

CIRCUITE INTEGRATE PE SCARĂ FOARTE LARGĂ

Decebal POPESCU

Vlad CIOBANU

Iacob PETRESCU

Adrian PETRESCU

2014

PREFATĂ

Integrarea pe Scară Foarte Mare a circuitelor electronice reprezintă una dintre tehnologiile de vârf ale industriei moderne. Cunoscută în engleză sub prescurtarea VLSI (Very Large Scale Integration) această tehnologie asigură componentele de bază și structurile funcționale necesare realizării unei game extrem de largi de produse și sisteme, pentru cele mai diverse aplicații, începând cu cele de uz casnic și terminând cu cele pentru industria aerospatială. Principalele avantaje ale produselor realizate în tehnologia VLSI se referă la implementarea unor sisteme cu o mare complexitate funcțională în capsule de mici dimensiuni, în condițiile unui consum mic de putere și a unei fiabilități extrem de ridicate. Materialul prezentat în acestă carte se bazează pe rezultatele grupului de cercetare din care fac parte: conf. dr. ing. Decebal Popescu, dr. ing. Iacob Petrescu și ing. drd. Vlad Ciobanu, pe teza de doctorat al acestuia din urmă, cât și pe suportul de curs pentru disciplina VLSI, elaborat de prof. dr. ing. Adrian Petrescu (www.csitsun.pub.ro). Fără utilizarea tehnologiei VLSI nu ar fi de conceput echipamentele întâlnite în bunurile de larg consum, între care se pot menționa:

- mașinile de spălat cu comandă programată, cuptoarele cu microunde, echipamentele audio de mare fidelitate, televizoarele, aparatele de fotografiat, ceasurile electronice;
- calculatoarele personale, calculatoarele personale ultramobile, tabletele, iPod-urile, jucăriile electronice, telefoanele mobile, sisteme de securitate;
- echipamentele medicale pentru măsurarea tensiunii arteriale, echipamente portabile pentru măsurarea și înregistrarea tensiunii, pulsului, electrocardiogramelor, echipamentele pentru asigurarea unei bune condiții fizice;

Autorii

CUPRINS

Prefață	5
1. Introducere	12
1.1. Realizarea elementelor active de tip BJT	19
1.2. Procesul de fabricare pentru MOSFET-uri	25
1.3. Fabricarea circuitelor VLSI - CMOS.	27
1.3.1. Introducere	27
1.3.2. Tehnologia de procesare a materialului de bază	28
1.4. Procesul CMOS simplificat.....	33
1.4.1. Un proces NMOS simplu	33
1.4.2. Un proces CMOS cu insula N	34
2. Tranzistoare MOS și CMOS	40
2.1. Principii generale	40
2.2. Caracteristicile tranzistoarelor MOS.....	42
2.3. Proprietăți specifice ale tranzistoarelor MOS	47
2.3.1. Efectul de substrat.....	47
2.3.2. Modulația în lungime a canalului.....	48
2.4. Modelele la semnal mic ale tranzistoarelor MOS	49
2.5. Porți logice C-MOS elementare.....	51
2.5.1. Poarta logică NU	51
2.6. Poarta logică ȘI-NU	57
2.7. Poarta logică SAU-NU.....	57
2.8. Porți logice MOS.....	58
2.9. Porți logice MOS complexe	61

CMOS VLSI

2.10. Regimul tranzitoriu al portilor logice.....	63
2.11. Circuite de memorie capacitive.....	65
3. Proiectarea portilor logice În VLSI.....	68
3.1. Proiectarea măștilor / şabloanelor	68
3.1.1. Proiectarea unui inversor.....	69
3.1.2. Reguli de proiectare	73
3.2. NAND2	77
3.3. NAND3	79
3.4. AND2	81
3.5. AND3	82
3.6. NOR2	84
3.7. NOR3	86
3.8. OR2	87
3.9. OR3	88
3.1. XOR2	90
3.2. XOR3	92
4. Circuite combinationale si cu memorie VLSI.....	94
4.1. Multiplexor 2:1.....	94
4.2. Memorie ROM	95
4.3. Bistabil de tip D	97
4.4. Registr	98
4.5. Reţele logice programabile	99
5. Circuite aritmetice VLSI.....	102
5.1. Sumator	102
5.2. Semisumator.....	103
5.3. Comparator.....	105
6. Proiectarea unui microprocesor Multichip.....	107
6.1. Proiectarea unității de prelucrare a datelor.....	109

0. Prefață

6.2. Proiectarea unității aritmetico-logice	110
6.2.1. Circuitul de comandă pentru liniile K_i , P_i și R_i din cadrul UAL.....	113
6.2.2. Registrele UAL	114
6.2.3. Registrul de ieșire OUT A, OUT B.....	115
6.2.4. Magistralele (BUS A, BUS B).....	115
6.2.5. Tristate (TS)	116
6.2.6. Circuitul de deplasare circulară.....	117
6.2.7. Unitatea de execuție	119
6.2.8. Conexiunea magistralei MAG A cu portul literal	120
6.2.9. Decodificatorul.....	121
6.2.10. Tabloul de registre generale	122
6.2.11. Comunicația cu mediul înconjurător. Circuitele de comandă pentru ploturile de I/E.....	122
6.2.12. Circuitul de intrare	124
6.2.13. Estimarea perioadei minime a ceasului plecând de la valorile parametrilor electrici pentru diverse straturi.....	125
6.2.14. Microprogramarea unității de execuție	126
6.2.15. Codificarea operațiilor în Unitatea de Execuție	126
6.2.16. Microprogramarea unității de execuție	127
6.2.17. Câmpul de condiționare	129
6.2.18. Câmpul de memorare	130
7. Implementarea în VLSI a sistemelor de conducere	132
7.1. Controlere digitale.....	132
7.1.1. Implementarea recursivă directă	134
7.1.2. Implementarea recursivă în cascadă.....	135
7.1.3. Implementarea recursivă în paralel	137
7.1.4. Implementarea nerecursivă	138
7.2. Implementarea controlerelor	139

CMOS VLSI

7.2.1. Controler recursiv direct	139
7.2.2. Controler nerecursiv.....	142
7.3. Conducerea unui robot mobil.....	143
7.4. Conducerea unui robot industrial (controler PID)	146
7.5. Conducerea unui robot prin metoda variabilelor de stare ...	152
7.6. Conducerea unui braț de robot prin metoda funcțiilor de transfer.....	155
8. Implementarea în VLSI a sistemelor cu predicție.....	157
8.1. Modele dinamice predictoare implementate VLSI	157
8.2. Conducerea cu model predictiv.....	159
8.3. Conducerea cu model predictiv a unui robot industrial	164
8.4. Estimarea parametrilor unui robot prin tehnici cu model predictor	168
9. Implementarea în VLSI a rețelelor neuronale.....	174
9.1. Modelul unei rețele neuronale.....	174
9.2. Implementarea unei rețele neuronale	180
9.2.1. Coeficientii de ponderare	180
9.2.2. Implementarea prin amplificatoare Gilbert.....	182
9.3. Conducerea unui robot prin rețele neuronale folosind modelul invers	185
9.4. Conducerea unui robot prin rețele neuronale folosind tehnica decuplării neliniare.....	187
10. Implementarea în VLSI a sistemelor fuzzy.....	189
10.1. Prinzipiul logicii fuzzy	189
10.2. Implementarea VLSI a componentelor fuzzy	198
10.2.1. Circuite de fuzificare.....	199
10.2.2. Circuit pentru realizarea funcției MAX	201
10.2.3. Circuit pentru realizarea funcției MIN	201
10.2.4. Circuit pentru realizarea funcției de defuzificare.....	202

0. Prefață

10.3. Conducerea dinamică a unui robot mobil în câmp cu potențial artificial	202
11. Bibliografie	205

1. INTRODUCERE

Începând cu inventarea tranzistorului, în anul 1947, tehnologia dispozitivelor semiconductoare a evoluat continuu. Din punctul de vedere al complexității, circuitele integrate s-au dezvoltat exponențial. Spre exemplu, primul microprocesor, pe 4 biți, apărut în anul 1971, avea circa 1700 de tranzistoare, iar în anul 1990 microprocesoarele pe 32 de biți aveau deja peste 150.000 de tranzistoare. Procesoarele moderne utilizează peste zeci și sute de milioane de tranzistoare (Tabelul Tabel 1.1). Toate aceste exemple demonstrează viabilitatea legii lui Moore, care afirmă faptul că numărul de tranzistoare plasate pe o singura pastilă se dublează la circa 18 luni.

Nume	Dată	Tranzistoare	Microni	Viteza ceasului	Lățimea datelor	MIPS
8080	1974	6000	6	2 MHz	8 biți	0.64
8088	1979	29000	3	5 MHz	16 biți, 8 biți mag.	0.33
80286	1982	134000	1.5	6 MHz	16 biți	1
80386	1985	275000	1.5	16 MHz	32 biți	5
80486	1989	1200000	1	25 MHz	32 biți	20
Pentium	1993	3100000	0.8	60 MHz	32 biți, 64 biți mag.	100
Pentium II	1997	7500000	0.35	233 MHz	32 biți, 64 biți mag.	Aprox. 300
Pentium III	1999	9500000	0.25	450 MHz	32 biți, 64 biți mag.	Aprox. 510
Pentium 4	2000	42000000	0.18	1.5 GHz	32 biți, 64 biți mag.	Aprox. 1700
Pentium 4 Prescott	2004	125000000	0.09	3.6 GHz	32 biți, 64 biți mag.	Aprox. 7000

Tabel 1.1 Evoluția în timp a numărului de tranzistoare pe pastilă pentru câteva procesoare Intel

Perfecționarea proceselor tehnologice în domeniul circuitelor integrate a permis, de asemenea, reducerea dimensiunilor dispozitivelor, ceea ce se poate exemplifica prin reducerea lungimii canalului tranzistorului

1. Introducere

elementar de la 5 μm , în 1985, la 0.35 μm , în 1997 și la 0.70 nm în 2005. În același timp au crescut dimensiunile discurilor din siliciul monocristalin, care reprezintă suportul pe care se realizează structurile larg integrate. Evoluția în timp a unor elemente definitorii pentru circuitele integrate se poate urmări în Tabelul 1.1.

Întârzierea în propagarea semnalelor s-a redus cu trei ordine de mărime în ultimii 20 de ani, ceea ce se reflectă în creșterea frecvenței ceasului microprocesoarelor de la circa 1MHz în 1975 la peste 1 GHz în anul 2000. În același timp s-au redus în mod continuu costurile de fabricație. Astfel, în cazul memoriilor RAM, costul pe bit s-a micșorat de la circa 1 cent, în 1970, la 10^{-4} - 10^{-5} cenți, în prezent. Actualmente, pentru circuitele integrate folosite în calculatoarele electronice, se folosesc numeroase tehnologii, care se pot grupa în tehnologii bipolare și tehnologii MOS.

Tehnologii bipolare:

- TTL (Transistor Transistor Logic)
 - o TTL-S (Schottky TTL);
 - o TTL-LS (Low-Power Schottky TTL);
 - o TTL-AS (Advanced Schottky TTL);
 - o TTL-ALS (Advanced Low-power Schottky TTL);
 - o FAST (Fairchild Advanced Schottky TTL);
- ECL (Emitter Coupled Logic);
- I2L (Integrated Injection Logic).

Tehnologii MOS:

- PMOS (MOS canal P);
- NMOS (MOS canal N):
 - o HMOS (High performance MOS).
- CMOS (Complementary MOS):
 - o HCMOS (High density CMOS),
 - o ACL (Advanced CMOS Logic).
- MNOS (Metal Nitride Oxide Semiconductor):
 - o FAMOS (Floating gate Avalanche injection MOS),
 - o FLTOX (FLOating gate Tunnel Oxide).

CMOS VLSI

Circuitele integrate care se folosesc în construcția calculatoarelor se plasează în categoriile: standard, specifice aplicațiilor (ASIC - Application Specific Integrated Circuits) și programabile/configurabile.

La rândul lor circuitele ASIC se împart în:

- Circuite personalizate la cerere (Semi-Custom)
 - o Circuite predifuzate (Gate Arrays).
- Circuite realizate la comandă (Custom):
 - o Circuite precaracterizate (Standard Cells),
 - o Circuite realizate complet la cerere (Full Custom).

Se amintește că tranzistorul a fost inventat în anul 1947 și că primele exemplare ocupau o suprafață de 3.5 mm^2 . La sfârșitul anilor 50 a apărut circuitul integrat care, grupând pe aceeași pastilă mai multe tranzistoare, a avut o evoluție spectaculoasă în sensul dublării numărului de componente pe pastilă, la fiecare 18 luni. Aceasta s-a datorat în primul rând numeroaselor perfecționări ale proceselor tehnologice, care au permis rezoluții de ordinul a $2.5 \mu\text{m}$ – $0.09 \mu\text{m}$. În continuare se vor da unele date privind tehnologiile circuitelor VLSI, în general, evoluția memoriilor și a procesoarelor.

Evoluția tehniciilor de fabricație a circuitelor integrate este unică în istoria industriei moderne. Tendințele privind creșterea vitezei, mărirea densității, cât și reducerea costului circuitelor integrate s-au menținut în mod constant, pe parcursul ultimilor 30 de ani. În continuare sunt ilustrate tendințele de scalare a tehnologiei.

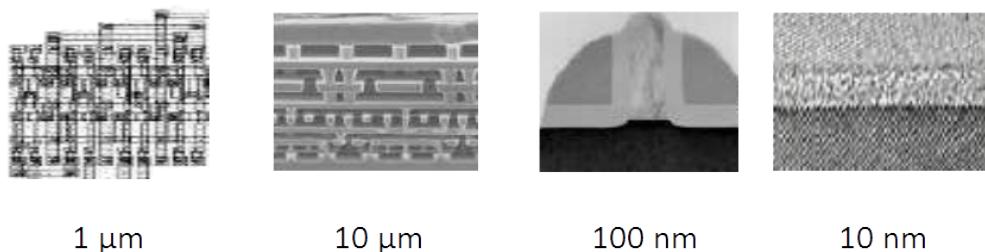


Figura 1.1 Structuri reprezentative pentru un circuit integrat la diverse niveluri de detaliere de la $10 \mu\text{m}$ la 1nm . (IBM, Fujitsu).

1. Introducere

Mai jos se prezintă evoluția în timp a complexității procesoarelor, ca număr de dispozitive pe un circuit integrat. Pentium IV, care se producea în 2003, avea circa 50.000.000 tranzistoare MOS, pe o pastila de 2x2 cm².

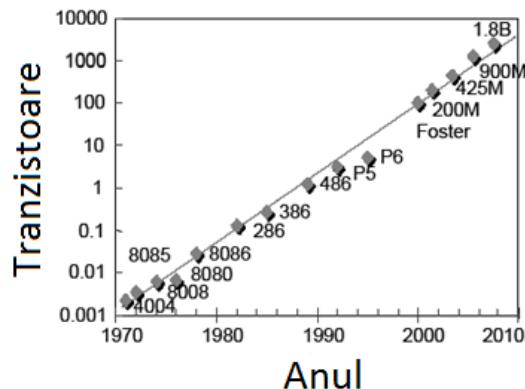


Figura 1.2 Complexitatea dispozitivelor din punctul de vedere al numărului de tranzistoare.

Începând cu memoria de 1Kb, realizată de către Intel, în 1971, memoriile semiconductoare au avut o evoluție susținută în termeni de capacitate și performanță (temp de acces): 256Mb în anul 2000, 1Gb în anul 2004, cu țintă de 16Gb, în 2008, conform ITRS (International Technology Roadmap for Semiconductor Technology).

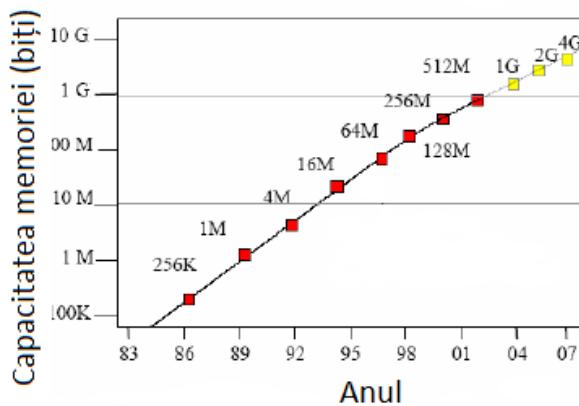


Figura 1.3 Evoluția capacitatății în biți a circuitelor de memorie (ITRS)

Organizarea la nivelul planului de amplasare a blocurilor componente ale unui microcontrolor industrial destinat aplicațiilor în

CMOS VLSI

industria automobilelor este prezentată mai jos. Pe lângă unitatea de prelucrare (procesor) microcontrolorul mai poseda diverse tipuri de memorii: EPROM, FLASH și RAM.

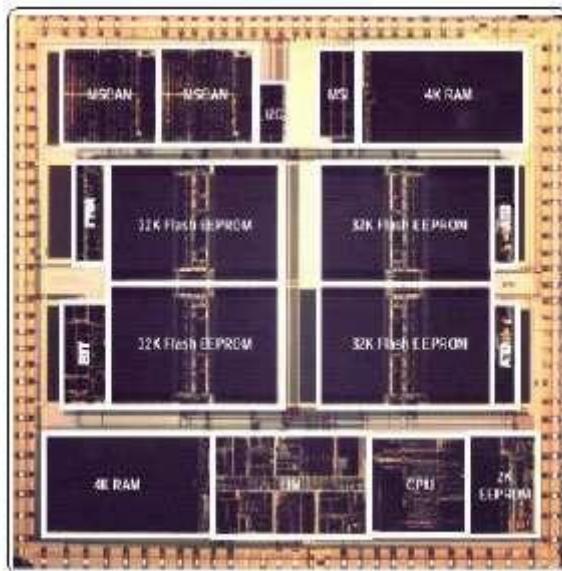


Figura 1.4 Planul de amplasare a blocurilor componente ale unui microcontrolor industrial.

În ceea ce privește reducerea dimensiunilor, se vor considera patru generații de tehnologii pentru circuitele integrate la nivel de:

- micrometru;
- submicrometru, 1990 - tehnologie $0.8 \mu\text{m}$;
- adânc submicrometru (deep submicron), 1995 – tehnologie $0.3 \mu\text{m}$;
- ultra-adânc submicrometru (ultra deep submicron) – tehnologie $0.1 \mu\text{m}$.

Conform figurii de mai jos cercetarea se află cu circa 5 ani înaintea producției de masă, în ceea ce privește tehnologia. În anul 2007 procesele litografice au coborât sub $0.07 \mu\text{m}$. Litografia, exprimată în μm , corespunde celor mai mici forme, care pot fi realizate pe suprafața unui circuit integrat.

Evoluția frecvenței ceasului pentru microprocesoarele și microcontroloarele performante a fost, de asemenea, influențată de reducerea dimensiunilor dispozitivelor integrate. Examinarea

1. Introducere

microprocesoarele Intel, destinate calculatoarelor personale (PC), și de la microcontroloarele Motorola, dedicate aplicațiilor din industria de automobile, pune în evidență două tendințe. Industria PC-urilor necesită procesoare extrem de rapide, care se caracterizează printr-o putere dissipată mare (30-100 W), în timp ce industria automobilelor solicita controloare încorporate, cu funcții numeroase și sofisticate, cu memorii de diverse tipuri și circuite de interfață capabile să asigure diferite protocoale de comunicații. Tendențele evoluției frecvențelor de lucru în cele două situații sunt asemănătoare.

Tabelul Tabel 1.2 prezintă parametrii mai importanți și evoluția lor odată cu perfecționarea tehnologiilor. Trebuie menționate creșterea numărului de straturi de metal, pentru interconectări, reducerea tensiunii de alimentare VDD, micșorarea grosimii stratului de oxid al porții, până la dimensiuni atomice. Se remarcă, de asemenea, creșterea dimensiunilor pastilei, cât și mărirea numărului de ploturi de I/E, disponibile pe o singură pastilă.

Litografia	Anul	Straturi de metal	Tensiunea de alimentare	Grosimea oxidului (nm)	Aria circuitului mm x mm	Ploturi de I/E
1.2 μm	1986	2	5.0	25	5 x 5	250
0.7 μm	1988	2	5.0	20	7 x 7	350
0.5 μm	1992	3	3.3	12	10 x 10	600
0.35 μm	1994	5	3.3	7	15 x 15	800
0.25 μm	1996	6	2.5	5	17 x 17	1000
0.18 μm	1998	6	1.8	3	20 x 20	1500
0.12 μm	2001	6 - 8	1.2	2	22 x 20	1800
90 nm	2003	6 - 10	1.0	1.8	25 x 20	2000
70 nm	2005	6 - 12	0.8	1.6	27 x 20	3000

Tabel 1.2 Parametrii mai importanți și evoluția lor odată cu perfecționarea tehnologiilor.

Circuitele pot fi de două tipuri: circuite discrete și circuite integrate. Circuitele discrete sunt formate din componente discrete precum:

- placă de bază;
- tranzistoare;
- diode;
- capacitoare;

CMOS VLSI

- rezistoare;
- fire de conectivitate.

Circuitele integrate sunt compuse din elemente pasive și elemente active. Elementele active (cele mai importante elemente dintr-un circuit integrat) sunt tranzistoarele care pot fi:

- tranzistoare bipolare (BJT - Bipolar Junction Tranzistor);
- tranzistoare cu efect de câmp (MOSFET - Field Effect Tranzistor). Aceste tranzistoare se folosesc de regulă în realizarea circuitelor cu o densitate foarte mare.

Pe același substrat se vor regăsi elemente active dar și elementele pasive care compun respectivul circuit integrat.

Spre exemplu, o memorie RAM de capacitate 4MB este formată din aproximativ 4.000.000 de tranzistoare. Acest circuit integrat este realizat folosind MOSFET-uri

Funcție de complexitatea circuitelor integrate ele se împart în:

- SSI - Small Scale Integrated Circuits - conțin un număr de tranzistoare cuprins între 10 - 100;
- MSI - Medium Scale Integrated Circuits - conțin un număr de tranzistoare cuprins între 100 - 1000;
- LSI - Large Scale Integrated Circuits - conțin un număr de peste 10.000 tranzistoare;
- VLSI - Very Large Scale Integration - conțin milioane de tranzistoare.

Actualmente, peste 95% din circuitele VLSI existente sunt fabricate din Siliciu (Si) deoarece acest material prezintă anumite proprietăți. Circuitele moderne trebuie să îndeplinească simultan condițiile de lucru pentru BJT și MOSFET, lucru care conduce la apariția de noi tehnologii precum BICMOS.

Materiale utilizate în VLSI se bazează pe Si monocristalin dopat cu impurități care asigură un surplus de goluri sau de electroni.

O dopare p normală - la 10^6 atomi de Si există 1 atom acceptor B, In, Ga cu $\rho = 2\Omega \text{ cm}$.

1. Introducere

O dopare p^+ - la 10^4 atomi de Si există 1 atom receptor de B, In, Ga cu $\rho = 0.05\Omega \text{ cm}$.

O dopare n normală – la 10^7 atomi de Si există 1 atom donor P, As cu $\rho = 5\Omega \text{ cm}$. Această dopare se realizează în zona canalului tranzistoarelor cu canal deschis obținându-se astfel o rezistență.

O dopare n^+ - la 10^4 atomi de Si există 1 atom donor P, As cu $\rho = 0.03\Omega \text{ cm}$.

1.1. Realizarea elementelor active de tip BJT

În cadrul acestor tipuri de elemente active, substratul trebuie să fie format dintr-un singur cristal. Substratul va trebui extins iar pentru realizarea acestei extinderi va trebui să cunoaștem orientarea cristalului. În vederea obținerii unui substrat de tip p, cristalul va trebui dopat; Si (element din grupa 4) va fi dopat cu Bor (B - element din grupa 3). Acest proces de dopaj este unul mai delicat deoarece trebuie obținută rezistivitatea impusă a substratului care este de $10 \Omega \text{ cm}$ (111).

Pasul 1 - Creșterea cristalului

În vederea realizării acestei etape, proprietățile asociate fiecărei orientări a cristalului sunt necesare a fi cunoscute aprioric. Orientarea este 111 aşa cum a fost definită mai sus. Suportul de lucru în VLSI este wafer-ul care este în fapt un disc foarte subțire. Grosimea acestui suport este de regulă aproximativ 500 microni. Diametrul cristalului este cel care va impune grosimea wafer-ului deoarece este foarte important ca stabilitatea mecanică să fie asigurată.

După procesul de dopare rezultă un substrat de tip p și o rezistivitate a wafer-ului de $10 \Omega \text{ cm}$, aşa cum se poate observa în Figura 1.5.



Figura 1.5 Reprezentarea unui substrat de tip p

Pentru realizarea unui element activ de tip BJT va trebui realizată creșterea și oxidarea cristalului în întreaga suprafață a substratului. Proprietatea Si de a se oxida ușor rezultând SiO_2 (dioxid de Si) este unul din principalele motive pentru care se folosește acest element în realizarea circuitelor integrate. Procesul de oxidare presupune creșterea temperaturii într-un mediu cu oxigen. Materialul rezultat (SiO_2) are proprietăți foarte bune ca dielectric precum și ca izolator. Totodată dioxidul de Si are și proprietăți foarte bune de mascare, proprietăți de care ne vom folosi în cadrul proceselor de realizare a elementelor active ale unui circuit integrat.

Pasul 2 - Oxidarea

Doparea se va realiza selectiv nefiind necesar să dopăm întreg blocul de Si și din acest considerent vom folosi dioxidul de siliciu pentru a masca doparea. Rezultatul acestui pas va fi un strat transparent (gen oglindă) care poate avea diverse culori: albastru, verde sau roz. Aceste culori rezultate sunt funcție de tehnica de oxidare folosită.

Procedeul de dopare selectivă a stratului de Si va conduce și la îndepărțarea mai de parte a stratului de dioxid de Si tot într-un mod selectiv. Acest procedeu de îndepărțare selectivă a stratului de dioxid de Si este prezentat în pasul 3.

Rezultatul obținut în urma acestui pas poate fi vizualizat în Figura 1.6.

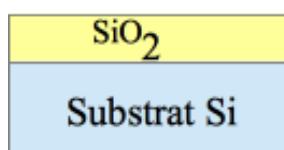


Figura 1.6 Rezultatul procesului de oxidare

Pasul 3 - Litografierea

Acest procedeu este necesar pentru îndepărțarea selectivă a stratului de dioxid de Si și presupune acoperirea întregului strat rezultat în urma procesului de oxidare cu un strat fotorezistiv aşa cum se prezintă în figura Figura 1.7.

1. Introducere

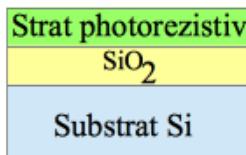


Figura 1.7 Reliefarea stratului fotorezistiv

Acest strat va intra în contact cu o mască (şablon) care va conține porțiuni transparente (Figura 1.8). Această fază este urmată de un proces de iradiere cu radiații UV, proces în urma căruia porțiunile transparente din cadrul şablonului sunt îndepărtate.



Figura 1.8 Reprezentarea unei măști

Porțiunile rămase în urma procesului de iradiere sunt cele care vor proteja stratul de dioxid de Si, aşa cum se poate observa în Figura 1.9.

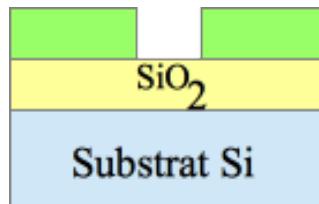


Figura 1.9 Rezultatul operațiunii de iradiere

Pentru înlăturarea porțiunilor de dioxid de Si se vor utiliza soluții fluoric acide (spre exemplu HF) care vor ataca stratul de dioxid de Si dar nu și stratul de Si (Figura 6). HF este un acid slab având o disociere constantă și mică comparativ cu acizii puternici $HF + H_2O \rightleftharpoons H_3O^+ + F^-$ [1]

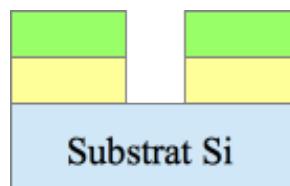


Figura 1.10 Efectul utilizării soluțiilor fluoric acide

CMOS VLSI

În ultimul pas se va realiza deschiderea ferestrei într-un oxid. În fapt se va înlătura stratul fotorezistent aşa cum se poate observa în Figura 1.11.

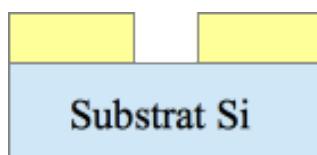


Figura 1.11 Înlăturarea stratului fotorezistent

Pasul 4 - Difuzia

Acest pas presupune realizarea unei dopări prin intermediul ferestrei realizându-se astfel o dopare n^+ . Doparea se va realiza cu arseniu realizându-se astfel un strat îngropat de tipul n^+ aşa cum se poate observa în Figura 1.8. Odată ce difuzia este realizată se va îndepărta stratul de dioxid de Si, regiunea activă fiind de tipul n^+ .

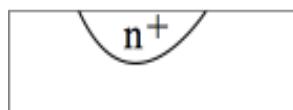


Figura 1.12 Realizarea unei dopări n^+ .

Pasul 5 - Epitaxy

După realizarea pașilor anterior descriși avem un substrat (dimensiunea sa este de doar câțiva microni) cu un monocristalin care trebuie îmbunătățit. Această îmbunătățire presupune adăugarea unui nou strat (cu o grosime mai mică decât primul strat) peste cel deja existent aşa cum se prezintă în Figura 1.13.

