **Logic Works**. Capitolul 2 este destinat abordării principalelor aspecte ale sintezei și analizei circuitelor logice combinaționale și descrierii diferitor implementări ale acestora. Pentru studiarea lor practică sunt descrise aplicații pentru cinci lucrări de laborator. În capitolul 3 este efectuată prezentarea teoretică și sunt descrise aspectele de sinteză ale principalelor structuri ale circuitelor logice secvențiale. Tot în acest capitol este prezentat materialul necesar îndeplinirii a două lucrări de laborator.

Îndeplinirea fiecărei lucrări de laborator presupune:

- asamblarea schemelor circuitelor numerice indicate de către profesor, sinteza cărora a fost efectuată anterior de către student în conformitate cu varianta sa;
- efectuarea analizei circuitelor asamblate și colectarea datelor pentru darea de seamă a lucrării.

În lucrare sunt descrise două modalități de îndeplinire a lucrărilor de laborator. Prima din ele prevede simularea și

analiza circuitelor cu ajutorul sistemului **Logic Works**, iar a doua - asamblarea circuitelor și analiza lor cu ajutorul standului de laborator.

Pentru a fi admis la îndeplinirea lucrării de laborator fiecare student trebuie să îndeplinească următoarele condiții:

- să fie îndeplinită tema pentru acasă, care constă din sinteza uneia sau mai multor scheme ale circuitelor numerice;
- să răspundă la întrebările de control puse de către profesor.

Lucrarea de laborator se consideră îndeplinită doar după ce studenții demonstrează profesorului funcționarea corectă a circuitelor asamblate.

Pentru fiecare lucrare, fiecare student pregătește o dare de seamă pe care o susține în fața profesorului. Darea de seamă include: foaia de titlu, tema, scopul lucrării, lucrul îndeplinit acasă, schemele tuturor circuitelor asamblate, tabelele obținute în rezultatul îndeplinirii lucrării respective.

Descrierea standului de laborator

Standul de laborator este destinat studierii practice a structurii și sintezei elementelor funcționale și unităților calculatoarelor numerice.

Standul de laborator este realizat în baza circuitelor integrate din familia K155 și este compus din: elemente ȘI-NU cu două, trei, patru și opt intrări (respectiv circuitele K155LA3, K155LA4, K155LA1 și K155LA2); elemente ȘI cu două intrări (circuitul K155LI1); elemente SAU cu două intrări (circuitul K155LL1); elemente NU (circuitul K155LN1); elemente SAU-NU cu două intrări (circuitul K155LE1); elemente 2-2ȘI-2SAU-NU și 4-

4ȘI-2SAU-NU cu posibilitatea de extindere prin elementul SAU) respectiv circuitele K155LR1, K155LR4); bistabilele JK (circuitul K155TV1); numărătorul binar-zecimal (circuitul K155IE5); numărătorul binar-reversibil (circuitul K155IE7); registrul de deplasare (circuitul K155IR1); sumatoare binare de patru ranguri (circuitul K155IM3).

Intrările și ieșirile circuitelor integrate sunt amplasate în partea din față a standului. Comunicațiile necesare între intrările și ieșirile circuitelor integrate se realizează cu ajutorul firelor de contact.

Standul este dotat cu un generator de impulsuri și poate funcționa și în regim “pas cu pas”.

1. SISTEMUL DE PROIECTARE DIGITALĂ LOGIC WORKS (versiunea 4)

Sistemul **Logic Works** este destinat proiectării și simulării circuitelor digitale. Asamblarea circuitelor se efectuează cu ajutorul elementelor logice, circuitelor combinaționale și secvențiale care se conțin în bibliotecile sistemului. Circuitul asamblat se analizează cu ajutorul diagramelor de timp și a elementelor de vizualizare a valorilor logice generate de circuit.

1.1. Interfața

Fereastra de lucru conține circuitele proiectate. Circuitele din **Fereastra de lucru** funcționează în acord cu informația de timp din **Fereastra de timp**.

Biblioteca conține toate elementele necesare pentru proiectare.

Fereastra de timp în stare activă afișează diagramele de timp ale circuitului proiectat.

Bara de instrucțiuni pentru proiectare este folosită pentru controlul cursorului și pentru modificarea statutului **Ferestrei de lucru**. Funcțiile din această bară permit deschiderea, salvarea și tipărirea documentului, folosirea cursorului pentru inserarea elementelor și trasarea liniilor, verificarea valorilor logice, marcarea elementelor, ștergerea liniilor și elementelor ș.a.

Bara de instrucțiuni pentru simulare este folosită pentru schimbarea frecvenței de generare a impulsurilor în timpul simulării. Instrucțiunile de simulare pot fi accesate și selectând meniul **Simulation** din **Bara de meniuri**.

1.2. Proiectarea de bază

Amplasarea elementelor. Pentru amplasarea elementelor în **Fereastra de lucru** se execută dublu clic pe numele elementului din bibliotecă. Versiunea elementului va apărea în dreptul cursorului. Elementul se amplasează în locația dorită și se execută clic. Elementul selectat va rămîne activ pentru o nouă amplasare pînă nu va fi dezactivat. Selectarea altui element se execută prin efectuarea dublului clic pe numele elementului corespunzător din bibliotecă și procedura se repetă. Pentru anularea selectării elementului se tastează **space bar** sau butonul din dreapta al mouseului. Pentru schimbarea direcției elementului se selectează instrucțiunea **Orientation** din meniul **Schematic**. Se poate alege una dintre cele opt opțiuni din meniul grafic care se va deschide. Toate elementele inserate vor avea aceeași direcție.

Trasarea liniilor. Pentru unirea elementelor logice între ele se apasă și se menține butonul din stînga al mous-ului la capătul pinului de intrare sau ieșire al unuia din elemente, sau la nodul de conexiune dintre firele circuitului. Cursorul se deplasează spre punctul dorit, care poate fi un alt pin, alt fir sau o locație liberă.

Fiți atenți, deoarece acest mod de trasare poate duce la apariția unor noduri nedorite în punctele de intersecție ale liniilor!

O altă metodă de trasare a liniilor constă în folosirea butoanelor din bara de instrumente pentru proiectare.

Editarea. Pentru deplasarea unei linii sau unui element se execută clic pe el. Butonul din stînga al mous-ului se ține apăsat și se deplasează spre locația dorită. Elementele selectate vor fi evidențiate cu negru, liniile cu galben. Pentru deplasarea mai multor elemente se apasă și se menține butonul din stînga al mous-ului pe o locație neocupată iar apoi se deplasează mous-ul pînă cand toate elementele dorite vor fi cuprinse în fereastra creată. Pentru a șterge un element sau o linie se execută clic pe el și se apasă tasta **Del**. Pentru a șterge un număr mai mare de elemente toate elementele se selectează și se apasă tasta **Del**.

1.3. Bara de instrucțiuni pentru proiectare

Funcțiile din această bară sunt folosite pentru schimbarea poziției elementelor și tipului de linie, executarea corectărilor necesare în circuite, inserarea textului, schimbarea aspectului **Ferestrei de lucru** și efectuarea probelor logice ale circuitului.

Bara de instrucțiuni pentru proiectare constă din cinci părți: **Gestionarea fișierelor**, **Deplasarea și modificarea elementelor**, **Modificarea cursorului**, **Modificarea Ferestrei de lucru** și **Tipul elementului**.

Gestionarea fișierelor. Primele patru butoane sunt folosite pentru gestionarea fișierelor. Începînd cu butonul din stînga aceste patru butoane sunt folosite pentru **crearea** fișierelor noi, **deschiderea** documentului existent, **salvarea** documentului activ și **tipărirea** documentului activ. Fiecare din aceste instrucțiuni, la fel ca și celelalte instrucțiuni de gestiune pot fi găsite în meniul **File**.

Modificarea elementelor. Următoarele cinci butoane sunt

folosite pentru deplasarea și modificarea elementelor.

Cut – ștergerea elementelor selectate în **Fereastra de lucru**.

Copy – copierea elementelor selectate în Clipboard.

Insert – inserarea conținutului din Clipboard în **Fereastra de lucru**.

Duplicate – dublarea elementelor selectate,

Get Info – obținerea și setarea informației despre elementul selectat. În acest caz doar un singur element poate fi selectat.

Modificarea cursorului. Următoarele șapte butoane servesc pentru modificarea cursorului.

Lupa. Prezintă imaginea centrată și mărită a locației selectate. Mărirea poate fi de asemenea efectuată și utilizând comanda **Magnify** din meniul **View**. Revenirea la dimensiunile normale poate fi executată utilizând comanda **Reduce to Fit** din meniul **View** sau butoanele de dirijare a dimensiunilor **Ferestrei de lucru**, examinate ulterior.

Probe. Butonul **Probe** are două funcții. În primul rând, el permite de a afla valoarea logică a pinului. Pentru aceasta cursorul se poziționează pe pinul dorit și se ține apăsat butonul din stînga al mous-ului. În al doilea rând, se poate atașa pinului valoarea logică dorită dacă se tastează această valoare în timpul procedurii precedente.

Atenție! În acest caz puteți provoca conflicte logice care pot duce la deteriorarea circuitului.

Pointer. Starea implicită a cursorului. Când nu este activă, cursorul poate funcționa în toate modurile descrise anterior.

Zap. În unele cazuri este nevoie de a șterge doar o porțiune de linie sau un element situat într-o regiune foarte aglomerată. Funcția **Zap** permite de a face aceasta cu o mai mare flexibilitate. Pentru aceasta se execută clic pe porțiunea de linie sau pe elementul dorit.

Text. Utilizând funcția **Text** se poate adăuga în schemă titluri, descrieri ale circuitelor și elementelor. La activarea butonului cursorul va apărea în formă de pix. În locația

selectată se va deschide o casetă de text, unde se va putea introduce textul dorit. Pentru redactarea textului se execută clic pe textul dorit.

Această funcție mai permite de a denumi un pin, semnal logic sau element. Numele pinului va avea culoarea albastră, numele elementului și al semnalului va avea culoarea roșie. Pentru denumirea semnalului logic e necesar de selectat linia respectivă.

Trasarea liniilor.

Draw signal. Permite trasarea liniilor de conexiune dintre elementele circuitului. Nu este necesar de început trasarea de la capătul pinului sau nodului spre deosebire de trasarea liniilor cu cursorul obișnuit al mous-ului.

Draw bus. Permite trasarea magistralelor.

Modificarea Ferestrei de lucru.

Zoom Out. Mărirea imaginii în **Fereastra de lucru.**

Zoom In. Micșorarea imaginii în **Fereastra de lucru.**

Fit to Window. Vizualizarea întregului circuit proiectat în **Fereastra de lucru.**

Normal Size. Revenirea la dimensiunile normale.

Tipul elementelor.

PROM/RAM/PLA Wizard. Această funcție permite programarea și cercetarea circuitelor de memorie permanentă, memorie cu acces aleatoriu și a matricelor logice programabile.

Parts Palette. Activează și dezactivează **Fereastra de vizualizare a bibliotecilor.**

1.4. Bibliotecile

Pentru îndeplinirea lucrărilor de laborator se vor folosi următoarele biblioteci:

Simulation Gates.clf. Această bibliotecă conține următoarele porți logice: **AND** (ȘI), **OR** (SAU), **NOT** (NU), **NAND** (ȘI-NU), **NOR** (SAU- NU), **XOR** (SAU-

EXSCLUSIV), **XNOR** (SAU-NU-EXSCLUSIV). Sunt accesibile porți logice cu numere diferite de intrări (**AND-2**, **AND-4**). Unele porți logice au intrări inversate (**AND-4(2-Inv)**).

Simulation IO.clf. Această bibliotecă conține mai multe dispozitive de intrare-ieșire care sunt folosite pentru introducerea și vizualizarea informației.

Seven-Segment Displays –display pe șapte segmente. Sunt disponibile mai multe variante color:

7-SegDisp-Color funcționează în felul următor: aplicând 1 logic la una dintre cele șapte intrări informaționale, segmentul corespunzător începe să lumineze.

7-SegDispInv-Color - la aplicarea valorii 0 logice la una dintre cele șapte intrări informaționale, segmentul corespunzător începe să lumineze.

Poate fi folosit pentru vizualizarea cifrelor zecimale utilizând un decodificator de șapte segmente din biblioteca **7400devs.clf** (74-46, 74-47, 74-48, 74-49).

LEDs (light –emitting diodes) – diode color luminescente. Pentru a conecta corect dioda este necesar de a uni cu pământul pinul inversat și celălalt pin cu dispozitivul care urmează să dirijeze dioda (1 logic – dioda luminează).

Comutatoarele.

- **SPDT pushbutton** (Single-Pole Double-Throw) - întrerupător care deține o anumită valoare și trece la altă valoare logică în momentul efectuării contactului cu ajutorul butonului din stînga al mous-ului.

- **SPDT switch** - comutator care transmite una dintre cele două valori pe care le deține în dependență de legătura de la intrarea comutatorului.

- **SPST switch** (Single-Pole Single-Throw) - comutator cu o singură intrare și o singură ieșire care poate fi închis sau deschis.

Binary Switch - reprezintă o varietate a **SPDT**

switch ale cărui intrări sunt deja conectate la 1 și 0 logic.

Binary Probe - conectat la orice linie a circuitului proiectat afișează valoarea binară în acest punct, fiind de ajutor la detectarea erorilor sau la testarea circuitului. În afară de valorile binare 0 și 1 pot apărea și altele:

- (X) - valoarea nu este cunoscută / valoare indiferentă;
- (Z) - ieșirea nu este conectată la intrare;
- (C) - condiție de conflict.

Hex Keyboard - tastatură hexazecimală cu ajutorul căreia pot fi introduse valorile hexazecimale de la 0 la F, prezentate în formă de cod binar pe patru biți.

Hex Display - display hexazecimal folosit pentru vizualizarea informației în formă hexazecimală.

Clock (semnalul de tact sau ceasul) este folosit la cercetarea circuitelor logice secvențiale.

Simulation Logic.clf. Această bibliotecă conține mai multe circuite combinaționale și secvențiale specializate care pot fi utilizate la proiectarea circuitelor digitale complexe: sumatoare (Adder-4, Adder-8, Adder-4 wo/Carry), numărătoare (Counter-4, Counter-4 Up wo/En), decodificatoare (Decoder-4, Decoder-8), multiplexoare (Mux-4, Mux-4 wo/En), bistabile (JK Flip Flop, D Flip Flop), registre (Reg-4, Reg-4 inv CLR) ș.a.

1.5. Simularea și sincronizarea

Bara de instrumente pentru simulare reprezintă interfața cu **Simulatorul** și cu **Fereastra de timp**. Instrucțiunile din această bară permit controlul asupra vitezei de simulare, statutului semnalelor din circuitul proiectat. Bara constă din trei părți de bază: **Statutul Semnalului**, **Statutul Ferestrei Temporale** și **Statutul Simulatorului**.

Statutul semnalului

Show/Hide Timing - activează sau dezactivează **Fereastra de timp**.

Add Signals to Timing - adaugă semnalul selectat în **Fereastra de timp** în cazul când opțiunea **Add Automatically** din meniul **Simulation** nu este activată.

Triggers - deschide meniul opțional de declanșare al semnalului.

Simulation Parameters - susține aceleași funcții ca și comanda **Simulation Parameters** din meniul **Simulation**.

Stick/unstick Signals - fixează semnalul la un anumit nivel logic. Semnalul fixat rămâne la acest nivel logic indiferent de schimbările care au loc în circuit.

Reset Simulator - resetează **Simulatorul**.

Clear Unknowns - elimină toate semnalele cu valori neidentificate.

Statutul Ferestrei Temporale

Următoarele trei butoane controlează scara **Ferestrei de timp**.

Zoom In - **Fereastra de timp** conține mai puține gradații.

Zoom Out - **Fereastra de timp** conține mai multe gradații.

Normal Size - revenirea la gradarea inițială.

Statutul Simulatorului dirijează viteza **Simulatorului** și constă din trei butoane, bara de viteză și un numărător.

Single Step - afișează în **Fereastra de timp** schimbările care au loc în circuit la un singur pas de simulare. Un pas de simulare reprezintă perioada minimă de timp în care în circuit se produce o oarecare schimbare. Deci pasul de simulare poate fi de orice lungime și, de obicei, reflectă timpul necesar pentru schimbarea valorii celei mai rapide variabile.

Stop Simulator - oprește **Simulatorul**, fără a-l reseta. Simularea poate fi restartată activând butonul **Run Simulator** sau selectând viteza de simulare de pe **Bara de viteză**. La fel simularea poate fi continuată în regimul **Single Step**.

Simulation Speed - reprezintă bara de viteză cu ajutorul căreia poate fi schimbată viteza de simulare a circuitului.

Run Simulator. Activarea acestui buton duce la rularea simulatorului la viteză maximă.

Simulation Counter. Această fereastră afișează pasul de simulare curent care coincide cu pasul din **Fereastra de timp**.

Fereastra de timp

Fereastra de timp afișează toate schimbările care se produc cu semnalele denumite. Aceste schimbări pot fi statice(manuale) și dinamice(automate).

