Curs 1: Sisteme de Numeratie. Coduri

Vlad-Cristian Miclea

Universitatea Technica din Cluj-Napoca

October 5, 2022

Cuprins

Cursul de ASDN

Informatii generale

- Vlad.Miclea@cs.utcluj.ro;
- *users.utcluj.ro/* $\sim$ *vmiclea* $\rightarrow$ Teaching
- Cursuri in H11 sau folosind platforma Teams
- Carte teorie + probleme
- Subiect destul de complex $\rightarrow$ Incremental
- Puneti intrebari oricand!!

Notare

- Examen final: 70 puncte
  - Examen scris sau probleme pe Moodle
  - Fara examen partial (Covid19)
- Test Lab: 30 puncte
  - Posibil: probleme/quiz-uri in timpul semestrului

Cursuri - program

1. Sisteme de numeratie si coduri
2. Aritmetica binara
3. Algebra booleana
4. Metode de minimizare a functiilor booleene
5. Circuite logice combinationale
6. Sinteza circuitelor digitale cu SSI, MSI, LSI and VLSI circuits
7. Circuite logice secventiale. Bistabile
8. Numaratoare
9. Registre. Memorii
10. Sinteza circuitelor digitale folosind bistabile
11. Sinteza circuitelor digitale folosind memorii, numaratoare etc.
12. Sisteme secventiale sincrone
13. Sinteza circuitelor digitale folosind circuite logice programabile
14. Probleme finale

Laboratorul de ASDN

Informatii generale

- Prezenta la laborator este obligatorie!!!
- Cerinta obligatorie: Cititi laboratorul si desenati circuitele!!

Simulator

- In general, la laborator veti folosi circuite reale
- Pentru anumite laburi, veti folosi simulatoare
- Implementarea se va face folosind Logisim (veti primi info de instalare)
- Mai mute detalii – la primul lab!

Laboratoare - program

1. Introducere
2. Circuite Logice Fundamentale
3. ActiveHDL – Editor and Simulator (I, II)
4. Minimizarea functiilor logice
5. Circuite Logice Combinationale CLC
6. Circuite Logice Combinationale MSI
7. Bistabile
8. Numaratoare I
9. Numaratoare II
10. Registre si Registre de deplasare
11. Implementarea circuitelor folosind FPGA (1)
12. Implementarea circuitelor folosind FPGA (2)
13. Recuperari
14. Test de laborator

Cuprins

Sistem de numeratie

Numerele in design-ul digital

- Datele sunt stocate/utilizate ca numere
- Informatia e **codificata** – e nevoie de o forma de reprezentare

Sistem de numeratie: Totalitatea regulilor de reprezentare a numerelor cu ajutorul unor simboluri numite cifre.

In functie de tipul reprezentarii

- SN Pozitionale – valoarea unei cifre e determinata de pozitia sa in cadrul numarului
- SN Nepozitionale – pozitia unei cifre are o alta semnificatie

Sistem de numeratie

Numarul **N**, in sistem pozitional, in baza **b** e reprezentat:

$$(N)_b = a_{q-1}b^{q-1} + ... + a_0 b^0 + ...a_{-p}b^{-p} = \sum_{i=-p}^{q-1} a_i b^i \quad (1)$$

Detalii

- Baza b e un nr intreg, in general $b > 1$
- Coeficientul (cifra) a_i e un intreg, $0 \leq a \leq b-1$
- Notatia $(N)_b$ inseamna "numarul N in baza b"
- Daca baza nu e specificata, in general e 10
- Complementul unei cifre a (notat $\bar{a}$ in baza b) e definit in eq. (2):

$$\bar{a} = (b-1) - a \quad (2)$$

Sistem de numeratie

Sisteme Binare

- Baza b e 2
- Cifrele posibile sunt 1 si 0 (”biti”)
- Complemente: $\bar{0} = 1$ si $\bar{1} = 0$

Alte sisteme utile

- Exista alte sisteme foarte intalnite
- Octal, Hexazecimal - de ce?