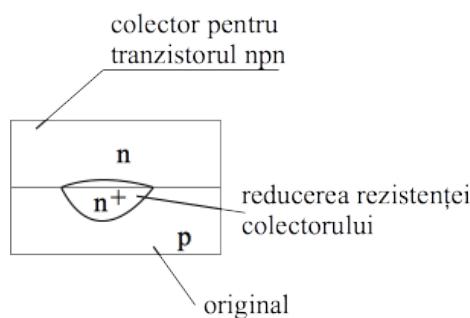


Figura 1.13 Adăugarea unui nou strat

1. Introducere

În mod uzual într-un chip există milioane de tranzistoare ceea ce înseamnă că toți colectorii sunt legați împreună adică toți colectorii vor fi în același strat epitaxy-al. Acest lucru este inacceptabil cu toate că în cazul utilizării tranzistoarelor discrete această problemă nu mai apare deoarece tranzistoarele individuale pot fi separate și utilizate.

În cazul utilizării unor circuite integrate, vom avea și componente pasive ceea ce conduce la imposibilitatea conectării împreună a colectorilor. Prin urmare dispozitivele adiacente vor trebui izolate ceea ce conduce la necesitatea determinării unei scheme de izolare a tranzistoarelor care se vor uni.

Tehnologiile mai vechi implică utilizarea unei joncțiuni pn inversate pentru realizarea izolației între tranzistoare deoarece o joncțiune pn inversată conduce la blocarea circulației curentului între două tranzistoare. În fapt se realizează o joncțiune pn izolatoare, aşa cum se poate observa în Figura 10.

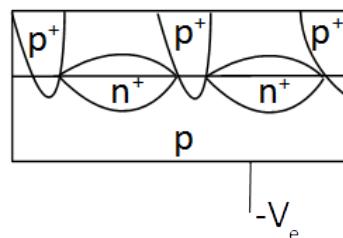
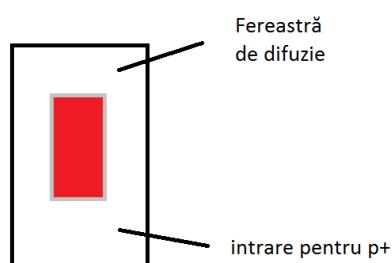


Figura 1.14 Joncțiune pn izolatoare

Masca necesară pentru realizarea acestei joncțiuni este prezentată în Figura 1.15.



CMOS VLSI

Figura 1.15 Masca pentru joncțiunea pn izolatoare

Din pașii prezentați până acum s-au realizat: colectorul și izolarea între dispozitive. Va mai trebui realizată baza și emitorul. Baza este realizată prin dopare înainte de realizarea emitorului deoarece emitorul este mai puternic dopat decât baza. Doparea bazei cu p presupune repetarea pașilor 2, 3 și 4. În acest fel se realizează o regiune de tip p ce reprezintă în fapt baza. Procesul de realizare a bazei unui tranzistor este prezentat în Figura 1.16.

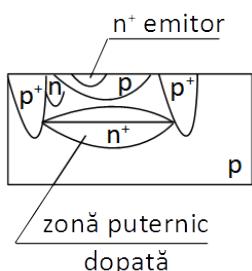


Figura 1.16 Realizarea bazei unui tranzistor care este o regiune p

Pasul 6 - Metalizarea

Acest ultim pas este necesar pentru realizarea contactului dintre tranzistor și lumea exterioară. Pentru realizarea contactelor în VLSI se folosește aluminiul. Aluminiul (Al) este un element tot din grupa III ceea ce înseamnă că Al va dota stratul de tip p rezultând un strat de tip p^+ . Problema cu folosirea aluminiului apare în momentul în care folosim straturi de tipul n (pentru straturile de tipul n^+ și p^+ nu apar probleme). Din acest motiv aluminiul este înlocuit cu cuprul (Cu) cu observația că și în acest caz este necesară folosirea unui mic "buzunar" de tipul n^+ .

Doparea se poate realiza prin difuzie sau printr-o metodă ce presupune implantarea de ioni. Actualmente metoda implantării de ioni este cea utilizată datorită flexibilității sale. Implantarea ionica constă în introducerea forțată a unor atomi ionizați sau molecule ionizate într-un material țintă, în condițiile în care li s-a imprimat o energie suficientă, pentru a penetra suprafața materialului. Sursa de ioni o constituie plasma, extracția acestora realizându-se cu ajutorul unor câmpuri puternice de CC sau RF. Întrucât permite controlul multor parametri în timpul procesului, implantarea ionica este utilizată în mod extensiv în fabricarea VLSI. De exemplu, adâncimea de pătrundere a atomilor este controlată direct prin

1. Introducere

potențialul de accelerare, în timp ce concentrația este controlată prin produsul curent × timp de expunere. Concentrațiile coborâte sunt folosite pentru controlul tensiunii de prag, atât la tranzistoare active, cât și la cele parazite. Concentrațiile mari sunt utilizate în formarea regiunilor de sursă și drenă, prin autoaliniere. Se poate afirma că implantarea ionica este direct responsabilă pentru succesul CMOS față de alte tehnologii. De exemplu, unul dintre motivele utilizării dispozitivelor PMOS, la începutul tehnologiei MOS, a fost controlul inadecvat al tensiunii de prag la fabricarea dispozitivelor NMOS. Implantarea ionică a făcut ca procesul NMOS să fie mult mai controlabil. Rezultatul procesului de metalizare este prezentat în Figura 1.17.

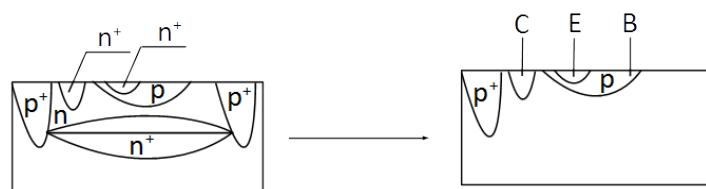


Figura 1.17 Metalizarea – C = colector, E= emitor, B = bază

1.2. Procesul de fabricare pentru MOSFET-uri

Pentru exemplificarea procesului de fabricare în cazul tranzistoarelor MOSFET se va porni de la un substrat de tipul n.

Pasul 1 – este identic cu pasul 1 de la procesul de fabricare al tranzistoarelor BJT. În cazul MOSFET-urilor, acest pas se numește oxidarea câmpului iar rezultatul acestui pas este prezentat în Figura 1.18.

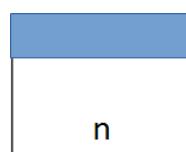


Figura 1.18 Oxidarea câmpului

Pasul 2 – Litografierea – acest pas presupune deschiderea unor ferestre în stratul de oxid (Figura 1.15). Prin intermediul acestor ferestre se va realiza dopajul sursei și a drenei. Procesul de dopaj nu se poate realiza pe

CMOS VLSI

dimensiunea exactă a ferestrei, astfel că vor apărea niște abateri de la dimensiunea ferestrei. Cu cât joncțiunea este realizată mai adânc cu atât abaterile de la dimensiunile ferestrei vor fi mai mari – apare efectul de margine.

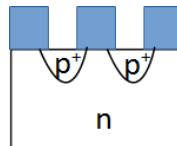


Figura 1.19 Doparea sursei și a drenei

După acest procedeu de dopare se va efectua o oxidare a sursei și a drenei (Figura 1.19) pentru a se putea realiza poarta – primul pas pentru realizarea porții este oxidarea sa. Conform Figurii 1.15 porțiunea dintre sursă și drene a fost deja supusă unui proces de oxidare, proces care nu rezolvă însă oxidarea porții din două motive:

- Grosimea stratului de oxid este prea mare (de regulă dimensiunea stratului este de 0.8 microni). Grosimea stratului trebuie însă să fie proporțională cu tensiunea de prag a MOSFET-ului, astfel că grosimea uzuală a stratului de oxid pentru tranzistoarele moderne trebuie să fie de câteva sute Å sau mai puțin.
- Calitatea oxidului este crucială pentru performanța tranzistorului, de aceea controlul calității oxidului este o operație critică în realizarea tranzistorului.

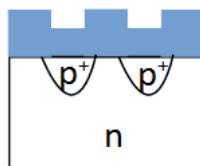


Figura 1.20 Pentru realizarea porții primul pas este realizarea oxidării porții

Pasul 3 – Oxidarea porții – În Figura 1.21 Realizarea sursei, porții și a drenei este prezentată oxidarea porții folosind culoarea portocalie.

1. Introducere

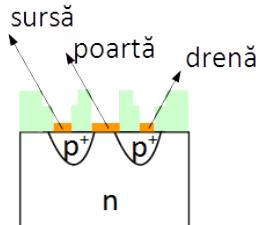


Figura 1.21 Realizarea sursei, porții și a drenei

Pasul 4 – Realizarea contactelor de metal – stratul de metal va trebui să fie deasupra stratului de oxid al porții și în contact cu stratul de Si în regiunile sursă și drenă. Pentru aceasta va trebui să se creeze o fereastră (Figura 1.18) și să se folosească aluminiul (reprezentat cu galben în Figura 1.22).

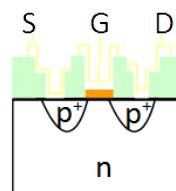


Figura 1.22 Cu galben este reprezentat rezultatul procesului de realizare a contactelor de metal

1.3. Fabricarea circuitelor VLSI - CMOS.

1.3.1. Introducere

Procesul fundamental de fabricație a unui circuit integrat constă în formarea selectivă a unor paturi/straturi de tip semiconductoare, dielectric și metal, pe suprafața placăteli de siliciu monocristalin. După realizare, aceste materiale devin componente active și pasive ale circuitului integrat. În acest subcapitol vor fi examineate procesele fundamentale legate de materiale, litografia și procedeele pentru crearea unui circuit integrat. Se va avea în vedere tehnologia CMOS cu două insule. Tratarea nu are un caracter exhaustiv, ci mai mult generic, ilustrând aspectele de detaliu comune celor mai multe procese CMOS.

1.3.2. Tehnologia de procesare a materialului de bază

1.3.2.1. Fabricarea plachetelor

În oricare proces VLSI placeta reprezintă materialul de la care se pornește. În tehnologia Si – CMOS, placeta se realizează dintr-un lingou de siliciu monocristalin. Lingoul se obține printr-un procedeu de tragere dintr-un creuzet, în care se află siliciu pur topit la o temperatură de circa 1475 °C. Cea mai frecventă metodă de obținere a lingoului din siliciu monocristalin se datorează lui Czochralski. Lingoul reprezintă un monocristal de Si, aproape fără defecte, cu o lungime de mai multe zeci de centimetri și cu un diametru de circa 10 cm. Pentru a împiedica apariția altor impurități, monocristalul este dopat N. Monocristalul este tăiat sub formă unor discuri/plachete cu o grosime, din considerente mecanice, de circa 300 μm , întrucât structurile electrice nu depășesc în grosime 10 μm . Placheta, de regulă de tip N, este acoperită cu un strat epitaxial de Si aproape intrinsec, înainte de a începe procesul de fabricare a circuitului integrat. Această prelucrare inițială conduce la creșterea rezistenței latch-up. Partea posterioară a placetei poate beneficia de un proces de implantare ionică pentru reducerea rezistenței electrice de contact, la împachetarea finală.

1.3.2.2. Oxidarea

Oxidarea se referă, de regulă, la creșterea sau depozitarea SiO_2 pe suprafața placetei. La o temperatură înaltă de 1000 °C și într-o atmosferă de O_2 , o placetă expusă se va oxida. Această structură de SiO_2 folosește Si de pe placetă, fiind plasată, atât în materialul placetei, cât și pe suprafața acesteia, după cum se va vedea în Figura 1.23.

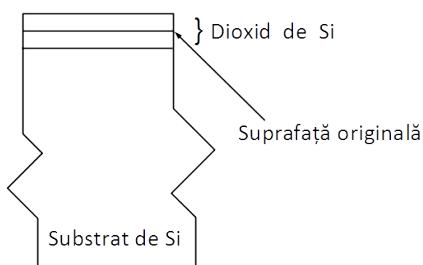


Figura 1.23 Procesul de oxidare în tehnologia CMSO

Acest proces crește un strat de SiO_2 de calitate, care poate fi folosit, fie ca dielectric izolator, fie ca oxid pentru poartă. În prezent, oxidul de

1. Introducere

poartă cu cele mai bune proprietăți electrice este format prin oxidare termică. Aceasta impune ca procesele care vor urma să aibă loc pe Si expus.

1.3.2.3. Depunerea de strat subțire

O serie de materiale utilizate în fabricarea circuitelor integrate își au originea în stare gazoasă. Aceste materiale sau straturi sunt constituite din siliciul policristalin, materialul principal pentru poarta tranzistorului MOS, cât și din izolatori dielectrici (SiO_2 și Si_3N_4), folosiți între diferitele straturi conductoare.

În practică se întâlnesc numeroase tehnici de depunere, care au multe elemente comune. Cea mai simplă, Depunerea Chimica de Vapori (DCV) la presiune atmosferică, este folosită pentru formarea siliciului policristalin. Placheta este încălzită în cuptor în prezența silanului (SiH_4), la o temperatură de circa 650°C , care este suficientă pentru a descompune molecula de SiH_4 și a depune un atom de Si pe suprafața plachetei. Pe suprafață expusă a plachetei se va forma un strat de $1\text{--}10 \mu\text{m}$ din cristale de siliciu cu diverse orientări.

O altă tehnică, având o importanță crescândă, este cea bazată pe depunere de materiale aflate în stare de plasmă. Plasma reprezintă un gaz neutru cu un număr egal de electroni și atomi ionizați sau molecule ionizate. Un material poate fi adus în stare de plasmă prin diverse procedee de excitare bazate pe câmpuri de radio-frecvență sau microunde. Plasma este generată în incinte vidate, la presiune joasă, ceea ce permite gazului să se descompună mai ușor. În timpul depunerii gazele necesare sunt introduse în incinta vidată la debit și presiune constante. Pentru a accelera electronii, care provoacă coliziunile, excitarea și ionizarea se folosesc de câmpuri de radio - frecvență. Generarea de radicali activi ușurează cerințele impuse pentru menținerea plachetei la temperatură înaltă, în timpul procesării. În Figura 1.24 se prezintă o instalație cu DCV și un reactor cu plasmă.

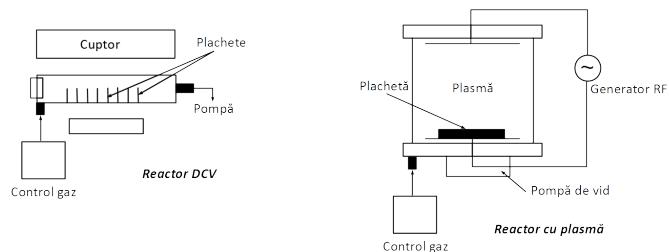


Figura 1.24 Reactor DCV și reactor cu plasmă

1.3.2.4. Difuzia

Difuzia reprezintă un proces termic de dopare N sau P a *Si* pentru a-i schimba caracteristicile electrice. Deși nu are loc o depunere sau o creștere de material, stratul care a suferit un proces de difuzie are un caracter critic în formarea dispozitivului CMOS. De exemplu, regiunile sursa și drena, ale unui tranzistor NMOS, sunt realizate în timpul unui proces de difuzie. În particular, atunci când se proiectează o poartă de bază, sunt folosite în mod intensiv straturile de difuzie. Difuzia are loc, de cele mai multe ori, plecând de la o sursă chimică în stare de vaporii, la temperatură înaltă. Mai recent, difuzia a fost cuplată cu implantarea ionică, pentru a obține regiuni mai bine izolate ale dispozitivelor.

1.3.2.5. Implantarea ionică

Implantarea ionică constă în introducerea forțată a unor atomi ionizați sau molecule ionizate într-un material țintă, în condițiile în care li s-a imprimat o energie suficientă, pentru a penetra suprafața materialului. Sursa de ioni o constituie plasma, extracția acestora realizându-se cu ajutorul unor câmpuri puternice de CC sau RF. Întrucât permite controlul multor parametri în timpul procesului, implantarea ionică este utilizată în mod extensiv în fabricarea VLSI. De exemplu, adâncimea de pătrundere a atomilor este controlată direct prin potențialul de accelerare, în timp ce concentrația este controlată prin produsul curent \times timp de expunere. Concentrațiile coborâte sunt folosite pentru controlul tensiunii de prag, atât la tranzistoarele active, cât și la cele parazite. Concentrațiile mari sunt utilizate în formarea regiunilor de sursă și drenă, prin autoaliniere. Se poate afirma că implantarea ionică este direct responsabilă pentru succesul CMOS față de alte tehnologii. De exemplu, unul dintre motivele utilizării dispozitivelor PMOS, la începutul tehnologiei MOS, a fost controlul inadecvat al tensiunii de prag la fabricarea dispozitivelor NMOS. Implantarea ionică a făcut ca procesul NMOS să fie mult mai fiabil.

1.3.2.6. Corodarea

Tehnicile examineate mai sus, cum ar fi DCV, pentru siliciul policristalin, acoperă întreaga plachetă, în timp ce, pentru realizarea unui dispozitiv este nevoie de o formă, de un şablon. Una din metodele de transfer selectiv al formelor constă în înlăturarea porțiunilor nemascate ale unui strat. Corodarea s-a realizat inițial în baie chimică (acid). Cât timp dimensiunile, avute în vedere pentru forme, sunt de ordinul a $10 \mu m$ sau mai mult, aceasta operează satisfăcător. În condițiile în care dimensiunile

1. Introducere

se reduc, tensiunea la suprafața materialului împiedică transferul efectiv al formei. De aceea corodarea uscată a devenit o metodă predominantă, care corespunde cerințelor VLSI. Corodarea uscată are loc din starea gazoasă, fiind ajutată de plasmă. De exemplu, CF_4 și O_2 , combinate în plasmă, pot genera un material extrem de eficient pentru corodarea Si , SiO_2 și a Si_3N_4 . Avantajele se referă la corodarea anisotropică, care asigură formarea unor pereti verticali, ceea ce reprezintă un element critic pentru DRAM-uri de mare densitate. Dificultățile asociate cu corodarea uscată se referă la selectivitate și la menținerea uniformității pe toată suprafața placetei. Corodarea uscată reprezintă un proces complicat, care depinde de o paletă largă de parametri.

1.3.2.7. Metalizarea

Metalizarea reprezintă unul dintre cele mai bine cunoscute procese de fabricație. Aluminiul sau aliajele de aluminiu sunt, de regulă, evaporate în vid sau împrăștiate de pe o țintă de Al, în prezența unei plasme. Pentru a crea traseele necesare de metal, trebuie să aibă loc un transfer de forme. În funcție de metal și strat, formarea este realizată printr-un proces litografic, folosind corodarea umedă sau uscată. În prezent se fac eforturi, care să conducă la găsirea unor aliaje capabile să asigure densități mai mari de curent.

1.3.2.8. Litografia

Litografia reprezintă etapa de bază în transferarea formelor geometrice de pe măști pe materialul de pe suprafața placetei. Formele materialului definesc ferestrele/tăieturile de contact, interconexiunile între diferitele straturi de metal, zonele de dielectric ale porților etc. Întreaga reprezentare geometrică a circuitului este redusă la o structură pe niveluri. De exemplu, nivelurile de metalizare sau de difuzie poartă numele de niveluri de măști. Circuitul integrat este realizat prin transferul secvențial al formelor de pe fiecare mască, nivel cu nivel, pe suprafața de Si. În funcție de tehnologia utilizată, masca poate fi pentru întreaga placeta sau pentru un circuit/structură de pe placetă. Pentru tehnologiile, care necesită precizie și rezoluție mai mari, masca corespunde unui circuit/structură, iar placeta este baleiată sub formă de rastru, pentru transpunerea formelor, în procesul de multiplicare a circuitelor de pe placetă. Procesul litografic este esențial pentru fabricație și se caracterizează printr-un nivel coborât, în sensul că, pe parcursul activității de proiectare a circuitului, detaliile procesului nu sunt avute în vedere de către proiectant, chiar dacă este vorba de un circuit

CMOS VLSI

specializat. Astfel, procesul litografic poate fi gândit ca un macro sau, la nivelul limbajului de asamblare, ca un microcod.

Etapele de bază ale unui proces litografic sunt:

- Pe suprafața plachetei se aplică o peliculă dintr-un polimer fotosensibil, care se usucă și este apoi supus unei radiații de ultraviolete printr-o mască fotografică, corespunzătoare formei dorite. Radiațiile ultraviolete se utilizează pentru a reduce fenomenul de difracție.
- În timpul expunerii polimerul reacționează cu radiația ultravioletă, fie prin întărirea, fie prin slăbirea lanțurilor macromoleculare. După expunere, suprafața plachetei este developată, ceea ce face ca ea să conțină imaginea, la nivelul peliculei de polimer, pe materialul fotosensibil. În funcție de polimer, în timpul developării, se înlătură, fie suprafața expusă, fie cea neexpusă.
- Placheta este plasată într-un mediu, care corodează zonele neprotejate de către forma realizată din polimer. Din această cauză, polimerul mai poartă numele de rezist sau fotorezist, în cazul în care se folosește lumina în procesul litografic.

Detalii privitoare la procesul litografic sunt date în Figura 1.25.

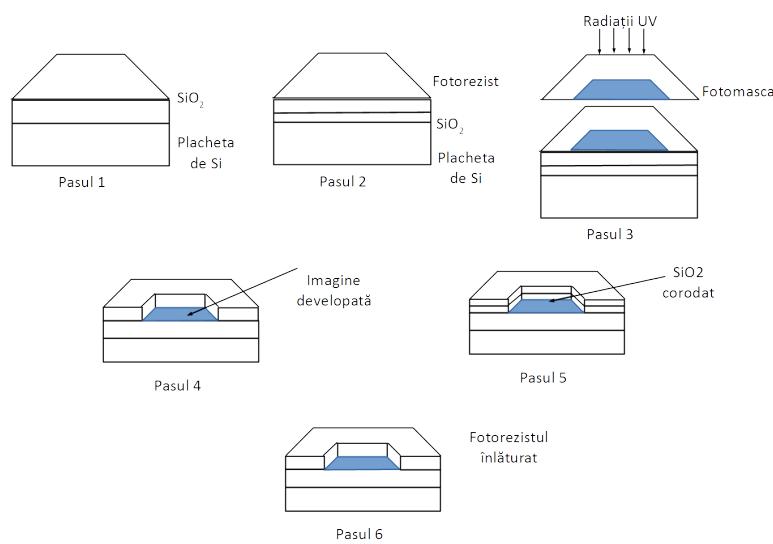


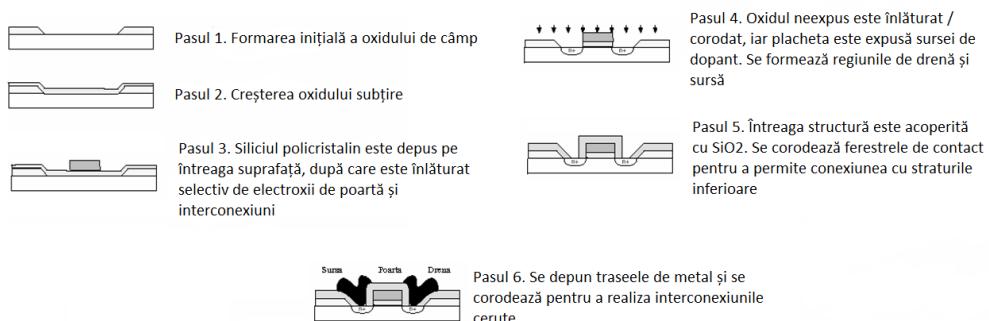
Figura 1.25 Etapele procesului de litografie

1. Introducere

1.4. Procesul CMOS simplificat

1.4.1. Un proces NMOS simplu

Înainte de a examina un proces CMOS, este util să se ilustreze pe scurt etapele fabricării tranzistorului de tip NMOS, conform celor de mai jos



Pasul 1 - Se formează oxidul de câmp SiO_2 inițial. Deschiderile în oxid definesc regiunile active.

Pasul 2 - Se crește oxidul subțire, definit adesea ca “thinox”.

Pasul 3 - Se depune siliciu policristalin pe întreaga suprafață. Aceasta este înlăturat selectiv din zonele electrozilor de poartă și ale interconexiunilor.

Pasul 4 - Oxidul subțire neexpus este îndepărtat/corodat, iar placeta este expusă sursei de dopare.

Pasul 5 - Întreaga structură este acoperită cu SiO_2 , iar ferestrele/tăieturile de contact sunt corodate, pentru a permite efectuarea conexiunilor cu straturile inferioare.

Pasul 6 - Metalul, pentru interconexiuni, este evaporat și apoi corodat, pentru a realiza interconexiunile finale.

Este important să se sublinieze faptul că jonețările de difuzie sunt realizate numai în regiunile în care poarta de siliciu policristalin nu maschează substratul inferior. Acesta este un proces cu autoaliniere. În

CMOS VLSI

această manieră, regiunile de drenă și sursă nu se extind sub poartă, ceea ce reduce capacitatea parazită, care ar fi putut degrada performanța. Inițial, dispozitivele MOS utilizau metal, în calitate de material pentru poartă. Aceste tehnologii, în mod inherent, erau mai lente, datorita capacităților parazite. Trebuie, de asemenea, subliniat faptul că, ori de câte ori, un traseu de siliciu policristalin intersectează un traseu de difuzie, se formează un tranzistor. Regiunile de drenă și sursă sunt marcate cu N+ sau n+, subliniind faptul că ele sunt puternic dopate, pentru a îmbunătăți contactele ohmice cu metalul, pentru a reduce rezistențele parazite și pentru a transfera rapid sarcina în canal, la aplicarea tensiunii pe poartă. Un alt avantaj este acela al reducerii dependenței capacității porții de tensiune.

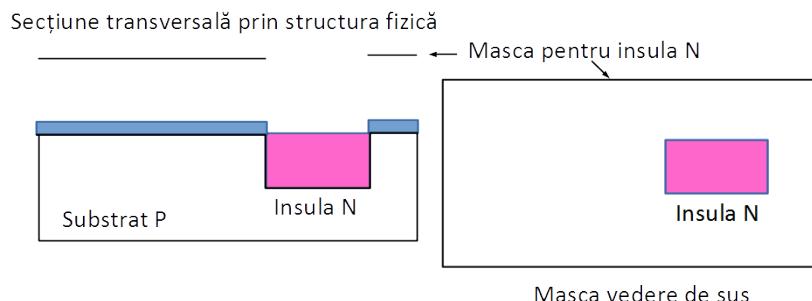
1.4.2. Un proces CMOS cu insula N

O abordare comună în fabricarea structurii CMOS, cu insulă N, este aceea de a porni de la o plachetă dopată ușor P și de a crea pe aceasta o insulă N, pe care se formează un tranzistor de tip P. În zona nativă P, a substratului se realizează tranzistorul N.

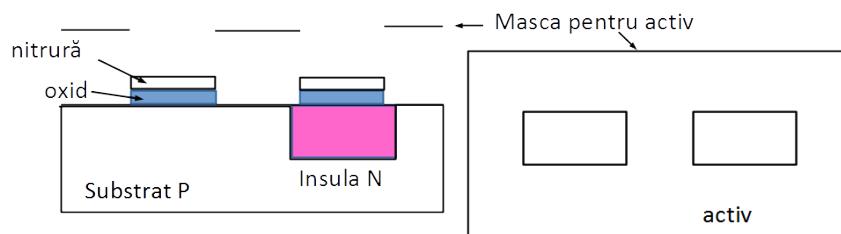
În cele ce urmează vor fi prezentate etapele principale ale unui proces CMOS, cu insulă de tip N. În realitate, etapele prelucrării plachetei sunt relativ complexe și depind de linia de fabricație. Fiecare etapă este ilustrată printr-o secțiune transversală în structura CMOS, cât și prin masca asociată etapei.

- În etapa (a) masca definește insula N, în care se va realiza tranzistorul P. Insula P este realizată prin implantare ionică sau prin difuzie. Implantarea ionică are avantajul de a crea insule de mică grosime, ceea ce satisfac procesele care presupun dimensiuni mici, în timp ce difuzia, realizându-se în toate direcțiile, cu cât este mai profundă, cu atât se răspândește lateral. Astfel, în cazul difuziei pot fi afectate structurile vecine, ceea ce impune zone mai mari de separare față de acestea, având drept consecință reducerea densității componentelor pe structură.

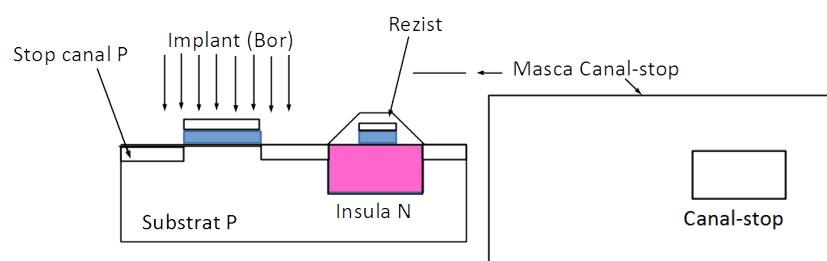
1. Introducere



- Masca următoare (b) poartă numele activă, întrucât ea definește zonele în care va fi prezent oxidul subțire, necesar realizării porților tranzistoarelor N și P, cât și a surselor și drenelor acestora, prin implantare ionică sau difuzie. Această mască mai poartă numele de mască pentru oxidul subțire sau mesa. În cadrul acestei etape se crește un strat subțire de SiO_2 , care se acoperă cu SiN , pentru a forma un strat de mascare, necesar următoarelor două etape.

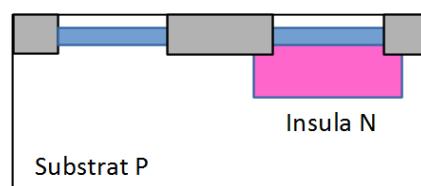


- În această etapă (c) se realizează implantarea pentru ceea ce se numește canal-stop. Aceasta folosește masca insulă-P, care este complementara măștii insulă-N, pentru a dopa P^+ substratul în zonele în care nu se află tranzistoare N, folosind o mască de fotorezist. Această dopare, cât și stratul gros de oxid, ce va acoperi aceste zone, vor împiedeca realizarea unei concuranțe între zonele drenă/sursă ale unor tranzistoare, care nu au nici o legătură între ele.