Simularea manuală. Pentru simularea manuală se folosesc comutatoarele și verificatoarele binare (**Binary Probe**). Deoarece Logic Works este un simulator bazat pe evenimente discrete de timp, orice schimbare a valorii variabilei de intrare produce o reînnoire a valorii variabilei respective. Această schimbare este reflectată în verificatorul binar sau în **Fereastra De timp**.

Simularea automată. Într-un circuit complex utilizarea manuală a comutatoarelor devine neavantajoasă și poate duce la erori. În aceste cazuri procesul de simulare poate fi automatizat utilizând **fișiere temporale**. Un fișier temporal reprezintă o metodă compactă și simplă de a indica valorile semnalelor de intrare în momente discrete de timp.

Un fișier temporal necesită cel puțin trei antete și un rând de date, cu un element din fiecare rând corespunzător fiecărui antet. Aceste antete sunt:

- **\$T** - indică momentele discrete de timp. În Logic Works măsurarea timpului începe de la 0 și fiecare unitate de timp este echivalentă cu o nanosecundă;

- **\$D** - indică diferența dintre valoarea timpului din rândul curent și cel următor.

- **\$O** – antetul este folosit pentru toate semnalele de intrare și este urmat de numele acestor semnale. În cazul când fișierul temporal este exportat toate semnalele, atât cele de intrare cât

și cele de ieșire sunt indicate cu ajutorul antetului \$O, urmat de numele fiecărui semnal.

În continuare este prezentat un exemplu al unui fișier temporal, creat pentru simularea unui circuit cu trei semnale de intrare: A, B și C

\$T	\$D	\$O A	\$O B	\$O C
0	10	0	0	0
10	10	0	0	1
20	10	0	1	0

Importarea și exportarea fișierelor temporale

Fișierul temporal cu extensia **TIM** poate fi creat în orice redactor de texte, de exemplu **Notepad** sau **Wordpad**. Pentru importarea fișierului temporal în **Simulator** selectați comanda **Import Timing** din meniul **Simulation**. După selectarea numelui fișierului temporal în **Fereastra De timp** va apărea o versiune întretăiată a semnalelor de intrare. La pornirea **Simulatorului** programul **Logic Works** va aplica valorile acestor semnale în locațiile necesare. Orice semnal de ieșire afișat în **Fereastra de timp** va fi reînnoit. În cazul când doriți să obțineți rezultatele simulării în formă de tabel puteți să exportați fișierul temporal. Pentru aceasta selectați comanda **Export Timing** din meniul **Simulation**. Acest fișier va conține semnalele de intrare și cele de ieșire.

Timpul de reținere a semnalelor

În **Logic Works** toate porțile logice la fel ca și circuitele logice mai complexe de tipul sumatoarelor, multiplexoarelor ș.a. au timpul de reținere egal cu o nanosecundă, ceea ce este departe de realitate. De exemplu, o poartă logică ȘI-NU cu două intrări are, în cel mai rău caz, o întârziere de 22 nanosecunde. În cazul când apare necesitatea de a studia comportamentul circuitelor în condiții mai reale, puteți să schimbați timpul de reținere.

Pentru schimbarea timpului de reținere la una sau

mai multe porți logice selectați-le pe toate, ținând apăsată tasta **CTRL**, apoi activați comanda **Simulation Parameters** din meniul **Simulation**. În fereastra ce va apărea introduceți valoarea necesară. Toate porțile selectate vor avea același timp de reținere.

Pentru a schimba timpul de reținere la circuitele din biblioteca **7400devs.clf** este necesar la început de deblocat porțile logice componente. Pentru aceasta apăsați butonul din dreapta al mous-ului pe circuit și selectați **Device Info**. În fereastra care va apărea desactivați caseta de validare **Lock Opening Subcircuit**. Acum puteți schimba parametrii porților logice componente ca și în cazul unor porți logice separate. Pentru aceasta executați dublu clic pe circuitul pe care doriți să-l modificați. Va apărea o nouă **Fereastră de lucru** cu toate porțile logice componente ale circuitului deblocat unde puteți executa modificările necesare.

2. CIRCUITELE LOGICE COMBINAȚIONALE

2.1. Particularitățile procesului de sinteză a circuitelor logice combinaționale

Orice circuit logic se caracterizează prin natura semnalelor de intrare, a celor de ieșire, prin clasele de funcții intrare-ieșire și prin natura prelucrărilor de date ce au loc în structura sa internă.

Circuitele logice se împart în două clase: combinaționale și secvențiale. Un circuit logic combinațional (CLC) se caracterizează prin aceea că starea ieșirilor sale la un moment dat depinde numai de starea intrărilor sale în acest moment. Legătura între starea intrărilor și starea ieșirilor circuitului este dată de funcțiile de transfer ale acestuia, denumite în acest caz funcții de comutare, care sunt funcții booleene (logice).

CLC este circuitul, care are n intrări ($x_1, x_2, x_3, \dots, x_n$) și m ieșiri ($y_1, y_2, y_3, \dots, y_m$), la care ieșirile pot fi exprimate numai

în dependență de variabilele de intrare:

$$y_1=f_1(x_1, x_2, x_3, \dots, x_n);$$

$$y_2=f_2(x_1, x_2, x_3, \dots, x_n);$$

.....

$$y_m=f_m(x_1, x_2, x_3, \dots, x_n).$$

Pentru că în acest model matematic nu intervin ca variabile independente timpul și nici mărimile de ieșire, rezultă, că în structura sa un CLC nu prezintă circuite de memorie și nici legături de reacție (variabilele de ieșire nu sunt aplicate la intrare).

Sinteza unui CLC se efectuează în următoarele etape:

- descrierea necesităților ce trebuie să le rezolve circuitul combinațional respectiv (prin text, desen, diagrame etc.);
- reprezentarea acestei descrieri sub forma unui tabel de adevăr;
- deducerea funcțiilor logice și minimizarea acestora;
- implementarea acestor funcții minimizate sub forma unor rețele de comutare prin intermediul circuitelor integrate;

Tabelul de adevăr conține $n+m$ coloane și 2^n rînduri. Fiecare rînd al tabelului reprezintă una din combinațiile posibile ale valorilor variabilelor și valorile funcțiilor pentru combinația respectivă.

Implementarea funcțiilor logice minimizate sub forma rețelelor de comutare poate fi realizată în forma canonică disjunctivă (ȘI/SAU), în forma canonică conjunctivă (SAU/ȘI) sau în orice altă formă normală, adică ȘI-NU/ȘI-NU, SAU/ȘI-NU, SAU-NU/SAU, ȘI/SAU-NU, ȘI-NU/ȘI, SAU-NU/SAU-NU.

Trecerea de la o formă normală la alta se efectuează prin utilizarea succesivă a formulelor lui De Morgan, avînd inițial forma canonică disjunctivă normală (ȘI/SAU) și forma canonică conjunctivă normală (SAU/ȘI) a funcției. De exemplu:

- a) din forma disjunctivă normală:
(forma ȘI/SAU):

$$\begin{aligned}
y &= x_2 \bar{x}_4 \vee \bar{x}_1 \bar{x}_3 x_4 \vee \bar{x}_1 \bar{x}_2 x_4 \vee x_1 \bar{x}_3 \bar{x}_4 = \\
&\text{(forma } \text{ȘI-NU/ȘI-NU):} \\
&= \overline{(x_2 \bar{x}_4)(\bar{x}_1 \bar{x}_3 x_4)(\bar{x}_1 \bar{x}_2 x_4)(x_1 \bar{x}_3 \bar{x}_4)} = \\
&\text{(forma SAU/ȘI-NU):} \\
&= \overline{(\bar{x}_2 \vee x_4)(x_1 \vee x_3 \vee \bar{x}_4)(x_1 \vee x_2 \vee \bar{x}_4)(\bar{x}_1 \vee x_3 \vee x_4)} = \\
&\text{(forma SAU-NU/SAU):} \\
&= \overline{(\bar{x}_2 \vee x_4) \vee (x_1 \vee x_3 \vee \bar{x}_4) \vee (x_1 \vee x_2 \vee \bar{x}_4) \vee (\bar{x}_1 \vee x_3 \vee x_4)}
\end{aligned}$$

b) din forma conjunctivă normală:

$$\begin{aligned}
&\text{(forma SAU/ȘI):} \\
y &= (\bar{x}_1 \vee \bar{x}_4)(x_2 \vee x_3 \vee x_4)(\bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4)(x_1 \vee x_2 \vee x_4) = \\
&\text{(forma } \text{ȘI-NU/ȘI):} \\
&= \overline{(x_1 x_4)(\bar{x}_2 \bar{x}_3 \bar{x}_4)(x_2 x_3 x_4)(\bar{x}_1 \bar{x}_2 \bar{x}_4)} = \\
&\text{(forma } \text{ȘI/SAU-NU):} \\
&= \overline{(x_1 x_4) \vee (\bar{x}_2 \bar{x}_3 \bar{x}_4) \vee (x_2 x_3 x_4) \vee (\bar{x}_1 \bar{x}_2 \bar{x}_4)} = \\
&\text{(forma SAU-NU/SAU-NU):} \\
&= \overline{(\bar{x}_1 \vee \bar{x}_4) \vee (x_2 \vee x_3 \vee x_4) \vee (\bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4) \vee (x_1 \vee x_2 \vee x_4)}
\end{aligned}$$

2.2. Lucrarea de laborator nr. 1

Sinteza circuitelor logice combinaționale

Scopul lucrării: studierea practică și cercetarea procesului de sinteză a circuitelor logice combinaționale.

Tema pentru acasă

1. Se efectuează minimizarea funcțiilor logice y_1 și y_2 conform variantei din tabelul 2.1. Pentru ambele funcții se efectuează sinteza circuitului logic în setul de elemente ȘI-NU.

2. Funcția y_1 se reprezintă în formă disjunctivă normală perfectă și forma conjunctivă normală perfectă. Pentru forma disjunctivă normală perfectă se efectuează sinteza circuitului logic în setul de elemente ȘI-NU.

3. Funcția y_2 se reprezintă în toate cele 8 forme normale.

Tabelul 2.1

Variante pentru îndeplinirea lucrării de laborator

Nr. variantului	Funcțiile logice
1	2
1	$y_1 = \vee(0,1,2,4,5,7,9,10,11,14,15)$ $y_2 = \vee(2,3,4,5,8,9,12,13)$
2	$y_1 = \vee(1,3,4,7,8,10,12,13,14)$ $y_2 = \vee(3,4,5,7,9,11,13,14,15)$
3	$y_1 = \vee(0,2,4,5,8,10,12,14)$ $y_2 = \vee(1,2,3,4,7,8,9,12,13,14)$

Tabelul 2.1 (Continuare)

1	2
4	$y_1=\vee(0,2,3,5,6,7,9,11,12,13,14)$ $y_2=\vee(1,2,4,5,6,8,9,11,14,15)$
5	$y_1=\vee(2,4,5,7,8,9,12,14,15)$ $y_2=\vee(0,1,2,7,8,10,11,14)$
6	$y_1=\vee(1,2,4,5,6,8,10,14,15)$ $y_2=\vee(0,1,2,5,6,7,9,11,12,13)$
7	$y_1=\vee(0,1,5,6,7,8,10,12,14,15)$ $y_2=\vee(1,2,4,8,9,10,11,12)$
8	$y_1=\vee(0,1,2,4,6,8,11,12,15)$ $y_2=\vee(0,1,2,5,6,7,8,9,12,13)$
9	$y_1=\vee(0,2,4,5,7,8,10,12,15)$ $y_2=\vee(2,3,4,5,7,8,9,11,12,14)$
10	$y_1=\vee(0,3,4,5,6,8,10,12,13)$ $y_2=\vee(4,5,6,7,9,11,12,13,14)$
11	$y_1=\vee(1,2,4,5,8,9,10,12,13,14,)$ $y_2=\vee(3,4,5,7,8,9,11,12,13)$
12	$y_1=\vee(0,1,2,4,5,8,9,12,14,)$ $y_2=\vee(1,2,3,5,6,8,10,11,12)$
13	$y_1=\vee(0,2,4,5,6,7,9,12,13,15)$ $y_2=\vee(2,3,4,5,7,8,9,10,11,)$
14	$y_1=\vee(0,2,3,4,6,9,10,11,13,14,15)$ $y_2=\vee(3,4,5,7,8,10,11,14,15,)$
15	$y_1=\vee(0,1,4,5,7,8,10,11,12)$ $y_2=\vee(1,3,5,6,7,9,10,12,15)$
16	$y_1=\vee(0,2,3,4,6,7,9,11,12,13)$ $y_2=\vee(3,4,5,8,9,11,12,14)$
17	$y_1=\vee(0,3,4,5,7,8,12,13,14)$ $y_2=\vee(2,4,5,6,8,10,11,15)$

Tabelul 2.1 (Continuare)

1	2
18	$y_1 = \vee(1,2,3,4,6,7,8,9,10)$ $y_2 = \vee(2,3,5,6,7,10,12,15)$
19	$y_1 = \vee(0,1,2,5,6,7,14,15)$ $y_2 = \vee(2,3,4,7,8,9,10,12,13,14,15)$
20	$y_1 = \vee(3,4,5,6,7,8,10,12,13)$ $y_2 = \vee(0,1,2,5,6,8,9,11,12,14)$
21	$y_1 = \vee(0,1,5,6,7,8,9,10,12,13)$ $y_2 = \vee(1,2,3,4,5,6,8,9,11,14,15)$
22	$y_1 = \vee(0,2,3,5,8,9,10,12,13)$ $y_2 = \vee(1,3,4,6,7,8,9,10,11,14)$

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.
2. Se asamblează și se reglează circuitul logic combinațional, care realizează două funcții din tema pentru acasă în setul de elemente ȘI-NU (la indicația profesorului).
3. Pentru circuitele asamblate se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele **NAND** cu numărul corespunzător de intrări. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de intrare-ieșire **Binary Probe** și **Hex Keyboard**.
2. Se asamblează circuitul logic combinațional în **Fereastra de lucru** și se verifică corectitudinea lui. Se studiază diagrama de timp. Un exemplu al circuitului asamblat este prezentat în figura 2.1.

3. Pentru circuitele asamblate se determină costul și timpul de reținere.

Întrebări

1. Care sunt particularitățile care caracterizează circuitele logice combinaționale?

2. Care sunt etapele de sinteză ale circuitelor logice combinaționale ?

3. Cum se calculează timpul de funcționare a unui circuit logic combinațional ?

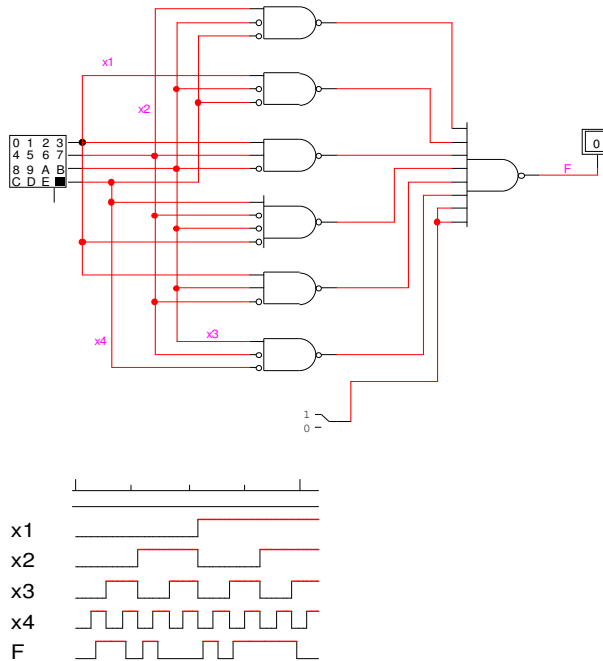


Fig. 2.1. Circuit logic combinațional asamblat în LogicWorks și diagrama lui de timp

2.3. Convertoarele de cod

Convertoarele de cod sunt elementele funcționale destinate transformării unui cod binar în altul. De obicei, aceste elemente funcționale reprezintă circuite logice combinaționale cu n intrări și m ieșiri. Aria tipurilor de convertoare de cod este foarte largă, iar valorile n și m ale lor pot coincide, dar pot fi și diferite, în dependență de tipul de coduri de la intrarea și, respectiv, de la ieșirea convertorului de cod.

În calitate de convertoare de cod, la care numărul de intrări coincide cu numărul de ieșiri pot servi cele care transformă codul direct al unui număr binar în codul lui invers, sau în cel complementar. Tot în această categorie intră convertoarele unui cod binar- zecimal în altul etc.

Convertoare de cod cu număr diferit de intrări și ieșiri sunt acelea care efectuează conversia numerelor dintr-un sistem de numerație în altul, convertoarele care transformă codul binar-zecimal de patru biți în codul pentru indicatoarele numerice de șapte segmente (șapte biți) ș.a.

