Proiectare logică

Curs 1

Introducere. Sisteme de numerație. Aritmetica binară

Cristian Vancea

E-mail: Cristian.Vancea@cs.utcluj.ro
Web: http://users.utcluj.ro/~vcristian/PL.html

Obiectivele cursului

- Operarea cu fundamentele matematice care stau la baza proiectării dispozitivelor numerice.
- Analiza și sinteza sistemelor logice combinaționale.
- Analiza și sinteza sistemelor logice secvențiale sincrone și asincrone.
- Aplicarea principiilor de proiectare logică și a tehnicilor descriptive.
- Utilizarea circuitelor programabile pentru implementarea dispozitivelor numerice.
- Analizarea constrângerilor temporale în cadrul circuitelor programabile.

Conținutul cursului

- Introducere. Sisteme de numerație. Aritmetică binară.
- Reprezentarea numerelor în calculator. Coduri binare. Detectarea erorilor.
- Algebra booleană. Funcții booleene.
- Minimizarea funcțiilor booleene.
- Analiza circuitelor logice combinaționale (CLC). Circuite SSI și MSI.
- Proiectarea CLC cu circuite MSI, LSI. Hazardul combinațional.
- Circuite logice secvențiale (CLS). Circuite basculante bistabile (CBB).
- Numărătoare.
- Registre, convertoare, memorii RAM.
- Metode de proiectare a circuitelor secvențiale sincrone cu bistabile, multiplexoare, decodificatoare, memorii, numărătoare.
- Sisteme secvențiale sincrone.
- Proiectarea cu dispozitive logice programabile.
- Instrumente software de implementare cu dispozitive logice programabile.

Nivele de abstractizare în sistemele de calcul

Sus

Aplicații software	Programe
Sisteme de operare	Drivere
Arhitectură	Instrucțiuni, registre
Micro-arhitectură	Căi de date Unități de control
Logică	Unități aritmetico-logice Numărătoare, memorii
Circuite digitale	Porți logice AND, OR, NOT
Circuite analogice	Amplificatoare de semnal, filtre
Dispozitive electronice	Tranzistoare, diode Rezistențe, condensatoare
Fizică	Electroni

Jos

Zona de interes pentru cursul de **Proiectare logică**

Obiectivele lucrărilor de laborator

- Operarea cu conceptele învățate la curs.
- Dezvoltarea de abilități practice de:
  - implementare a circuitelor numerice cu aplicațiile din suita Xilinx ISE și Logisim.
  - testare/simulare a funcționării circuitelor numerice pe plăci didactice și în mediul de implementare.

Notă$_1$: Lucrările de laborator sunt componente obligatorii ale disciplinei Proiectarea logică.

Notă$_2$: Lucrările de laborator se studiază în prealabil !!!

Notă$_3$: Conținutul se găsește pe site sau pe Teams!

Conținutul Laboratorului

- Utilizarea mediului de dezvoltare Xilinx ISE.
- Circuite logice fundamentale.
- Utilizarea mediului de dezvoltare Logisim.
- Circuite logice combinaționale SSI/MSI/complexe.
- Bistabile, numărătoare, registre.
- Familia de circuite FPGA. Sinteza circuitelor numerice cu FPGA.

Bibliografie

[1] R. H. Katz, G. Borriello, “Contemporary Logic Design”, 2nd edition, Pearson, 2004.

[2] L. Văcariu, O. Creț, “Probleme de Proiectare Logică a sistemelor numerice / Logic Design Problems for digital systems”, Ediția a 2-a, U.T. Press, 2013.

[3] J. F. Wakerly, “Digital Design Principles and Practices”, 5th edition, Pearson, 2018.

[4] C. H. Roth, L. L. Kinney, “Fundamentals of Logic Design”, 7th edition, enhanced edition, Cengage Learning, 2020.

[5] W. Wolf, “FPGA-based System Design”, Prentice Hall, 2004.

[6] S. L. Harris, D. M. Harris, “Digital Design and Computer Architecture”, RISC-V edition, Morgan–Kaufmann, 2021.

[7] M. M. Morris, “Digital Design with an introduction to the Verilog, VHDL, HDL, and SystemVerilog”, 6th edition, global edition, Pearson, 2018.