CMOS VLSI

- După implantarea canal-stop, masca din fotorezist este înlăturată, ceea ce permite definirea regiunilor active de către structura SiO₂/SiN mascată anterior. În continuare este format stratul gros de oxid, în zonele în care stratul de SiN este absent. Stratul de oxid se formează, atât în direcție verticală, cât și pe orizontală/lateral, sub structura SiO₂/SiN (d). Această extindere laterală poartă numele de “cioc de pasăre”, datorită formei pe care o capătă, și are ca efect reducerea dimensiunilor zonelor active. Astfel, lățimea canalului unui tranzistor va fi mai mică decât cea presupusă la realizarea măștii. Această tehnică, de realizare a stratului gros de oxid, poartă numele de LOCOS (Local Oxidation Of Silicon). În scopul reducerii efectului menționat mai sus, s-au propus diferite metode, printre care și cea numită SWAMI (Side Wall Masked Isolation). Un alt aspect important este cel legat de planaritatea interfeței între oxidul subțire, de poartă, și oxidul gros, de câmp. În cazurile în care diferențele de cote între cele două straturi sunt mari, există pericolul fisurării traseelor de metal, care se depun în zonele de interfață între cele două straturi de oxid. Pentru a preîntâmpina un asemenea fenomen, se recurge la o serie de tehnici de “planarizare”. Una dintre tehnici constă în precordarea plachetei substrat, pe o adâncime egală cu jumătatea grosimii stratului gros de oxid, în zonele care vor fi acoperite de către acesta. În continuare va fi format stratul de oxid LOCOS, în condițiile unei “planarități” satisfăcătoare.



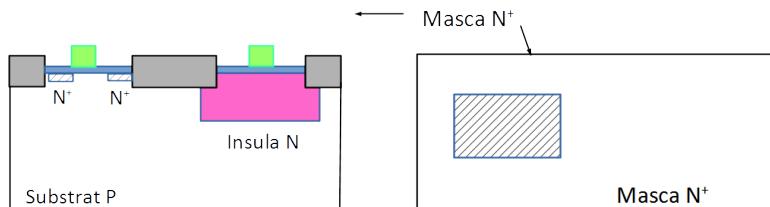
- În această etapă, se va efectua o ajustarea a tensiunii de prag a tranzistorului *N*, folosind masca de fotorezist pentru insula *P*. Procesele de fabricație curente presupun o dopare *N*⁺ a siliciului policristalin. În condițiile proceselor corespunzătoare dispozitivelor de mici dimensiuni, concentrațiile normale de dopare au ca rezultat tensiuni de prag de 0.5 ÷ 0.7 V, pentru dispozitivele de tip *N*, și de – 1.5 ÷ 2.0 V, pentru dispozitivele de tip *P*. Astfel, pentru dispozitivele de tip *P* ajustarea tensiunii de prag se va face într-o măsură mai mare decât cea a tensiunii de prag, pentru dispozitivele de tip *N*. Aceasta

1. Introducere

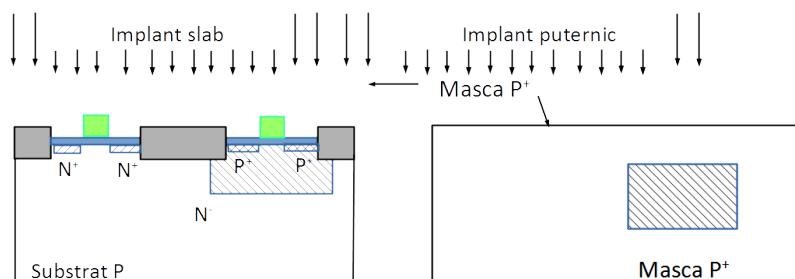
se realizează prin introducerea unui strat suplimentar, încărcat negativ, la interfața între siliciu și oxid. În acest mod canalul de la interfața siliciu/oxid se deplasează în siliciu, în adâncime, formând un dispozitiv cu “canal îngropat”. După aceasta este format oxidul de poartă.

- Definirea porții implică acoperirea suprafeței cu siliciu policristalin, după care are loc îndepărarea acestuia, prin corodare, pentru a obține forma dorită, în cazul de față un “U” întors (e). După cum este cunoscut, în cazul porții din siliciu policristalin, are loc o autoaliniere a regiunilor drenă-sursă.
- În continuare se utilizează o mască N+, pentru a specifica zonele de difuzie și de siliciu policristalin, care vor fi implantate N+ (f). Dacă zona N+ se află pe substratul de tip P, atunci se formează un tranzistor cu canal N. În cazul când zona N+ este plasată pe o insulă de tip N, se va obține un contact ohmic la insulă. Acesta are, evident, un caracter rezistiv. Această mască mai poartă numele de select, deoarece selectează acele regiuni în care se formează tranzistoarele de tip N. După cum este cunoscut, în cazul proceselor, care presupun dimensiuni reduse pentru dispozitive, se face simțit efectul “electronilor fierbinți”. Aceștia, datorită energiei pe care o posedă, pot disloca goluri la drenă, care sunt preluate de către substratul încărcat negativ, ceea ce duce la apariția unui curent de substrat. Electronii fierbinți au efecte negative asupra timpului de reîmprospătare, la memoriile dinamice, asupra zgomotului și a fenomenului de “latch-up”. În cazul când aceștia penetreză oxidul porții, apare și un curent de poartă, care poate degrada tensiunea de prag, curentul de subprag și transconductanța. Pentru a evita asemenea fenomene se recurge la un proces de formare a zonelor sursă/drenă în două etape. Mai întâi se realizează, în aceste zone, o structură puțin adâncă, ușor dopată N (LDD - Light Doped Drain structure), ca în figura (g), în zonele neacoperite de siliciul policristalin. În continuare se crește un oxid de “spațiere”, peste traseul de siliciu policristalin al porții, după care are loc o implantare N+. Aceasta nu va afecta zona aflată la limita porții, sub oxid. După înlăturarea oxidului de “spațiere” rezultă o structură mult mai rezistentă la efectele electronilor fierbinți. Procesele curente de $0.25 \mu m$ nu vor utiliza tehnica LDD.

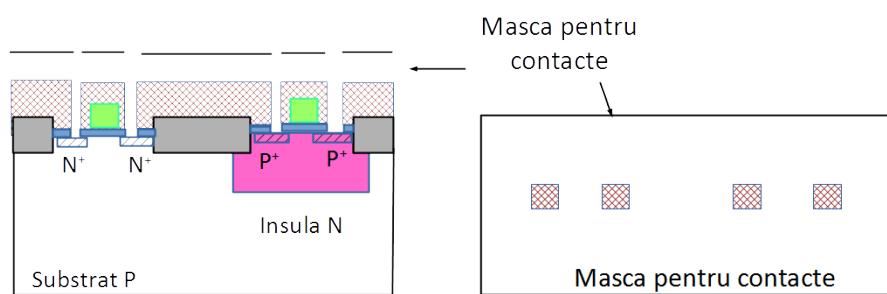
CMOS VLSI



- Următoarea etapă utilizează complementul măștii N^+ , cât și o mască suplimentară. Absența unei regiuni N^+ , în zonele acoperite de către oxidul subțire, specifică faptul că acestea vor fi zone de difuzie P^+ sau zone active. Zona activă P , pe o insulă N , definește fie tranzistoare de tip P , fie fire (h). O difuzie de tip P^+ , pe un substrat P , permite realizarea unui contact *ohmic*. După aceasta se depune un strat de SiO_2 . În cazul tranzistoarelor de tip P , nu se pune problema unei etape LDD, întrucât, în acest caz, purtătorii fierbinți nu au aceleași efecte ca în cazul tranzistoarelor de tip N .

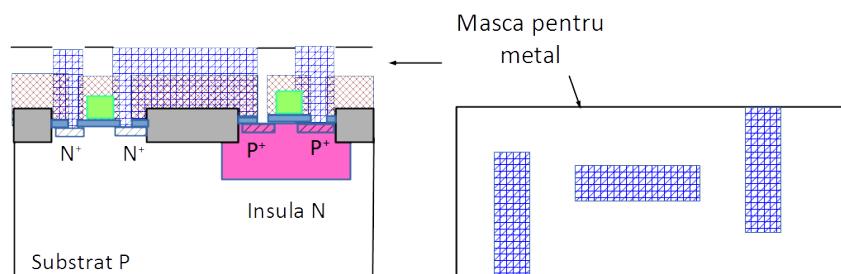


- În această etapă se definesc tăieturile de contact, ceea ce presupune corodarea SiO_2 până la stratul cu care trebuie să se realizeze contactul (i). Aceasta permite, în cadrul următoarei etape, să se efectueze contacte cu regiunile de difuzie sau siliciu policristalin.



1. Introducere

- Pentru realizarea contactelor se face o metalizare a suprafeței, urmată de o corodare selectivă.



- Ultima etapă, neilustrată printr-un desen, se referă la *pasivizarea* întregii suprafețe a siliciului, cu practicarea unor tăieturi, prin corodare, în zonele ploturilor de contact. Pasivizarea constă în acoperirea structurii, cu un strat de sticlă, în scopul protejării acesteia față de eventualele contaminări, care ar modifica în mod nedorit comportarea circuitului.

2. TRANZISTOARE MOS ȘI CMOS

2.1. Principii generale

Tranzistorul MOSFET (sau prescurtat MOS) poate fi analizat ca un sistem pe căruia terminale se pot defini semnale de intrare, de ieșire sau de control. În cele ce urmează se vor lua în considerare notațiile și terminologia literaturii anglo-saxone, notații frecvent întâlnite în cataloge și literatura de specialitate.

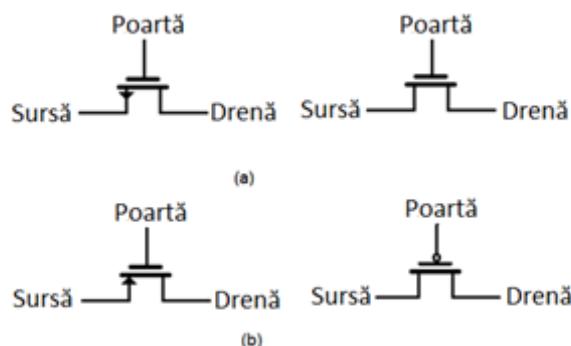


Figura 2.1 Simbolurile tranzistorilor MOS: (a) cu canal n ; (b) cu canal p

În Figura 2.1 este prezentată simbolistica utilizată pentru un tranzistor MOS cu canal n (n-MOS) (Figura 2.1a) și cu canal p (Figura 2.1 b), în care se remarcă cele trei terminale S (Source - Sursă), D (Drain - Drenă) și G (Gate - Poartă).

În Figura 2.2 este prezentată structura simplificată a unui tranzistor MOS. Pe un substrat p sunt implementate două zone cu dopare în exces n⁺ ce formează sursa și respectiv drena tranzistorului și o zonă conductoare, puternic dopată cu polisiliciu, ce formează grila. O peliculă de SiO₂ izolează grila de substrat. Lungimea grilei pe traseul drenă-sursă este notată cu L_d iar lățimea este notată prin w. Lungimea efectivă de difuzie va fi:

2. Tranzistoare MOS Si CMOS

$$L_D = (L_d - L_e)/2 \quad 2.1$$

iar grosimea stratului de oxid în zona porții se va nota prin t_{ox} . Într-o analiză completă a unui tranzistor MOS apare și potențialul substratului p, în mod curent aflat la cel mai negativ potențial aplicat. Prezența substratului este pusă în evidență, într-o reprezentare simbolică, pentru un MOS-n, ca în Figura 2.2.

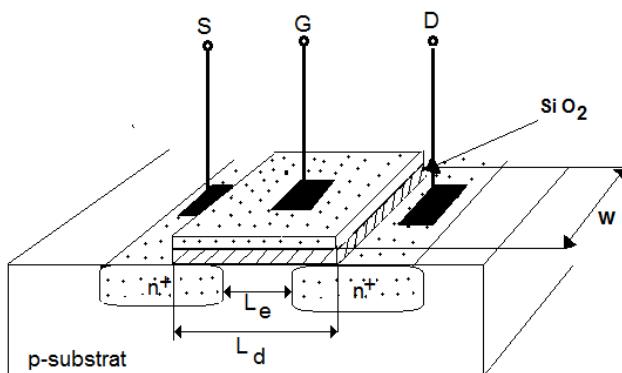


Figura 2.2 Structura unui tranzistor MOS

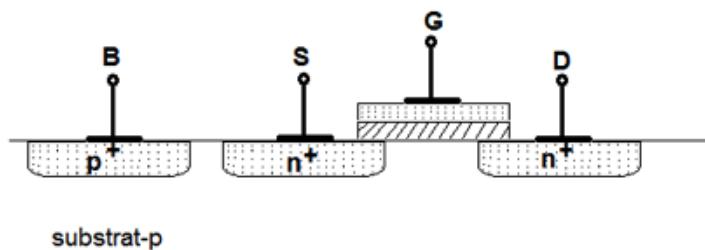


Figura 2.3 Polarizarea unui MOS-n

În contextul implementării unor tranzistoare CMOS, MOS-p se obține prin complementarea zonelor de la MOS-n.

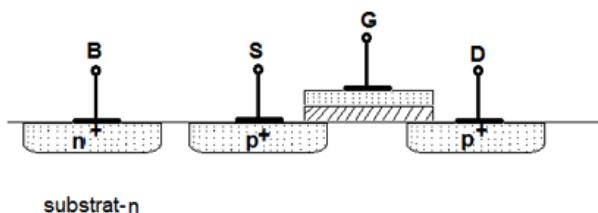


Figura 2.4 Polarizarea unui MOS-p

2.2. Caracteristicile tranzistoarelor MOS

Caracteristicile intrare-ieșire ale unui MOS se pot obține prin analiza transferului de sarcini, pentru diferite nivele de potențial aplicate pe electrozii acestei structuri.

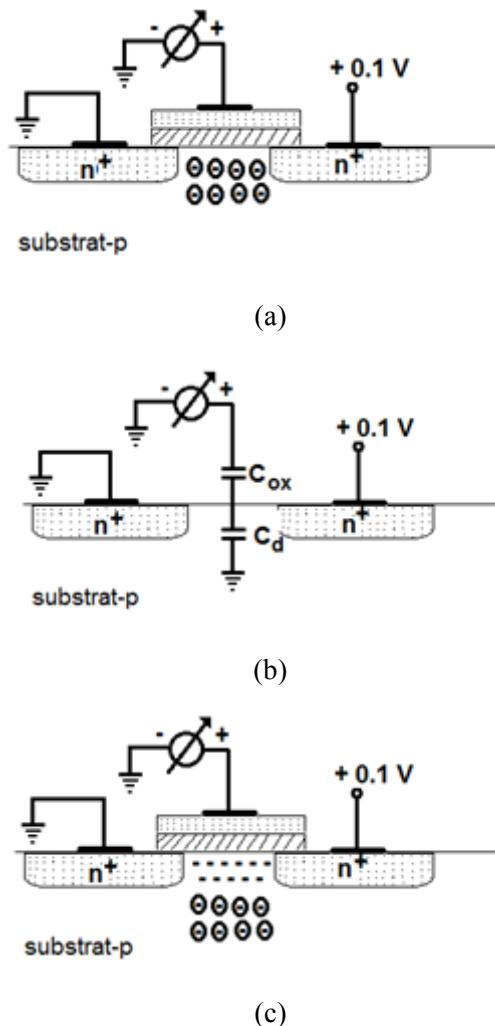


Figura 2.5 Polarizarea tranzistorului MOS pe poartă: a) Formarea zonei de respingere; b) Capacitățile echivalente pe poartă; c) Formarea sarcinilor

Pentru un potențial pe poartă ușor pozitiv, goulurile din substratul-p sunt respinse determinând apariția unor ioni negativi. În această fază, nu există purtători de sarcină, electroni, care să determine apariția unui curent

2. Tranzistoare MOS și CMOS

între drenă și sursă (Figura 2.5 a). La o creștere ușoară a potențialului pe poartă, efectul capacativ al porții (Figura 2.5 b) va determina apariția unei zone de electroni (Figura 2.5 c) și deci un curent drenă-sursă (numai în prezența unui potențial drenă-sursă). Tranzistorul intră în conducție (state-on) [2,4] iar tensiunea de prag pe poartă la care se produce această deblocare este V_{TH} . Acest potențial de prag este extrem de important în identificarea caracteristicilor tranzistorului și, tehnologic, poate fi modificat prin doparea în exces a substratului cu p⁺ [3-5]. O analiză similară poate fi realizată și pe un MOS-p.

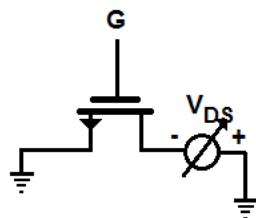


Figura 2.6 Polarizarea circuitului de drenă

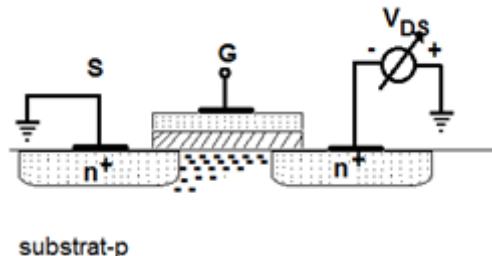


Figura 2.7 Distribuția sarcinilor pentru o polarizare pe drenă

Pentru ridicarea caracteristicii de drenă, se va considera circuitul din Figura 2.6 în care potențialul V_{DS} este crescut treptat, pentru diferite tensiuni aplicate pe poartă, V_{GS} . Considerând o distribuție a potențialului $V(x)$ de-a lungul canalului drenă-sursă, determinat de V_{DS} , curentul de drenă are forma [2],

$$I_D = w C_{0x} [V_{GS} - V(x) - V_{TH}] \mu_n \frac{dV}{dx} \quad 2.2$$

unde $V(0) = 0, V(L) = V_{DS}$. Integrând această ecuație și ținând cont că I_D este constant, se obține

CMOS VLSI

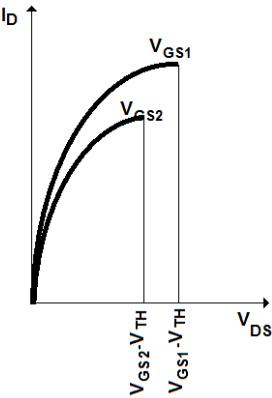
$$I_D = \mu_n \frac{w}{L} C_{0x} [(V_{GS} - V_{TH})V_{DS} - \frac{1}{2}V_{DS}^2] \quad 2.3$$

Alura caracteristicii $I_D = f(V_{DS})$ este dată de o parabolă (Figura 2.8). Punctul de maxim este atins pentru valori

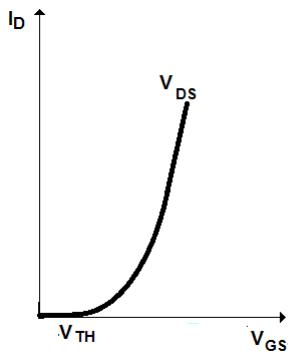
$$V_{DS} = V_{GS} - V_{TH} \quad 2.4$$

Iar acest maxim are valoarea

$$I_{Dmax} = \frac{1}{2} \mu_n \frac{w}{L} C_{0x} (V_{GS} - V_{TH})^2 \quad 2.5$$



(a)



(b)

Figura 2.8 a) Caracteristica $I_D = f(V_{DS})$ în regim de triodă.
b) Caracteristica $I_D = f(V_{GS})$

Dacă

2. Tranzistoare MOS Si CMOS

$$V_{DS} < V_{GS} - V_{TH} \quad 2.6$$

se spune că tranzistorul operează în regim de triodă. Se remarcă dependența acestor caracteristici de parametrii tehnologici, C_{0x}, w, L . Dacă,

$$V_{DS} < 2(V_{GS} - V_{TH}) \quad 2.7$$

se obține o dependență liniară a lui I_D în raport cu V_{DS} ,

$$I_D \approx \mu_n \frac{w}{L} C_{0x} (V_{GS} - V_{TH}) V_{DS} \quad 2.8$$

În acest caz, tranzistorul funcționează ca o rezistență de valoare:

$$R_{on} = \frac{1}{\mu_n \frac{w}{L} C_{0x} (V_{GS} - V_{TH})} \quad 2.9$$

deci cu o caracteristică aproximată de forma celei din Figura 2.9. Utilizarea unui tranzistor MOS în regim de rezistență variabilă este extrem de importantă într-un domeniu larg de aplicații.

Dacă potențialul V_{DS} depășește valoarea critică dată de (2.7),

$$V_{DS} > 2(V_{GS} - V_{TH}) \quad 2.10$$

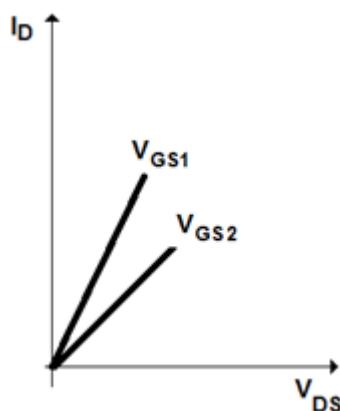


Figura 2.9 Caracteristica aproximată a tranzistorului MOS pentru V_{DS} mic
tranzistorul intră într-un regim de saturatie [2-4], valoarea acestui curent rămânând constantă la valoarea

$$I_D = \frac{1}{2} \mu_n \frac{w}{L} C_{0x} (V_{GS} - V_{TH})^2 \quad 2.11$$

Caracteristica globală $I_D = f(V_{DS})$ este prezentată în Figura 2.10.

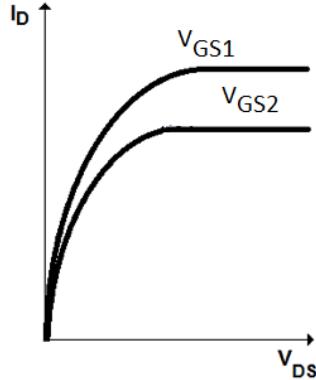


Figura 2.10 Caracteristica $I_D = f(V_{DS})$

Caracteristica de saturatie permite utilizarea tranzistoarelor MOS ca surse de curent constant. În Figura 2.12 sunt ilustrate aceste regimuri, atât pentru tranzistoarele MOS-n cât și MOS-p. Operarea tranzistoarelor MOS în saturatie este caracterizată prin transconductanță asociată g_m .

$$g_m = \frac{\partial I_D}{\partial V_{GS}} = \mu_n \frac{W}{L} C_{0x} (V_{GS} - V_{TH}) \quad 2.12$$

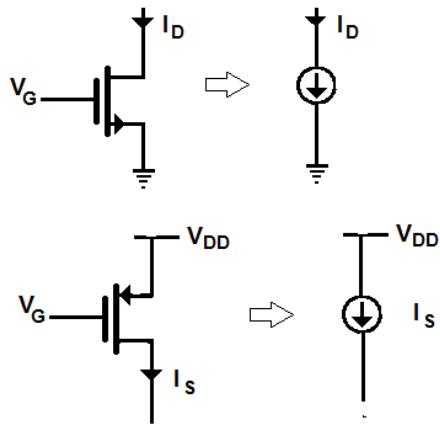


Figura 2.11 Tranzistoare MOS în regim de curent constant

Acest parametru definește exact comportarea tranzistoarelor MOS în acest regim funcțional, de aceea este interesantă comportarea acestui

2. Tranzistoare MOS și CMOS

parametru în raport cu celelalte mărimi. În Figura 2.12 sunt prezentate câteva caracteristici: $g_m = f(V_{GS} - V_{TH})$, și $i_s = f(I_D)$.

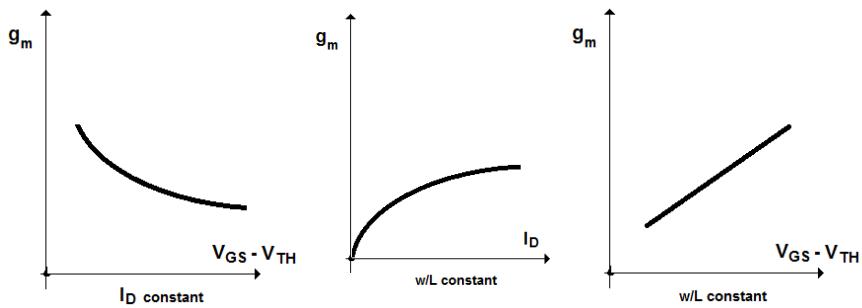


Figura 2.12 Caracteristicile transconductanței g_m

2.3. Proprietăți specifice ale tranzistoarelor MOS

În multe situații practice, modificările impuse unor parametrii funcționali sau constructivi determină abateri de la regimurile convenționale discutate în paragraful precedent. În cele ce urmează vor fi analizate câteva probleme de acest tip.

2.3.1. Efectul de substrat

În analizele anterioare s-a presupus că substratul-p (se va considera un tranzistor MOS-n) este conectat la masă (potențial zero). Dacă potențialul de polarizare V_B devine negativ, acesta va modifica comportarea tranzistorului. În acest caz, mai multe goluri vor fi atrase spre electrodul de sursă ceea ce va determina o creștere a numărului de ioni negativi în zona canalului drenă-sursă. În acest caz, formarea purtătorilor de sarcină în zona canalului se va restricționa, deci un potențial negativ V_B va determina o creștere a potențialului de prag V_{TH} (Figura 2.13).

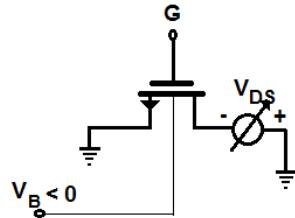


Figura 2.13 Polarizarea negativă a substratului

Se poate demonstra că această creștere este dată de o relație de forma

$$V_{TH} = V_{TH0} + \gamma(V_B) \quad 2.13$$

2.3.2. Modulația în lungime a canalului

În paragrafele anterioare s-a discutat despre comportarea tranzistoarelor MOS și s-a considerat că lungimea canalului drenă-sursă, un parametru important în toate evaluările caracteristicilor, este constant. În realitate, se demonstrează că această lungime este variabilă și depinde de tensiunea V_{DS} aplicată. Cu cât potențialul drenei crește, lungimea efectivă a canalului scade cu o valoare relativă dată de:

$$\frac{\Delta L}{L} = \lambda V_{DS} \quad 2.14$$

Ceea ce va determina reevaluarea curentului I_D

$$I_D \approx \frac{1}{2} \mu_n \frac{W}{L} C_{0x} (V_{GS} - V_{TH})^2 (1 + \lambda V_{DS}) \quad 2.15$$

Relația (2.15) determină o evoluție oarecum diferită în zona de saturare, panta caracteristicii fiind ușor pozitivă (Figura 2.14).

2. Tranzistoare MOS Si CMOS

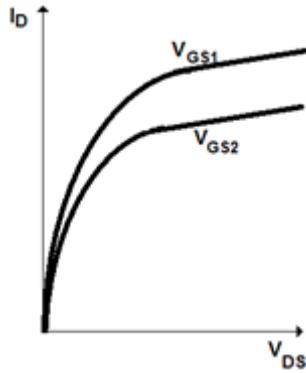


Figura 2.14 Caracteristica $I_D = f(V_{DS})$ în condițiile modulației în lungime a canalului

2.4. Modelele la semnal mic ale tranzistoarelor MOS

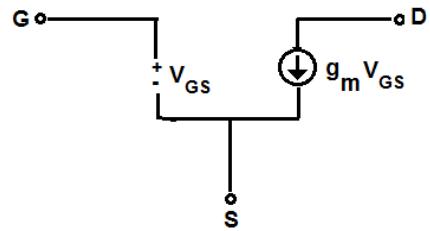
Analiza comportării acestor tranzistoare în jurul unor puncte de funcționare, fie ca urmare a unor semnale aplicate pe electroziile de control, fie ca urmare a unor perturbații necontrolabile, necesită utilizarea modelelor la semnal mic.

Modelul ideal la semnal mic este prezentat în Figura 2.15 a. Întrucât curentul I_D este dependent de tensiunea V_{GS} , acesta va fi produs de un generator de curent încorporat în circuitul de ieșire. Circuitul de intrare conține tensiunea de polarizare V_{GS} . Modelul din Figura 2.15 b introduce încă o componentă de curent I_D care ține cont de efectul de modulație în lungime a canalului. În unele cazuri, în locul acestui generator se preferă introducerea unei rezistențe liniare [2,3] (Figura 2.15 c).

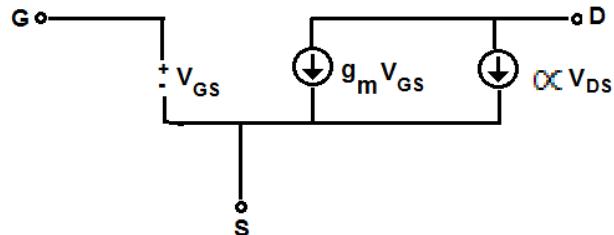
$$r_o = \frac{\partial V_{DS}}{\partial I_D} \quad 2.16$$

Efectul de substrat este prezent prin introducerea potențialului V_{BS} în circuitul de sursă (Figura 2.15 d)

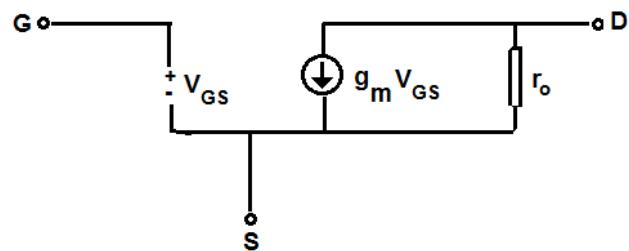
CMOS VLSI



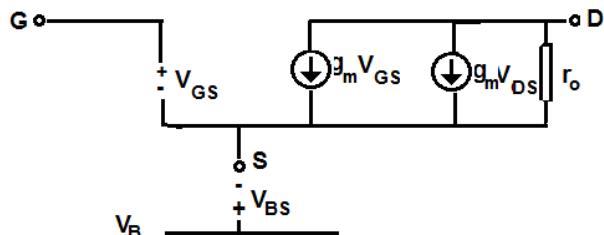
(a)



(b)



(c)



(d)

Figura 2.15 Modele la semnal mic

2. Tranzistoare MOS și CMOS

2.5. Porți logice C-MOS elementare

Spre deosebire de porțile logice realizate în tehnologii bipolare, tehnologiile MOS și CMOS permit implementarea acestor porți în configurații simple. Vom ilustra această tehnică pe câteva porți logice elementare.