Sinteza convertoarelor de cod, indiferent de tipul lor, are loc în felul următor:

- elaborarea tabelului de adevăr cu următorii parametri: numărul de variabile este egal cu numărul biților codului, care se aplică la intrările convertorului, iar numărul funcțiilor cu numărul de biți ai codului, care trebuie obținut la ieșirile convertorului de cod. Funcțiile logice pot fi parțial determinate, dacă numărul combinațiilor codului de intrare este mai mic decât 2^n .
- minimizarea tuturor funcțiilor din tabelul de adevăr.
- depistarea conjuncțiilor comune ale formelor minimale ale tuturor funcțiilor, pentru a evita dublarea elementelor logice, care realizează părți comune ale mai multor funcții.

- implementarea funcțiilor minimize prin circuite integrale digitale.

Vom ilustra cele descrise mai sus printr-un exemplu de sinteză a unui convertor de cod binar-zecimal. Tabelul de adevăr care descrie structura convertorului de cod 87 (-2) (-7) → 4221 este prezentat în tabelul 2.2.

Tabelul 2.2

Tabelul de adevăr al convertorului de cod binar-zecimal

Nr. crt.	87(-2)(-4)				4221			
	x_4	x_3	x_2	x_1	f_4	f_3	f_2	f_1
0	0	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0	1
2	1	0	1	1	0	0	1	0
3	0	1	0	1	0	0	1	1
4	1	0	0	1	0	1	1	0
5	0	1	1	0	1	0	0	1
6	1	0	1	0	1	1	0	0
7	0	1	0	0	1	1	0	1
8	1	0	0	0	1	1	1	0
9	1	1	1	1	1	1	1	1
10	0	0	0	1	*	*	*	*
11	0	0	1	0	*	*	*	*
12	0	0	1	1	*	*	*	*
13	1	1	0	0	*	*	*	*
14	1	1	0	1	*	*	*	*
15	1	1	1	0	*	*	*	*

În figura 2.2. sunt prezentate diagramele Karnaugh pentru minimizarea funcțiilor f_4, f_3, f_2, f_1 .

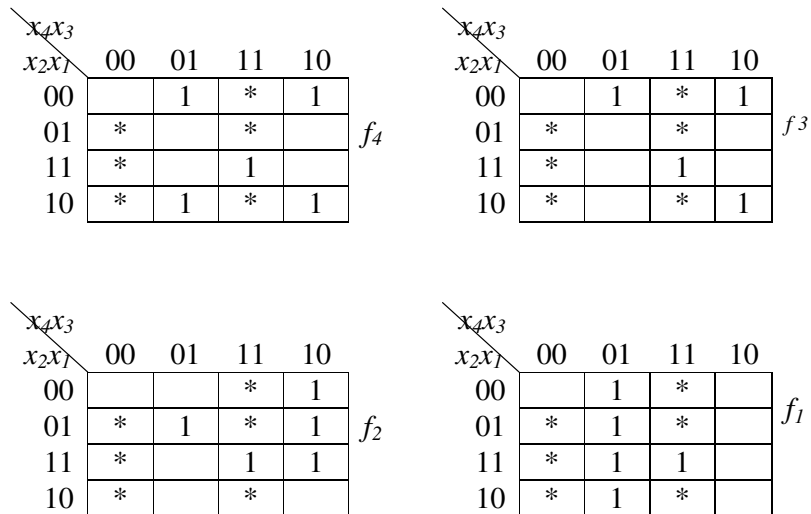


Fig. 2.2. Diagramele Karnaugh pentru minimizarea funcțiilor f_4, f_3, f_2, f_1

În rezultatul minimizării au fost obținute următoarele funcții logice:

$$\begin{aligned}
 f_4 &= x_3 \bar{x}_1 \vee x_4 \bar{x}_1 \vee x_4 x_3; \\
 f_3 &= x_3 \bar{x}_2 \bar{x}_1 \vee x_4 \bar{x}_2 \vee x_4 \bar{x}_1 \vee x_4 x_3; \\
 f_2 &= x_4 \bar{x}_2 \vee \bar{x}_2 x_1 \vee x_4 x_1; \\
 f_1 &= x_3.
 \end{aligned}
 \tag{2.1}$$

Luând în considerare conjuncțiile comune, funcțiile f_4, f_3, f_2, f_1 pot fi scrise în felul următor:

$$\begin{aligned}
 f_4 &= x_3 \bar{x}_1 \vee z_1; \\
 f_3 &= x_3 \bar{x}_2 \bar{x}_1 \vee z_1 \vee z_2; \\
 f_2 &= \bar{x}_2 x_1 \vee x_4 x_1 \vee z_2; \\
 f_1 &= x_3.
 \end{aligned}
 \tag{2.2}$$

unde :

$$\begin{aligned}
 z_1 &= x_4 \bar{x}_1 \vee x_4 x_3; \\
 z_2 &= x_4 \bar{x}_2.
 \end{aligned}
 \tag{2.3}$$

Schema convertorului de cod 8 7 (-2) (-7) \rightarrow 4 2 2 1 este prezentată în figura 2.3.

2.4. Lucrarea de laborator nr. 2

Sinteza convertoarelor de cod

Scopul lucrării: studierea practică a metodelor de sinteză a convertoarelor de cod.

Tema pentru acasă

1. Să se efectueze sinteza unui convertor de cod binar-zecimal în altul conform variantei din tabelul 2.3 (la indicația profesorului).

2. Funcțiile să se reprezinte în forma disjunctivă normală perfectă și forma disjunctivă minimală. Pentru forma minimală să se prezinte schema în setul de elemente ȘI-NU.

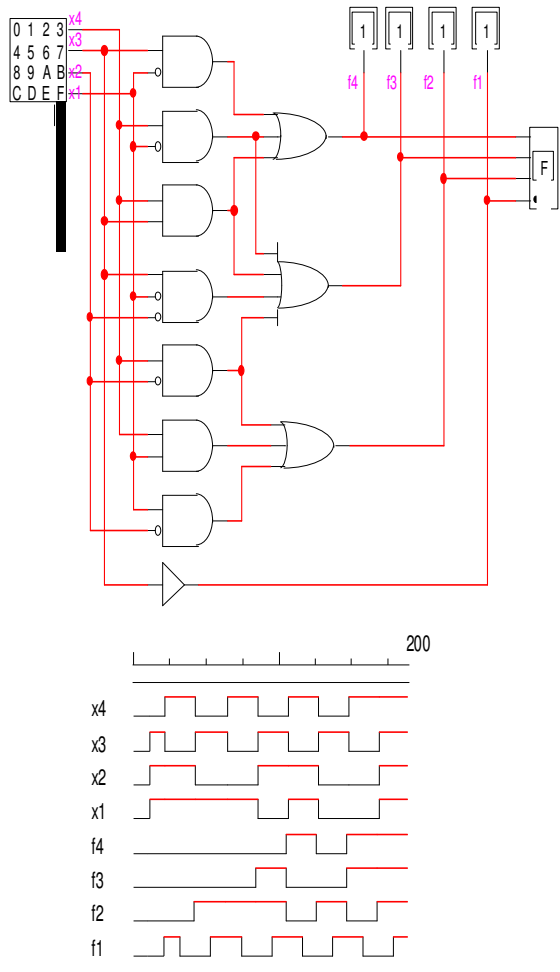


Fig. 2.3. Circuitul convertorului de cod 87(-2)(-7)→4221 și diagrama lui de timp

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.
2. Se assemblează și se reglează schema convertorului de cod binar-zecimal din tema pentru acasă în setul de elemente ȘI-NU.
3. Pentru circuitele asamblate se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele **NAND** cu numărul corespunzător de intrări. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de intrare-ieșire **Binary Probe** și **Hex Keyboard**.
2. Se assemblează schema convertorului de cod binar-zecimal din tema pentru acasă în setul de elemente ȘI-NU în **Fereastra de lucru** și se verifică corectitudinea lui. Se studiază diagrama de timp.
3. Pentru circuitul asamblat se determină costul și timpul de reținere.

Întrebări

1. Care este raportul dintre numărul de intrări și numărul de ieșiri ale convertoarelor de cod ?
2. Enumerați etapele și specificul procesului de sinteză a convertoarelor de cod.

Tabelul 2.3

Variante pentru îndeplinirea lucrării de laborator

Nr. var.	Codul binar-zecimal intrare	Codul binar-zecimal ieșire	Nr. var.	Codul binar-zecimal intrare	Codul binar-zecimal ieșire
1.	8 4 2 (-1)	4 4 2 1	16.	4 4 2 1	8 4 2 (-1)
2.	8 4 2 (-3)	5 2 1 1	17.	5 2 1 1	8 4 2 (-3)
3.	8 4 1 (-2)	5 2 2 (-1)	18.	5 2 2 (-1)	8 4 1 (-2)
4.	8 3 2 (-4)	5 3 2 (-1)	19.	5 3 2 (-1)	8 3 2 (-4)
5.	8 4 2 (-5)	5 2 2 1	20.	5 2 2 1	8 4 2 (-5)
6.	8 4 1 (-6)	5 3 1 (-1)	21.	5 3 1 (-1)	8 4 1 (-6)
7.	8 4 2 (-3)	5 3 2 (-1)	22.	5 3 2 (-1)	8 4 2 (-3)
8.	8 7 (-2)(-4)	3 3 2 1	23.	3 3 2 1	8 7 (-2)(-4)
9.	8 6 (-1)(-4)	4 2 2 1	24.	4 2 2 1	8 6 (-1)(-4)
10.	8 5 (-2)(-4)	4 3 1 1	25.	4 3 1 1	8 5 (-2)(-4)
11.	8 4 3 (-6)	4 3 2 (-1)	26.	4 3 2 (-1)	8 4 3 (-6)
12.	8 6 1 (-4)	4 3 2 1	27.	4 3 2 1	8 6 1 (-4)
13.	8 5 2 (-4)	4 4 1 (-2)	28.	4 4 1 (-2)	8 5 2 (-4)
14.	8 4 3 (-2)	4 4 2 (-1)	29.	4 4 2 (-1)	8 4 3 (-2)
15.	8 4 2 1	4 4 3 (-2)	30.	4 4 3 (-2)	8 4 2 1

2.5. Decodificatoarele și codificatoarele

Decodicatorul este un element funcțional, care reprezintă un circuit logic combinațional și este destinat decodificării cuvintelor binare aplicate la intrările lui. Dacă notăm numărul de intrări ale decodicatorului prin n și numărul de ieșiri prin m , atunci relația dintre aceste numere pentru un decodicator complet este de $m=2^n$. Fiecărei combinații de

variabile de intrare, care se mai numesc și variabile de selecție îi corespunde o singură ieșire, care este activă când combinația respectivă se aplică la intrare, celelalte ieșiri fiind inactive.

Tabelul de adevăr la sinteza unui decodificator complet are dimensiunile de $n+m$ coloane și 2^n rînduri. În primele n coloane sunt reprezentate toate 2^n combinații posibile ale variabilelor, care pot fi aplicate la intrările decodificatorului, iar în celelalte m sunt reprezentate valorile funcțiilor logice care descriu ieșirile decodificatorului. Specificul acestui tabel constă în faptul că fiecare funcție poate avea valoarea egală cu unu doar pentru o singură combinație a variabilelor de intrare, iar pentru celelalte valorile ei sunt egale cu zero. De aceea, este inutilă minimizarea acestor funcții și, în consecință, fiecare din ele va fi egală cu o conjuncție a variabilelor de intrare, iar schema unui decodificator complet va include m elemente logice ȘI cu n intrări fiecare.

Relația dintre numărul de intrări și ieșiri poate fi și $m < 2^n$. În acest caz decodificatorul se numește incomplet și cheltuielile de aparat pot fi micșorate dacă la sinteza decodificatorului se iau în considerare combinațiile neutilizate. În acest caz sinteza schemei decodificatorului practic se reduce la minimizarea a m funcții logice parțial determinate. Particularitățile acestor funcții sunt următoarele: numai pentru o singură combinație funcția este egală cu unu, pentru $m-1$ combinații valoarea ei este egală cu zero, iar pentru 2^n-m combinații funcția nu este determinată.

Vom ilustra cele descrise mai sus printr-un exemplu de sinteză a unui decodificator binar-zecimal. În tabelul 2.4. este prezentată codificarea cifrelor zecimale cu ajutorul codului 842(-3). Tot aici este și tabelul de adevăr pentru cele 10 funcții, care descriu structura decodificatorului.

Tabelul 2.4

Tabelul de adevăr al decodificatorului pentru codul binar-zecimal 842(-3)

Cifra zecimală	Codul				Funcțiile									
	8	4	2	-3										
	x_4	x_3	x_2	x_1	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	1	0	0	0	0	0	0	0
3	0	1	1	1	0	0	0	1	0	0	0	0	0	0
4	0	1	0	0	0	0	0	0	1	0	0	0	0	0
5	1	0	0	1	0	0	0	0	0	1	0	0	0	0
6	0	1	1	0	0	0	0	0	0	0	1	0	0	0
7	1	0	1	1	0	0	0	0	0	0	0	1	0	0
8	1	0	0	0	0	0	0	0	0	0	0	0	1	0
9	1	1	0	1	0	0	0	0	0	0	0	0	0	1

Combinățiile pentru care funcțiile nu sunt determinate: 0001, 0011, 1010, 1100, 1110, 1111.

În figura 2.4 sunt prezentate digramele Karnaugh pentru minimizarea funcțiilor $y_0 - y_9$.

x_4x_3 x_2x_1	00	01	11	10	
00	1		*		y_0
01	*				
11	*		*		
10			*	*	

x_4x_3 x_2x_1	00	01	11	10	
00			*		y_1
01	*	1	*		
11	*				
10			*	*	

Fig. 2. 4. Diagramele Vetch-Karnaugh pentru minimizarea funcțiilor $y_0 - y_9$

x_4x_3					
x_2x_1	00	01	11	10	
00			*		y_2
01	*				
11	*		*		
10	1		*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00			*		y_3
01	*				
11	*	1	*		
10			*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00		1	*		y_4
01	*				
11	*		*		
10			*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00			*		y_5
01	*				
11	*		*		
10			*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00			*		y_6
01	*				
11	*		*		
10		1	*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00			*		y_7
01	*				
11	*		*	1	
10			*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00			*	1	y_8
01	*				
11	*		*		
10			*	*	

x_4x_3					
x_2x_1	00	01	11	10	
00			*		y_9
01	*		1		
11	*		*		
10			*	*	

Fig. 2. 4. (Continuare)

În rezultatul minimizării au fost obținute următoarele funcții logice:

$$\begin{aligned}
 y_0 &= \bar{x}_4 \bar{x}_3 \bar{x}_2 & y_1 &= \bar{x}_4 \bar{x}_2 x_1 & y_2 &= \bar{x}_4 \bar{x}_3 x_2 & y_3 &= x_3 x_2 x_1 \\
 y_4 &= x_3 \bar{x}_2 \bar{x}_1 & y_5 &= \bar{x}_3 \bar{x}_2 x_1 & y_6 &= x_3 x_2 \bar{x}_1 & y_7 &= x_4 x_2 \\
 y_8 &= x_4 \bar{x}_1 & y_9 &= x_4 x_3
 \end{aligned}$$

Schema decodicatorului binar-zecimal 842(-3) și diagrama lui de timp sunt prezentate în figura 2.5.

Costul decodicatorului elaborat în baza acestor funcții logice va fi de 27 unități Quine, spre deosebire de cazul clasic când costul ar fi fost de 40 unități Quine.

Codificatorul este un element funcțional, care reprezintă un circuit logic combinațional și este destinat codificării prin m biți ai uneia din n intrări active dintr-un număr maxim de 2^m intrări.

La sinteza codificatoarelor trebuie de ținut cont de faptul, că concomitent nu pot fi active două sau mai multe intrări, de aceea la sinteza codificatoarelor în tabelul de adevăr fiecare combinație a variabilelor de intrare poate avea valoarea egală cu unu doar pentru o singură variabilă și zero pentru toate celelalte. În acest caz sinteza codicatorului se reduce la reprezentarea fiecărei ieșiri prin disjuncția variabilelor de intrare, care determină egalitatea cu unu a funcției respective. Cele descrise sunt ilustrate mai jos printr-un exemplu de sinteză a unui codificator pentru codul binar-zecimal 8 5(-2)(-4). Tabelul 2.5 reprezintă tabelul de adevăr pentru sinteza acestui codificator.