Evaluare

- Examen (E)
  - onsite: scris
  - online: scris și/sau oral
- Test laborator (L)
  - onsite: scris
  - online: scris și/sau oral

Nota = round((E + L)/10), L<=30p, E<=70p

round() – rotunjire la cel mai apropiat întreg

Condiții de promovare:

- Laborator: L>=14p și 2 din 3 probleme peste 5p
- Examen: promovare laborator și E>=32p

Cuprins

- Sisteme de numerație
- Aritmetica binară

Sisteme de numerație

- Sistem de numerație: reguli de reprezentare a numerelor cu ajutorul cifrelor.
- Scop: codificarea informației pentru a putea fi prelucrată de sistemele de calcul.

Baza de numerație 2 – sistem de numerație binar

2 cifre (biți): 0 și 1

Poziția 3 2 1 0 . -1 -2 (numerotare crescătoare dreapta -> stânga; punctul între 0 și -1)

$N_2 = \mathbf{1\ 0\ 1\ 0\ .\ 0\ 1} \Rightarrow N_{10} = \mathbf{1}\text{x}2^{3} + \mathbf{0}\text{x}2^{2} + \mathbf{1}\text{x}2^{1} + \mathbf{0}\text{x}2^{0} + \mathbf{0}\text{x}2^{-1} + \mathbf{1}\text{x}2^{-2} =$

$= 8 + 2 + 0.25 = 10.25$

Baza 2 => sistem de numerație **pozițional**

Sisteme de numerație

Baza de numerație 8 – sistem de numerație octal

8 cifre: 0, 1, 2, 3, …, 7

Poziția 2 1 0 . -1 -2 (numerotare crescătoare dreapta -> stânga; punctul între 0 și -1)

$N_8 = \mathbf{3\ 1\ 7\ .\ 1\ \ 3} \Rightarrow N_{10} = \mathbf{3}\text{x}8^{\mathbf{2}} + \mathbf{1}\text{x}8^{\mathbf{1}} + \mathbf{7}\text{x}8^{\mathbf{0}} + \mathbf{1}\text{x}8^{\mathbf{-1}} + \mathbf{3}\text{x}8^{\mathbf{-2}} =$

$= \mathbf{3}\text{x}64 + \mathbf{1}\text{x}8 + \mathbf{7}\text{x}1 + \mathbf{1}\text{x}0.125 + \mathbf{3}\text{x}0.015625 =$

$= 207.171875$

Baza 8 => sistem de numerație **pozițional**

Sisteme de numerație

Baza de numerație 16 – sistem de numerație hexazecimal

16 cifre: 0, 1, 2, 3, ..., 7, 8, 9, A, B, C, D, E, F

(A=10, B=11, C=12, D=13, E=14, F=15)

Poziția 2 1 0 . -1 (numerotare crescătoare dreapta -> stânga; punctul între 0 și -1)

N_{16} = **F 4 C . 5** => N_{10} = **15**x16^{2} + **4**x16^{1} + **12**x16^{0} + **5**x16^{-1} =

= **15**x256 + **4**x16 + **12**x1 + **5**x0.0625 = 3916.3125

Baza 16 => sistem de numerație **pozițional**

Sistem de numerație **pozițional** => **contează poziția în număr**

Sistem de numerație **nepozițional** => **nu contează poziția în număr**

Sisteme de numerație

Conversia în baza 10 – sistem de numerație zecimal

Baza de pornire: **b**

$N_b = c_n c_{n-1} \ldots c_1 c_0 . c_{-1} c_{-2} \ldots c_{-m}$

c_i – cifra de pe poziția **i**

n+1 cifre – partea întreagă

m cifre – partea fracționară

$$N_{10} = c_n x b^n + c_{n-1} x b^{n-1} + \ldots + c_1 x b^1 + c_0 x b^0 + c_{-1} x b^{-1} + c_{-2} x b^{-2} + \ldots + c_{-m} x b^{-m}$$

Sisteme de numerație

Conversia din baza 10

- în baza 2

Partea întreagă

Ex_1: $N_{10}=57$

Regulă: Se împarte numărul la 2, apoi împărțiri repetate la 2 ale **câtului** până când **cât** = **0**.