2.5.1. Poarta logică NU

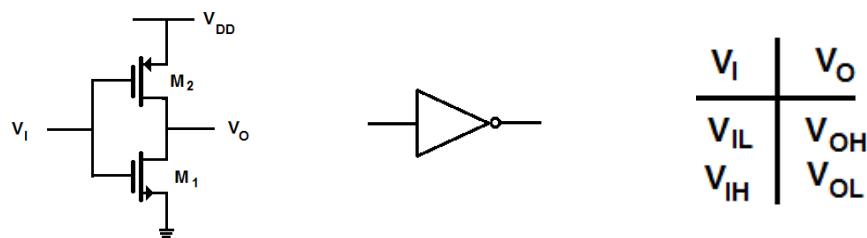


Figura 2.16 Poarta logică NU

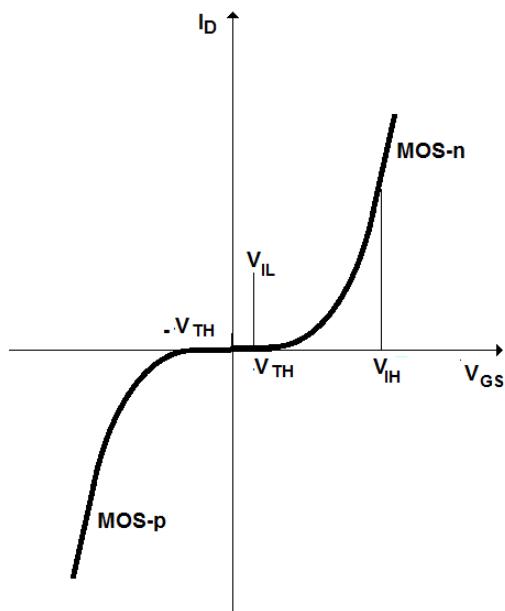


Figura 2.17 Caracteristica porții NU

Poarta se obține prin legarea în serie a două tranzistoare C-MOS (Figura 2.16), în care un MOS-n (M₁) are în sarcină un MOS-p (M₂). Se vor

CMOS VLSI

consideră nivele logice de tip L (low) pentru $V_L < V_{TH}$ și valori de tip H (High) în care $V_H \approx V_{DD}$. Caracteristica comună a celor două tranzistoare este prezentată în Figura 2.17, iar tabelul de operare logică poate fi urmărit în Figura 2.16. Pentru un potențial de intrare de tip L, tranzistorul M_1 este blocat iar M_2 este în conducție, deci potențialul de ieșire va fi de tip H. Dacă nivelul de intrare este H, M_1 este în conducție, M_2 este blocat iar potențialul ieșirii va fi de tip L.

Pentru analiza funcționării în curent continuu se vor examina caracteristicile curent-tensiune pentru cele două tipuri de tranzistoare NMOS și PMOS, din Figura 2.18 și Figura 2.19. Se consideră că la un tranzistor de tip N curentul intră prin drenă, iar la un tranzistor P curentuliese prin drenă.

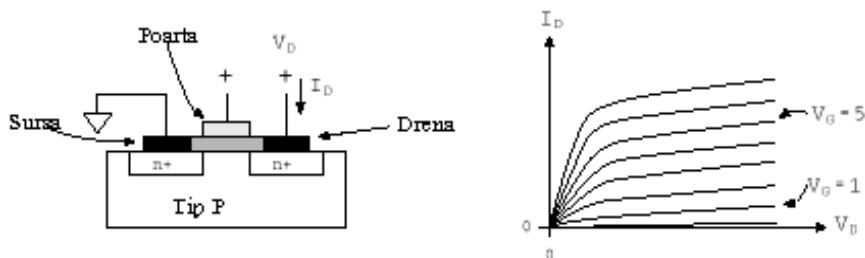


Figura 2.18 Caracteristica curent-tensiune pentru NMOS

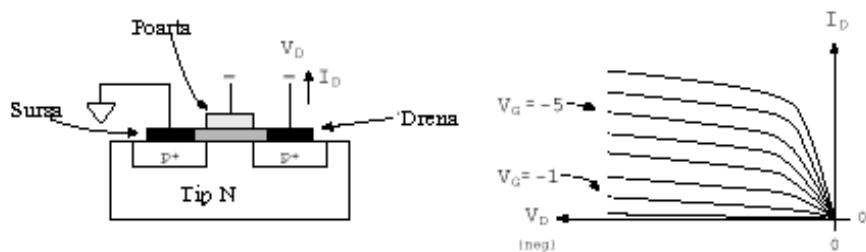


Figura 2.19 Caracteristica curent-tensiune pentru PMOS

În continuare se vor suprapune caracteristicile I-V ale celor două tipuri de dispozitive pentru a efectua o analiză a sarcinii. Plecând de la schema inversorului se constată că același curent ($I_{DSN} = I_{DSP}$) traversează, în același sens, cele două tranzistoare. Din acest motiv caracteristicile I-V ale celor două tranzistoare pot folosi aceeași axă pentru curenți. V_{DD} se poate scrie astfel: $V_{DD} = V_{DSN} + (-V_{DSP})$. Rescriind ecuația: $V_{DSP} = V_{DSN} - V_{DD}$ se

2. Tranzistoare MOS Si CMOS

observă că tensiunea drenei tranzistorului PMOS diferă de cea a tranzistorului NMOS cu o valoare egală cu V_{DD} , ceea ce permite utilizarea aceleiași axe pentru tensiuni, însă deplasată cu V_{DD} . Cele două familii de caracteristici I-V sunt prezentate în Figura 2.20.

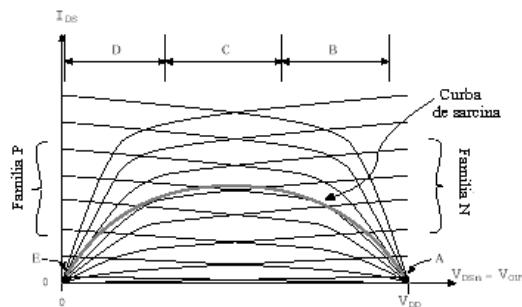


Figura 2.20 Caracteristici I-V

În continuare se urmărește găsirea caracteristicii de transfer. Pe curba de sarcină vor fi evidențiate două cazuri. Primul caz apare când $V_{IN} = V_{DD}$, care este marcat pe Figura 2.21 cu E. În acest punct tranzistorul P este blocat, iar tranzistorul N funcționează în regiunea liniară, ceea ce va plasa inversorul în regiunea E. Al doilea caz apare atunci când $V_{IN} = GND$, situație marcată cu A. În acest punct tranzistorul P funcționează în regiunea liniară, tranzistorul N este blocat, iar inversorul se găsește în regiunea A. Analiza poate fi continuată pentru toate punctele caracteristicii de transfer. Curba de sarcină, prezentată în Figura 2.21 va reprezenta soluția pentru familiile de caracteristici I-V ale celor două tipuri de tranzistoare. În această analiză, în curent continuu, s-a neglijat curentul, care ar ieși sau intra prin terminalul notat cu V_{OUT} .

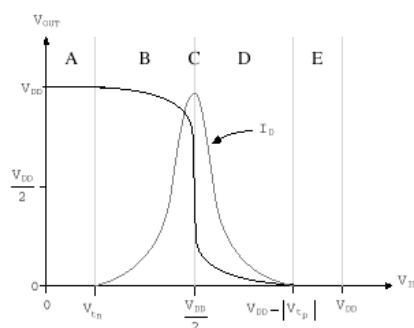


Figura 2.21 Caracteristica de transfer

CMOS VLSI

Caracteristica de transfer de mai sus are cinci regiuni distincte marcate prin A, ..., E, care vor fi analizate în continuare:

Regiunea A: Tranzistorul N este blocat, tranzistorul P este în regiunea liniară: $V_{IN} < V_{tn}$ și $V_{OUT} = V_{DD}$.

Regiunea B: Tranzistorul P rămâne în regiunea liniară, iar tranzistorul N trece în saturatie. Astfel, primul tranzistor se comportă ca o rezistență, iar cel de-al doilea operează ca o sursă de curent:

$$V_{tn} < V_{IN} < \frac{V_{DD}}{2}$$

$$\begin{aligned} V_{OUT} \\ = (V_{IN} - V_{tn}) \\ + \sqrt{\left(V_{IN} - V_{tp}\right)^2 - 2\left(V_{IN} - \frac{V_{DD}}{2} - V_{tp}\right)V_{DD} - \frac{\beta_n}{\beta_p}(V_{IN} - V_{tn})^2} \end{aligned}$$

Regiunea C: Expresia pentru V_{OUT} se poate stabili prin egalarea curentilor de drenă pentru cele două tranzistoare. Tranzistorul N rămâne în saturatie, iar tranzistorul P trece în saturatie. Panta mare a caracteristicii face ca ieșirea să varieze foarte mult la o mică modificare a intrării.

$$V_{IN} = \frac{V_{DD}}{2}$$

$$V_{IN} - V_{tn} < V_{OUT} < V_{IN} - V_{tn}$$

Regiunea D: Tranzistorul PMOS rămâne în saturatie, jucând rolul unei surse de curent, iar tranzistorul NMOS trece în regiunea liniară, comportându-se ca o rezistență.

$$\begin{aligned} \frac{V_{DD}}{2} < V_{IN} \leq V_{DD} - V_{tp} \\ V_{UT} = (V_{IN} - V_{tn}) + \sqrt{(V_{IN} - V_{tn})^2 - \frac{\beta_p}{\beta_n}(V_{IN} - V_{DD} - V_{tp})^2} \end{aligned}$$

Regiunea E: Tranzistorul NMOS rămâne în regiunea liniară, iar tranzistorul PMOS este blocat.

$$V_{IN} > V_{DD} + V_{tp}; V_{OUT} = 0$$

2. Tranzistoare MOS Si CMOS

Raportul $\frac{\beta_n}{\beta_p}$

Atunci când se proiectează un inversor este de dorit ca procesul de comutare să aibă loc la o valoare egală cu jumătatea tensiunii de alimentare. Astfel, comutarea trebuie să aibă loc la $V_{IN} = \frac{V_{DD}}{2}$. Punctul la care inversorul comută este dependent de valorile amplificărilor tranzistoarelor (β). Valoarea lui β este calculată cu ajutorul expresiei $\beta = \frac{(\mu\epsilon/t_{ox})*W}{L}$ unde:

- μ - este mobilitatea;
- ϵ - permitivitatea oxidului;
- t_{ox} – grosimea stratului de oxid;
- W – lățimea canalului;
- L – lungimea canalului.

Marginea de zgomot

Marginea de zgomot a inversorului poate fi stabilită pe baza caracteristicii de transfer. Marginea de zgomot reprezintă abaterea maximă a unui semnal față de valoarea normală, înainte ca el să fie recunoscut ca un alt semnal. Pentru calculul marginii de zgomot se folosesc următoarele notații, conform Figura 2.22.

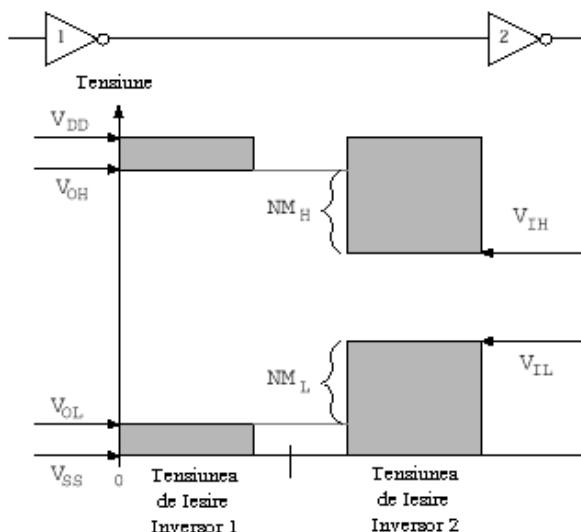


Figura 2.22 Calculul marginii de zgomot

CMOS VLSI

- V_{IL} – tensiunea cea mai mare, care poate fi considerată ca intrare de nivel coborât,
- V_{IH} – tensiunea cea mai coborâtă, care poate fi considerată intrare de nivel înalt,
- V_{OL} – tensiunea cea mai coborâtă, care poate fi generată ca ieșire,
- V_{OH} – tensiunea cea mai ridicată, care poate fi generată ca ieșire.

Valorile V_{IL} și V_{IH} apar acolo unde caracteristica de transfer are amplificarea egală cu unu.

Acestea sunt punctele de pe caracteristica de transfer unde zgometul are efect critic. Întrucât amplificarea în modul este egală cu unu, în aceste două puncte ieșirea se modifică în aceeași măsură ca și intrarea. Dacă tensiunea semnalului de zgomet este mai mică decât V_{IH} , pentru un semnal de nivel ridicat, ieșirea va comuta de la nivel coborât la nivel ridicat. Dacă tensiunea semnalului de zgomet este mai mare decât V_{IL} , pentru un semnal de nivel coborât, ieșirea va comuta de la nivel ridicat la nivel coborât, datorită amplificării supraunitare.

Marginile de zgomet pot fi definite astfel:

- $N_{ML} = V_{IL} - V_{OL}$ – pentru valoare coborâtă. Nivel logic 0
- $N_{MH} = V_{OH} - V_{IH}$ – pentru valoare ridicată. Nivel logic 1

Valoarea lui N_{ML} și N_{MH} este de aproximativ $2V$, pentru $\frac{\beta_n}{\beta_p} = 1$,
 $V_{tn} = |V_{tp}| = 1V$ și $V_{DD} = 5V$.

Este interesant de observat că marginea de zgomet va descrește pe măsură ce V_{DD} descrește. Acest fapt poate fi constatat în ecuația ce descrie cele două margini de zgomet. De asemenea, tensiunile de prag sunt deja foarte apropiate.

2. Tranzistoare MOS și CMOS

2.6. Poarta logică řI-NU

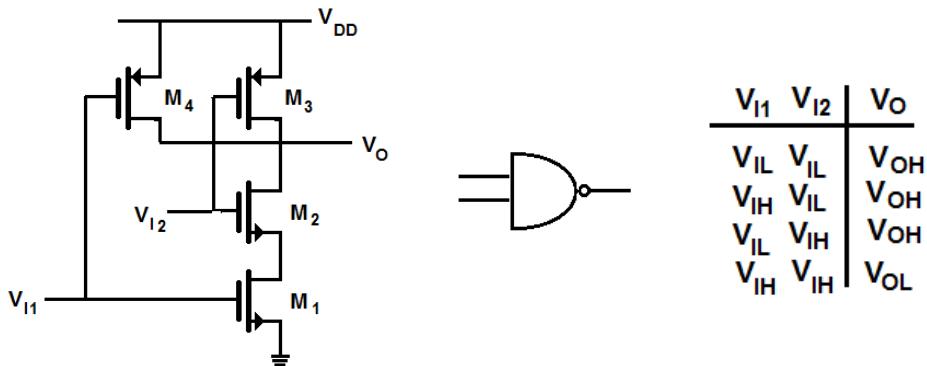


Figura 2.23 Poarta řI-NU

Configurația acestei porți este prezentată în Figura 2.23. Funcționarea porții se bazează pe același principiu ca acela discutat la poarta NU. Tabelul de funcționare logică este ilustrat pentru cele două variabile de intrare.

2.7. Poarta logică SAU-NU

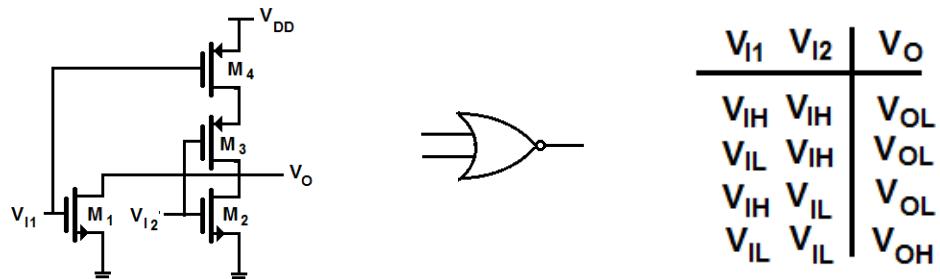


Figura 2.24 Poarta SAU-NU

Funcția logică SAU se obține prin legarea în paralel a tranzistoarelor MOS-n (Figura 2.24) iar complementarea este realizată prin legarea în sarcină, în serie, a tranzistoarelor MOS-p.

2.8. Porți logice MOS

În paragraful precedent s-au prezentat câteva soluții constructive de implementare ale porților logice folosind tranzistoare C-MOS, unele operând ca elemente inversoare iar celelalte jucând rolul de sarcină activă. În afara acestor metode, există posibilitatea implementării acestor porți folosind numai tranzistoare MOS-n sau MOS-p. Pentru exemplificare se va studia poarta logică de tip NU implementată cu circuite MOS-n (Figura 2.25).

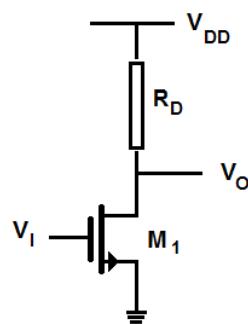


Figura 2.25 Poarta NU cu rezistență de sarcină

În sarcina tranzistorului M_1 se plasează rezistența de polarizare R_D . Caracteristica globală $V_o = f(V_I)$ este reprezentată în Figura 2.26.

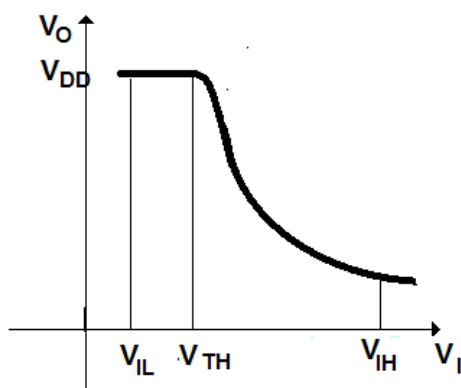


Figura 2.26 Caracteristica $V_o = f(V_I)$ pentru circuitul inversor

2. Tranzistoare MOS și CMOS

Pentru valori ale potențialului de intrare $V_{IL} < V_{TH}$, tranzistorul este blocat, potențialul de ieșire fiind $V_O \approx V_{DD}$. Pentru valori $V_I > V_{TH}$, tranzistorul intră în conducție, potențialul de ieșire se obține printr-un divizor rezistiv,

$$V_O = \frac{R_{ON}}{R_D + R_{ON}} V_{DD} \quad 2.17$$

unde R_{ON} este rezistența drenă-sursă a tranzistorului în conducție. Pentru a obține funcția logică NU, potențialul de ieșire trebuie să fie:

$$V_O = V_{OL} \quad 2.18$$

ceea ce implică condiția:

$$R_D \gg R_{ON} \quad 2.19$$

În cele mai multe tehnologii MOS este dificil să construim rezistențe suficiente de mari care să verifice inegalitatea de mai sus. Din acest motiv, acestea sunt înlocuite cu tranzistoare MOS aflate în regimuri de funcționare „de tip diodă”(Figura 2.27).

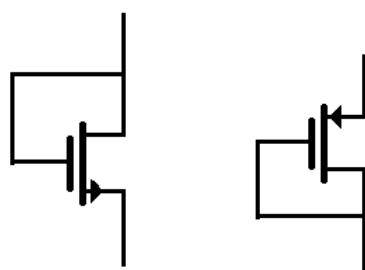


Figura 2.27 Tranzistoare în „regim de diodă”

În Figura 2.27 sunt prezentate două tranzistoare MOS-n și MOS-p la care conectarea directă poartă-drenă determină un astfel de regim de funcționare. O poartă logică NU cu o astfel de configurație de sarcină este prezentată în Figura 2.28.

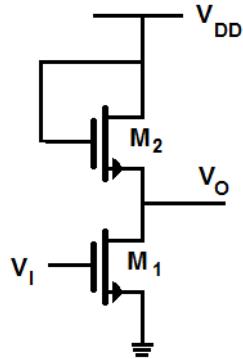


Figura 2.28 Poarta NU cu sarcina MOS-n în „regim de diodă”

Tranzistorul M_2 operează în regim de diodă cu o rezistență echivalentă:

$$R_{DS} \approx \frac{1}{g_m} \quad 2.20$$

Un efect similar se obține prin utilizarea unui tranzistor MOS-p în regim de diodă (Figura 2.29).

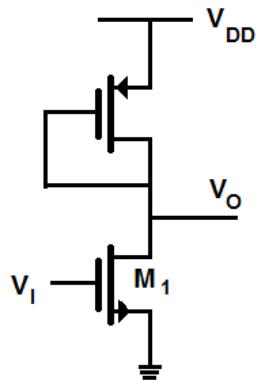


Figura 2.29 Poarta NU cu sarcina MOS-p în „regim de diodă”

Pentru obținerea unei mai bune adaptări de impedanță pe ieșire, de multe ori se preferă menținerea tranzistorului M_2 în saturare printr-o polarizare pe poartă aleasă în mod convenabil (Figura 2.30).

2. Tranzistoare MOS Si CMOS

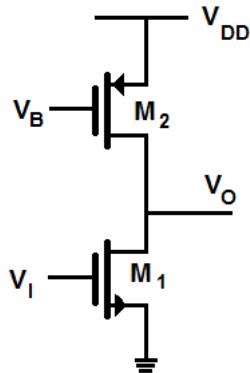


Figura 2.30 Poarta NU cu sarcina în regim de saturație

Pentru acest circuit, impedanța de ieșire pentru $V_I = V_{IH}$ (ambele tranzistoare sunt în saturație) va fi:

$$r_o = \frac{1}{g_m} (r_{o1} II r_{o2}) \quad 2.21$$

Pentru anumite regimuri funcționale se poate prefera utilizarea tranzistorului de sarcină în regim de triodă ceea ce va determina evaluarea corespunzătoare a rezistenței de ieșire.

2.9. Porți logice MOS complexe

Procedura analizată mai sus permite implementarea unor porți logice cu complexitate ridicată prin simpla conexiune în paralel sau serie a tranzistoarelor de intrare și folosind ca sarcină un tranzistor ce funcționează într-unul din regimurile considerate favorabile. În Figura 2.31, Figura 2.32 și Figura 2.33 sunt prezentate operatorii logici SAU-NU, ȘI-NU respectiv ȘI-SAU-NU.

CMOS VLSI

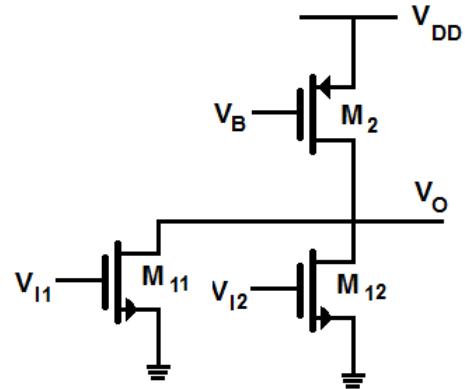


Figura 2.31 Poarta logică SAU-NU

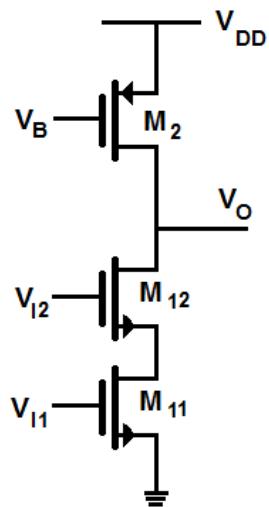


Figura 2.32 Poarta logică řI-NU

2. Tranzistoare MOS și CMOS

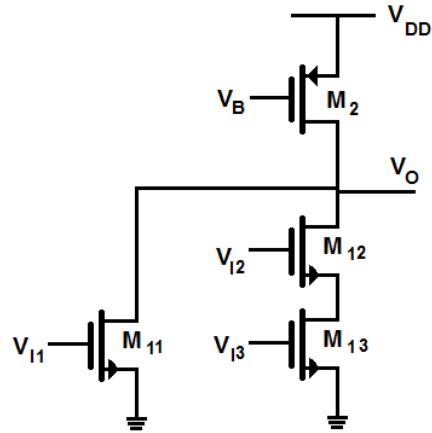


Figura 2.33 Poarta logică ȘI-SAU-NU

2.10. Regimul tranzistoriu al porților logice

Aprecierea regimurilor de funcționare a unor porți MOS sau C-MOS cere imperios determinarea timpilor de comutare, evaluarea fronturilor de comutare la semnale de tip treaptă aplicate pe intrare. În Figura 2.34 este prezentat circuitul experimental pentru testarea unei astfel de porți considerând pe intrare un generator ideal formator de semnal standard dreptunghiular. În Figura 2.35 poate fi urmărită alura semnalului pe ieșire pentru un semnal pe intrare neidealizat, similar celui oferit de alte porți ce operează ca sursă.

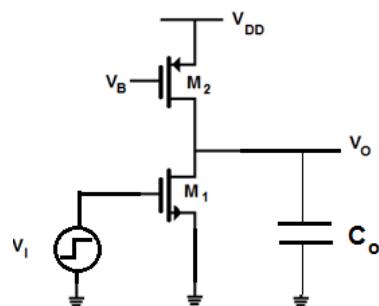


Figura 2.34 Circuitul de comutație al unei porți NU

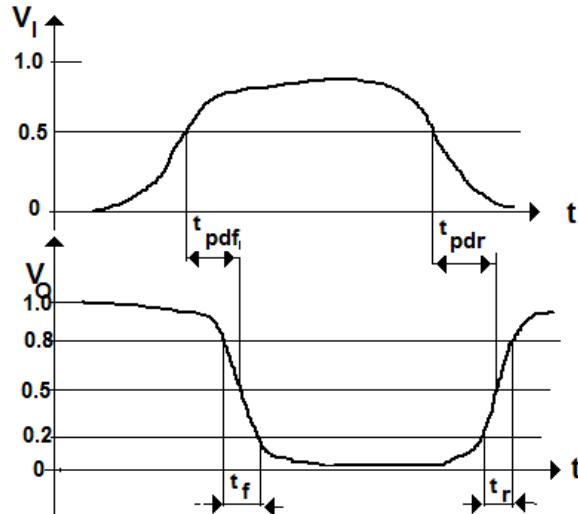


Figura 2.35 Nivelele critice de semnal și timpii de comutare

Conform standardelor internaționale, răspunsul în timp și fronturile de semnal se evaluatează în raport cu nivele de semnal de referință. Sunt măsurate astfel fronturile t_f și t_r și se evaluatează timpul de propagare mediu ca:

$$t_c = \frac{1}{2}(t_f + t_r) \quad 2.22$$

Evaluarea acestor timpi se obține pe baza capacităților parazite ce apar între electrozi de control G, D, S și a rezistențelor create în diferitele regimuri de funcționare ale tranzistorului. În Figura 2.36 sunt prezentate capacitățile parazite C_{DG} , C_{GS} și capacitatea de sarcină C_o . Capacitatea între drenă și sursă este, în mod curent, neglijabilă. Valorile acestor capacități depind de suprafața tehnologică de implementare. Considerând lungimea canalului constantă, aceste capacități vor fi direct proporționale cu lățimea w a ariei de implementare. De asemenea, conform tehnologiei utilizate, rezistența asociată este invers proporțională cu lățimea w .

2. Tranzistoare MOS Si CMOS

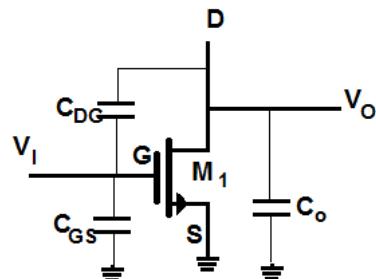


Figura 2.36 Capacitățile unui tranzistor MOS-n

Evaluarea timpilor de comutare se obține prin calculul constantelor de timp:

$$\begin{aligned}\tau_f &= R_{on} C_{on} \\ \tau_r &= R_{off} C_{off}\end{aligned}\quad 2.23$$

unde R_{on} , C_{on} , R_{off} , C_{off} sunt rezistențele și capacitatările echivalente în diferite regimuri funcționale (ele sunt foarte diferite în funcție de regimul de funcționare).

2.11. Circuite de memorie capacitive

În memoriile de mare capacitate folosite în tehnologiile VLSI, sunt utilizate în mod frecvent capacități ca elemente de memorie cărora li se asociază circuite cu tranzistoare cu rol de comutator, ce asigură procesul de încărcare sau descărcare al capacității. Procesul este ilustrat în Figura 2.37. Elementul de memorie este reprezentat de capacitatea C , iar procesul de încărcare se obține prin închiderea comutatorului K . Acest comutator este înlocuit cu un tranzistor MOS-n controlat pe poartă printr-un semnal de clock.

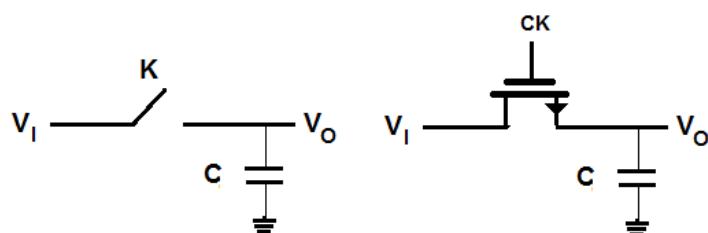


Figura 2.37 Tranzistor MOS-n în regim de comutator

Procesul este ilustrat în Figura 2.38 și Figura 2.39. În Figura 2.38 capacitatea C este încărcată la un potențial V_{DD} ce servește ca sursă de drenă. Prin aplicarea unui potențial pe poartă (clock), tranzistorul conduce în regim de saturatie (initial) și capacitatea C se descarcă. Ulterior, când potențialul de polarizare al drenei scade, tranzistorul conduce în regim de triodă și descărcarea capacității C continuă până ce atinge potențial 0. În Figura 2.39 se produce încărcarea capacității C la un potențial $+V_A$. În acest caz, borna conectată la potențialul înalt V_A reprezintă drena tranzistorului, iar borna conectată la capacitate va fi sursa. Procesul de încărcare se va realiza exponențial, tranzistorul funcționând ca simplu element rezistiv. Timpul de încărcare sau descărcare al acestui circuit definește viteza de comutare, parametru esențial pentru memoriile de mare viteză.