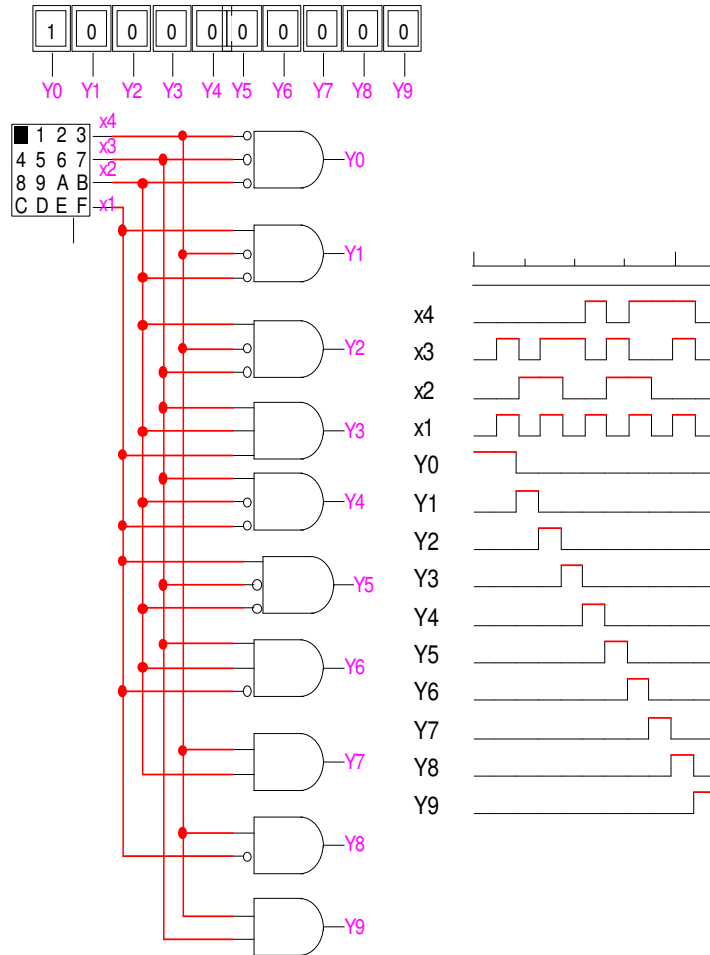


Fig. 2.5. Schema decodificatorului binar-zecimal 842(-3) și diagrama lui de timp

Tabelul 2.5

Tabelul de adevăr al codificatorului pentru codul binar-zecimal 85(-2)(-4)

Cifra zecimală	Intrările										Ieșirile			
											8	5	-2	-4
	x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	f_4	f_3	f_2	f_1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	1	0	1
2	0	0	1	0	0	0	0	0	0	0	1	0	1	1
3	0	0	0	1	0	0	0	0	0	0	0	1	1	0
4	0	0	0	0	1	0	0	0	0	0	1	0	0	1
5	0	0	0	0	0	1	0	0	0	0	0	1	0	0
6	0	0	0	0	0	0	1	0	0	0	1	0	1	0
7	0	0	0	0	0	0	0	1	0	0	1	1	1	1
8	0	0	0	0	0	0	0	0	1	0	1	0	0	0
9	0	0	0	0	0	0	0	0	0	1	1	1	0	1

Setul de funcții care realizează codificatorul din tabelul de mai sus este următorul:

$$\begin{aligned}
 f_4 &= x_2 \vee x_4 \vee x_6 \vee x_7 \vee x_8 \vee x_9; \\
 f_3 &= x_1 \vee x_3 \vee x_5 \vee x_7 \vee x_9; \\
 f_2 &= x_2 \vee x_3 \vee x_6 \vee x_7; \\
 f_1 &= x_1 \vee x_2 \vee x_4 \vee x_7 \vee x_9.
 \end{aligned}
 \tag{2.4}$$

Folosind legile lui De Morgan, transformăm relațiile 2.4 pentru setul de elemente ȘI-NU:

$$\begin{aligned}
 f_4 &= \overline{\bar{x}_2 \& \bar{x}_4 \& \bar{x}_6 \& \bar{x}_7 \& \bar{x}_8 \& \bar{x}_9}; \\
 f_3 &= \overline{\bar{x}_1 \& \bar{x}_3 \& \bar{x}_5 \& \bar{x}_7 \& \bar{x}_9}; \\
 f_2 &= \overline{\bar{x}_2 \& \bar{x}_3 \& \bar{x}_6 \& \bar{x}_7}; \\
 f_1 &= \overline{\bar{x}_1 \& \bar{x}_2 \& \bar{x}_4 \& \bar{x}_7 \& \bar{x}_9}.
 \end{aligned}
 \tag{2.5}$$

Schema codificatorului binar-zecimal 85(-2)(-4) și diagrama lui de timp sunt prezentate în figura 2.6.

2.6. Lucrarea de laborator nr. 3

Sinteza decodificatoarelor și codificatoarelor

Scopul lucrării: studierea practică a structurii și a metodelor de sinteză a decodificatoarelor și codificatoarelor.

Tema pentru acasă

1. Efectuați sinteza unui decodificator complet cu trei variabile de intrare.
2. Efectuați sinteza unui decodificator binar-zecimal conform variantei din tabelul 2.6 (la indicația profesorului).
3. Efectuați sinteza unui codificator binar-zecimal conform variantei din tabelul 2.6 (la indicația profesorului).

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.
2. Se assemblează și se reglează schema unui decodificator binar-zecimal din tema pentru acasă în setul de elemente ȘI-NU.

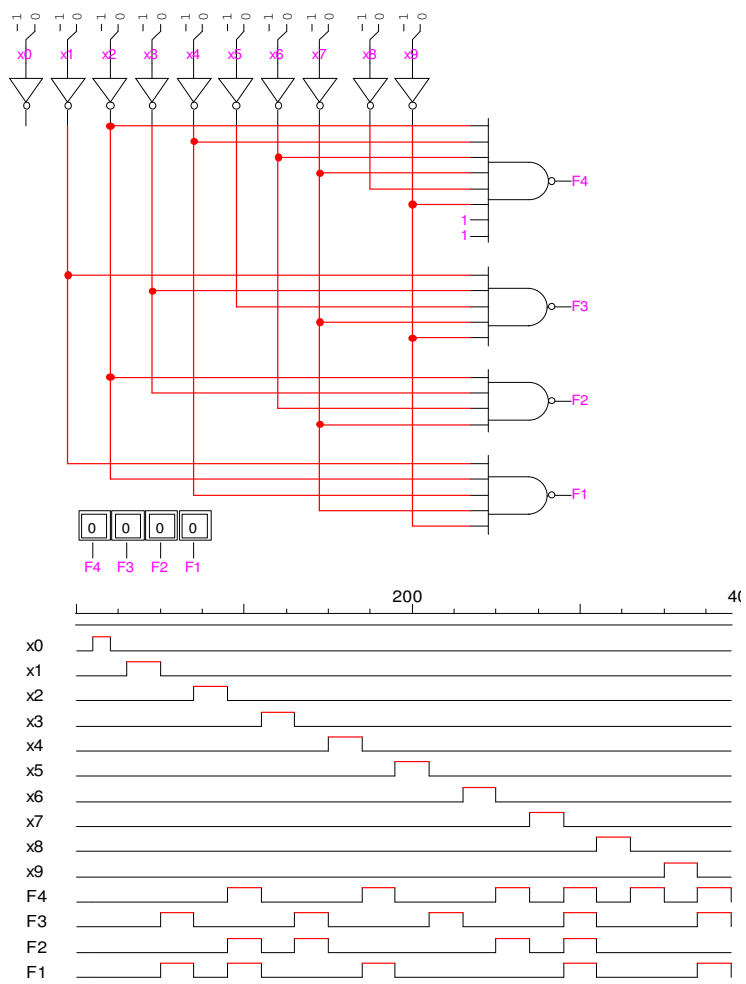


Fig. 2.6. Schema codificatorului binar-zecimal 85(-2)(-4) și diagrama lui de timp

3. Se assemblează și se reglează schema unui codificator binar-zecimal din tema pentru acasă în setul de elemente ȘI-NU.

4. Pentru circuitele asamblate se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele **NAND** cu numărul corespunzător de intrări. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de intrare-ieșire **Binary Probe**, **Hex Keyboard** și **Binary Switch**.

2. Se assemblează schema unui decodificator binar-zecimal din tema pentru acasă în setul de elemente ȘI-NU în **Fereastra de lucru** și se verifică corectitudinea lui. Se studiază diagrama de timp.

3. Se assemblează schema unui codificator binar-zecimal din tema pentru acasă în setul de elemente ȘI-NU în **Fereastra de lucru** și se verifică corectitudinea lui. Se studiază diagrama de timp.

4. Pentru circuitele asamblate se determină costul și timpul de reținere.

Tabelul 2.6

Variante pentru îndeplinirea lucrării de laborator

Nr. crt.	Codul binar-zecimal		Nr. crt.	Codul binar-zecimal	
	Decodificator	Codificator		Decodificator	Codificator
1	2	3	4	5	6
1.	8 7 (-2)(-4)	5 3 2 (-1)	16.	3 3 2 1	8 6 1 (-4)
2.	8 6 (-1)(-4)	5 3 1 (-1)	17.	4 2 2 1	8 6 (-1)(-4)
3.	8 5 (-2)(-4)	3 3 2 1	18.	4 3 1 1	8 7 (-2)(-4)
4.	8 4 3 (-6)	4 2 2 1	19.	4 3 2 (-1)	8 5 (-2)(-4)
5.	8 6 1 (-4)	4 3 1 1	20.	4 3 2 1	8 6 1 (-4)

Tabelul 2.6 (Continuare)

1	2	3	4	5	6
6.	8 5 2 (-4)	4 3 2 (-1)	21.	4 4 1 (-2)	8 5 2 (-4)
7.	8 4 3 (-2)	4 3 2 1	22.	4 4 2 (-1)	8 4 3 (-2)
8.	8 4 2 1	4 4 1 (-2)	23.	4 4 3 (-2)	8 4 2 1
9.	8 4 2 (-1)	4 4 2 (-1)	24.	4 4 2 1	8 4 2 (-1)
10.	8 4 2 (-3)	4 4 3 (-2)	25.	5 2 1 1	8 4 2 (-3)
11.	8 4 1 (-2)	4 4 2 1	26.	5 2 2 (-1)	8 4 1 (-2)
12.	8 3 2 (-4)	5 2 1 1	27.	5 3 2 (-1)	8 3 2 (-4)
13.	8 4 2 (-5)	5 2 2 (-1)	28.	5 2 2 1	8 4 2 (-5)
14.	8 4 1 (-6)	5 3 2 (-1)	29.	5 2 2 (-1)	8 4 1 (-6)
15.	8 4 1 (-2)	5 2 2 1	30.	5 3 2 (-1)	8 4 1 (-2)

Întrebări

1. De ce depinde numărul de funcții logice, care trebuie minimizate la sinteza decodificatoarelor incomplete și care sunt particularitățile lor?

2. Care este raportul dintre numărul de intrări și numărul de ieșiri ale decodificatoarelor complete și incomplete.

2.7. Sumatoarele binare

Sumatoarele se includ în clasa de circuite logice combinaționale, în care semnalele prelucrate sunt asociate unor numere. Sumatoarele execută operația de adunare a două numere și operația de scădere, care constă în sumarea descăzutului cu codul complementar al scăzătorului.

Realizarea structurii tuturor sumatoarelor pornește de la două scheme logice de sumare pe un bit, cunoscute în literatura de specialitate ca schema semisumatorului și respectiv schema sumatorului complet. Un rang al unui sumator binar complet are

două intrări de date ale cifrelor operanzilor din rangul respectiv și o intrare de transport din rangul vecin mai puțin semnificativ, și produce la ieșire rezultatul sumei în rangul respectiv și bitul de transport în rangul următor mai semnificativ. Tabelul de adevăr care descrie funcționarea unui rang al sumatorului complet și simbolul de reprezentare al acestuia este prezentat în figura 2.7.

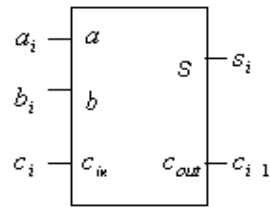
În rangul i sunt sumați doi biți a_i și b_i (valorile binare din rangul i ale ambilor operanzi) prezenți la cele două intrări de date, precum și bitul de transport c_i de la rangul vecin mai puțin semnificativ. Se generează două ieșiri s_i și c_{i+1} , care sunt rezultatul sumei în rangul i și respectiv cifra de transport în rangul următor mai semnificativ. Din tabelul de adevăr putem deduce, că sistemul de funcții logice, cu ajutorul căruia se descrie structura unui rang a sumatorului binar complet este următoarea:

$$s_i = \bar{a}_i b_i \bar{c}_i \vee a_i \bar{b}_i \bar{c}_i \vee \bar{a}_i \bar{b}_i c_i \vee a_i b_i c_i = a_i \oplus b_i \oplus c_i;$$

$$c_{i+1} = a_i b_i \vee a_i c_i \vee b_i c_i.$$

a_i	b_i	c_i	s_i	c_{i+1}
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

a)



b)

Fig. 2.7. Sumatorul complet: a) tabelul de adevăr; b) simbolul de reprezentare

Pe baza sumatorului complet se poate realiza simplu structura unui sumator de n biți. Pentru $n=4$ structura este cea din figura 2.8. Se observă propagarea succesivă a semnalului pe traseul bitului de transport. În consecință, rezultatul va fi disponibil la ieșire doar după ce semnalul corespunzător bitului de transport va parcurge întreg traseul.

La sumatoarele cu propagarea succesivă a transportului, lanțul de propagare va introduce o întârziere maximă în cazul când transportul produs în rangul cel mai puțin semnificativ al sumatorului se propagă prin toate rangurile lui pînă ajunge la cel mai semnificativ. Este evident că timpul maximal de funcționare a sumatorului cu transport succesiv crește în dependență liniară de numărul său de ranguri. Intervalul de timp mare de sumare este prețul plătit de sumatoarele cu transport succesiv pentru simplitatea structurii.

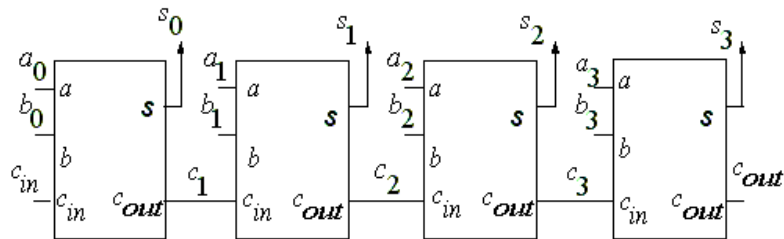


Fig. 2.8. Structura sumatorului cu transport succesiv

Sumatorul este una din cele mai importante componente ale oricărui calculator, de aceea viteza lui de lucru reprezintă un criteriu de calitate primordial. Pentru creșterea vitezei de lucru a sumatoarelor au fost propuse și implementate diverse măsuri de natură tehnologică și arhitecturală. Măsurile luate în plan tehnologic pot fi de exemplu:

- a) circuitele logice pe care le parcurge semnalul de transport

trebuie proiectate în așa fel, încât să fie redus timpul de propagare;

b) se operează cu semnalul de transport inversat între intrare și ieșire;

Aceste măsuri însă nu sunt suficiente atunci, când trebuie realizate sumatoare pentru cuvinte cu un număr mare de biți. Soluția în acest caz este arhitecturală și presupune renunțarea la transportul succesiv în favoarea celui anticipat (accelerat). Aceasta de fapt înseamnă redefinirea funcției logice a acelei părți a sumatorului, care formează semnalul de transport.

Relația de definiție a transportului poate fi scrisă sub forma:

$$c_{i+1} = a_i b_i \vee a_i c_i \vee b_i c_i = a_i b_i \vee (a_i \vee b_i) c_i.$$

Notînd $G_i = a_i b_i$ și $P_i = a_i \vee b_i$, relația de definiție a transportului devine:

$$c_{i+1} = G_i \vee P_i c_i .$$

Funcția de generare $G_i=1$ indică faptul că din rangul i al sumatorului se generează o cifră de transport egală cu 1 indiferent de valoarea cifrei de transport, care vine din rangul vecin mai puțin semnificativ.

Funcția de propagare $P_i=1$ indică faptul că prin rangul i al sumatorului se va propaga valoarea transportului c_i de la ieșirea din rangul vecin mai puțin semnificativ.

Structura sumatorului cu transport anticipat este compusă din două module: modulul de sumare propriu-zisă, alcătuit din mai multe sumatoare de un rang și modulul (schema) de transport anticipat, care generează simultan semnalele de transport pentru toate rangurile sumatorului. Pentru cuvinte cu lungimea de patru biți schema sumatorului cu transport anticipat este prezentată în figura 2.9. Modulul de sumare este format din patru sumatoare complete de un rang.