Sisteme de numeratie

Sistemul Octal

- Baza b e 8
- Cifrele posibile sunt 0..7

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Sisteme de numeratie

Sistemul Octal

- Baza b e 8
- Cifrele posibile sunt 0..7

Sistemul Hexazecimal

- Baza b e 16
- 16 cifre: $0 - 9$ si $A - F$
- Foarte util pentru reprezentarea folosind un singur caracter

Reprezentarea Byte

- **1 byte = 8 biti!!!**
- Care e intervalul de numere?
- Cel mai utilizat pentru reprezentare

Hexadecimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Conversia bazei de numeratie

- De multe ori, se doreste trecerea dintr-o baza b_1 in baza b_2
- In sisteme pozitionale: se foloseste un set de inmultiri si impartiri
- Exista 2 cazuri principale:
  - $b_1 < b_2$
  - $b_1 > b_2$
- https://www.rapidtables.com/convert/number/base-converter.html

Conversia bazei de numeratie

Cazul 1: $b_1 < b_2$

- $(N)_{b1}$ e exprimat ca un polinom cu coeficienti in baza b_1
- Rezultatul (alaturi de operatiile asociate: adunare si inmultire) e evaluat in baza b_2
- Exemplu: $b_1 = 3$, $b_2 = 10$ si $(N)_3 = 2120.1$

$$\begin{aligned}(N)_{10} &= 2 \times 3^3 + 1 \times 3^2 + 2 \times 3^1 + 0 \times 3^0 + 1 \times 3^{-1} \\ &= 54 + 9 + 6 + 0 + 0.3 = 69.3\end{aligned} \tag{3}$$

Conversia bazei de numeratie

Cazul 2: $b_1 > b_2$

- Operatiile aritmetice sunt realizate in baza b_1
- Exista 2 algoritmi diferiti, pentru partea Intreaga si partea Fractionara

Case 2.1: Partea Intreaga

- Numarul se imparte la baza b_2
- Resulta un Cat si un Rest
- Restul e stocat
- Catul e impartit mai departe la baza b_2
- Algoritmul continua pana cand **Catul e** 0
- Partea Intreaga a numarului convertit e obtinuta prin citirea Resturilor in ordine **inversa** (incepand de la ultima impartire)

Conversia bazei de numeratie

Cazul 2: $b_1 > b_2$

- Operatiile aritmetice sunt realizate in baza b_1
- Exista 2 algoritmi diferiti, pentru partea Intreaga si partea Fractionara

Case 2.2: Fractional part

- Numarul se inmulteste cu baza b_2
- Rezulta un nou numar, cu parte Intreaga si parte Fractionara
- Partea Intreaga se stocheaza
- Noua parte Fractionara e mai departe inmultita cu baza b_2
- Algoritmul continua pana cand **Precizia dorita e obtinuta**
- Partea Fractionara a numarului convertit e obtinuta prin citirea partilor Intregi rezultate, in ordine **directa** (incepand cu prima inmultire)

Conversia bazei de numeratie

Exemplu: $b_1 = 10$, $b_2 = 4$ si $(N)_{10} = 347.4$

Partea Intreaga

$$347 \div 4 = 86 \; rest \; 3$$
$$86 \div 4 = 21 \; rest \; 2$$
$$21 \div 4 = 5 \; rest \; 1$$
$$5 \div 4 = 1 \; rest \; 1$$
$$1 \div 4 = 0 \; rest \; 1$$

Rezultat (intreg): $(11123)_4$

Partea Fractionara

$$0.4 \times 4 = 1.6 \rightarrow 1$$
$$0.6 \times 4 = 2.4 \rightarrow 2$$
$$0.4 \times 4 = 1.6 \rightarrow 1$$
$$0.6 \times 4 = 2.4 \rightarrow 2$$
$$...$$