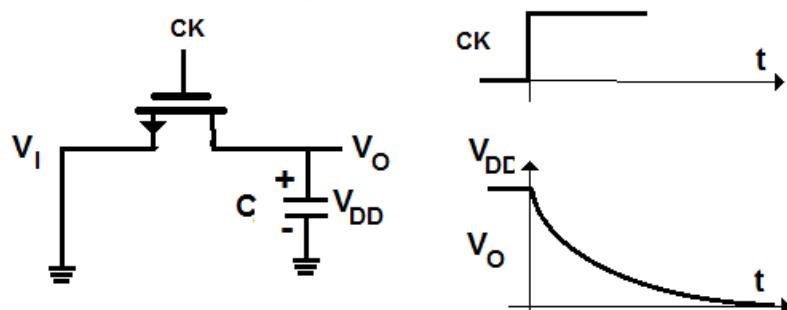


Figura 2.38 Procesul de descărcare

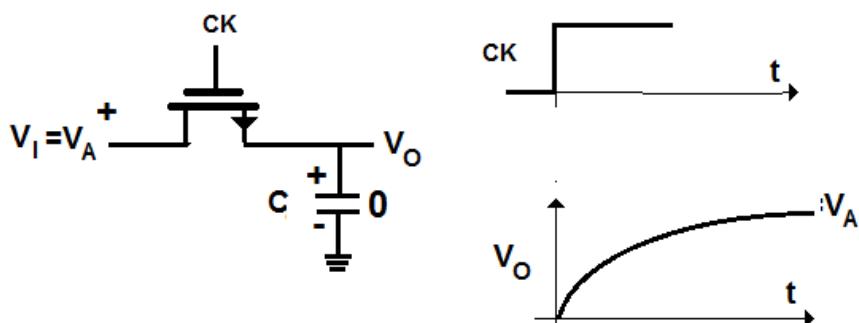


Figura 2.39 Procesul de încărcare

2. Tranzistoare MOS Si CMOS

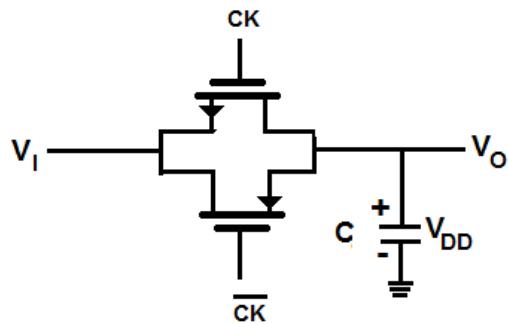


Figura 2.40 Comutatorul realizat cu două tranzistoare complementare

O soluție tehnologică mult mai bună, cu performanțe certe în ceea ce privește viteza de comutare, este prezentată în Figura 2.40. În acest caz, funcția de comutare este realizată în ambele sensuri prin utilizarea a două tranzistoare complementare, MOS-n, MOS-p, acestea funcționând pe valorile complementate ale ceasului.

3. PROIECTAREA PORȚILOR LOGICE ÎN VLSI

3.1. Proiectarea măștilor / şabloanelor

Schema unui inversor CMOS a fost prezentată într-un paragraf anterior. În continuare se vor examina, din punct de vedere geometric, şabloanele necesare pentru realizarea unui inversor CMOS, cu ajutorul unui proces de fabricație plecând de la un substrat de tip P. Fabricarea dispozitivelor VLSI implică crearea mai multor straturi din materiale cu diverse proprietăți electrice, depuse unul peste celălalt. În figurile de mai jos se prezintă:

- codul culorilor folosit pentru identificarea diverselor straturi (Tabel 3.1)
- geometria simplificată a unui inversor CMOS, la nivelul diverselor straturi (Figura 3.2)
- secțiuni transversale prin structura inversorului CMOS (Figura 3.3)

	Cod	Nume	Culoare	Simbol
1.	cn	N well (insula N)	dreptunghi cu laturile verzi	
2.	dn	N+ difuzie	dreptunghi sau pătrat complet verde	
3.	dp	P+ difuzie	dreptunghi hașurat cu verde	
4.	co	tăietură de contact	dreptunghi complet alb	
5.	m1	metal 1	dreptunghi complet albastru	
6.	po	poli	dreptunghi colorat roșu	
7.	Vi	via	dreptunghi alb	
8.	m2	metal 2	dreptunghi colorat cu albastru și hașurat	

Tabel 3.1 Straturile procesului CMOS

3. Proiectarea porților logice În VLSI

3.1.1. Proiectarea unui inversor

Inversorul reprezintă unitatea de bază pentru orice circuit digital.

In	Out
1	0
0	1

Tabel 3.2 Tabela de adevăr pentru inversor

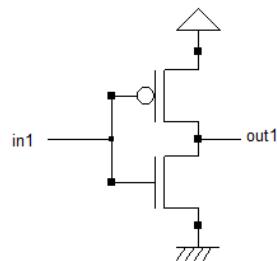


Figura 3.1 Implementare folosind tranzistoare pmos și nmos

Substratul este de tip P. Pentru crearea tranzistorului de tip P se realizează, printr-un proces de difuzie, o insulă N (Nwell). Zonele în care vor fi plasate tranzistoarele de tip N și P vor fi dopate prin difuzie cu impurități de tip N (N^+ Diffusion) și P (P^+ Diffusion). Separarea celor două straturi de metal1 și metal2 se efectuează printr-un strat de dioxid de siliciu. Zonele în care cele două straturi se suprapun poartă numele de Via m1/m2.

În Figura 3.2, tranzistorul din dreapta, de tip PMOS, este realizat pe o insulă de tip N, în timp ce tranzistorul din stânga, de tip N, este creat pe substratul de tip P.

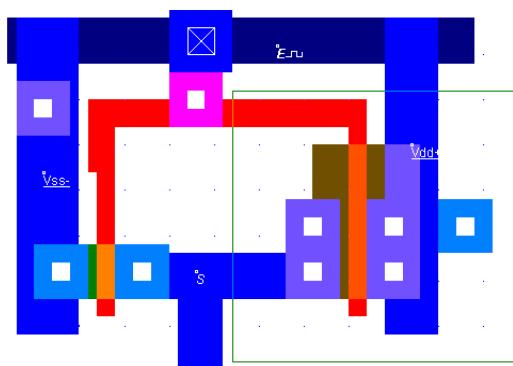


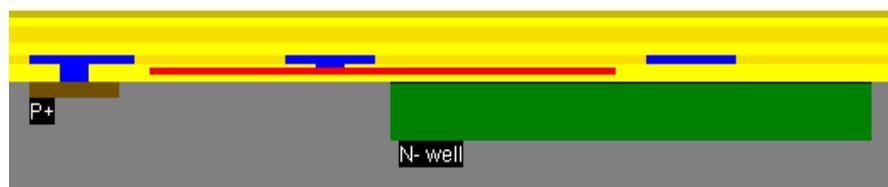
Figura 3.2 Geometria inversorului CMOS

CMOS VLSI

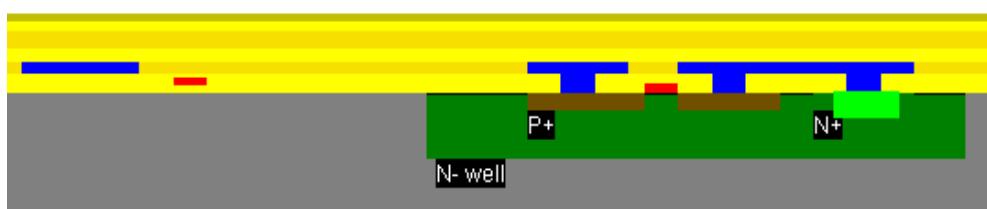
La extrema dreaptă se află un strat de metal1, care asigură tensiunea de alimentare V_{DD} . Traseul de metal1, de la limita din stânga realizează conexiunea la masa, GND. Intrarea V_{IN} este realizată prin traseul din siliciu policristalin, care traversează zonele porților celor două tranzistoare și care este conectat la stratul de metal1, acesta din urmă fiind conectat la stratul metal2.



a. Secțiune transversală prin structura inversorului, la nivelul metal2



b. Secțiune transversală prin structura inversorului la nivelul contactului metal1-siliciu policristalin.



c. Secțiune transversală prin structura inversorului la nivelul canalului tranzistorului P. Se poate observa în partea din stânga conectarea insulei N, la traseul metal1 (V_{DD}) prin zona puternic difuzată N^+

3. Proiectarea porților logice În VLSI



d. Secțiune transversală prin structura inversorului la nivelul canalelor celor două tranzistoare N și P. Se poate observa în stânga conexiunea substratului la GND prin zona de difuzie P⁺.

Figura 3.3 Secțiuni transversale prin structura inversorului CMOS

Pentru realizarea unui inversor trebuie să se creeze, pe substrat, mai multe straturi de materiale cu proprietăți electrice diferite. Modurile în care vor fi depuse aceste materiale pe substrat vor fi discutate într-un alt capitol. Deocamdată se pleacă de la premisa că ele pot fi depuse pe substrat, în zonele specificate de către proiectant. Crearea unui inversor CMOS presupune următorii pași:

- se folosește un substrat de siliciu ușor dopat P, pe care se vor depune celelalte straturi;
- se crește un strat de dioxid de siliciu și se creează tăieturile necesare în acesta;
- se creează o insulă N în substrat, prin doparea cu ioni de P sau As, insula N fiind necesară pentru realizarea tranzistorului P;
- se crește un strat subțire de dioxid de siliciu, având rolul de izolator, în zonele în care se vor crea tranzistoarele P și N;
- se plasează un strat de siliciu policristalin peste stratul subțire de dioxid, pentru a forma porțile celor două tranzistoare;
- se implantează/difuzează o regiune N⁺ în substratul P, pentru a forma sursa și drena tranzistorului de tip N și, în mod asemănător, se implantează/difuzează o regiune P⁺ în insula N, pentru a forma sursa și drena tranzistorului P;
- peste întreaga structură se crește un strat de oxid și se deschid în acesta tăieturile de contact, pentru ca stratul de metall1, care se va depune, în continuare, să realizeze conexiunile cu diversele straturi, în vederea: aplicării tensiunii de alimentare, conectării la masă, la intrare și ieșire;

CMOS VLSI

- în cazul în care procesul de fabricație permite crearea celui de-al doilea strat de metal se va depune un strat de oxid pe întreaga suprafață și se vor crea tăieturi de contact în zonele (via) în care metal2 și metal1 trebuie să vină în contact.

Pentru crearea formelor strukturilor menționate mai sus se utilizează procese litografice complexe, care au fost deja discutate.

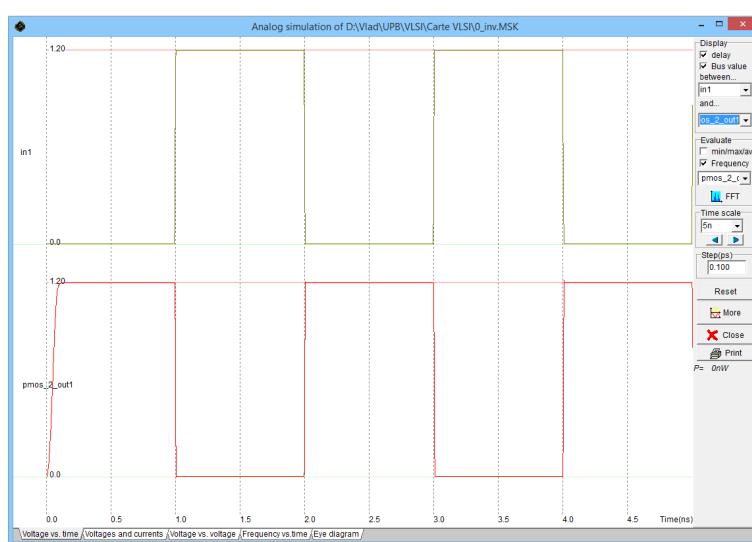


Figura 3.4 Simularea inversorului (V_{in}/t și V_{out}/t)

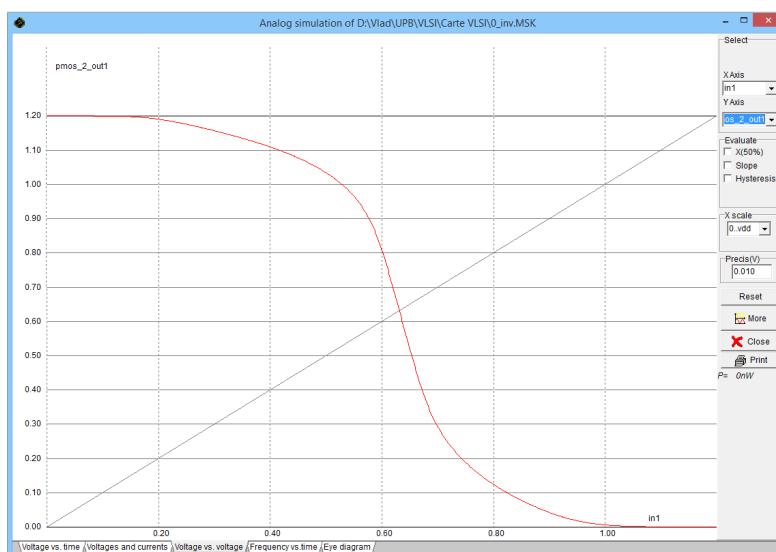


Figura 3.5 Simularea inversorului (V_{out}/V_{in})

3. Proiectarea porților logice În VLSI

3.1.2. Reguli de proiectare

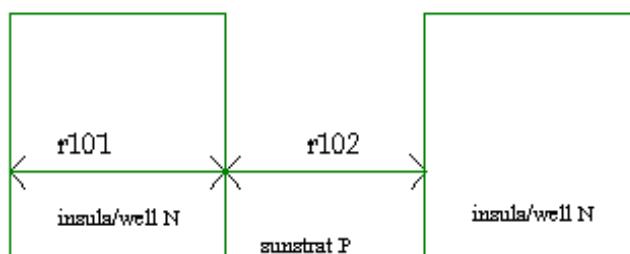
Când se realizează desenele şabloanelor unor circuite integrate se urmăreşte ca acestea să ocupe o suprafaţă cât mai mică. Procesele de fabricaţie impun însă o serie de restricţii referitoare la dimensiunile minime ale unor trasee, cât şi la distanţele între traseele din acelaşi material sau din materiale diferite, chiar dacă se află pe niveluri diferite.

Plecând de la elementele ce caracterizează diferite procese de fabricaţie în termenii dimensiunilor şi distanţelor minime au fost generate o serie de reguli de proiectare. Aceste reguli reprezintă un ghid în proiectare, prin care se urmăreşte reducerea ariei ocupate de un circuit, garantându-se însă funcţionarea corectă.

Specialiştii au căutat să lege regulile de proiectare de un factor ce poate caracteriza unele procesele tehnologice din acest domeniu. Acest factor, care poartă numele de rezoluţie a procesului, este notat cu λ şi este influenţat de o serie de factori legaţi de proces: precizia de aliniere a măştilor, precizia controlului de corodare ş.a. De exemplu, în cazul procesului ATMEL-ES2 2-metal CMOS, rezoluţia este egală cu $0,8 \mu m$.

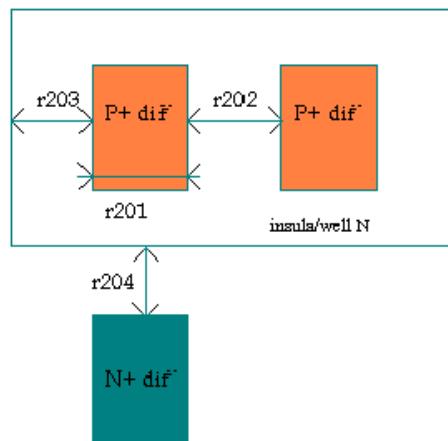
Regulile de proiectare pot fi exprimate în valori absolute (μm) sau sub formă elativă în raport cu rezoluţia λ . Astfel lăţimea minimă a unui traseu de siliciu policristalin este de $1,6 \mu m$, în cazul procesului $0,8 \mu m$ CMOS. În forma relativă, lăţimea minimă a unui traseu de siliciu policristalin va fi egală cu 2. În continuare vor fi prezentate egulile de proiectare în forma relativă, pentru procesul ATMEL-ES2 2-metal $0,8 \mu m$ CMOS.

Insula N



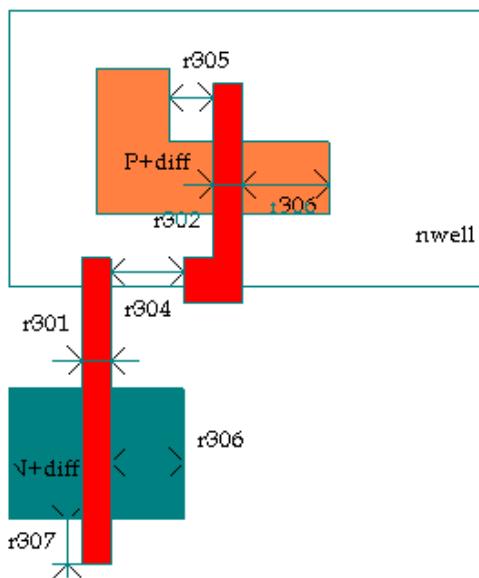
- r101 dimensiunea minimă a insulei: 12
- r102 distanța minimă între insule: 12

Difuzie



- r_{201} dimensiunea minimă a zonei de difuzie: 4
- r_{202} distanța minimă între două zone de difuzie: 4
- r_{203} extensia insulei față de difuzie: 6
- r_{204} distanța minimă între o zonă de difuzie și o insulă: 6

Siliciu policristalin

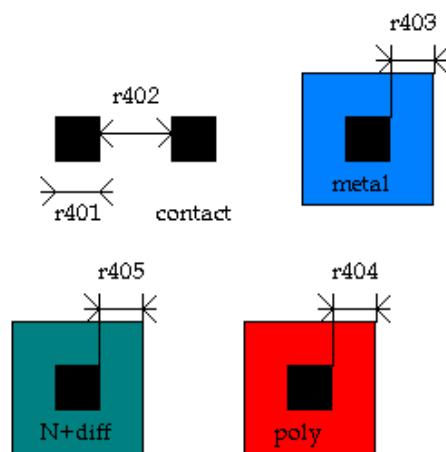


- r_{301} lățimea traseului de siliciu policristalin: 2
- r_{302} lățimea porții de siliciu policristalin pe difuziune P^+ : 2
- r_{303} lățimea porții de siliciu policristalin pe difuziune N^+ : 2

3. Proiectarea porților logice În VLSI

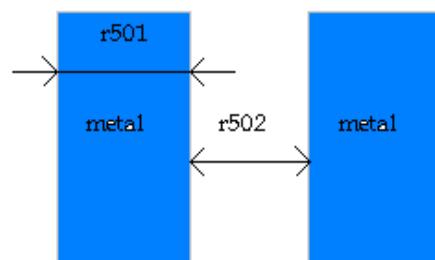
- r304 distanță minimă între două trasee de siliciu policristalin: 3
- r305 distanță minimă între un traseu de siliciu policristalin și un strat de difuzie: 2
- r306 extensia unui strat de difuzie în raport cu stratul de siliciu policristalin: 4
- r306 extensia unui strat de siliciu policristalin în raport cu stratul de difuzie: 2

Contact



- r401 lățimea contactului: 2
- r402 distanță minimă între două contacte: 3
- r403 extensia metalului față de tăietura de contact: 2
- r404 extensia siliciului policristalin față de tăietura de contact: 2
- r405 extensia difuziei față de tăietura de contact: 2

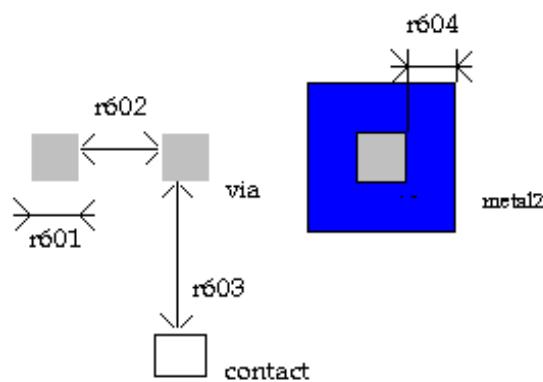
Metal



CMOS VLSI

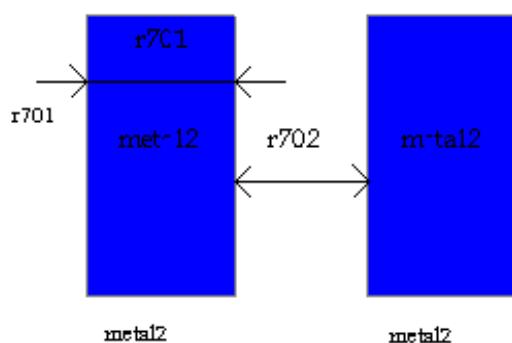
- r501 dimensiunea minimă a metalului1: 3
- r302 distanța minimă între două zone de metal1: 3

Via



- r601 lățimea zonei via: 3
- r602 distanța minimă între două zone via: 3
- r603 distanta minima între via și contact: 3
- r604 extensia metal1 peste via: 2
- r605 extensia metal2 peste via: 2

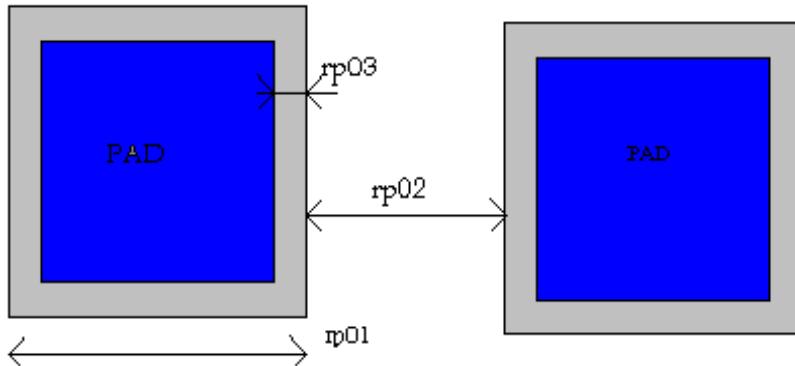
Metal 2



- r701 dimensiunea minimă a zonei de metal2: 5
- r702 distanța minimă între două zone de metal2: 5

3. Proiectarea porților logice În VLSI

Plotul



- rp01 dimensiunea minimă $100\mu m$
- rp02 distanță minimă între două ploturi: $100 \mu m$
- rp03 deschiderea în pasivizare față de via: $5 \mu m$
- rp04 deschiderea în pasivizare față de metale: $5 \mu m$
- rp05 distanță minimă între plot și zone active fără legătură cu plotul: $20 \mu m$

3.2. NAND2

Se consideră un circuit NAND cu două intrări in1, in2 și ieșirea out1. Tabela de adevăr corespunzătoare circuitului este:

in1	in2	out1
0	0	1
0	1	1
1	0	1
1	1	0

Tabel 3.3 Tabela de adevăr pentru NAND2

Tranzistoarele PMOS au sursele conectate la VDD și drenele la VOUT (ieșirea out1). Tranzistoarele NMOS sunt conectate în serie, cel de la nivelul cel mai de jos având sursa conectată la masă, iar cel de la nivelul cel mai de sus drena conectată la VOUT (ieșirea out1).

CMOS VLSI

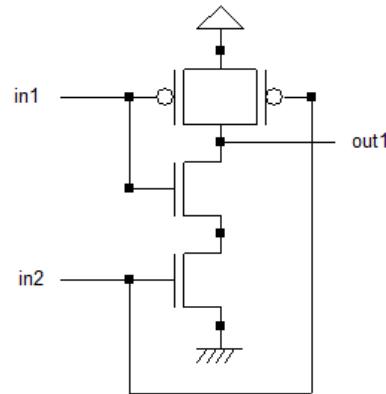


Figura 3.6 Implementarea NAND2 folosind tranzistoare pmos si nmos

Figura 3.7 reprezintă desenul măștilor porții NAND cu două intrări. Această celulă posedă câteva trăsături notabile. Mai întâi, traseele VDD și GND formează câte o bară care se extinde transversal la nivelul superior și la nivelul inferior. Aceasta permite plasarea alăturată a unor asemenea celule. În acest mod se vor forma liniile de alimentare și de masă, care se extind de la stânga la dreapta, fără alte conexiuni externe. Ca rezultat, dimensiunea liniei și spațiile între ele trebuie standardizate.

Figura 3.8 prezintă simularea layout-ului propus pentru circuitul NAND cu două intrări.

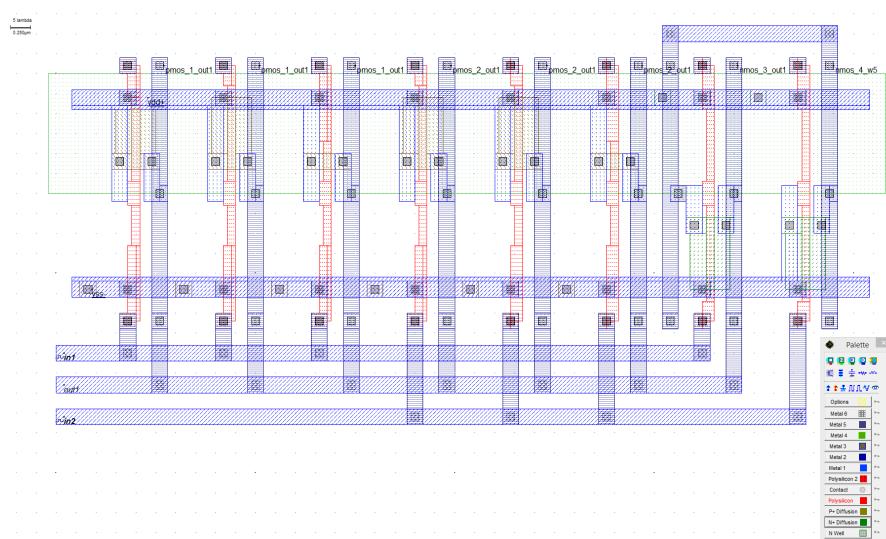


Figura 3.7 Layout-ul circuitului NAND2

3. Proiectarea porților logice În VLSI

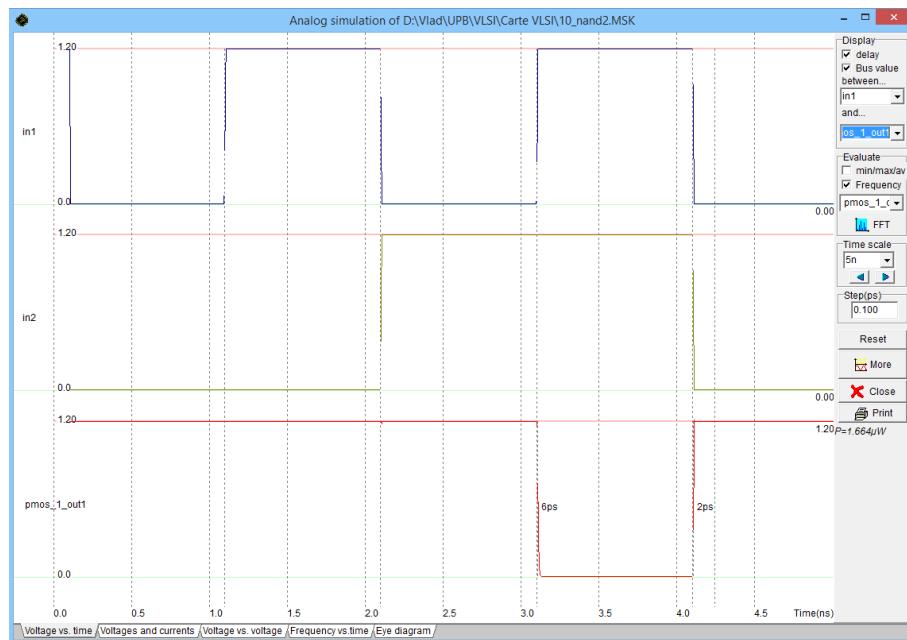


Figura 3.8 Simularea NAND2

3.3. NAND3

Pentru circuitul NAND3, se consideră trei intrări: in1, in2, in3 și o singură ieșire out1, având tabelul de adevăr conform Tabel 3.4.

in1	in2	in3	out1
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Tabel 3.4 Tabelă de adevăr pentru NAND3

Circuitul NAND3 este o extensie a circuitului prezentat în secțiunea anterioară (NAND2), cu observația apărând un nou tranzistor nmos cuplat în

CMOS VLSI

serie și un nou tranzistor pmos cuplat în paralel. Schema circuitului de implementarează funcția NAND3 este prezentată în Figura 3.9.