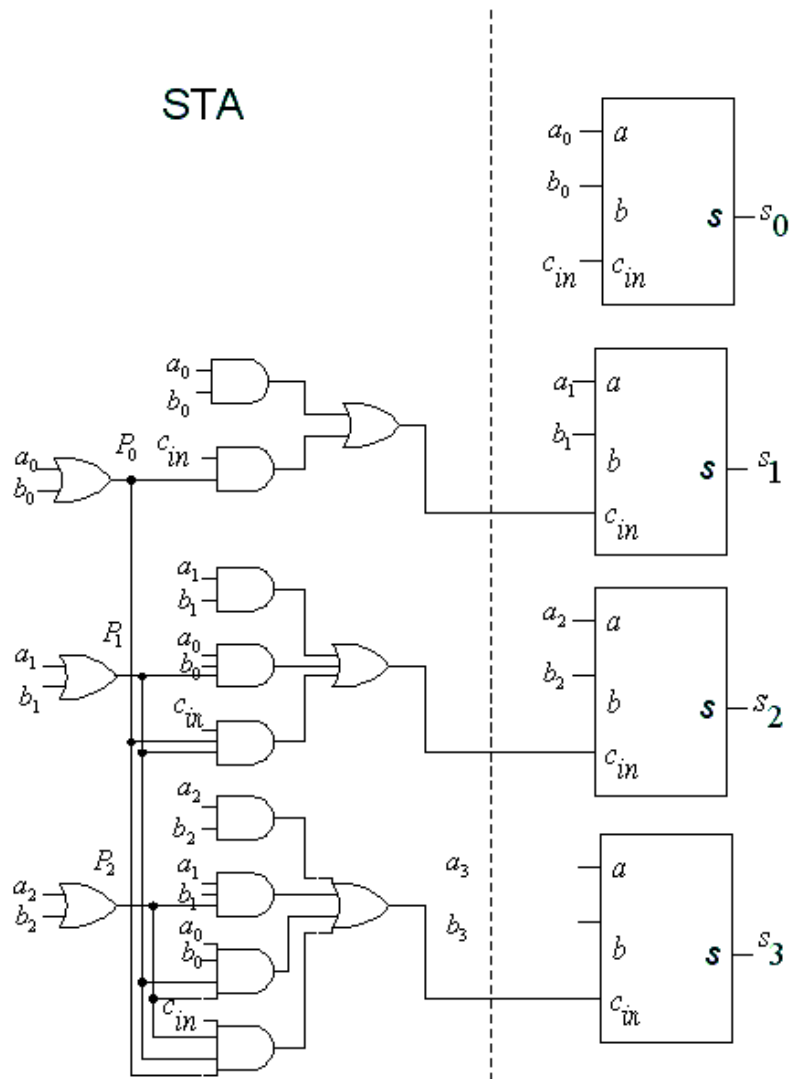


Fig. 2.9. Structura sumatorului cu transport anticipat

Generarea semnalelor de transport se bazează pe următoarele relații:

$$c_1 = G_0 \vee P_0 c_{IN};$$

$$c_2 = G_2 \vee P_1 G_0 \vee P_1 P_0 c_{IN};$$

$$c_3 = G_2 \vee P_2 G_1 \vee P_2 P_1 G_0 \vee P_2 P_1 P_0 c_{IN}.$$

Timpul total de sumare al unui sumator cu transport anticipat este egal cu timpul de sumare al unui sumator de un rang plus întârzierea introdusă de schema transportului anticipat, nedepinzând de numărul de ranguri.

2.8. Lucrarea de laborator nr. 4

Sinteza sumatoarelor binare

Scopul lucrării: însușirea deprinderilor practice de sinteză a sumatorului complet și a sumatorului binar cu transport succesiv și anticipat.

Tema pentru acasă

1. De efectuat sinteza sumatorului complet în setul de elemente ȘI-NU.
2. De efectuat sinteza sumatorului binar pe 4 biți cu transport anticipat în setul de elemente ȘI-NU.

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.

2. Se asamblează și se reglează schema sumatorului binar complet.

3. Se asamblează și se reglează schema sumatorului binar pe 4 biți cu transport anticipat. Se realizează prin intermediul acestei scheme un exemplu de adunare și unul de scădere (la indicația profesorului).

4. Pentru circuitele asamblate se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele **NAND și XOR** cu numărul corespunzător de intrări. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de intrare-ieșire **Binary Probe, Hex Keyboard și Binary Switch**.

2. Se asamblează două scheme ale sumatorului binar complet:

a) în setul de elemente ȘI-NU;

b) utilizând elemente XOR și SI-NU.

Se verifică corectitudinea lor. Se studiază diagramele de timp.

3. Se asamblează schema sumatorului binar pe 4 biți cu transport anticipat utilizând elemente XOR pentru fiecare rang de sumare și elemente ȘI-NU pentru transportul anticipat. Se verifică corectitudinea schemei. Se studiază diagrama de timp.

4. Pentru circuitele asamblate se determină costul și timpul de reținere.

Întrebări

1. Determinați timpul maximal de sumare pentru structurile din figurile 2.8 și 2.9, considerînd că reținerea semnalului într-un element logic este egală cu τ .

2. Care sunt restricțiile care intervin în sinteza schemei transportului anticipat odată cu creșterea lungimii operanzilor?

2.9. Comparatoarele

Comparatoarele sunt incluse în clasa circuitelor logice combinaționale care asigură compararea cuvintelor binare. Comparatorul elementar reprezintă un circuit combinațional simplu capabil să detecteze egalitatea a două cuvinte binare, aplicate la intrările de date, furnizînd la ieșire valoarea respectivă a unui bit special de egalitate. Evident, că cuvintele de la intrările de date ar putea să nu fie egale. În cazul cînd cuvintele comparate sunt numere, atunci prin comparare se va putea stabili nu doar dacă sunt egale sau nu cuvintele respective, ci și care din ele este mai mare sau mai mic. În acest caz comparatorul va fi mai complex și va produce trei semnale de ieșire: bitul de superioritate, bitul de inferioritate și bitul de egalitate.

Cele mai simple sunt comparatoarele, la care unul dintre cuvintele, ce se compară este cunoscut dinainte, adică este o constantă. De aceea ele se numesc comparatoare cu constante și pot fi folosite atît la detectarea egalității, cît și a inegalității cuvintelor comparate. Structura acestor comparatoare este foarte simplă și sinteza lor se efectuează în felul următor. Pentru detectarea egalității se folosește un element logic ȘI, numărul de intrări al căruia este egal cu numărul de biți al cuvintelor comparate. Cuvîntul ce se compară cu constanta se aplică la intrările elementului logic în felul următor. Dacă valoarea constantei în rangul i este egală cu 1, atunci rangul respectiv se aplică direct, iar dacă această valoare este 0, atunci rangul respectiv se aplică printr-un inversor. În consecință la ieșirea elementului ȘI respectiv vom avea 1 logic doar atunci, cînd cuvîntul aplicat va coincide cu constanta respectivă.

Pentru a efectua sinteza unui comparator de detectare a inegalității unui cuvînt cu o constantă se folosește un element logic SAU, numărul de intrări al căruia este egal cu numărul de biți al cuvintelor comparate. Aplicarea la intrările

elementului logic al cuvântului ce se compară cu constanta se efectuează în felul următor. Dacă valoarea constantei în rangul i este egală cu 0, atunci rangul respectiv se aplică direct, iar dacă această valoare este 1, atunci rangul respectiv se aplică printr-un inversor. La ieșirea elementului SAU în acest caz vom avea 1 logic atunci, când cuvântul aplicat nu va coincide cu constanta respectivă și 0 doar în cazul, când cuvintele comparate vor coincide.

Sinteza unui comparator, care este destinat de a stabili dacă un număr oarecare face sau nu parte dintr-un interval, se efectuează prin metoda clasică, adică tabelul de adevăr al funcției care exprimă ieșirea comparatorului, se completează cu 1 pentru combinațiile ce exprimă valorile numerelor din intervalul respectiv și cu 0 pentru celelalte combinații. După aceea funcția respectivă se minimizează.

Sinteza comparatoarelor, destinate stabilirii care din două numere este mai mare, este practic imposibilă dacă se folosesc metodele clasice. Să presupunem, că este necesară sinteza unui comparator, care ar stabili egalitatea sau care din două numere de opt biți este mai mare sau mai mic. Metoda clasică de sinteză ar necesita construirea unui tabel de adevăr cu $2^{(8+8)}=65536$ rânduri și ulterioara minimizare a funcției (funcțiilor) respective. În asemenea cazuri soluția este utilizarea metodei de decompoziție a problemei, care presupune soluționarea prin fragmentare. Datorită fragmentării rezolvarea și soluționarea unei probleme complexe se reduce la formularea și soluționarea unor probleme mai simple. Să exemplificăm cele expuse, prezentînd în continuare sinteza comparatorului cu trei ieșiri.

Fie că avem de comparat două cuvinte binare $A=a_3a_2a_1a_0$ și $B=b_3b_2b_1b_0$. Sinteza directă ar necesita scrierea formelor canonice pentru trei funcții $F_{A=B}$ – de egalitate, $F_{A>B}$ – de superioritate, $F_{A<B}$ – de inferioritate dintr-un tabel de adevăr cu $2^{(4+4)} = 256$ rânduri. Practic sinteza se va realiza prin compararea separată a cifrelor de rang 3, 2, 1, 0. Pentru aceasta este necesară sinteza unui

element de comparare, care compară două cuvinte de un bit, producând trei ieșiri. Pentru aceste ieșiri vom obține funcțiile f_e – de egalitate, f_s – de superioritate și f_i – de inferioritate. Apoi cu ajutorul elementului proiectat se va construi un comparator de patru biți.

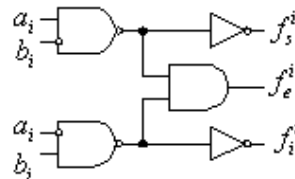
În figura 2.10,a este prezentat tabelul de adevăr pentru funcțiile f_e , f_s și f_i . Aceste trei funcții se reprezintă cu ajutorul următoarelor expresii logice:

$$\begin{aligned} f_e &= \bar{a}_i \bar{b}_i \vee a_i b_i = \overline{\bar{a}_i b_i} \wedge \overline{a_i \bar{b}_i}; \\ f_s &= a_i \bar{b}_i; \\ f_i &= \bar{a}_i b_i. \end{aligned} \quad (2.6)$$

Circuitul care realizează funcțiile logice de mai sus este prezentat în figura 2.10,b.

a_i	b_i	f_e	f_s	f_i
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

a)



b)

Fig. 2.10. Elementul de comparare a doi biți
a) tabelul de adevăr; b) schema logică

Folosind relațiile (2.6) putem scrie cele trei funcții logice în conformitate cu care funcționează comparatorul de patru biți:

- relația de egalitate $A=B$ se exprimă prin funcția logică:

$$F_{A=B} = f_{e3} f_{e2} f_{e1} f_{e0}, \quad (2.7)$$

deoarece relația $A=B$ presupune că $a_3=b_3$, $a_2=b_2$, $a_1=b_1$ și $a_0=b_0$;

- relația de superioritate $A>B$ presupune că $a_3>b_3$; sau $a_3=b_3$ și $a_2>b_2$; sau $a_3=b_3$ și $a_2=b_2$ și $a_1>b_1$; sau $a_3=b_3$ și $a_2=b_2$ și $a_1=b_1$ și $a_0>b_0$ ceea ce conduce la funcția logică:

$$F_{A>B} = f_{s3} \vee f_{e3}f_{s2} \vee f_{e3}f_{e2}f_{s1} \vee f_{e3}f_{e2}f_{e1}f_{s0} \quad (2.8)$$

relația de inferioritate $A < B$ presupune că $a_3 < b_3$; sau $a_3 = b_3$ și $a_2 < b_2$; sau $a_3 = b_3$ și $a_2 = b_2$ și $a_1 < b_1$; sau $a_3 = b_3$ și $a_2 = b_2$ și $a_1 = b_1$ și $a_0 < b_0$ de unde rezultă funcția logică:

$$F_{A<B} = f_{i3} \vee f_{e3}f_{i2} \vee f_{e3}f_{e2}f_{i1} \vee f_{e3}f_{e2}f_{e1}f_{i0} \quad (2.9)$$

Schema comparatorului paralel care realizează funcțiile (2.7), (2.8) și (2.9) este prezentată în figura 2.11 (este notat prin C_i elementul pentru compararea cifrelor de rangul i).

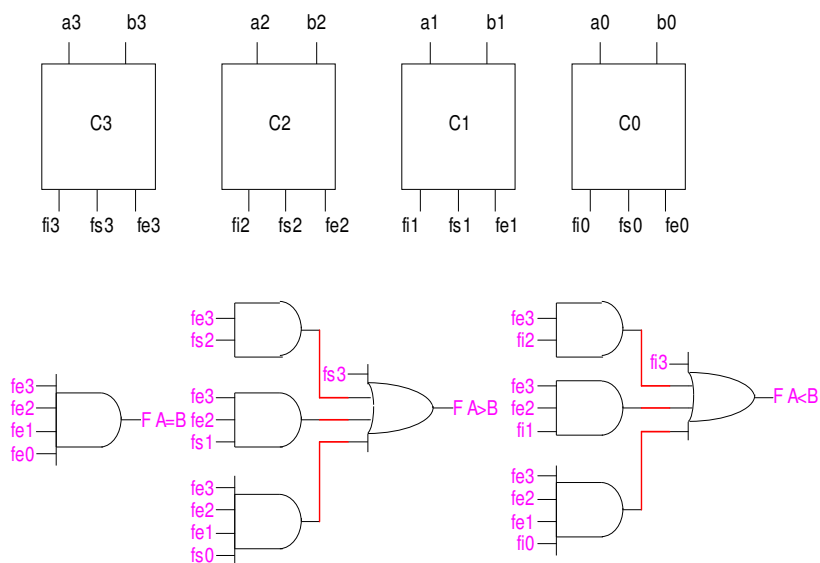


Fig. 2.11. Modulul comparator paralel de patru biți

Același circuit este realizat în varianta succesivă în figura 2.12.

Altă modalitate de sinteză a comparatorului presupune realizarea operației $(A-B)$ pe un sumator, după care urmează

analiza rezultatului obținut. După cum se știe, sumatoarele execută operația de scădere prin sumarea descăzutului la codul complementar al scăzătorului. De aceea, cuvântul A se va aplica direct la una dintre cele două intrări de date ale sumatorului, iar pentru a obține codul complementar al lui B acesta trebuie aplicat la a doua intrare de date, fiind în prealabil inversat, iar la intrarea de transport a celui mai puțin semnificativ rang al sumatorului trebuie aplicat unu logic.

Pentru a stabili cum se va determina relația dintre cuvintele comparate cu ajutorul sumatorului vom lua ca exemplu două numere pozitive (bitul semnului lipsește) A și B cu lungimea de patru biți. Efectuăm operația de scădere și analizăm rezultatele obținute pentru toate cele trei cazuri posibile: $A > B$, $A = B$, $A < B$.

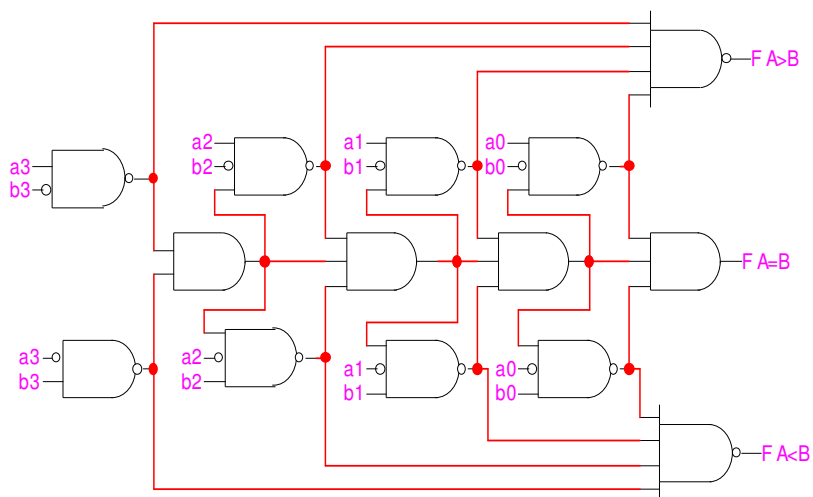


Fig. 2.12. Modulul comparator succesiv de patru biți

$A > B$	$A = B$	$A < B$
$A = 11_{(10)} = 1011_{(2)}$	$A = 11_{(10)} = 1011_{(2)}$	$A = 9_{(10)} = 1001_{(2)}$
$B = 10_{(10)} = 1010_{(2)}$	$B = 11_{(10)} = 1011_{(2)}$	$B = 12_{(10)} = 1100_{(2)}$
$(A-B) \Rightarrow 1011$	$(A-B) \Rightarrow 1011$	$(A-B) \Rightarrow 1001$
+ 0101	+ 0100	+ 0011
_____1	_____1	_____1
01 001	01 000	00 101
C_{out} S	C_{out} S	C_{out} S

Din exemplul prezentat rezultă că:

- relația de superioritate $A > B$ are loc când cifra transportului următor $C_{out}=1$ și suma $S \neq 0$;
- relația de egalitate $A = B$ are loc când suma $S = 0$.
- relația de inferioritate $A < B$ este adevărată când cifra transportului următor $C_{out}=0$.

Schema comparatorului obținută în urma analizei efectuate mai sus pentru două cuvinte de patru biți este prezentată în figura 2.13.