57 : 2 = **28** rest **1**

28 : 2 = **14** rest **0**

14 : 2 = **7** rest **0**

7 : 2 = **3** rest **1**

3 : 2 = **1** rest **1**

1 : 2 = **0** rest **1**

Se scriu resturile de la stânga la dreapta în **ordinea inversă apariției** (de jos în sus) => N_2=**111001**

Sisteme de numerație

Conversia din baza 10

- în baza 2

Partea întreagă

Ex_2: $N_{10}=10$

10 : 2 = **5** rest **0**

5 : 2 = **2** rest **1**

2 : 2 = **1** rest **0**

1 : 2 = **0** rest **1**

N_2=**1010**

Sisteme de numerație

Conversia din baza 10

- în baza 2

Partea fracționară

Ex_3: $N_{10}=0.21$

Regulă: Stabilim precizia **m**. Înmulțiri repetate cu 2 ale **părții fracționare** de **m** ori. Se reține **partea întreagă**.

Precizia **m**=4 => 4 înmulțiri:

1: 0.21 x 2 = **0**.**42**

2: **0.42** x 2 = **0**.**84**

3: **0.84** x 2 = **1**.**68**

4: **0.68** x 2 = **1**.**76**

Se scriu părțile întregi de la stânga la dreapta în **ordinea apariției** (de sus în jos) => N_2=0.**0011**

Sisteme de numerație

Conversia din baza 10

- în baza 2

Partea fracționară

Ex_4: N_{10}=0.125

Precizia 6 biți (m=6):

1: 0.125 x 2 = **0.25**

2: **0.25** x 2 = **0.5**

3: **0.5** x 2 = **1.0** (dacă partea fracționară e **0** ne putem opri)

4: **0.0** x 2 = **0.0**

5: **0.0** x 2 = **0.0**

6: **0.0** x 2 = **0.0**

N_2=0.**001000…** = 0.**001**

Sisteme de numerație

Conversia din baza 10

- în baza 2

Ex$_5$: N_{10}=10.125 (cu 4 biți de precizie)

Partea întreagă: 10 => **1010**

Partea fracționară: 0.125 => 0.**0010**

N_2=**1010.0010**

Sisteme de numerație

Conversia din baza 10

- în baza 2

Ex_6: N_{10}=23.47 (cu 3 biți de precizie)

23 : 2 = **11** rest **1**

11 : 2 = **5** rest **1**

5 : 2 = **2** rest **1**

2 : 2 = **1** rest **0**

1 : 2 = **0** rest **1**

0.47 x 2 = **0.94**

0.**94** x 2 = **1.88**

0.**88** x 2 = **1.76**

Partea întreagă: 23 => **10111**

Partea fracționară: 0.47 => 0.**011**

N_2=**10111.011**

Sisteme de numerație

Conversia din baza 4

- în baza 2

Se codifică cifrele în binar pe **2** biți:

$$\begin{cases} 0_4 \Rightarrow \mathbf{00}_2 \\ 1_4 \Rightarrow \mathbf{01}_2 \\ 2_4 \Rightarrow \mathbf{10}_2 \\ 3_4 \Rightarrow \mathbf{11}_2 \end{cases}$$

1 0 . 3 2

$N_4 = \mathbf{10.32} \Rightarrow N_2 = \mathbf{01}\ \mathbf{00}\ .\ \mathbf{11}\ \mathbf{10} = 100.111$

(zerourile la capete se elimină)

Sisteme de numerație

Conversia din baza 8

- în baza 2

Se codifică cifrele în binar pe **3** biți:

0_8 => **000_2**
1_8 => **001_2**
2_8 => **010_2**
3_8 => **011_2**
4_8 => **100_2**
5_8 => **101_2**
6_8 => **110_2**
7_8 => **111_2**

2 3 . 4 7

N_8 = **23**.**47** => N_2 = **010 011** . **100 111** = 10011.100111

(zerourile la capete se elimină)

Sisteme de numerație

Conversia din baza 16

- în baza 2

Se codifică cifrele în binar pe **4** biți:

0_{16} => $\mathbf{0000_2}$	8_{16} => $\mathbf{1000_2}$
1_{16} => $\mathbf{0001_2}$	9_{16} => $\mathbf{1001_2}$
2_{16} => $\mathbf{0010_2}$	A_{16} => $\mathbf{1010_2}$
3_{16} => $\mathbf{0011_2}$	B_{16} => $\mathbf{1011_2}$
4_{16} => $\mathbf{0100_2}$	C_{16} => $\mathbf{1100_2}$
5_{16} => $\mathbf{0101_2}$	D_{16} => $\mathbf{1101_2}$
6_{16} => $\mathbf{0110_2}$	E_{16} => $\mathbf{1110_2}$
7_{16} => $\mathbf{0111_2}$	F_{16} => $\mathbf{1111_2}$