Rezultat (fractionar): $(1212..)_4$

Numarul Rezultat: $(11123.1212...)_4$

Conversia bazei de numeratie

Cazuri speciale de conversie

- Conversia din octal/hexazecimal in binar (si invers)
- Doar grupam/separam biti (cate 3 sau 4)
- Octal/hexa in binar (atentie la 0-uri pt formatare)

$$(123.4)_8 = (001\,010\,011.100)_2 = (1010011.1)_2$$
$$(2C5F)_{16} = (0010\,1100\,0101\,1111)_2 = (1011000101111)_2$$

- Binar in octal/hexa

$$(1010110.0101)_2 = (001\,010\,110.\,010\,100)_2 = (126.24)_8$$
$$(1011100110101010.1)_2 = (0010\,1110\,0110\,1010.\,1000)_2 = (2E6A.8)_{16}$$

Cuprins

Coduri Binare

Generalitati

- In viata reala, folosim coduri zecimale (precum si in interfata om-masina)
- Numerele zecimale sunt si ele reprezentate in binar in sistemele digitale
- Fiecare cifra zecimala (0-9) e reprezentata pe ? biti?
- Codurile binare:
  - Ponderate
  - Neponderate

Coduri binare ponderate

- Fiecare cifra binara are o pondere
- Numarul e codificat prin insumarea ponderilor cifrelor 1

$$N = \sum_{i=0}^{K-1} a_i b_i \tag{4}$$

- unde K este numarul de cifre si $a_i \in \{0, 1\}$
- caz particular al eq. (1) (slide 9)

Coduri binare ponderate

BCD

- Codificarea normala binara
- Ponderile corespund puterilor lui 2: $1, 2, 4, 8, \ldots$

Decimal	b_3	b_2	b_1	b_0
D	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Coduri binare ponderate

BCD

- Codificarea normala binara
- Ponderile corespund puterilor lui 2: $1, 2, 4, 8, \ldots$

Coduri auto-complementare

- Conditie: suma ponderilor trebuie sa fie 9
- Complementul unui nr N e $9 - N$
- Coduri auto-complementare pozitive (2421)

Decimal	b_3	b_2	b_1	b_0
D	2	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	0	1
8	1	1	1	0
9	1	1	1	1

Coduri binare ponderate

BCD

- Codificarea normala binara
- Ponderile corespund puterilor lui 2: $1, 2, 4, 8, ...$

Coduri auto-complementare

- Conditie: suma ponderilor trebuie sa fie 9
- Complementul unui nr N e $9 - N$
- Coduri auto-complementare pozitive (2421)
- Coduri auto-complementare negative

Decimal	b_3	b_2	b_1	b_0
D	6	4	2	-3
0	0	0	0	0
1	0	1	0	1
2	0	0	1	0
3	1	0	0	1
4	0	1	0	0
5	1	0	1	1
6	0	1	1	0
7	1	1	0	1
8	1	0	1	0
9	1	1	1	1

Coduri binare neponderate

Aceste tipuri de coduri urmaresc alte reguli.

Excess3

- Obtinut prin adunarea 0011 (reprezentarea binara a cifrei 3) la fiecare numar in reprezentare BCD
- Rezulta un cod auto-complementar
- Nu contine combinatia 0000

Decimal	b_3	b_2	b_1	b_0
0	0	0	1	1
1	0	1	0	0
2	0	1	0	1
3	0	1	1	0
4	0	1	1	1
5	1	0	0	0
6	1	0	0	1
7	1	0	1	0
8	1	0	1	1
9	1	1	0	0

Coduri binare neponderate

Gray

- Codificare ciclica: cifrele succesive difera printr-o singura cifra binara
- Codificare reflectiva: codificarea de n-biti va fi formata prin reflectarea codului pe $n-1$-biti si adaugarea unui bit suplimentar pe prima pozitie
- Ex: codificare pe 2-biti obtinuta prin reflectarea a 2 coduri de 1-bit