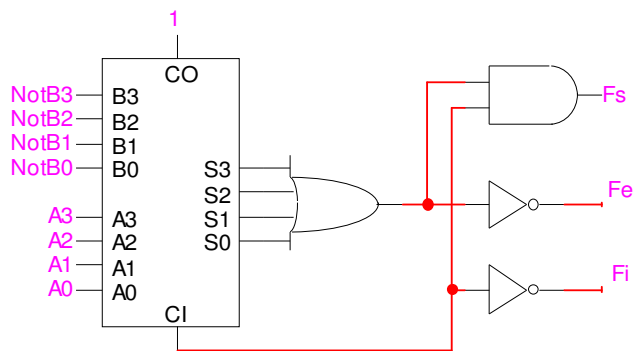


Fig. 2.13. Comparatorul de patru biți în baza sumatorului

2.10. Lucrarea de laborator nr. 5

Sinteza comparatoarelor

Scopul lucrării: însușirea deprinderilor practice de sinteză a diferitor tipuri de comparatoare.

Tema pentru acasă

1. Efectuați sinteza unui comparator cu cinci intrări și trei ieșiri conform variantei proprii din tabelul 2.7, având în vedere că:
I ieșire – compararea la egalitate cu o constantă;
II ieșire – compararea la inegalitate cu o constantă;
III ieșire – depistarea intervalului.
2. Efectuați sinteza modulului comparator de patru biți, utilizând setul de elemente SI-NU.
3. Efectuați sinteza comparatorului de opt biți în baza sumatorului

Tabelul 2.7

Variante pentru îndeplinirea lucrării de laborator

Nr. crt.	I ieșire	II ieșire	III ieșire	Nr. crt.	I ieșire	II ieșire	III ieșire
1	2	3	4	5	6	7	8
1.	1.	30	10-19	16.	16.	15	7-15
2.	2.	29	11-19	17.	17.	14	7-16
3.	3.	28	12-18	18.	18.	13	7-17
4.	4.	27	10-20	19.	19.	12	8-19
5.	5.	26	1-11	20.	20.	11	8-20
6.	6.	25	2-10	21.	21.	10	8-18

Tabelul 2.7 (Continuare)

1	2	3	4	5	6	7	8
7.	7.	24	3-12	22.	22.	9	9-19
8.	8.	23	4-12	23.	23.	8	9-24
9.	9.	22	14-19	24.	24.	7	9-15
10.	10.	21	1-14	25.	25.	6	9-26
11.	11.	20	13-20	26.	26.	5	10-21
12.	12.	19	16-29	27.	27.	4	10-22
13.	13.	18	1-26	28.	28.	3	10-23
14.	14.	17	1-27	29.	29.	2	15-24
15.	15.	16	14-28	30.	30.	1	16-25

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.
2. Se asamblează și se reglează schema comparatorului cu cinci intrări și trei ieșiri.
3. Se asamblează și se reglează schema modulului comparator de patru biți. Se efectuează compararea a două cuvinte binare de patru biți (la indicația profesorului).
4. Se asamblează și se reglează schema comparatorului de opt biți în baza sumatorului. Se efectuează compararea a două cuvinte binare de opt biți (la indicația profesorului).
5. Pentru circuitele asamblate se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele **NAND** cu numărul corespunzător de intrări. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de

intrare-ieșire **Binary Probe**, **Hex Keyboard** și **Binary Switch**.

2. Se asamblează schema comparatorului cu cinci intrări și trei ieșiri în **Fereastra de lucru** și se verifică corectitudinea ei. Se studiază diagrama de timp.

3. Se asamblează schema modulului comparator de patru biți. Se efectuează compararea a două cuvinte binare de patru biți (la indicația profesorului). Se studiază diagrama de timp.

4. Se asamblează schema comparatorului de opt biți în baza sumatorului. Se efectuează compararea a două cuvinte binare de opt biți (la indicația profesorului). Se studiază diagrama de timp.

5. Pentru circuitul asamblat se determină costul și timpul de reținere.

Întrebări

1. Demonstrați că funcția de egalitate f_e din relația (2.6) poate fi descrisă cu relația $f_e = \overline{a_i b_i} \vee \overline{a_i b_i}$.

2. Care este viteza de lucru a comparatorului din figura 2.11? Depinde ea de numărul de ranguri?

3. Efectuați o analiză comparativă a schemelor din figurile 2.11 și 2.12.

4. E posibilă oare sinteza comparatorului numai cu funcțiile de superioritate f_s și inferioritate f_i , excluzând funcția de egalitate? Dacă da, atunci scrieți relațiile pentru funcțiile logice $F_{A=B}$, $F_{A>B}$, $F_{A<B}$ și caracterizați eventualul comparator sub aspectul costului de implementare și al vitezei de lucru. Comparați-l cu cel din figura 2.11.

5. Modificați modulul comparator din figura 2.11 astfel, încât să devină posibilă expandarea acestuia pentru compararea cuvintelor de 4 biți.

3. CIRCUITELE LOGICE SECVENȚIALE

3.1. Particularitățile procesului de sinteză a circuitelor logice secvențiale

Circuitele logice secvențiale (CLS) se caracterizează prin faptul că în orice moment de timp vectorul de ieșire al circuitului depinde nu numai de semnalele de la intrare din acel moment (ignorând timpul de propagare) ci și de semnalele de la intrare aplicate în momentele de timp anterioare. Această dependență se datorează prezenței în CLS a unor elemente de memorie, care reprezintă mai multe stări logice stabile, comandate prin intrări și participă la formarea semnalelor de ieșire. Rezultă că un CLS conține o schemă combinațională completată cu o structură de memorare.

Sinteza unui CLS se efectuează în următoarele etape:

- descrierea necesităților ce trebuie să le rezolve circuitul respectiv (prin text, desen, diagrame etc.);
- reprezentarea acestei descrieri sub forma unui tabel de tranziție;
- deducerea funcțiilor logice și minimizarea acestora;
- implementarea acestor funcții minimizate sub forma unor rețele de comutare prin intermediul circuitelor integrate.

Tabelul de tranziție se deosebește de tabelul de adevăr prin faptul că aici trebuie de luat în considerare și factorul timpului. Acest tabel include următoarele părți componente:

- valoarea variabilelor de intrare la orice moment de timp t ;
- valoarea funcțiilor (vectorilor) de ieșire la momentul de timp t ;
- valoarea funcțiilor (vectorilor) de ieșire, care vor rezulta la momentul de timp $t+1$ în urma aplicării semnalelor de intrare în momentul de timp precedent t .

În calitate de valori ale funcțiilor logice, care trebuie

deduse și minimizate, se iau valorile funcțiilor de ieșire la momentul $t+1$, iar în calitate de variabile ale acestor funcții se iau valorile variabilelor de intrare la momentul t și valorile funcțiilor de ieșire la momentul t , adică toate datele care asigură tranziția circuitului respectiv de la starea care o are în momentul t la starea care trebuie s-o aibă în momentul $t+1$.

3.2. Circuitele basculante bistabile

Circuitele basculante bistabile (CBB) sunt circuite elementare secvențiale și se caracterizează prin faptul că tot timpul prezintă una din cele două posibile stări stabile. Aceste două stări codificabile prin cele două valori binare (0 sau 1) se utilizează, de regulă, în calculatoarele numerice, ca suport pentru memorarea unui bit.

Există mai multe tipuri de circuite basculante bistabile sau prescurtat bistabile. Primul ca ordine a apariției și cel mai simplu este bistabilul de tip RS. Funcționarea acestuia este descrisă de tabelul 3.1.

Tabelul 3.1

Tabelul de tranziție al bistabilului RS

S	R	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	*
1	1	1	*

În acest tabel S (Set) este intrarea de scriere, R (Reset) - intrarea de ștergere, Q_t -starea bistabilului în momentul t , iar Q_{t+1} este starea bistabilului în momentul $t+1$, care rezultă în urma aplicării valorilor semnalelor respective la intrare și stării bistabilului în momentul t . Combinația $S=R=1$ este interzisă, deoarece în urma aplicării ei starea bistabilului va fi incertă din cauza asimetriei reale a elementelor logice folosite la implementarea bistabilului.

Din tabelul 3.1 ușor se poate deduce că:

$$Q_{t+1} = S \vee \overline{R}Q_t, \quad (3.1)$$

cu condiția

$$RS = 0. \quad (3.2)$$

Cu ajutorul formulelor De Morgan expresia (3.1) poate fi ușor transformată în una din următoarele două forme:

$$Q_{t+1} = S \vee \overline{R \vee \overline{Q}_t}, \quad (3.3)$$

$$Q_{t+1} = \overline{\overline{S} \wedge \overline{R} \overline{Q}_t}. \quad (3.4)$$

Expresiile (3.3) și (3.4) pot fi implementate cu ajutorul a două elemente logice SAU-NU (figura 3.1, a) sau cu ajutorul a două elemente logice ȘI-NU (figura 3.1, b) și în așa fel se obține schema bistabilelor RS asincrone.



Fig. 3.1. Circuitul bistabil RS asincron:

a) în setul ȘI-NU; b) în setul SAU-NU

În foarte multe aplicații este utilizat bistabilul *RS* sincron (figura 3.2). Elementele logice 1 și 2 formează bistabilul propriu-zis, iar elementele 3 și 4 realizează sincronizarea cu un semnal de ceas, *CLK*. Acest semnal activează intrările *S* și *R* numai când se află în starea logică 1, inhibându-le pe durata de timp cât se află în stare logică 0.

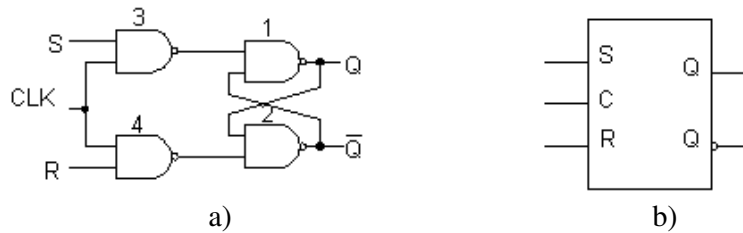


Fig. 3.2. Circuitul bistabil *RS* sincron:
a) schema logică; b) simbolul

Pornind de la bistabilul *RS*, se obține bistabilul de tip *D* în cazul când se stabilește condiția $S = \bar{R}$. În acest caz semnalul de la ieșire va fi identic cu cel de la intrare. Funcționarea bistabilului de tip *D* este descrisă de relația:

$$Q_{t+1} = D,$$

în care *D* este intrarea de date. Această ultimă relație reflectă capacitatea bistabilului de tip *D* de a păstra direct o informație, aplicația principală a circuitului fiind memorarea cuvintelor binare. Cea mai simplă schemă a circuitului bistabil *D* sincron este prezentată în figura 3.3. Faptul că informația de la intrare este transmisă la ieșire și apoi intrarea este inactivată a condus la denumirea de latch (lacăt) atribuită bistabilelor de tip *D*.



Fig. 3.3. Circuitul bistabil D sincron:

a) schema; b) simbolul

Existența restricției $S=R=1$ constituie o dificultate pentru proiectanți, de aceea, foarte des se recurge la utilizarea circuitului bistabil de tip JK . Bistabilul JK își păstrează funcționalitatea și în cazul când $S=R=1$. Această combinație a semnalelor de la intrare se folosește pentru inversarea stării bistabilului. Funcționarea bistabilului JK este prezentată în tabelul de tranziție 3.2, iar funcția logică ce rezultă din acest tabel este reflectată de expresia logică (3.5).

$$Q_{t+1} = J\bar{Q}_t \vee \bar{K}Q_t, \quad (3.5)$$

Tabelul 3.2

Tabelul de tranziție al bistabilului JK

J	K	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Din tabelul 3.2 se observă că intrările J și K pot fi tratate ca intrări S și respectiv R . Însă în cazul, când $J=K=1$, bistabilul își schimbă starea anterioară ($0 \rightarrow 1$, respectiv $1 \rightarrow 0$).

Bistabilul JK se realizează, de obicei, în baza bistabilului RS într-o formă cunoscută sub numele de master-slave (stăpîn-sclav). Circuitul JK de tip master-slave (fig. 3.4) conține două bistabile RS .

Analizînd modul de funcționare a circuitului se poate constata că informația de la intrare este transferată la ieșire pe frontul negativ al semnalului de ceas. Asemenea circuite sunt denumite în literatura de specialitate ca bistabile flip-flop.

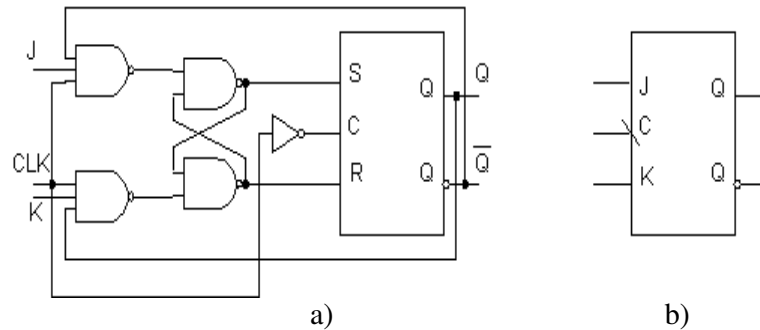


Fig. 3.4. Circuitul bistabil JK sincron:
a) schema; b) simbolul

Bistabilul JK este universal, avînd aplicații multiple. După cum s-a menționat mai sus, el poate funcționa în regim de bistabil RS . Dacă intrările J și K sunt conectate între ele printr-un inversor, circuitul devine bistabil D , cu intrarea D pe linia J . Dacă $J=K=1$, atunci bistabilul JK devine bistabil de tip T , care basculează la fiecare impuls de ceas.

3.3. Registrele

Registrele se includ în categoria elementelor funcționale secvențiale și sunt destinate memorării și procesării cuvintelor binare. Componenta de bază a oricărui registru o constituie bistabilele. Structura generală a unui registru este prezentată în figura 3.5. și este constituită din n bistabile (în cazul de față de tip D), avînd un semnal de ceas CLK comun pentru toate bistabilele (CB). Intrarea de ștergere \overline{CLR} , activă pe zero logic, este prezentă la majoritatea registrelor și permite resetarea celor n bistabile. Intrările S_1 și S_0 comandă cele n comutatoare logice (CM), asigurînd astfel selecția regimului de lucru al registrului. Comutatoarele, în dependență de codul de selecție, pot asigura conectarea intrărilor bistabilelor în trei moduri: la ieșirea B_i din stînga, din dreapta sau la intrarea de date D . În dependență de conectarea intrărilor bistabilelor, registrul poate încărca un cuvînt binar în cod paralel sau succesiv.

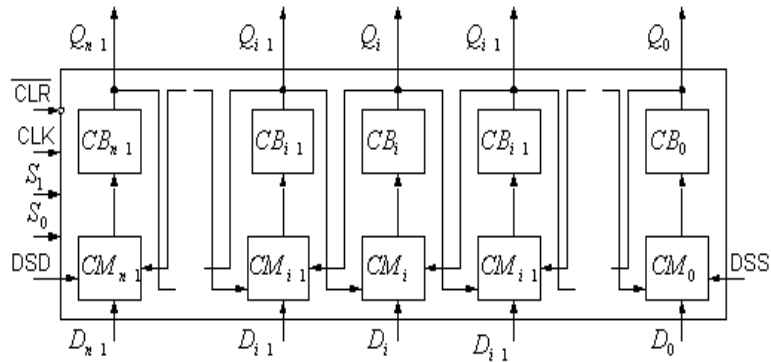


Fig. 3.5. Structura generală a unui registru

În regimul de încărcare paralel, cuvântul pentru înscriere se aplică la intrările de date D_{n-1}, \dots, D_0 – portul de intrare. Cuvântul înscris este accesibil la ieșirile Q_{n-1}, \dots, Q_0 – portul de ieșire. Încărcarea datelor în registru se realizează la aplicarea semnalului de ceas.

În regimul succesiv de încărcare a datelor cuvântul binar poate fi deplasat spre dreapta sau spre stînga. Pentru înscrierea succesivă se folosesc două intrări de date: spre dreapta DSD și respectiv spre stînga DSS .

Registru capabil să deplaseze datele atît la stînga, cît și la dreapta se numește registru cu deplasare bidirecțională. Acest registru poate fi utilizat nu numai pentru memorarea unui cuvînt binar, ci și pentru procesarea lui, deoarece deplasarea spre dreapta cu i poziții este echivalentă cu operația de împărțire a cuvîntului la 2^i iar deplasarea spre stînga – cu operația de înmulțire cu 2^i .

Cuvîntul de n biți înscris pe intervalul de n tacte prin intrarea DSD , respectiv DSS este pierdut secvențial bit cu bit la ieșirea Q_0 , respectiv Q_{n-1} pe următorul interval de n tacte. Dacă însă se conectează ieșirea Q_0 la DSD , respectiv Q_{n-1} la DSS se obține structura de registru în inel sau registru cu deplasare ciclică. Într-un asemenea registru cuvîntul înscris inițial este recirculat în interiorul registrului.