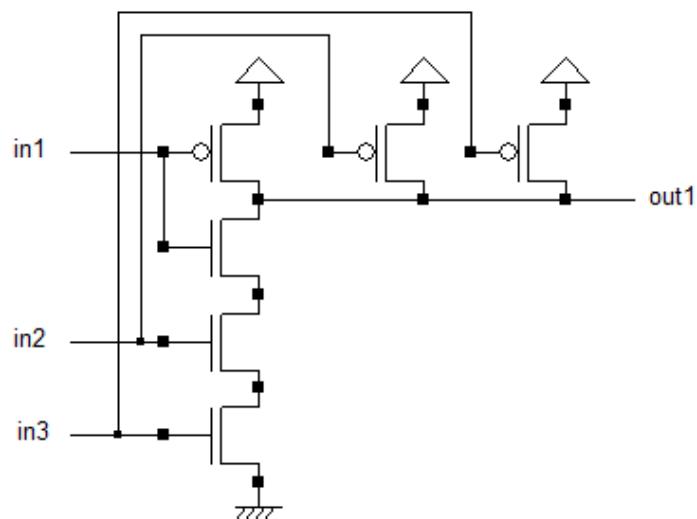


Figura 3.9 Implementarea circuitului NAND3 folosind tranzistoare nmos și pmos

Figura 3.10 reprezintă desenul măștilor porții NAND cu trei intrări, în timp ce simularea circuitului este descrisă de Figura 3.11.

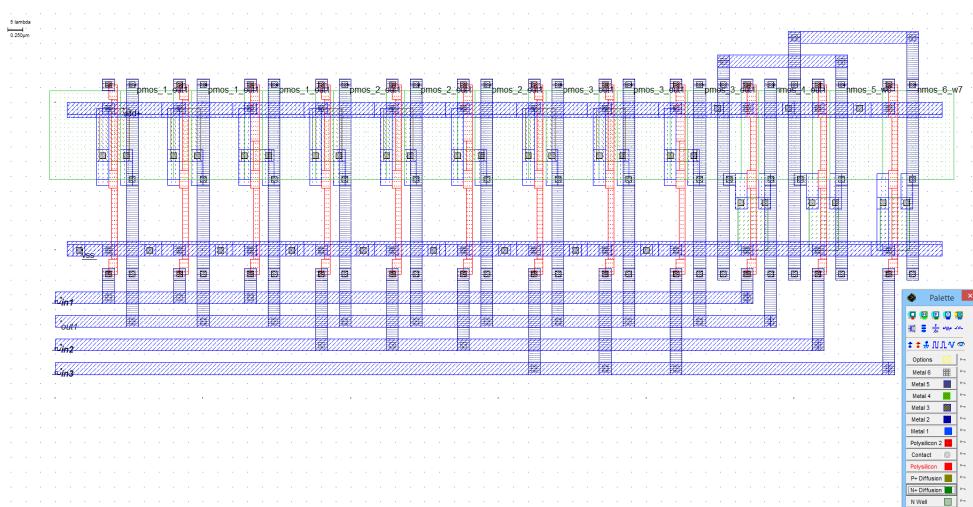


Figura 3.10 Layout-ul circuitului NAND3

3. Proiectarea porților logice În VLSI

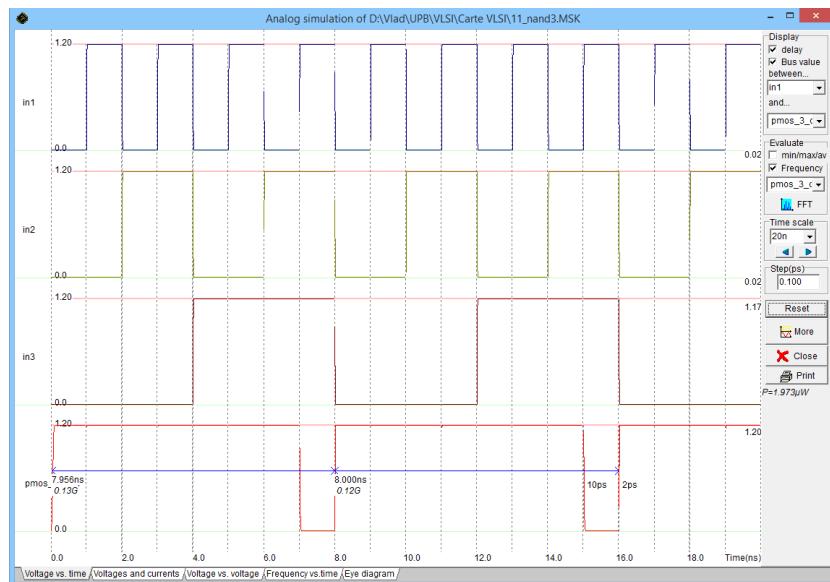


Figura 3.11 Simularea circuitului NAND3

3.4. AND2

Se consideră un circuit NAND cu două intrări in1, in2 și ieșirea out1. Tabela de adevăr corespunzătoare circuitului este:

in1	in2	out1
0	0	0
0	1	0
1	0	0
1	1	1

Tabel 3.5 Tabela de adevăr pentru NAND2

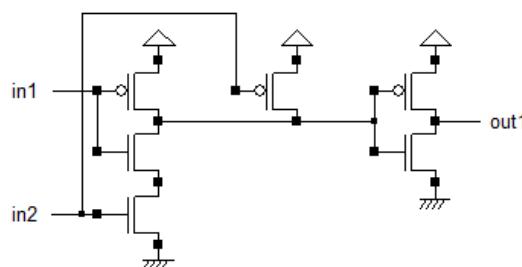


Figura 3.12 Implementarea circuitului AND2 folosind tranzistoare nmos și pmos

CMOS VLSI

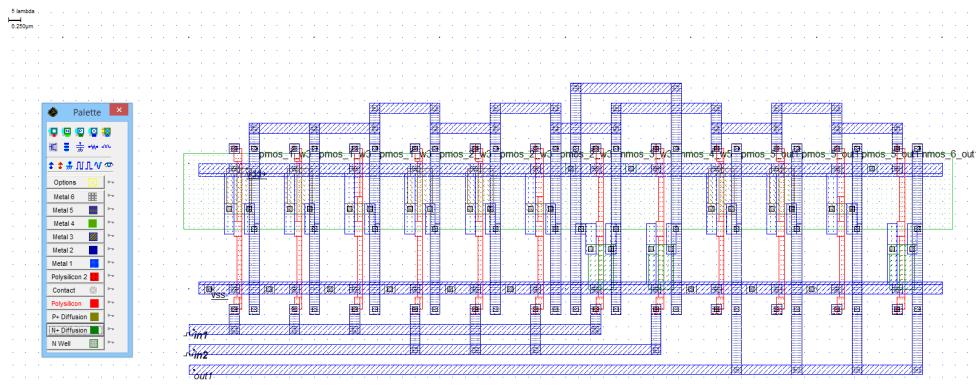


Figura 3.13 Layout-ul circuitului AND2

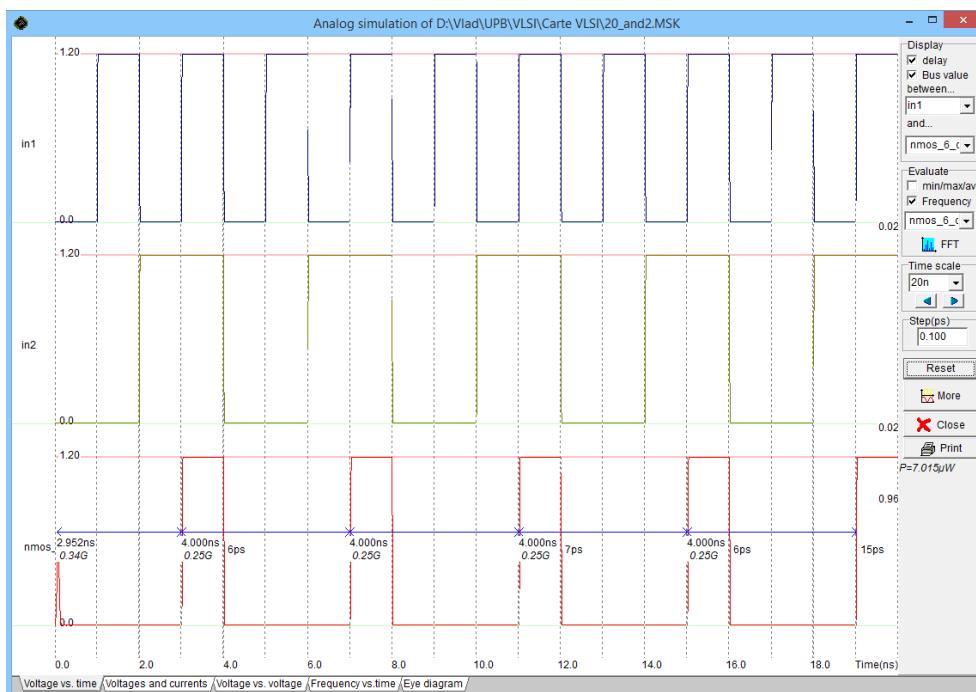


Figura 3.14 Simularea circuitului AND2

3.5. AND3

Pentru circuitul AND3, se consideră trei intrări: in1, in2, in3 și o singură ieșire out1.

3. Proiectarea porților logice În VLSI

in1	in2	in3	out1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Tabel 3.6 Tabelă de adevăr pentru NAND3

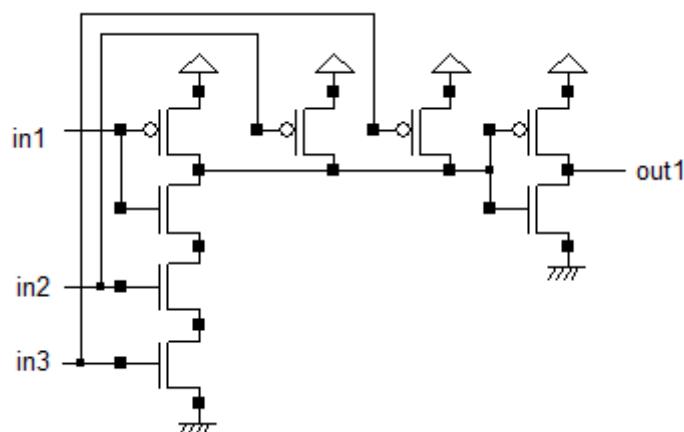


Figura 3.15 Implementarea circuitului AND3 folosind tranzistoare nmos și pmos

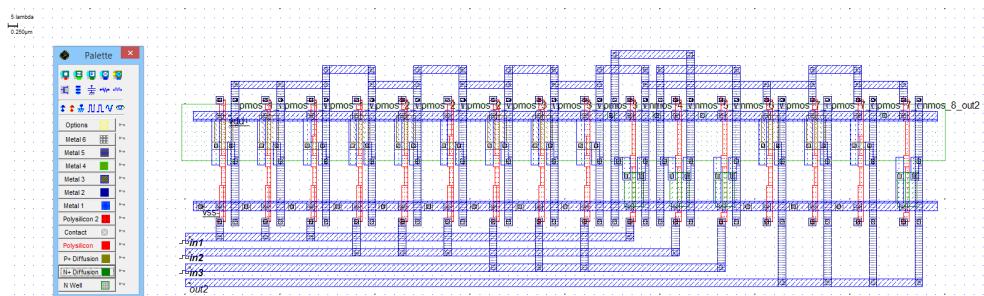


Fig. 3.1 Layout-ul circuitului AND3

CMOS VLSI

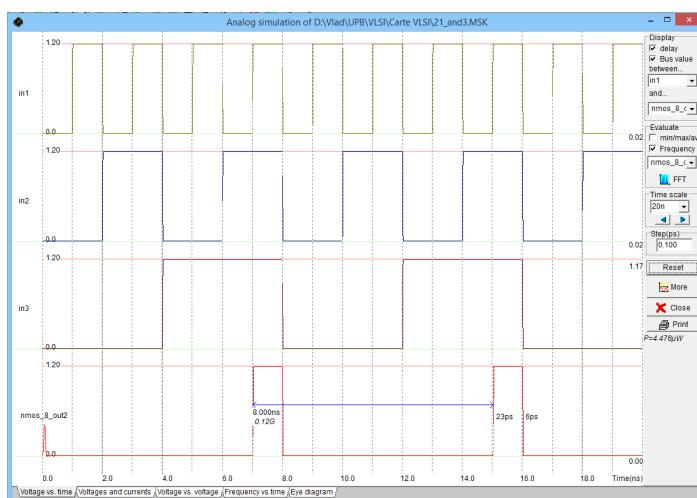


Figura 3.16 Simularea circuitului AND3

3.6. NOR2

Se consideră un circuit NOR cu două intrări in1, in2 și ieșirea out1. Tabela de adevăr corespunzătoare circuitului este:

in1	in2	out1
0	0	1
0	1	0
1	0	0
1	1	0

Tabel 3.7 Tabela de adevăr pentru NOR2

În cazul circuitelor NOR, tranzistoarele PMOS sunt conectate în serie iar cele NMOS sunt conectate în paralel.

Desenul măștilor pentru poarta NOR cu două intrări prezintă aceleași caracteristici ca și cel pentru poarta NAND. Trebuie remarcat faptul că tranzistoarele de tip P, fiind în serie, trebuie să aibă dimensiuni mai mari decât tranzistoarele de tip N, pentru a asigura tempi de creștere și cădere egali.

Un alt aspect se referă la faptul că schema circuitului la nivelul tranzistoarelor reprezintă amplasarea relativă a acestora din urmă. Desenele

3. Proiectarea porților logice În VLSI

reale ale măștilor încearcă, de regulă, să minimizeze aria ocupată. În acest scop se utilizează conexiuni și orientări ale tranzistoarelor cât mai convenabile.

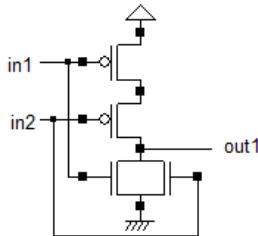


Figura 3.17 Implementarea circuitului NOR2 folosind tranzistoare nmos și pmos

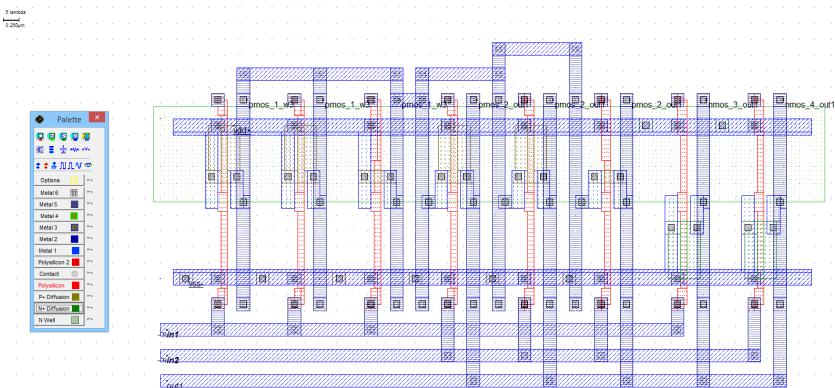


Figura 3.18 Layout-ul circuitului NOR2

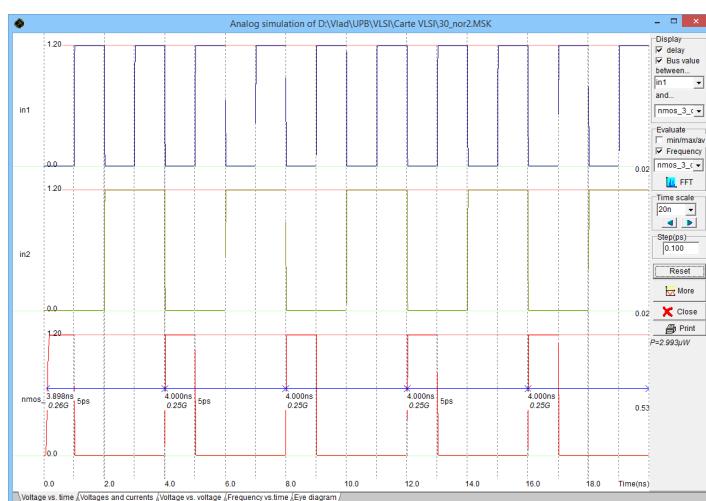


Figura 3.19 Simularea circuitului NOR2

3.7. NOR3

Pentru circuitul NOR3, se consideră trei intrări: in1, in2, in3 și o singură ieșire out1.

in1	in2	in3	out1
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Tabel 3.8 Tabelă de adevăr pentru NOR3

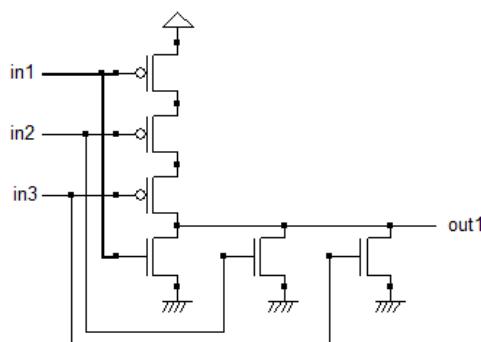


Figura 3.20 Implementarea circuitului NOR3 folosind tranzistoare nmos și pmos

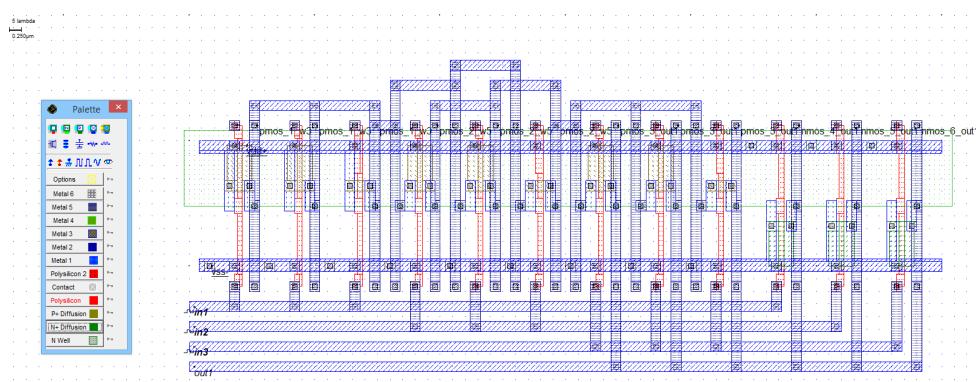


Figura 3.21 Layout-ul circuitului NOR3

3. Proiectarea porților logice În VLSI

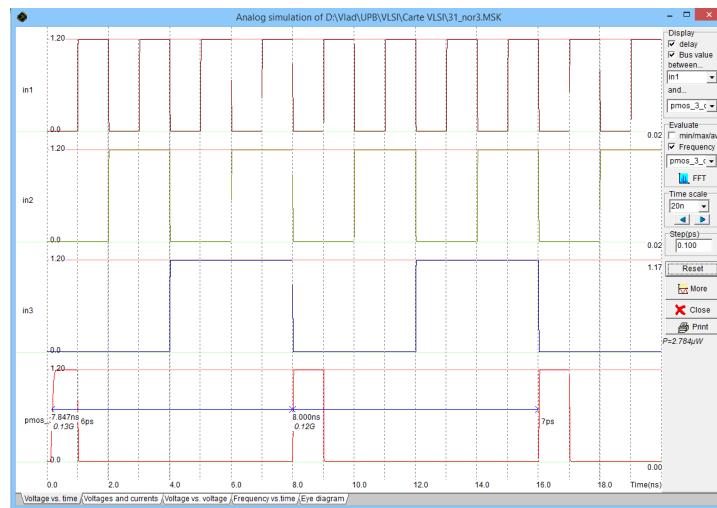


Figura 3.22 Simularea circuitului NOR3

3.8. OR2

Se consideră un circuit OR cu două intrări in1, in2 și ieșirea out1. Tabela de adevăr corespunzătoare circuitului este:

in1	in2	out1
0	0	0
0	1	1
1	0	1
1	1	1

Tabel 3.9 Tabela de adevăr pentru OR2

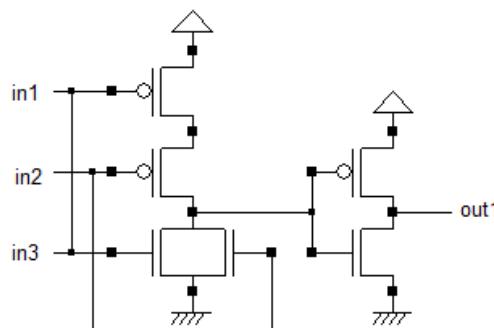


Figura 3.23 Implementarea circuitului OR2 folosind tranzistoare nmos și pmos

CMOS VLSI

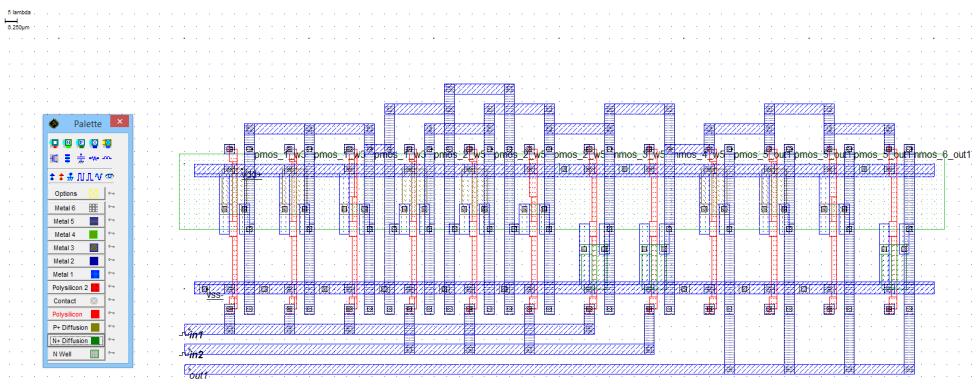


Fig. 3.2 Layout-ul circuitului OR2

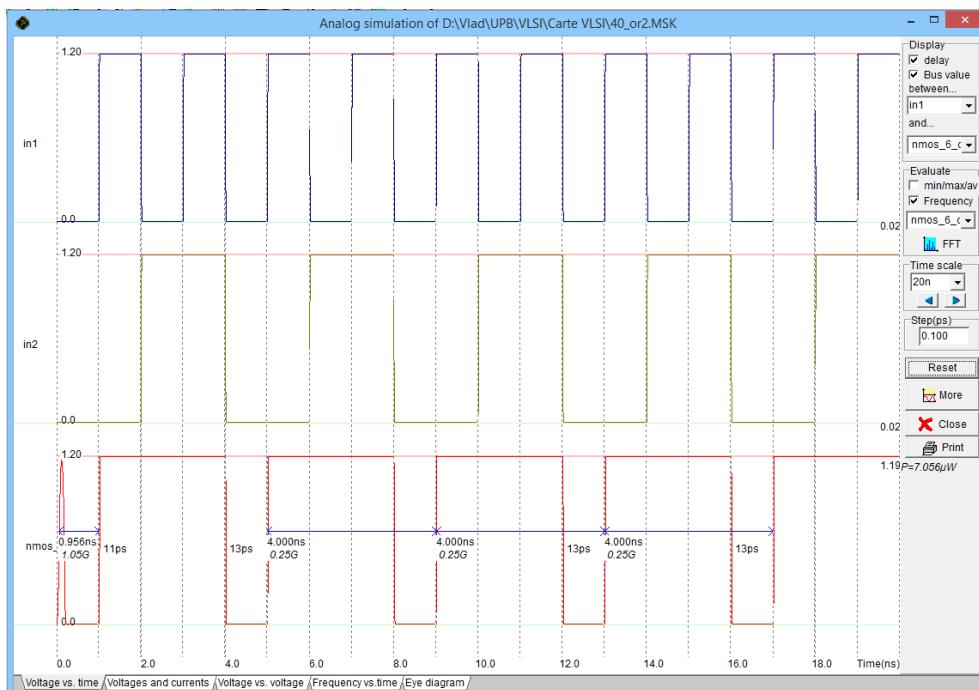


Figura 3.24 Simularea circuitului OR2

3.9. OR3

Pentru circuitul OR3, se consideră trei intrări: in1, in2, in3 și o singură ieșire out1.

3. Proiectarea porților logice În VLSI

in1	in2	in3	out1
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Tabel 3.10 Tabelă de adevăr pentru OR3

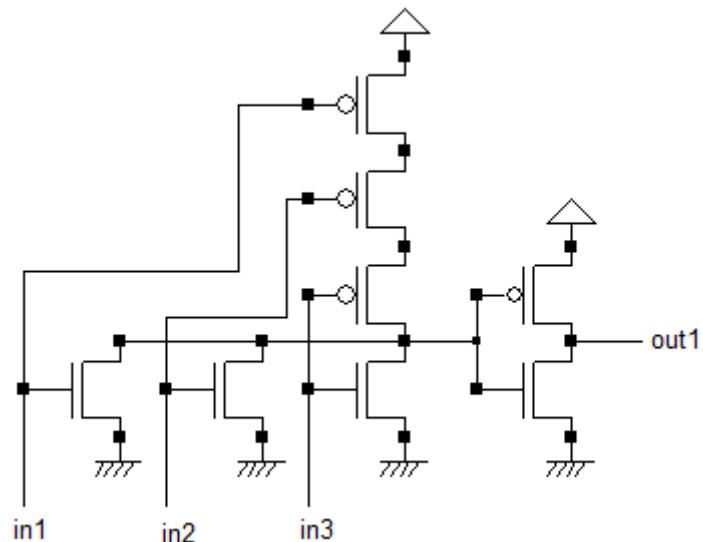


Figura 3.25 Implementarea circuitului OR3 folosind tranzistoare nmos și pmos

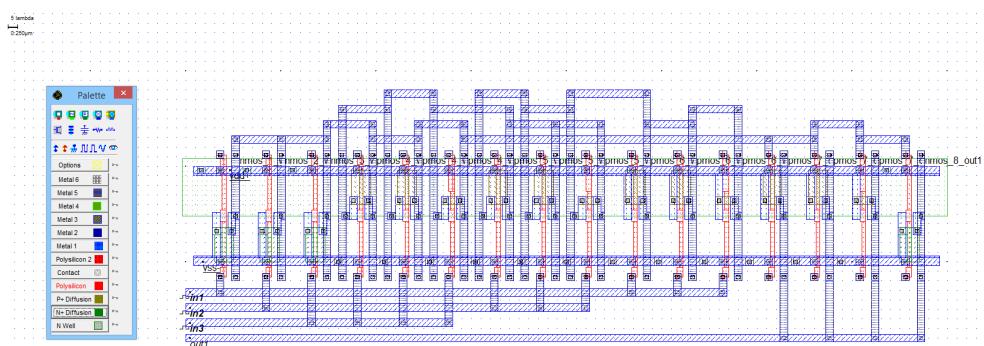


Figura 3.26 Layout-ul circuitului OR3

CMOS VLSI

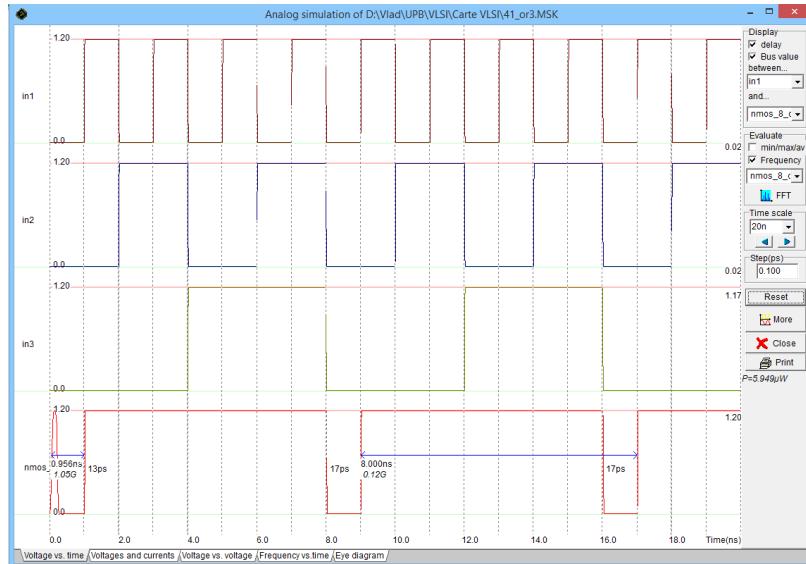


Figura 3.27 Simularea circuitului OR3

3.1. XOR2

Se consideră un circuit XOR cu două intrări in1, in2 și ieșirea out1. Tabela de adevăr corespunzătoare circuitului este:

in1	in2	out1
0	0	0
0	1	1
1	0	1
1	1	0

Tabel 3.11 Tabela de adevăr pentru XOR2

Poarta XOR are numeroase utilizări în proiectarea circuitelor numerice. Funcția îndeplinită de acest circuit nu se realizează direct, ca în cazul circuitului NAND. De aceea, există mai multe versiuni ale circuitului XOR. Unele versiuni se caracterizează printr-o arie ocupată mai mică, și prin folosirea porților de transmisie. Întrucât puterea de comandă la ieșire derivă din intrări, circuitul mai poartă numele de XOR pasiv. Alte configurații se bazează pe porți statice CMOS. Această soluție asigură un timp de răspuns mai rapid, dar utilizează mai multe tranzistoare în proiectare.