A **5** . **2** **4**

N_{16} = **A5.24** => N_2 = **1010 0101** . **0010 0100** = 10100101.001001

(zerourile la capete se elimină)

Sisteme de numerație

Conversia din baza 2

- în baza 4

Se ține cont de codificarea: $\begin{cases} 00_2 \Rightarrow \mathbf{0}_4 \\ 01_2 \Rightarrow \mathbf{1}_4 \\ 10_2 \Rightarrow \mathbf{2}_4 \\ 11_2 \Rightarrow \mathbf{3}_4 \end{cases}$

Se grupează biții câte **2** începând de la punctul fracționar. Se adaugă zerouri la capete dacă este necesar. Acestea nu afectează valoarea.

01 00 . 11 10

$N_2 = 100.111 \Rightarrow N_4 = \mathbf{1}\ \ \mathbf{0}\ .\ \mathbf{3}\ \ \mathbf{2}$

Sisteme de numerație

Conversia din baza 2

- în baza 8

Se ține cont de codificarea:

$$\begin{cases} 000_2 \Rightarrow \mathbf{0}_8 \\ 001_2 \Rightarrow \mathbf{1}_8 \\ 010_2 \Rightarrow \mathbf{2}_8 \\ 011_2 \Rightarrow \mathbf{3}_8 \\ 100_2 \Rightarrow \mathbf{4}_8 \\ 101_2 \Rightarrow \mathbf{5}_8 \\ 110_2 \Rightarrow \mathbf{6}_8 \\ 111_2 \Rightarrow \mathbf{7}_8 \end{cases}$$

Se grupează biții câte **3** începând de la punctul fracționar. Se adaugă zerouri la capete dacă este necesar. Acestea nu afectează valoarea.

010 011 . 100 111

N_2 = 10011.100111 => N_8 = **2** **3** . **4** **7**

Sisteme de numerație

Conversia din baza 2

- în baza 16

Se ține cont de codificarea:

$0000_2 \Rightarrow \mathbf{0}_{16}$	$1000_2 \Rightarrow \mathbf{8}_{16}$
$0001_2 \Rightarrow \mathbf{1}_{16}$	$1001_2 \Rightarrow \mathbf{9}_{16}$
$0010_2 \Rightarrow \mathbf{2}_{16}$	$1010_2 \Rightarrow \mathbf{A}_{16}$
$0011_2 \Rightarrow \mathbf{3}_{16}$	$1011_2 \Rightarrow \mathbf{B}_{16}$
$0100_2 \Rightarrow \mathbf{4}_{16}$	$1100_2 \Rightarrow \mathbf{C}_{16}$
$0101_2 \Rightarrow \mathbf{5}_{16}$	$1101_2 \Rightarrow \mathbf{D}_{16}$
$0110_2 \Rightarrow \mathbf{6}_{16}$	$1110_2 \Rightarrow \mathbf{E}_{16}$
$0111_2 \Rightarrow \mathbf{7}_{16}$	$1111_2 \Rightarrow \mathbf{F}_{16}$

Se grupează biții câte **4** începând de la punctul fracționar. Se adaugă zerouri la capete dacă este necesar. Acestea nu afectează valoarea.

1010 0101 . 0010 0100

$N_2 = 10100101.001001 \Rightarrow N_{16} = \mathbf{A} \quad \mathbf{5} \; . \; \mathbf{2} \quad \mathbf{4}$

Aritmetica binară

Adunarea fără semn

- Se adună bit cu bit de la dreapta la stânga:
  - La prima poziție **transportul** este 0.
  - Dacă la o poziție **i** rezultatul depășește 1 apare **transport** către poziția următoare **i+1**.