În continuare vom exemplifica sinteza unui registru de patru biți cu încărcare paralelă și deplasare bidirecțională. Pentru o generalizare completă se va asigura, de asemenea, păstrarea cuvîntului încărcat în registru. Pentru a realiza păstrarea datelor, comutatoarele trebuie să aibă cîte o intrare suplimentară conectată la ieșirea bistabilului din același rang. Funcționarea unui asemenea registru este descrisă în tabelul 3.3.

Notînd prin J_i, K_i intrările de setare, respectiv de resetare ale bistabilelor cu ieșirile Q_i și folosind tabelul 3.1, se pot obține relațiile (3.6) care descriu funcționarea registrului în cele patru moduri de operare.

Tabelul 3.3

Tabelul de funcționare al registrului de patru biți

Regim de lucru	Intrări					Ieșiri				t
	s_1	s_0	D_i	DSD	DSS	Q_3	Q_2	Q_1	Q_0	
Păstrare	0	0	*	*	*	Q_3	Q_2	Q_1	Q_0	$t + 1$
Deplasare stînga	0	1	*	*	DSS	Q_2	Q_1	Q_0	DSS	
Deplasare dreapta	1	0	*	DSD	*	DSD	Q_3	Q_2	Q_1	
Încărcare paralelă	1	1	D_i	*	*	D_3	D_2	D_1	D_0	

$$\begin{aligned}
J_3 &= \bar{K}_3 = \bar{s}_1 \bar{s}_0 Q_3 \vee \bar{s}_1 s_0 Q_2 \vee s_1 \bar{s}_0 DSD \vee s_1 s_0 D_3, \\
J_2 &= \bar{K}_2 = \bar{s}_1 \bar{s}_0 Q_2 \vee \bar{s}_1 s_0 Q_1 \vee s_1 \bar{s}_0 Q_3 \vee s_1 s_0 D_2, \\
J_1 &= \bar{K}_1 = \bar{s}_1 \bar{s}_0 Q_1 \vee \bar{s}_1 s_0 Q_0 \vee s_1 \bar{s}_0 Q_2 \vee s_1 s_0 D_1, \\
J_0 &= \bar{K}_0 = \bar{s}_1 \bar{s}_0 Q_0 \vee \bar{s}_1 s_0 DSS \vee s_1 \bar{s}_0 Q_1 \vee s_1 s_0 D_0.
\end{aligned} \tag{3.6}$$

Schema registrului sintetizat este dată în figura 3.6.

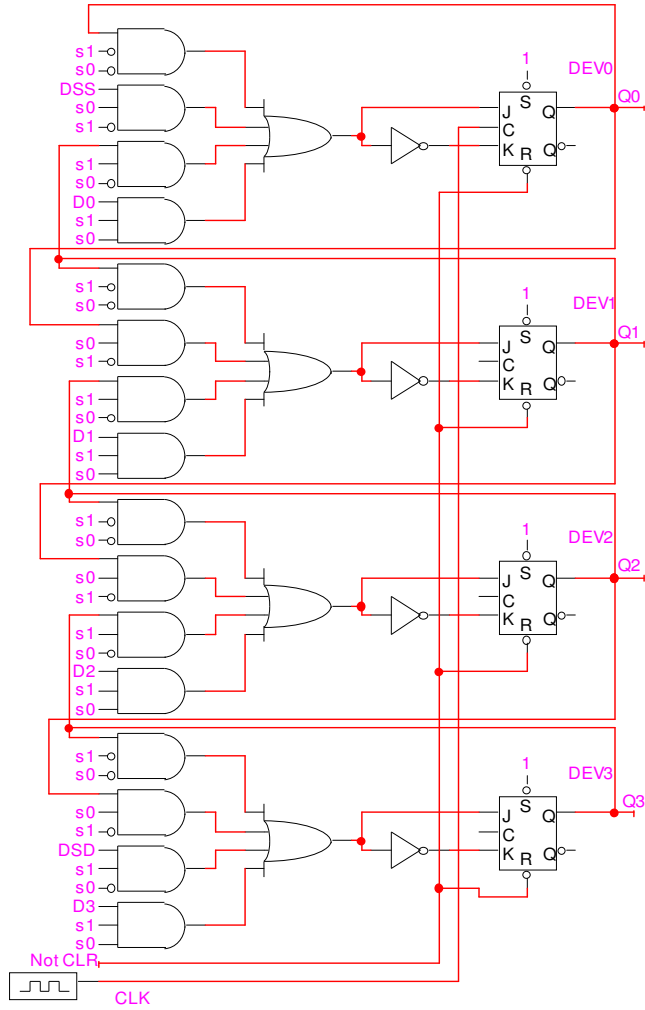


Fig. 3.6. Registru cu deplasare bidirecțională

3.4. Lucrarea de laborator nr. 6

Sinteza registrelor

Scopul lucrării: studierea registrelor și metodelor lor de proiectare.

Tema pentru acasă

Conform variantei indicate de profesor efectuați sinteza unui registru pe 4 biți, setul ȘI-NU, care realizează operațiile definite în tabelul 3.4.

Tabelul 3.4

Variante pentru îndeplinirea lucrării de laborator

Nr. var.	Bista-bilul	Modurile de operare	Nr. var.	Bista-bilul
1	2	3	4	5
1	JK	Încărcare paralelă Deplasare aritmetică stînga	13	D
2	JK	Încărcare paralelă Deplasare aritmetică dreapta	14	D
3	JK	Menținere Deplasare aritmetică stînga	15	D
4	JK	Menținere Deplasare aritmetică dreapta	16	D
5	JK	Încărcare paralelă Deplasare ciclică stînga	17	D
6	JK	Încărcare paralelă Deplasare ciclică stînga	18	D

Tabelul 3.4 (Continuare)

1	2	3	4	5
7	JK	Menținere Deplasare ciclică stînga	19	D
8	JK	Menținere Deplasare ciclică dreapta	20	D
9	JK	Încărcare paralelă Deplasare aritmetică stînga	21	D
10	JK	Încărcare paralelă Deplasare aritmetică dreapta	22	D
11	JK	Menținere Deplasare aritmetică stînga	23	D
12	JK	Menținere Deplasare aritmetică dreapta	24	D

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.
4. Se assemblează și se reglează schema registrului (conform variantei). Se verifică pe un exemplu concret corectitudinea funcționării lui.
5. Pentru circuitul asamblat se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele necesare. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de intrare-ieșire **Binary Probe**, **Hex Keyboard**, **Binary Switch** și **Clock**.
2. Se assemblează schema registrului (conform variantei) în **Fereastra de lucru** și se verifică corectitudinea ei. Se studiază diagrama de timp.

3. Pentru circuitul asamblat se determină costul și timpul de reținere.

Întrebări

1. Demonstrați că circuitul bistabil JK (fig. 3.4,a) se comportă ca un bistabil de tip T dacă $J=K=1$.

2. Realizați conversia bistabilelor $SR \rightarrow T$ și $D \rightarrow T$. Prezentați schemele posibile.

3. Avem un registru de patru biți realizat cu bistabile de tip D în care este înscris un cuvânt binar. Realizați conexiunile necesare astfel, încât să se asigure într-un singur tact deplasarea la dreapta cu două ranguri.

4. Propuneți schema unui registru cu bistabile de tip D care să asigure realizarea operației de înmulțire cu 2 a cuvântului binar înscris în registru.

5. Modificați schema registrului din figura 3.6, utilizând circuite multiplexoare pentru realizarea relațiilor (3.6).

3.5. Numărătoarele

O clasă importantă de circuite logice secvențiale o reprezintă numărătoarele (contoarele). Destinația numărătorului este înregistrarea numărului de impulsuri aplicate la intrarea lui și divizarea frecvenței acestor impulsuri. Componenta de bază a numărătoarelor sunt bistabilele. Un numărător are M stări distincte. Tranziția între stările succesive se produce în urma impulsurilor aplicate la intrarea numărătorului. După aplicarea unui număr de M impulsuri, numărătorul revine în starea inițială (de exemplu în starea zero). Un astfel de circuit reprezintă un numărător modulo M . Relația dintre modulo M și numărul de bistabile n , ce intră în componența numărătorului este următoarea: $n = \lceil \log_2 M \rceil$. Dacă numărul n nu este întreg, atunci el se rotunjește

pînă la cel mai apropiat număr întreg mai mare decît cel obținut.

Numărătoarele se clasifică după două criterii: în dependență de ordinea numărării și în dependență de faptul cum își schimbă starea bistabilele. Dacă numărătorul realizează o succesiune de stări codificate în ordine crescătoare, atunci el se numește numărător direct, iar dacă succesiunea stărilor este în ordine descrescătoare, atunci el se numește numărător invers. Numărătorul care realizează atât numărarea directă, cît și cea inversă se numește reversibil. Numărătoarele se divizează în sincrone sau asincrone, în dependență de faptul cum își schimbă starea bistabilele ce intră în componența lor. Dacă bistabilele își schimbă starea concomitent, atunci numărătoarele se numesc sincrone, iar dacă starea lor se schimbă succesiv numărătoarele se numesc asincrone.

Pentru un numărător modulo 8 vor fi necesare trei bistabile, succesiunea de numărare în cele două sensuri este prezentată în tabelul 3.5.

Tabelul 3.5

Tabelul de stări al numărătorului modulo 8

Nr. de impulsuri	Numărare directă			Numărare inversă		
	Rang			Rang		
	2	1	0	2	1	0
0	0	0	0	1	1	1
1	0	0	1	1	1	0
2	0	1	0	1	0	1
3	0	1	1	1	0	0
4	1	0	0	0	1	1
5	1	0	1	0	1	0
6	1	1	0	0	0	1
7	1	1	1	0	0	0
8	0	0	0	1	1	1

Analiza succesiunii cifrelor de rang 0 conduce la concluzia că succesiunea respectivă poate fi realizată în baza bistabilelor de tip T , care basculează la fiecare impuls de ceas. Cu alte cuvinte bistabilul T este un numărător modulo 2. Extinzând analiza tabelului 3.5 pentru rangurile 1, 2, ..., se poate observa că un numărător modulo $M=2^n$ se obține prin înserierea a n bistabile de tip T . Într-o asemenea înseriere fiecare din bistabilele de tip T când revine în starea inițială (zero) trebuie să transmită un semnal de comandă la intrarea de ceas a bistabilului aflat în rangul vecin mai semnificativ. Impulsurile de numărare trebuie aplicate la intrarea de ceas a bistabilului de rang 0.

Schema numărătorului asincron, direct modulo 8 obținut prin înserierea a trei bistabile JK, ajustate la bistabile T prin conectarea de unu logic la intrările J și K , este prezentată în figura 3.7.

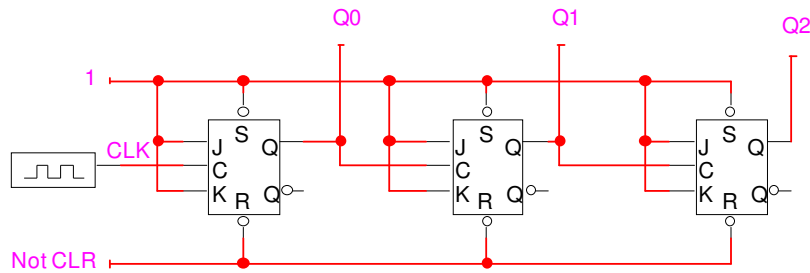


Fig. 3.7. Numărător asincron modulo 8

Pe frontul negativ al primului impuls de numărare bistabilul Q_0 se va comuta din 0 în 1, iar Q_1 și Q_2 vor rămâne în starea de zero logic. Pe frontul negativ al celui de-al doilea impuls de numărare Q_0 se va comuta din 1 în 0. Deoarece ieșirea Q_0 comandă intrarea de ceas a bistabilului Q_1 , atunci frontul negativ la ieșirea Q_0 va duce la comutarea lui Q_1 din 0 în 1, Q_2 rămânând în starea zero. Deci, se observă că pe fiecare front negativ al impulsului de numărare basculează Q_0 , Q_1 se inversează

pe frontul negativ al ieșirii lui Q_0 , iar Q_2 se inversează pe frontul negativ al lui Q_1 . Pentru a realiza numărarea în sens invers conectarea bistabilelor din figura 3.7 se face astfel, încât intrarea de ceas a fiecărui bistabil să fie comandată de ieșirea inversă a bistabilului din rangul vecin mai puțin semnificativ.

Numărătorul din figura 3.7 este asincron, deoarece, după cum s-a menționat mai sus, comutarea bistabilelor este succesivă în timp. Un numărător asincron are structura simplă dar și rapiditatea lui este relativ mică, fiind determinată de comutarea succesivă a bistabilelor.

La numărătoarele sincrone impulsul de numărare se aplică simultan la intrările de ceas ale tuturor bistabilelor și acestea se vor comuta sincron. Oricare bistabil al unui numărător sincron se comută simultan cu celelalte la aplicarea impulsului de numărare în starea, dependentă de valoarea semnalelor la intrările lui.

Se poate observa din analiza succesiunii directe (tabelul 3.5) că oricare bistabil trebuie să-și inverseze starea numai atunci, când toate bistabilele de rang inferior sunt în starea de unu logic. Pentru un numărător sincron modulo 16 condițiile de basculare se stabilesc conform relațiilor:

$$\begin{aligned} T_0 &= 1, \\ T_1 &= Q_0, \\ T_2 &= Q_0 Q_1, \\ T_3 &= Q_0 Q_1 Q_2. \end{aligned} \quad (3.7)$$

Schema numărătorului este prezentată în figura 3.8. În acest numărător este implementată propagarea paralelă a transportului.

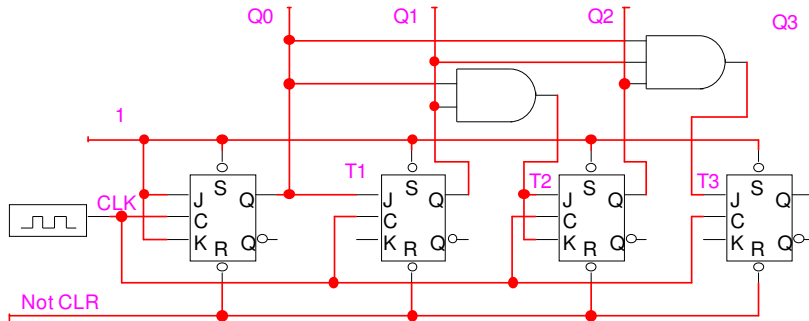


Fig. 3.8. Numărător sincron cu transport paralel

Condițiile (3.7) pot fi rescrise astfel:

$$\begin{aligned}
 T_0 &= 1, \\
 T_1 &= T_0 Q_0 = Q_0, \\
 T_2 &= T_1 Q_1, \\
 T_3 &= T_2 Q_2.
 \end{aligned}
 \tag{3.8}$$

Implementarea relațiilor (3.8) conduce la structura numărătorului sincron cu transport succesiv, care are avantajul utilizării de elemente logice ȘI numai cu două intrări, dar și dezavantajul conectării succesive a elementelor ȘI.

Dacă se dorește o funcționare reversibilă, numărătorul trebuie să conțină un semnal de comandă Up/Down. Acest semnal va determina sensul de numărare direct/invers prin selectarea fie a ieșirilor directe Q , fie a ieșirilor inverse \bar{Q} . O structură reversibilă se obține simplu dacă la ieșirea fiecărui bistabil este conectat un comutator logic care va selecta pentru numărarea directă ieșirea Q , iar pentru numărarea inversă ieșirea \bar{Q} , după cum este arătat în figura 3.9.

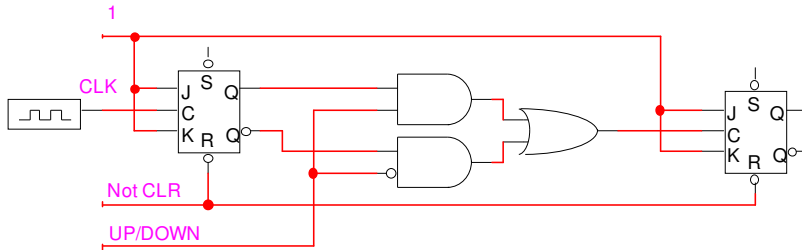


Fig. 3.9. Numărător asincron reversibil

Pînă acum au fost prezentate numărătoarele asincrone și sincrone modulo $M=2^n$. În continuare vom prezenta sinteza numărătoarelor modulo K , $2^{n-1} < K < 2^n$.

Structura unui numărător asincron modulo K se obține dintr-un numărător asincron modulo 2^n la care se adaugă un decodificator. În momentul cînd starea binară a numărătorului are valoarea $K_{(10)}$ decodificatorul trebuie să activeze semnalul asincron de ștergere CLR pentru a reseta bistabilele numărătorului. În figura 3.10 este prezentat un numărător asincron modulo 10.