3. Proiectarea porților logice În VLSI

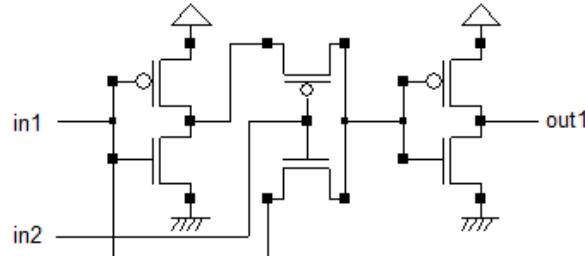


Figura 3.28 Implementarea circuitului XOR2 folosind tranzistoare nmos și pmos

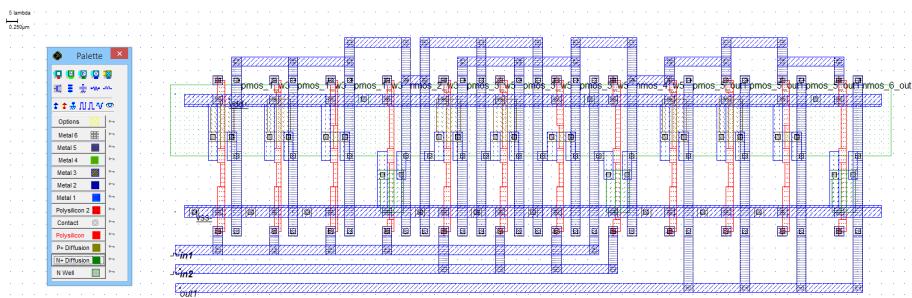


Figura 3.29 Layout-ul circuitului XOR2

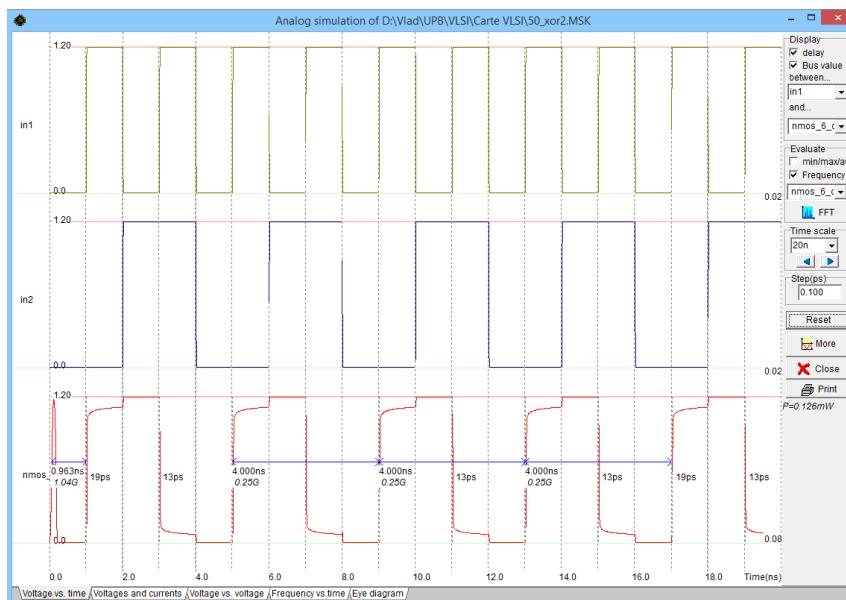


Figura 3.30 Simularea circuitului XOR2

3.2. XOR3

Pentru circuitul XOR3, se consideră trei intrări: in1, in2, in3 și o singură ieșire out1.

in1	in2	in3	out1
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Tabel 3.12 Tabelă de adevăr pentru XOR3

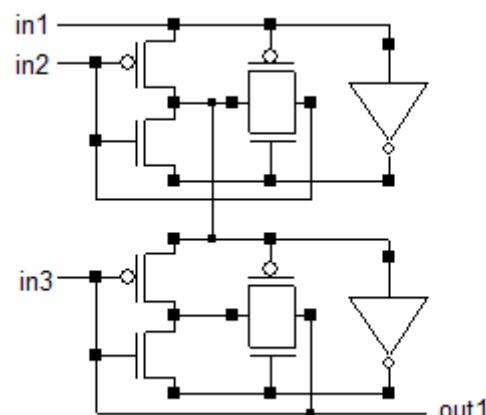


Figura 3.31 Implementarea circuitului XOR3 folosind tranzistoare nmos și pmos

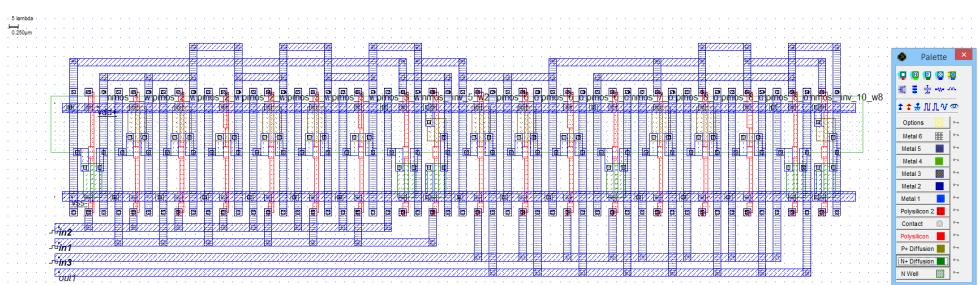


Figura 3.32 Layout-ul circuitului XOR3

3. Proiectarea porților logice În VLSI

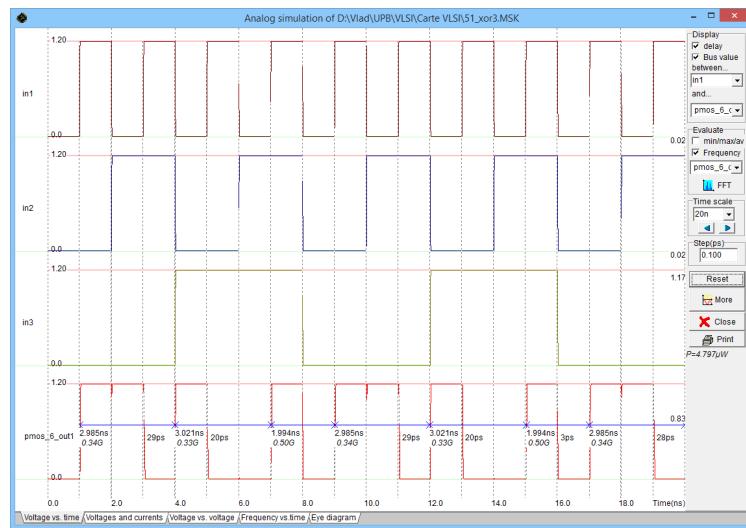


Figura 3.33 Simularea circuitului XOR3

4. CIRCUITE COMBINATIONALE SI CU MEMORIE VLSI

4.1. Multiplexor 2:1

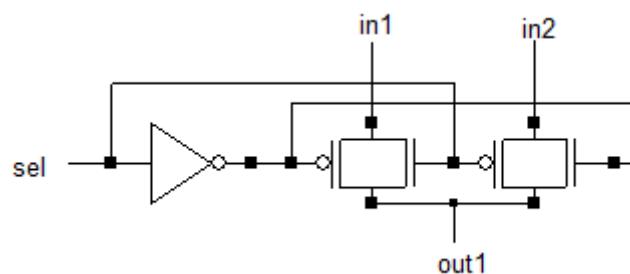


Figura 4.1 Implementarea MUX 2:1

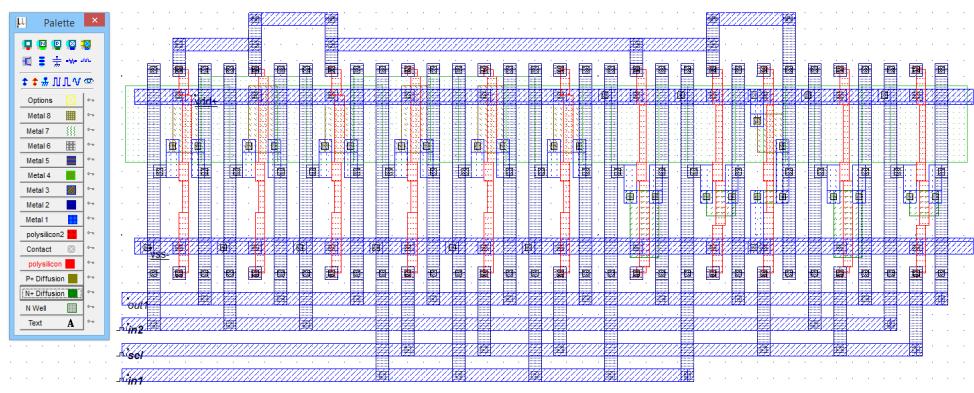


Figura 4.2 Layout-ul circuitului MUX 2:1

4. Circuite combinatoriale si cu memorie VLSI

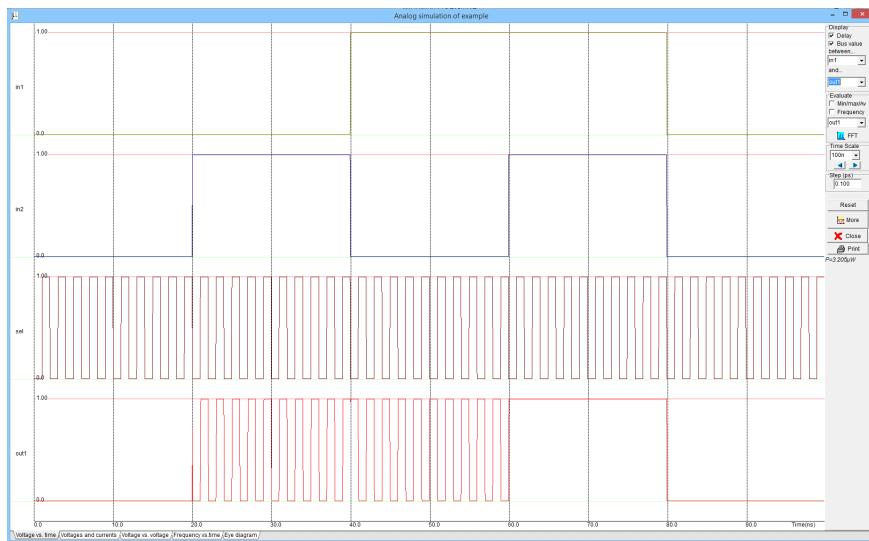


Figura 4.3 Simularea circuitului MUX 2:1

4.2. Memorie ROM

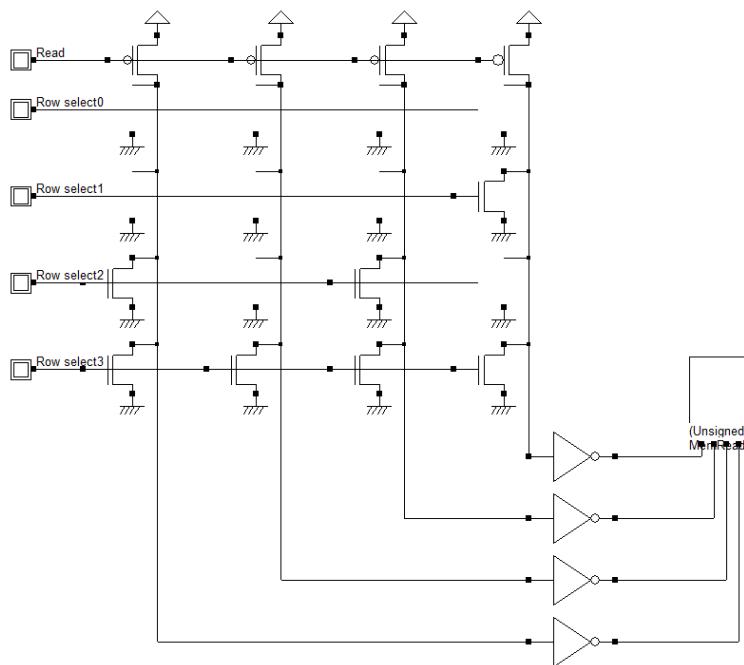


Figura 4.4 Implementarea unei memorii ROM 4x4

CMOS VLSI

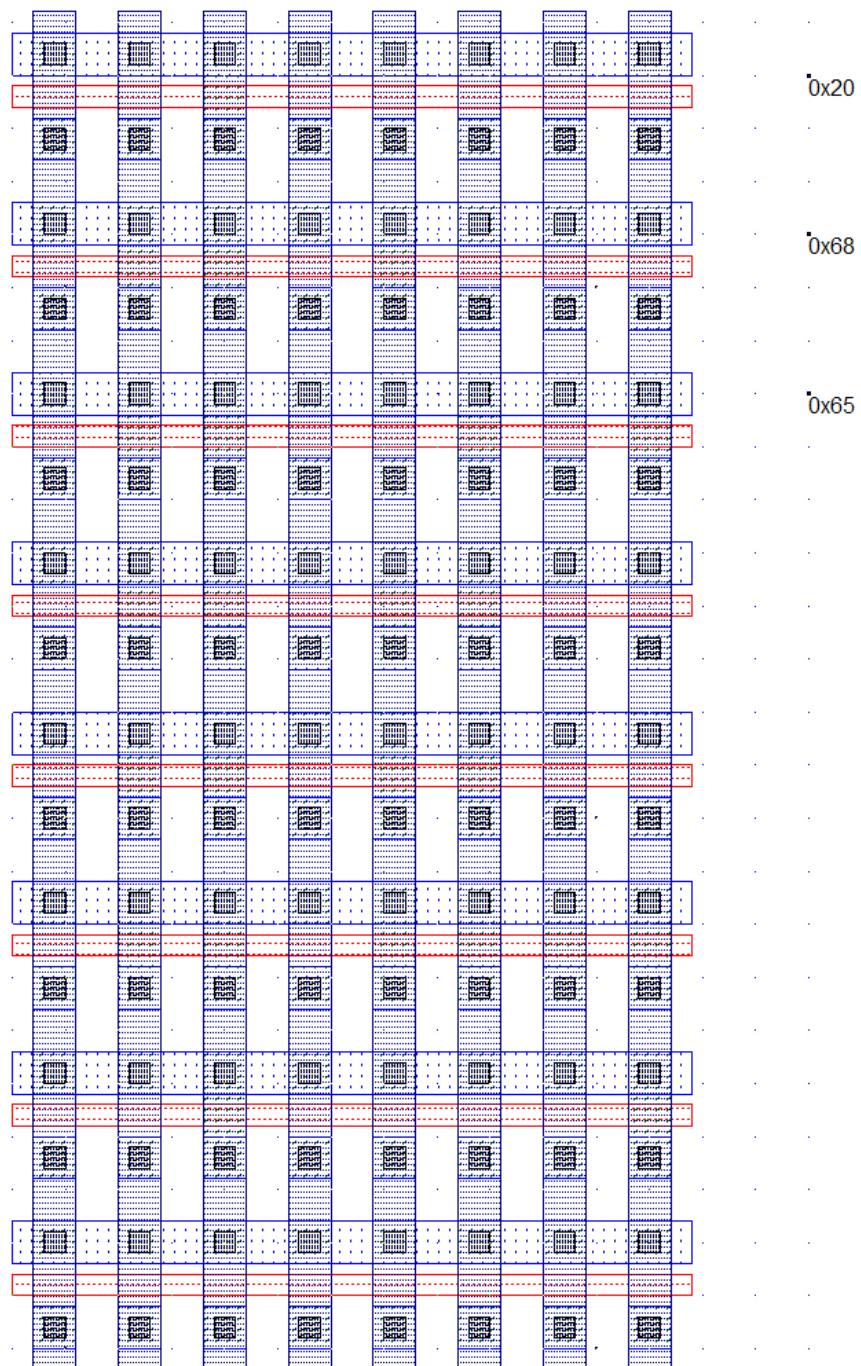


Figura 4.5 Memorie ROM programata cu textul Hello

4. Circuite combinationale si cu memorie VLSI

4.3. Bistabil de tip D

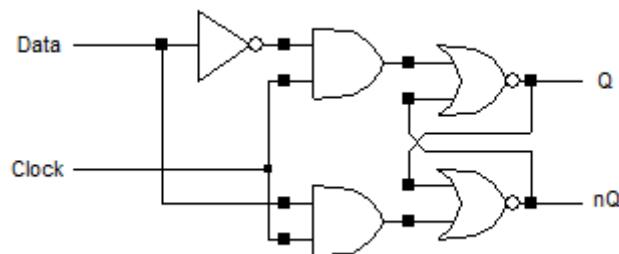


Figura 4.6 Implementare bistabil de tip D

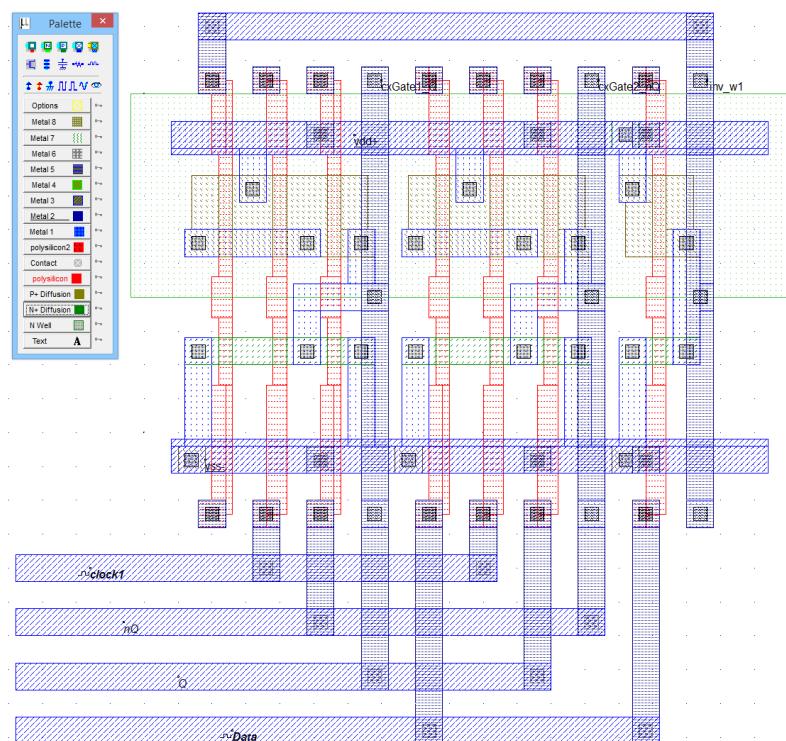


Figura 4.7 Layout bistabil de tip D

CMOS VLSI

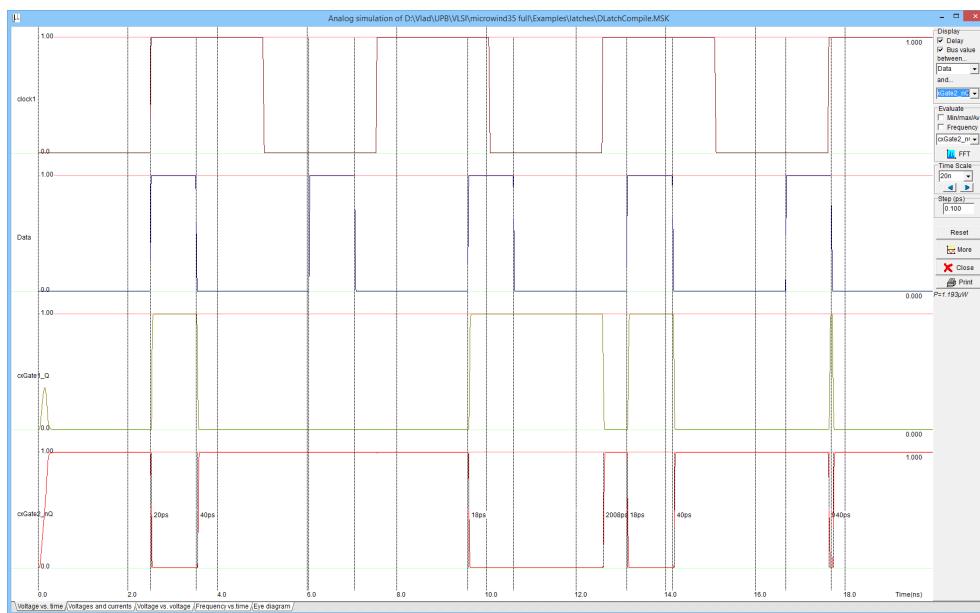


Figura 4.8 Simularea unui bistabil de tip D

4.4. Registru

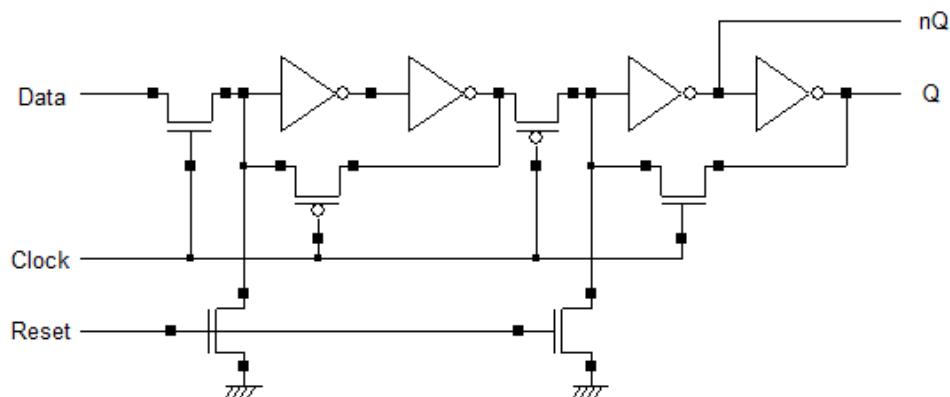


Figura 4.9 Implementare registru

4. Circuite combinatoriale si cu memorie VLSI

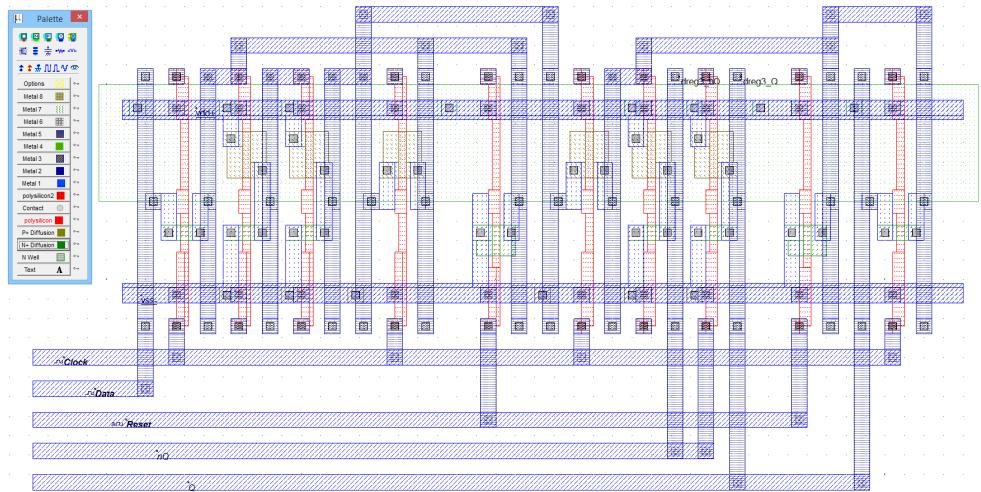


Figura 4.10 Layout registru

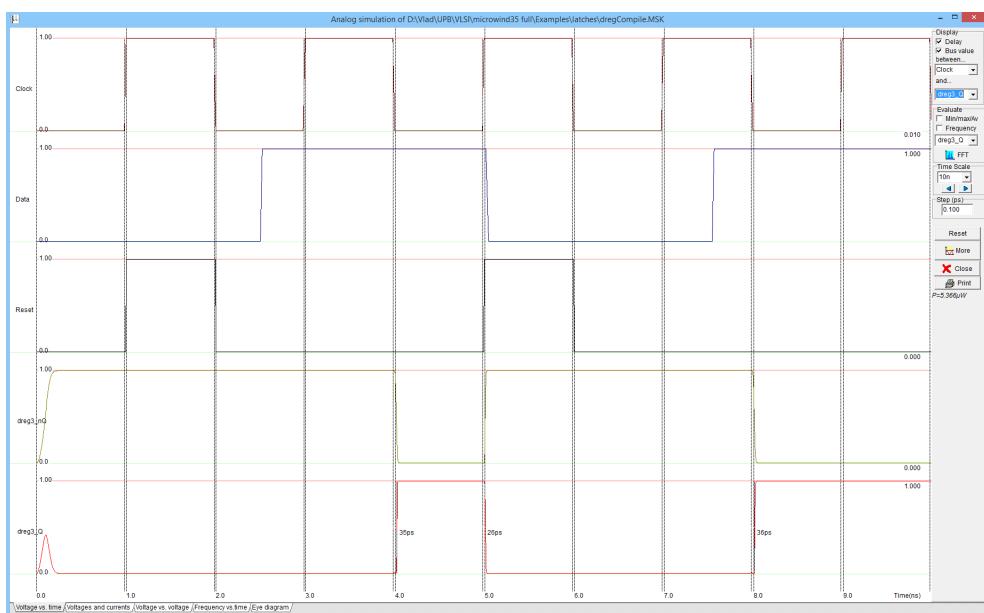


Figura 4.11 Simulare registru

4.5. Rețele logice programabile

Implementarea funcțiilor logice trebuie efectuată fără a cunoaște aplicarea funcției într-o structură regulată. Rețelele programabile

(Programmable Logic Array - PLA) permit maparea funcțiilor logice neregulate în structuri regulate. Funcțiile logice pot fi modificate substanțial fără schimbări ale proiectului sau ale măștii PLA. Utilizarea memorilor pentru stocarea tabelelor de adevăr nu este convenabilă deoarece ocupă un volum foarte mare.

Schema bloc al unui PLA este prezentată în Figura 4.12, în care ariile SI și SAU sunt realizate folosind circuite NOR.

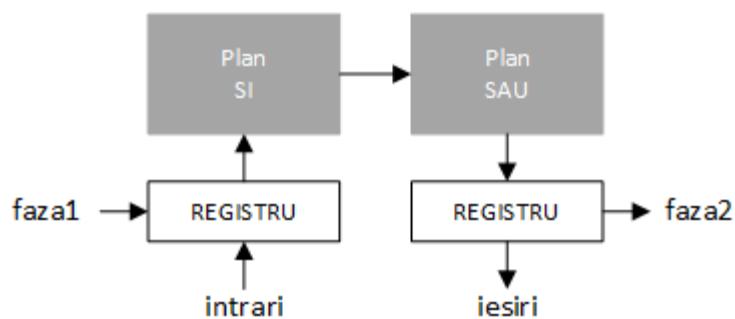


Figura 4.12 Schema bloc a unei rețele logice programabile

Pentru exemplificarea implementării unui sistem de funcții logice, cu ajutorul unui PLA, se consideră următorul exemplu, plecând de la sistemul descris în sistemul de ecuații (4.1) ce are ca implicantă primă: $A, \bar{B} \cdot \bar{C}, \bar{A} \cdot \bar{B} \cdot C, \bar{A} \cdot B \cdot \bar{C}$.

$$\begin{aligned}
 Z_1 &= A; \\
 Z_2 &= A + \bar{A} \cdot \bar{B} \cdot C; \\
 Z_3 &= \bar{B} \cdot \bar{C}; \\
 Z_4 &= \bar{A} \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot \bar{C}.
 \end{aligned} \tag{4.1}$$

Implementarea rețelelor logice programabile corespunzătoare ecuațiilor date este evidențiată în Figura 4.13.