0	0
0	1
1	1
1	0

Decimal	b_3	b_2	b_1	b_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1

Detectia si corectia codurilor

Generalitati

- Codificarea e f importanta la transmiterea informatiei
- In sistemele numerice, informatia se poate altera in procesul de transmitere
- Trebuie sa asiguram corectitudinea informatiei transmise
  - Detectam daca informatia a fost modificata
  - Corectam codul transmis

Coduri detectoare de erori – CDE

Aparitia unei singure erori transforma un cuvant de cod valid in cuvant de cod invalid.

Metoda bitului de paritate

- Adaugam un singur bit supllimentar la fiecare cuvant
- Acest bit ne va "spune" daca numarul de biti de 1 in cuvantul transmis a fost par sau impar
- Exemplu: trimitem cuvantul (bitstring) 1011
  - Pentru paritate impara: vom adauga un 1, noul cuvant fiind 1 1011
  - Pentru paritate para: vom adauga un 0, noul cuvant fiind 0 1011

Coduri detectoare de erori

Codul "2 din 5"

- Foloseste ponderile 1, 2, 4, 7
- Exceptie: codificarea pentru cifra 0
- Ponderea asociata bit-ului 0 – va spune daca numarul de valori de 1 e par sau impar

Decimal	0	1	2	4	7
0	0	0	0	1	1
1	1	1	0	0	0
2	1	0	1	0	0
3	0	1	1	0	0
4	1	0	0	1	0
5	0	1	0	1	0
6	0	0	1	1	0
7	1	0	0	0	1
8	0	1	0	0	1
9	0	0	1	0	1

Coduri corectoare de erori – CCE

Un cod e corector de erori (CCE) daca intotdeauna cuvantul de cod corect poate fi dedus din cuvantul eronat.

Coduri Hamming

- Coduri corectoare de erori singulare: permit corectarea unei singure erori!
- Diferenta minima intre 2 cuvinte de cod (numarul de biti diferiti) trebuie sa fie 3
- Relatia lui Hamming: Numarul de "biti de control" (biti suplimentari, necesari pt corectie) e dat de:

$$2^k \geq m + k + 1$$

- m e numarul de biti de informatie utila
- k e numarul de biti de control

Coduri corectoare de erori

Coduri Hamming - exemplu

- Codul Hamming pentru un mesaj de lungime $m = 4$, in codificare BCD
- Daca aplicam relatia lui Hamming $\rightarrow k = 3$
- Bitii de control vor fi inserati pe pozitiile cu valoare de putere a lui 2

1	2	3	4	5	6	7
c_1	c_2	b_1	c_3	b_2	b_3	b_4

- Bitii de control sunt generati folosind relatiile:

$$c_1 = b_1 \oplus b_2 \oplus b_4$$
$$c_2 = b_1 \oplus b_3 \oplus b_4$$
$$c_3 = b_2 \oplus b_3 \oplus b_4$$

- unde $\oplus$ e operatorul xor – ce calculeaza acest operator?

Coduri corectoare de erori

Codificare Hamming - exemplu

- $b_1 b_2 b_3 b_4 = 0100$ – (trimitem 4, in BCD)
- Calculam bitii de control: $c_1 = 1$; $c_2 = 0$; $c_3 = 1$.
- Mesajul trimis va fi: 1001100

Concluzii

Rezumat

- Sisteme de Numeratie
  - Binar, Octal, Hexazecimal
- Conversia bazei de numeratie
  - 2 cazuri, in functie de relatia dintre baze
- Coduri binare
  - Ponderate: BCD, Auto-complementare
  - Neponderate: Excess3, Gray
- Coduri detectoare de erori
- Coduri corectoare de erori

Saptamana viitoare

- Reprezentarea numerelor
- Aritmetica binara

Multumesc pentru atentie!