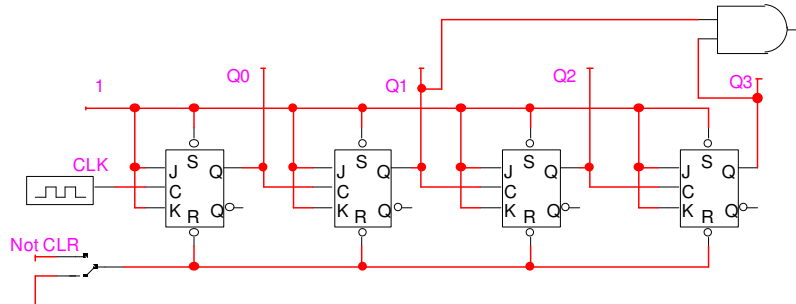


Fig. 3.10. Numărător asincron modulo 10

Un circuit numărător este în fond un automat de tip

Moore, la care vectorul de ieșire este determinat de vectorul de stare. De aceea sinteza unui numărator sincron modulo M se efectuează în câteva etape:

- se determină numărul de bistabile ale număratorului conform relației $n = \lceil \log_2 M \rceil$;
- se elaborează tabelul de tranziție al număratorului, în care se completează coloanele pentru starea prezentă a număratorului (momentul t), starea următoare (momentul $t+1$) și valorile ce trebuie aplicate la intrările tuturor bistabilelor pentru a asigura tranziția număratorului din starea de la momentul t la starea de la momentul $t+1$.
- din tabelul de tranziție al număratorului se obțin funcțiile de instalare a bistabilelor, care se minimizează;
- în baza funcțiilor minimizezate se elaborează circuitele de conexiune a bistabilelor între ele în urma implementării cărora se obține schema număratorului.

La sinteza unui numărator modulo M , din cele 2^n stări posibile un număr de $2^n - M$ sunt stări neutilizate (ilegale) și se consideră ca nedeterminate în diagrama Veitch-Karnaugh. Se poate întâmpla, însă, ca număratorul la pornire sau sub influența unor semnale parazitare să nimerească în una din aceste stări. Dacă după câteva tacturi număratorul poate ajunge într-o stare legală, atunci funcționarea lui de mai departe este corectă. Însă se poate întâmpla ca număratorul să nu poată ieși din stările ilegale, decât numai printr-o nouă pornire. Pentru a evita aceste cazuri, sinteza număratorului trebuie efectuată în așa fel, ca în tabelul de tranziție să se facă tranziția spre starea inițială din oricare din stările ilegale la următorul impuls de numărare.

Mai jos este prezentată sinteza unui numărator direct, sincron modulo 10 în baza bistabilelor JK . Numărul de bistabile necesar realizării acestui numărator este $n = \lceil \log_2 10 \rceil = 4$. În tabelul de tranziție 3.6 aceste bistabile sunt notate respectiv: Q_3, Q_2, Q_1, Q_0 .

Tabelul 3.6

Tabelul de tranziție al numărătorului sincron modulo 10

Ieșirile bistabilelor		Funcțiile de instalare a bistabilelor							
$(Q_3Q_2Q_1Q_0)_t$	$(Q_3Q_2Q_1Q_0)_{t+1}$	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0000	0001	0	*	0	*	0	*	1	*
0001	0010	0	*	0	*	1	*	*	1
0010	0011	0	*	0	*	*	0	1	*
0011	0100	0	*	1	*	*	1	*	1
0100	0101	0	*	*	0	0	*	1	*
0101	0110	0	*	*	0	1	*	*	1
0110	0111	0	*	*	0	*	0	1	*
0111	1000	1	*	*	1	*	1	*	1
1000	1001	*	0	0	*	0	*	1	*
1001	0000	*	1	0	*	0	*	*	1
1010	0000	*	1	0	*	*	1	0	*
1011	0000	*	1	0	*	*	1	*	1
1100	0000	*	1	*	1	0	*	0	*
1101	0000	*	1	*	1	0	*	*	1
1110	0000	*	1	*	1	*	1	0	*
1111	0000	*	1	*	1	*	1	*	1

Tabelul 3.7

Tabelul de tranziție al bistabilului JK

Q_t	Q_{t+1}	J	K
0	0	0	*
0	1	1	*
1	0	*	1
1	1	*	0

Tabelul de tranziție al numărătorului este elaborat

utilizând tabelul de tranziție al bistabilului JK modificat după cum este prezentat în tabelul 3.7. Funcțiile de instalare a bistabilelor sunt minimizate cu ajutorul diagramelor Veitch-Karnaugh după cum este arătat în figura 3.11, iar schema numărătorului este prezentată în figura 3.12.

Q_3Q_2				
Q_1Q_0	00	01	11	10
00			*	*
01			*	*
11		1	*	*
10			*	*

$$J_3 = Q_2 Q_1 Q_0$$

Q_3Q_2				
Q_1Q_0	00	01	11	10
00	*	*	1	
01	*	*	1	1
11	*	*	1	1
10	*	*	1	1

$$K_3 = Q_2 \vee Q_1 \vee Q_0 = \overline{\overline{Q_2 Q_1 Q_0}}$$

Q_3Q_2				
Q_1Q_0	00	01	11	10
00		*	*	
01		*	*	
11	1	*	*	
10		*	*	

$$J_2 = \overline{Q_3} Q_1 Q_0$$

Q_3Q_2				
Q_1Q_0	00	01	11	10
00	*		1	*
01	*		1	*
11	*	1	1	*
10	*		1	*

$$K_2 = Q_3 \vee Q_1 Q_0 = \overline{\overline{Q_3} \overline{Q_1} \overline{Q_0}}$$

Q_3Q_2				
Q_1Q_0	00	01	11	10
00				
01	1	1		
11	*	*	*	*
10	*	*	*	*

$$J_1 = \overline{Q_3} Q_0$$

Q_3Q_2				
Q_1Q_0	00	01	11	10
00	*	*	*	*
01	*	*	*	*
11	1	1	1	1
10			1	1

$$K_1 = Q_3 \vee Q_0 = \overline{\overline{Q_3} \overline{Q_0}}$$

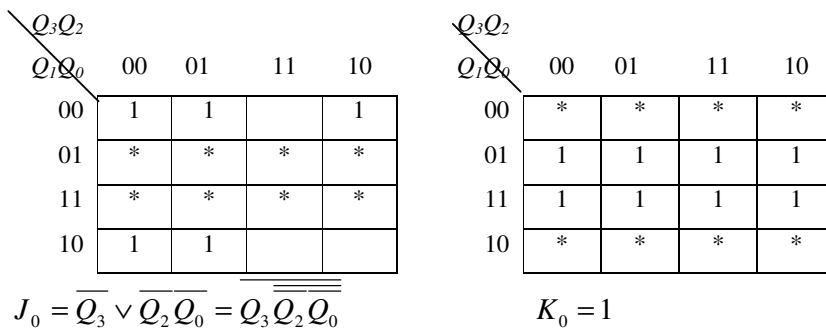


Fig. 3.11. Diagramele Veitch-Karnaugh pentru minimizarea funcțiilor de instalare a bistabilelor

6. Lucrarea de laborator nr.7

Sinteza numărătoarelor

Scopul lucrării: studierea diferitor tipuri de numărătoare și a metodelor lor de proiectare.

Tema pentru acasă

1. Executați sinteza numărătorului sincron incomplet în baza bistabilelor JK sau D , setul SI-NU, conform variantei indicate de profesor (tab. 3.8).
2. Executați sinteza numărătorului asincron incomplet în baza bistabilelor T , conform variantei indicate de profesor (tab. 3.9).
3. Executați sinteza numărătorului reversibil asincron modulo 32 în baza bistabilelor T .

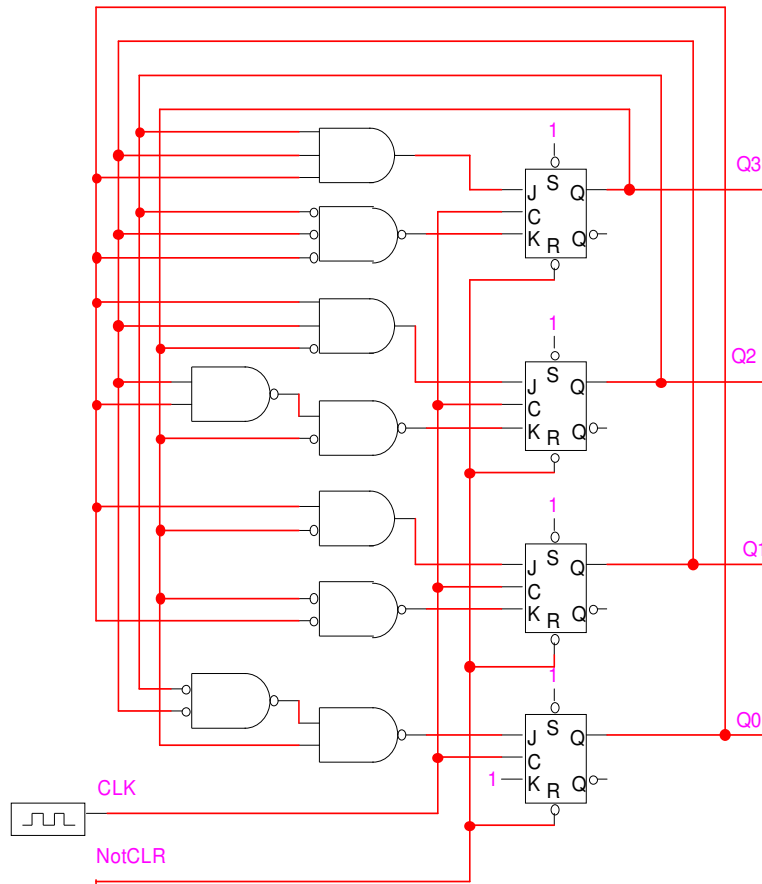


Fig. 3.12. Numărător sincron modulo 10

Tabelul 3.8

Variante pentru sinteza numărătorului sincron incomplet

Nr. var.	Bistabilul	Tipul numărătorului	Nr. var.	Bistabilul
1.	JK	Direct, modulo 9	13	D
2.	JK	Direct, modulo 11	14	D
3.	JK	Direct, modulo 12	15	D
4.	JK	Direct, modulo 13	16	D
5.	JK	Direct, modulo 14	17	D
6.	JK	Direct, modulo 15	18	D
7.	JK	Invers, modulo 10	19	D
8.	JK	Invers, modulo 11	20	D
9.	JK	Invers, modulo 12	21	D
10.	JK	Invers, modulo 13	22	D
11.	JK	Invers, modulo 14	23	D
12.	JK	Invers, modulo 15	24	D

Tabelul 3.9

Variante pentru sinteza numărătorului asincron incomplet

Nr. var.	Tipul numărătorului	Nr. var.	Tipul numărătorului
1	2	3	4
1.	Direct, modulo 17	15.	Invers, modulo 17
2.	Direct, modulo 18	16.	Invers, modulo 18
3.	Direct, modulo 19	17.	Invers, modulo 19
4.	Direct, modulo 20	18.	Invers, modulo 20
5.	Direct, modulo 21	19.	Invers, modulo 21

Tabelul 3.9 (Continuare)

1	2	3	4
6.	Direct, modulo 22	20.	Invers, modulo 22
7.	Direct, modulo 23	21.	Invers, modulo 23
8.	Direct, modulo 24	22.	Invers, modulo 24
9.	Direct, modulo 25	23.	Invers, modulo 25
10.	Direct, modulo 26	24.	Invers, modulo 26
11.	Direct, modulo 27	25.	Invers, modulo 27
12.	Direct, modulo 28	26.	Invers, modulo 28
13.	Direct, modulo 29	27.	Invers, modulo 29
14.	Direct, modulo 30	28.	Invers, modulo 30

Desfășurarea lucrării

a) la standul de laborator:

1. Se verifică corectitudinea funcționării circuitelor integrate ale standului de laborator.

2. Se asamblează și se reglează schema numărătorului sincron incomplet în baza bistabilelor JK sau D (p. 1 din tema pentru acasă).

3. Se asamblează și se reglează schema numărătorului asincron incomplet în baza bistabilelor T (p. 2 din tema pentru acasă).

4. Se asamblează și se reglează schema numărătorului reversibil asincron modulo 32 în baza bistabilelor T (p. 3 din tema pentru acasă).

5. Pentru circuitele asamblate se determină costul și timpul de reținere.

b) în LogicWorks:

1. Din biblioteca de elemente **Simulation Gates.clf** se selectează elementele necesare. Din biblioteca **Simulation IO.clf** se selectează dispozitivele de intrare-ieșire **Binary Probe**,

Hex Keyboard, Binary Switch și Clock.

2. Se assemblează schema numărătorului sincron incomplet în baza bistabilelor JK sau D în **Fereastra de lucru** și se verifică corectitudinea ei. Se studiază diagrama de timp.

3. Se assemblează schema numărătorului asincron incomplet în baza bistabilelor T în **Fereastra de lucru** și se verifică corectitudinea ei. Se studiază diagrama de timp.

4. Se assemblează schema numărătorului reversibil asincron modulo 32 în baza bistabilelor T în **Fereastra de lucru** și se verifică corectitudinea ei. Se studiază diagrama de timp.

5. Pentru circuitele asamblate se determină costul și timpul de reținere.

Întrebări:

1. Prezentați schemele numărătoarelor asincrone modulo 8 cu numărare directă și inversă în baza bistabilelor de tip T cu comutare pe front crescător.

2. Când apare timpul de propagare maxim într-un numărător asincron modulo 2^n cu numărare directă?

3. Analizați succesiunea inversă din tabelul 3.5 și scrieți relațiile de calculare a condițiilor de basculare pentru un numărător sincron modulo 2^n .

4. Prezentați schema numărătorului sincron modulo 16 cu transport succesiv. Comparați sub aspectul problemelor de implementare și al timpului de propagare maxim numărătoarele sincrone cu transport succesiv și paralel.

BIBLIOGRAFIE

1. Alexandru Valachi ș. a. Analiza, sinteza și testarea dispozitivelor numerice. – Iași, Editura Nord-Est: 1993.
2. Alin Dan Potorac. Bazele proiectării circuitelor numerice. – București, Matrix Rom: 2002.
3. Gheorghe Toacșe, Dan Nicula. Electronică digitală. - București, Editura Teora: 1996.
4. Gheorghe Dodescu. Informatica. - București, Editura Pedagogică: 1987.
5. I. Spânulescu, S. Spânulescu. Circuite integrate digitale și sisteme cu microprocesoare. – București, Editura Victor: 1996.
6. V. Gîscă, S. Zaporojan, V. Sudacevschi. Analiza și sinteza dispozitivelor numerice. Îndrumar de laborator.- Chișinău, 1999.
7. Alexandru Valachi, Mihai Bîrsan. Tehnici numerice și automate. - Iași, Junimea: 1986.
8. А. Л. Савельев. – Прикладная теория цифровых автоматов. Москва, Высшая школа: 1992.
9. К. Г. Самофалов и др. Прикладная теория цифровых автоматов. – Киев, Вища школа: 1987.

Anexă

**MINISTERUL EDUCAȚIEI ȘI ȘTIINȚEI AL
REPUBLICII MOLDOVA**

Universitatea Tehnică a Moldovei

Catedra Calculatoare

**REFERAT
LA LUCRAREA DE LABORATOR NR. 1**

Tema: Sinteza circuitelor logice combinaționale

Elaborat studentul gr. C-033

Dorinel Munteanu

Verificat

conf.univ.,dr. Vasile Gîscă

Chișinău
2005

CUPRINS

	Prefață.....	3
	Descrierea standului de laborator.....	4
1.	Sistemul de proiectare digitală Logic Works (versiunea 4).....	5
1.1.	Interfața.....	5
1.2.	Proiectarea de bază.....	6
1.3.	Bara de instrucțiuni pentru proiectare.....	7
1.4.	Bibliotecile.....	9
1.5.	Simularea și sincronizarea.....	11
2.	Circuitele logice combinaționale.....	15
2.1.	Particularitățile procesului de sinteză a circuitelor logice combinaționale.....	15
2.2.	Lucrarea de laborator nr.1. <i>Sinteza circuitelor logice combinaționale</i>	18
2.3.	Convertoarele de cod.....	22
2.4.	Lucrarea de laborator nr.2. <i>Sinteza convertoarelor de cod</i>	25
2.5.	Decodificatoarele și codificatoarele.....	28
2.6.	Lucrarea de laborator nr.3. <i>Sinteza decodificatoarelor și codificatoarelor</i>	35
2.7.	Sumatoarele binare.....	38
2.8.	Lucrarea de laborator nr.4. <i>Sinteza sumatoarelor binare</i>	43
2.9.	Comparatoarele.....	45
2.10.	Lucrarea de laborator nr.5. <i>Sinteza comparatoarelor</i> ...	51
3.	Circuitele logice secvențiale.....	54
3.1.	Particularitățile procesului de sinteză a circuitelor logice secvențiale.....	54
3.2.	Circuitele basculante bistabile.....	55

3.3.	Registrelle.....	60
3.4.	Lucrarea de laborator nr.6. <i>Sinteza registrelor</i>	64
3.5.	Numărătoarele.....	66
3.6.	Lucrarea de laborator nr.7. <i>Sinteza numărătoarelor</i>	75
	Bibliografie	80
	Anexă	81