Ex$_1$:

Transport (T) 1 1 1

A 01110+ = 14_{10}+

B 00111 = 7_{10}

R 10101 = 21_{10}

Ex$_2$:

Transport (T) 1

A 00110+ = 6_{10}+

B 10101 = 21_{10}

R 11011 = 27_{10}

Reguli adunare pe biți

$A_i + B_i + T_i$	T_{i+1} R_i
0 + 0 + 0	0 0
0 + 0 + 1	0 1
0 + 1 + 0	0 1
0 + 1 + 1	1 0
1 + 0 + 0	0 1
1 + 0 + 1	1 0
1 + 1 + 0	1 0
1 + 1 + 1	1 1

Aritmetica binară

Adunarea fără semn

- Depășirea numărului de biți – **overflow**: poziția cea mai semnificativă generează transport la stânga.

Ex$_1$:

Transport (T) 1 1 1

A	1010+	=	10_{10}+
B	1111	=	15_{10}
R	1001	≠	25_{10}

Ex$_2$:

Transport (T) 1

A	110+	=	6_{10}+
B	101	=	5_{10}
R	011	≠	11_{10}

Reguli adunare pe biți

$A_i + B_i + T_i$	T_{i+1} R_i
0 + 0 + 0	0 0
0 + 0 + 1	0 1
0 + 1 + 0	0 1
0 + 1 + 1	1 0
1 + 0 + 0	0 1
1 + 0 + 1	1 0
1 + 1 + 0	1 0
1 + 1 + 1	1 1

Aritmetica binară

Scăderea fără semn

- Se scade bit cu bit de la dreapta la stânga:

  - La prima poziție **împrumutul** este 0.

  - Dacă la o poziție **i** rezultatul devine negativ apare **împrumut** către poziția următoare **i+1**.

Ex$_1$:

Împrumut (Î)		1 1 1		
	A	01110-	=	14_{10}-
	B	00111	=	7_{10}
	R	00111	=	7_{10}

Ex$_2$:

Împrumut (Î)		1		
	A	10110-	=	22_{10}-
	B	00101	=	5_{10}
	R	10001	=	17_{10}

Reguli scădere pe biți

A_i - B_i - $Î_i$	$Î_{i+1}$ R_i
0 - 0 - 0	0 0
0 - 0 - 1	1 1
0 - 1 - 0	1 1
0 - 1 - 1	1 0
1 - 0 - 0	0 1
1 - 0 - 1	0 0
1 - 1 - 0	0 0
1 - 1 - 1	1 1

$-1_{10} = 11_2$
$-2_{10} = 10_2$ în Complement față de 2 pe n=2 biți (Curs 2)

Aritmetica binară

Scăderea fără semn

- Depășirea numărului de biți – **underflow**: când A<B poziția cea mai semnificativă generează împrumut la stânga.

Ex$_1$:

Împrumut (Î) 1 1 1 1

A	1010-	=	10_{10}-
B	<u>1111</u>	=	<u>15_{10}</u>
R	1011	≠	-5_{10}

Ex$_2$:

Împrumut (Î) 1 1

A	101-	=	5_{10}-
B	<u>111</u>	=	<u>7_{10}</u>
R	110	≠	-2_{10}

Reguli scădere pe biți

A_i - B_i - $Î_i$	$Î_{i+1}$ R_i
0 - 0 - 0	0 0
0 - 0 - 1	1 1
0 - 1 - 0	1 1
0 - 1 - 1	1 0
1 - 0 - 0	0 1
1 - 0 - 1	0 0
1 - 1 - 0	0 0
1 - 1 - 1	1 1

Aritmetica binară

Înmulțirea fără semn

- Obs: $\forall N_2$ număr binar => $N_2 \times 0_2 = 0_2$ și $N_2 \times 1_2 = N_2$
- **Metoda**: se adună rezultatele înmulțirii primului termen cu fiecare bit din al doilea termen, deplasate la stânga.
  - Deplasarea se face cu un număr de biți identic cu poziția bitului cu care se face înmulțirea.
  - Poziția se numerotează de la dreapta la stânga începând cu zero.

Ex:

```
A      1010x    =    10(10)x
B      0101     =     5(10)
    1                50(10)
        1010
       0000
      1010
     0000
R    0110010    =    2^5+2^4+2^1=32+16+2=50(10)
```

$$A:\ 1010 \times \equiv 10_{10} \times$$
$$B:\ \underline{0101} \equiv \underline{\ 5_{10}}$$
$$50_{10}$$

$$R\ 0110010 \equiv 2^5+2^4+2^1=32+16+2=50_{10}$$