4. Circuite combinatoriale si cu memorie VLSI

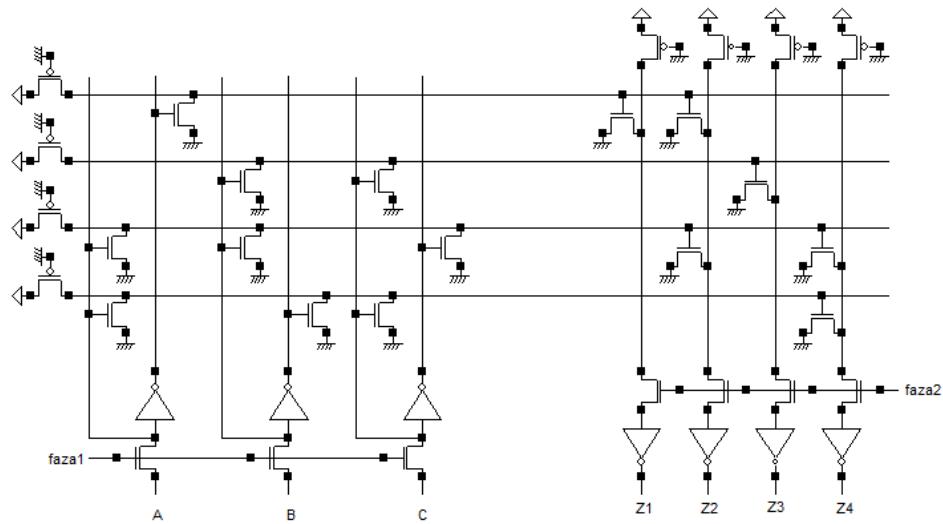


Figura 4.13 Implementare PLA

5. CIRCUITE ARITMETICE VLSI

5.1. Sumator

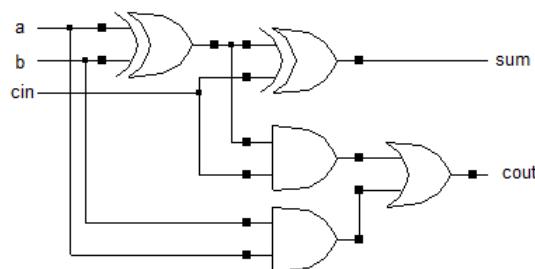


Figura 5.1 Implementarea sumator

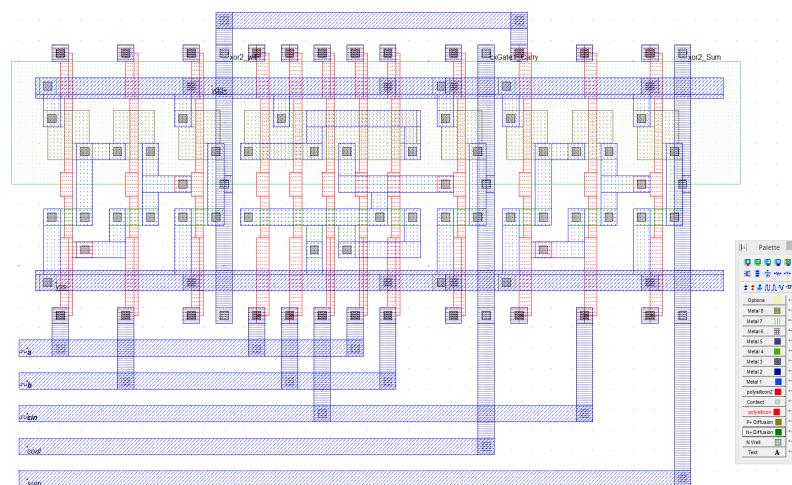


Figura 5.2 Layout sumator

5. Circuite aritmetice VLSI

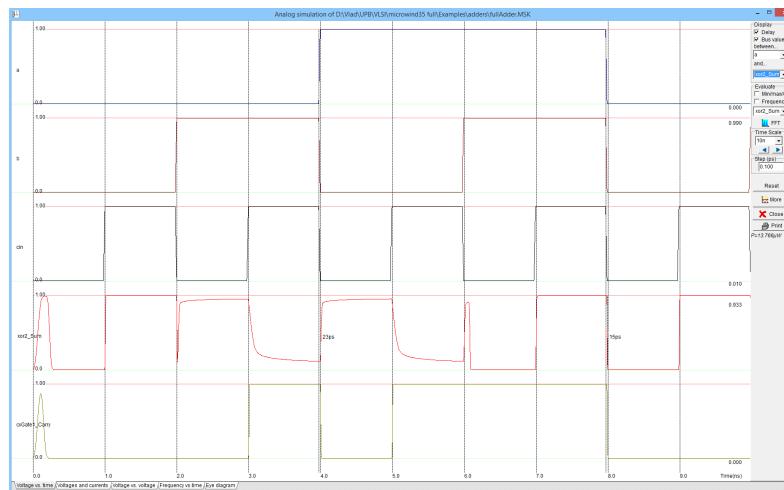


Figura 5.3 Simularea sumatorului

5.2. Semisumator

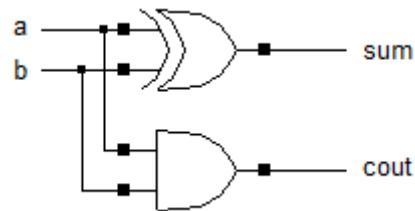


Figura 5.4 Implementarea circuitului semisumator

CMOS VLSI

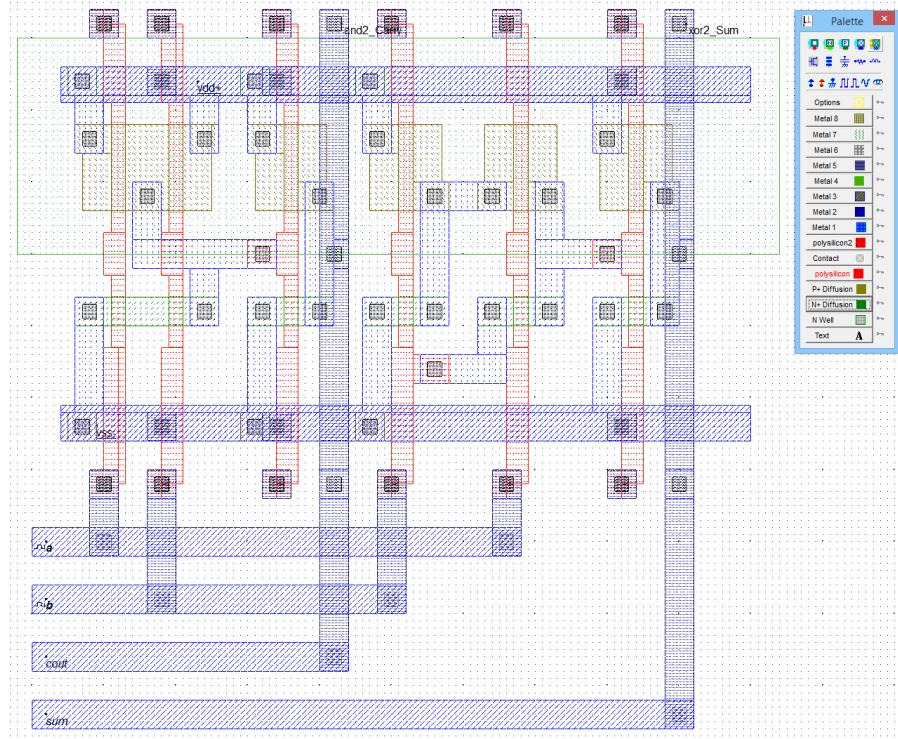


Figura 5.5 Layout-ul semisumatorului

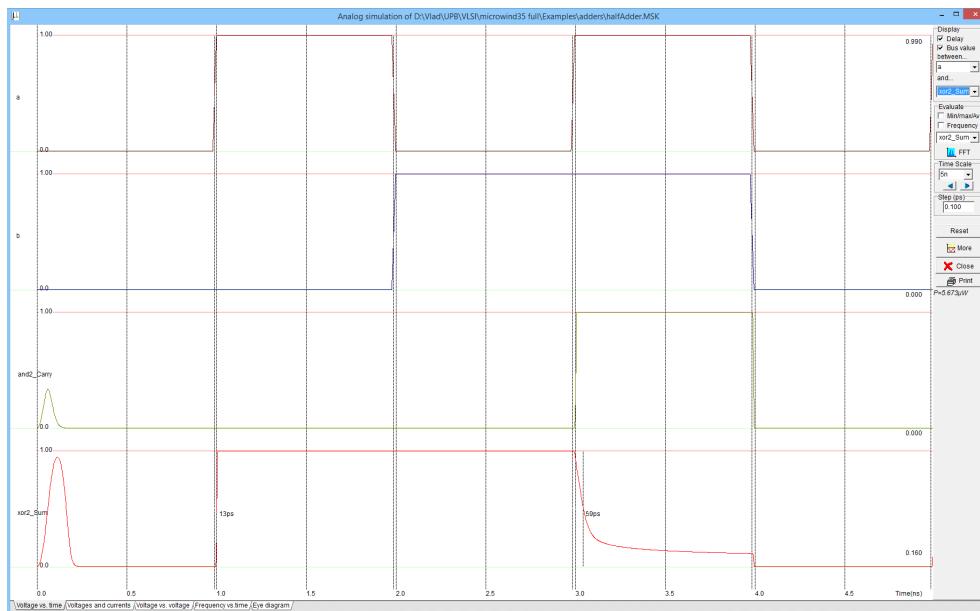


Figura 5.6 Simularea semisumatorului

5. Circuite aritmetice VLSI

5.3. Comparator

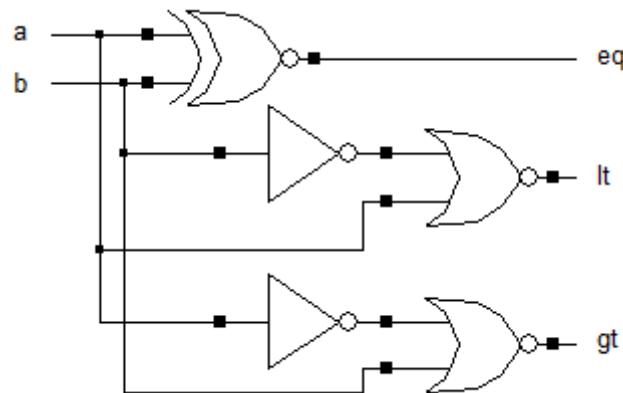


Figura 5.7 Implementare comparator

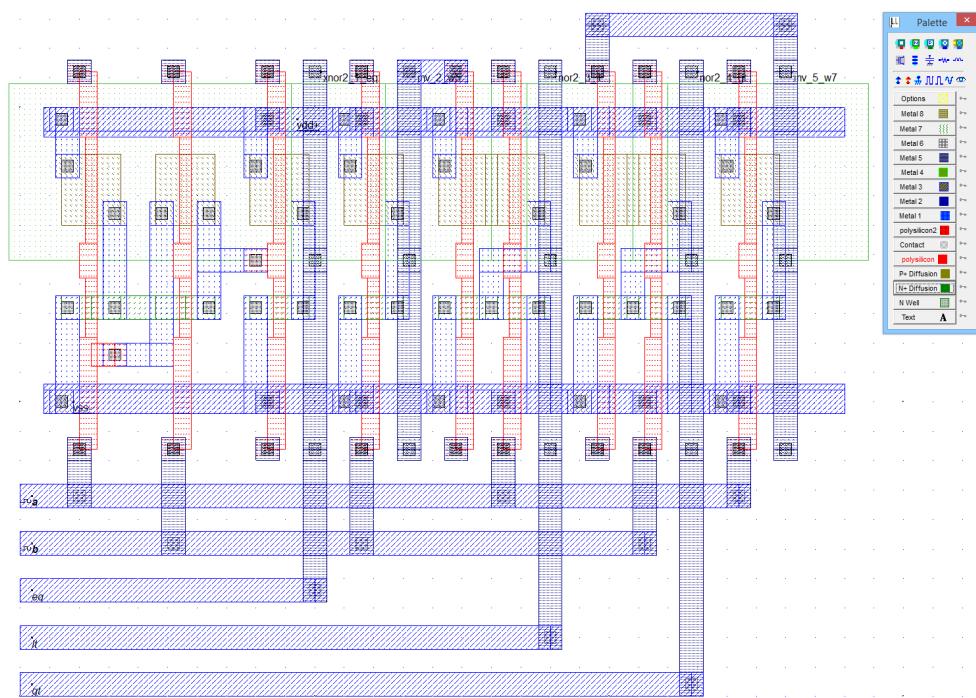


Figura 5.8 Layout comparator

CMOS VLSI

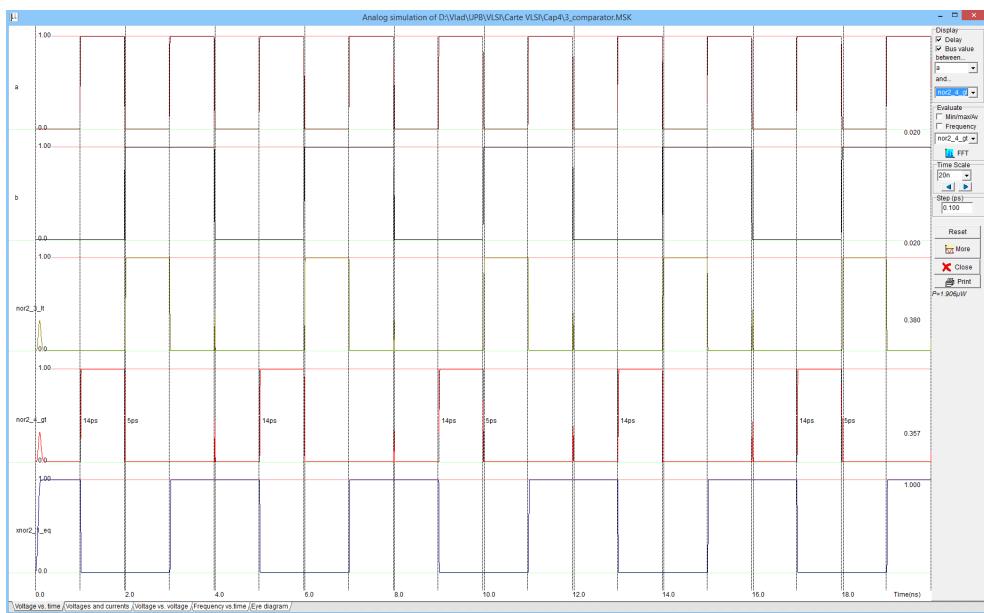


Figura 5.9 Simularea comparatorului

6. PROIECTAREA UNUI MICROPROCESOR MULTICHP

În cadrul acestui capitol se va detalia proiectarea în VLSI a unui procesor a cărui schemă bloc este prezentată în Figura 1. Microprocesorul propus este un procesor universal microprogramabil. Procesorul este compus din următoarele blocuri funcționale:

- Unitatea de execuție – UEx
- Unitatea de comandă – UC
- Unitatea de management a memoriei – UMM
- Unitatea de interfață cu magistrala – UIM

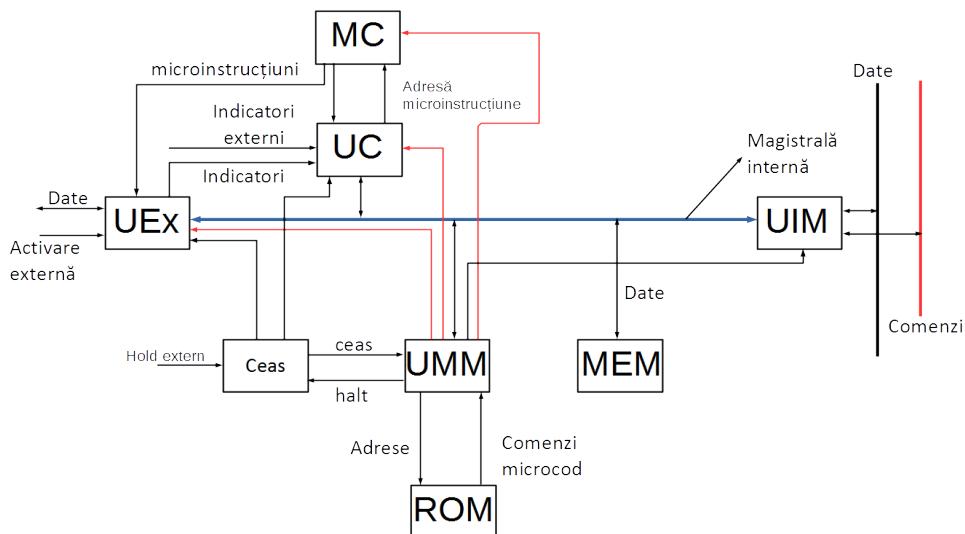


Figura 6.1 Schema bloc a procesorului multichip

Unitatea de prelucrare execută operații aritmetice sub control microprogramat și constă din:

- Un tablou de registre generale – 16 registre generale de 16 biți
- O rețea de deplasare
- O unitate aritmetico-logică

Unitatea de control a procesorului este compusă dintr-un controler de microprogram și logica aferentă în vederea realizării calculului adresei microinstrucțiunii următoare. Controler-ul de microprogram posedă o stivă pentru stocarea contorului de microprogram și o stivă pentru contorii de ciclii în cazurile implementării de cicluri *do* sau *for* imbricate.

Circuitul pentru managementul memoriei – este utilizat pentru furnizarea adreselor necesare accesării memoriei de date și memoriei de instrucțiuni. Acest circuit va furniza și semnalele pentru comunicația celorlalte unități conectate la magistrala internă. Totodată, acest circuit va asigura partaționarea și gestionarea memoriei pentru structuri de date diferite realizând partații pentru stive, cozi, liste înlănuite, tablouri, asigurând și indicatorii de stivă plină etc.

Unitatea de interfațare cu magistrala – asigură comunicarea asincronă cu alte dispozitive sau procesoare conectate la magistrala sistemului.

Circuitul de ceas – generează un semnal bifazic, el putând fi operat din exterior sau controlat de către UMM în cazul unor operații care necesită o durată de execuție mai mare decât operațiile obișnuite.

Sincronizarea – procesorul funcționează cu un ceas bifazic așa cum se poate observa în Figura 2. Pe durata ceasului φ_1 datele sunt transferate de la un subsistem la altul în cadrul aceluiași chip. Faza φ_2 a ceasului este folosită pentru realizarea comunicării între chip-urile microprocesorului. În cadrul unității de execuție, operațiile aritmetico-logice se vor executa în φ_2 . Pentru ca operațiile să se efectueze în φ_2 microinstrucțiunea trebuie să fie adusă în φ_1 anterior.

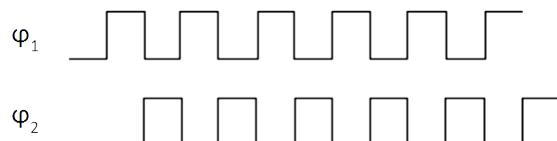


Figura 6.2 Ceas bifazic

6. Proiectarea unui microp procesor Multichip

6.1. Proiectarea unității de prelucrare a datelor

Proiectarea unității de prelucrare a datelor (sau unității de execuție) va trebui să se realizeze astfel încât să răspundă la următoarele cerințe:

1. Să se interconecteze ușor într-o configurație multiprocesor;
2. Să suporte o structură de comandă microprogramată astfel încât setul de instrucțiuni să fie definitivat ulterior;
3. Să permită realizarea unor operații pe câmpuri de lungime variabilă;
4. Să fie cât mai rapidă

Pentru realizarea cerinței 1, se vor prevedea două porturi. Un port va fi necesar pentru realizarea conectivității cu memoria respectiv cu circuitul de interfațare cu magistrala de sistem. Un al doilea port este necesar pentru realizarea comunicării cu mediul extern eventual cu alte sisteme.

Pentru realizarea cerinței 3 se va utiliza o rețea de deplasare cu mai mulți biți simultani.

Pentru realizarea cerinței 4, se vor utiliza două magistrale interne A și B cât și un tablou de registre generale de tip bipartit.

Structura generală a unității de prelucrare a datelor poate fi observată în Figura 6.3.

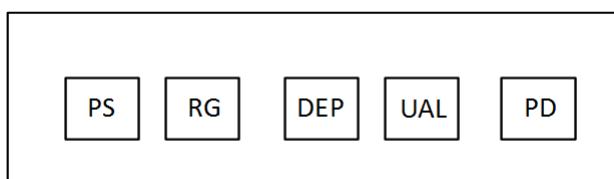


Figura 6.3 Structura generală. PS, PD reprezintă portul stânga / portul dreapta; RG reprezintă registrele generale; DEP = rețea de deplasare cu mai mulți biți simultan; UAL reprezintă unitatea aritmetico-logică.

Magistralele interne sunt realizate pe trasee de polisiliciu și operează în regim de preîncărcare. Pe durata lui φ_2 ele vor fi preîncărcate la V_{DD} iar pe durata lui φ_1 când trebuie să se transfere date, ele vor fi "trase" rapid la masă dacă valoarea 0 va trebui transferată sau rămân pe 1 dacă va trebui transferată valoarea 1.

Întrucât microprocesorul este realizat în tehnologie NMOS care presupune durata fronturilor pozitive mai mare ca durata fronturilor negative, preîncărcarea magistralelor reduce întârzierea pe fronturile pozitive. Cele două magistrale traversează structura orizontală. Perpendicular pe structura traseelor metalice vor fi aduse semnalele de comandă.

6.2. Proiectarea unității aritmetico-logice

UAL poate avea performanțele afectate negativ de propagarea transportului. Pentru a preîntâmpina acest lucru se va folosi schema de propagare a transportului numită lanțul Manchester aşa cum este prezentat în Figura 6.4.

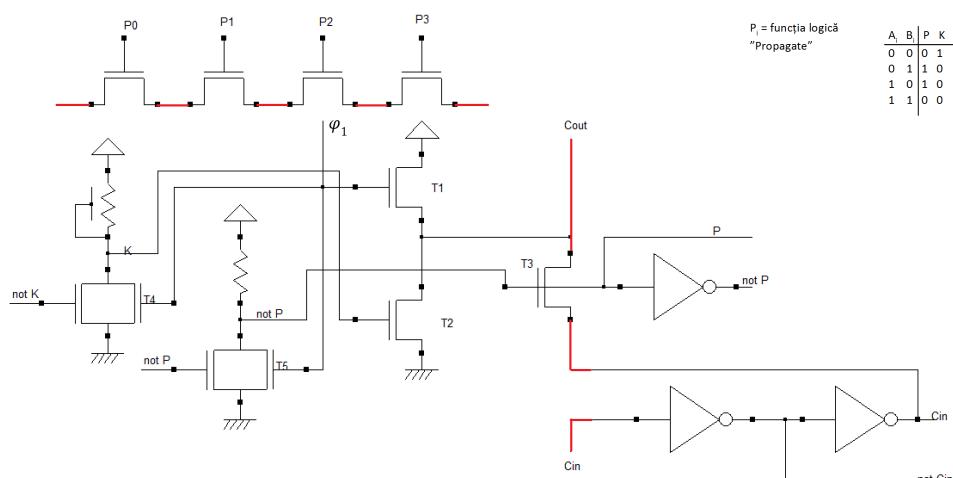


Figura 6.4 UAL

În φ_1 , T_1 va fi deschis și traseul de polisiliciu se va încărca la V_{DD} . T_2 și T_3 sunt blocate deoarece etajele care controlează grilele lui T_2 și T_3 au ieșirile la nivel coborât deoarece tranzistoarele T_4 și T_5 sunt deschise și atunci ieșirile K și P sunt la nivel coborât.

În φ_2 dacă transportul spre etajul următor trebuie blocat, K va fi la nivel ridicat, T_2 va conduce și lanțul va fi conectat la masă prin T_2 . Dacă etajul propagă transportul atunci T_3 va fi deschis. Presupunând că C_{in} este la nivel ridicat implică că lanțul Manchester va fi la nivel ridicat. Dacă C_{in} este

6. Proiectarea unui micropresor Multichip

la nivel coborât atunci lanțul Manchester va fi conectat la masă prin intermediul tranzistorului ultimului etaj inversor care furnizează C_{in} , valoarea lui C_{in} fiind 0.

Implementarea funcțiilor \bar{K} și \bar{P} se va realiza prin intermediul rețelelor formate din tranzistoare de trecere și este prezentată în Figura 6.5.

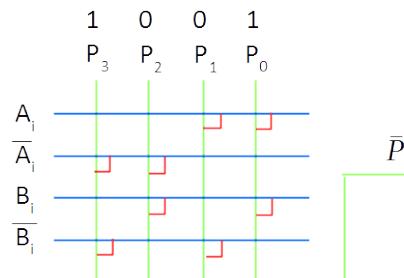
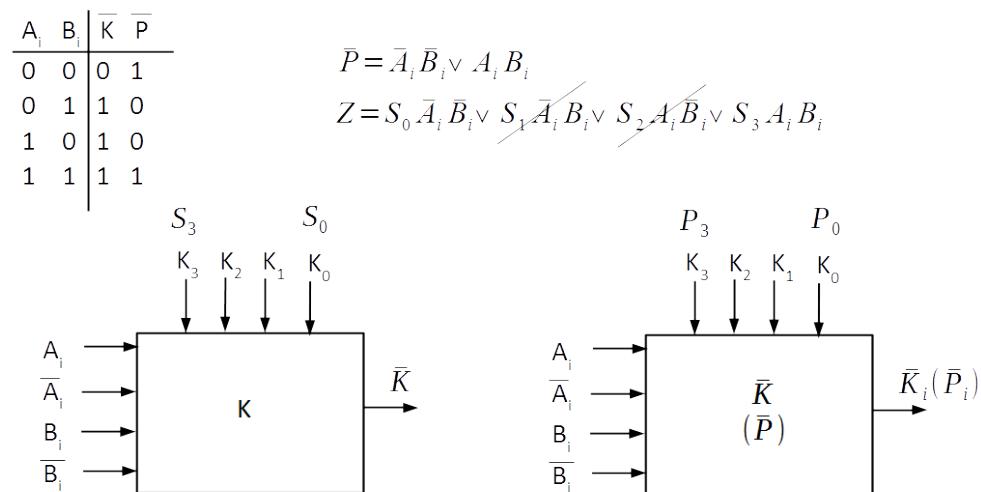


Figura 6.5 Implementarea funcțiilor \bar{K} și \bar{P} precum și circuitul folosit pentru realizarea funcțiilor \bar{K} și \bar{P}

Având implementarea pentru \bar{K} și \bar{P} se poate stabili circuitul pentru propagarea transportului ca subsistem, circuit prezentat în Figura 6.6.

CMOS VLSI

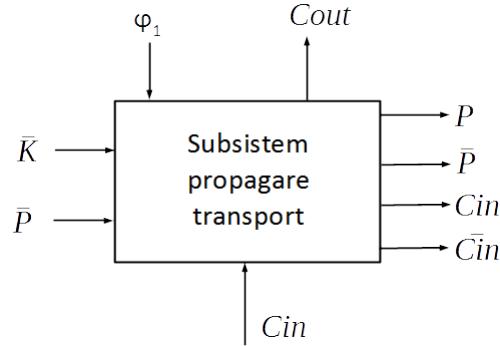


Figura 6.6 Circuit pentru propagarea transportului ca subsistem

Conform celor prezentate până acum, putem realiza o unitate aritmetico-logică de bază și anume o unitate aritmetico-logică pe 4 biți aşa cum se poate observa în Figura 6.7.

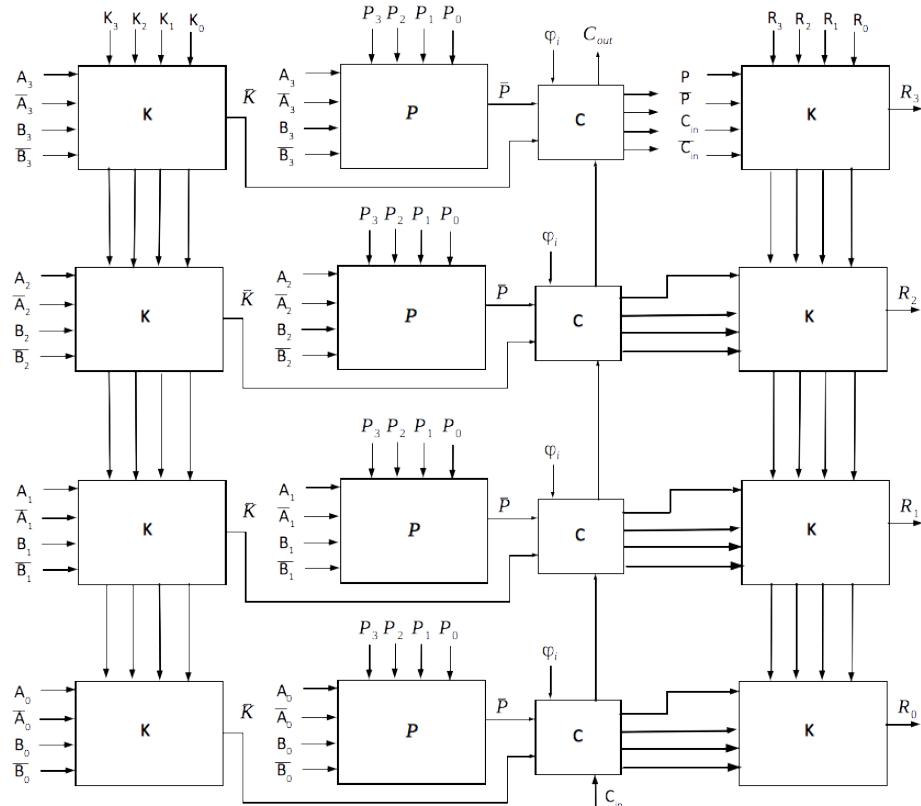


Figura 6.7 Unitate aritmetico-logică pe 4 biți

6. Proiectarea unui micropresor Multichip

6.2.1. Circuitul de comandă pentru liniile K_i , P_i și R_i din cadrul UAL

Semnalele de comandă vor fi aplicate pe trasee verticale metalice. Din analiza funcționării sistemului se observă că traseele de comandă pot fi preîncărcate pe φ_1 deoarece UAL va opera pe φ_2 , iar preîncărcarea nu va afecta în nici un mod operarea corectă a rețelei UAL. Implementarea circuitului de comandă poate fi observată în Figura 6.8.

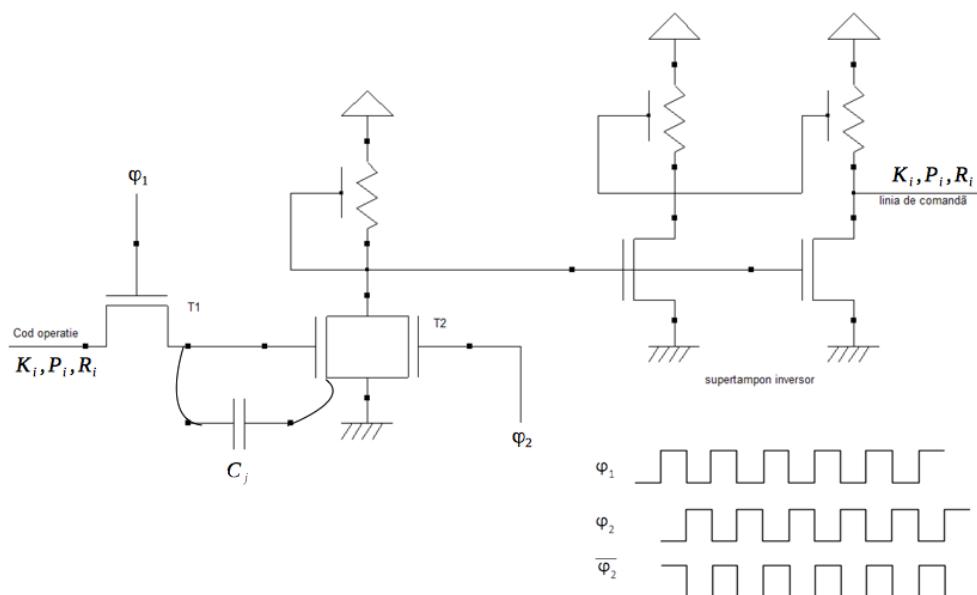


Figura 6.8 Circuitul de comandă pentru UAL (K , P și R)

Liniile K_i , P_i și R_i de comandă sunt controlate de circuitul supertampon de tip inversor. Fronturile pozitive de la ieșirea supratamponului au aceeași întârziere ca și fronturile negative deoarece grila tranzistorului trage-sus al celui de al doilea etaj al supratamponului nu mai este conectată la sursă ci la o tensiune care variază în antifază cu tensiunea sursei. Astfel tensiunea între grila tranzistorului trage-sus și sursa acestui tranzistor va fi: $V_{gs} = V_{DD} + 0,8V_{DD} \cong 2V_{DD}$. Considerând curentul la saturatie rezultă: $I_{ds} \cong (V_{gs})^2$, adică curentul de preîncărcare va fi de patru ori mai mare decât în cazul unui inversor obișnuit - $\frac{z_{pn}}{z_{pd}} = \frac{4}{1}$. Curentul din tranzistorul trage-sus este de patru ori mai mare decât curentul din tranzistorul trage-jos.

La apariția lui φ_1 codul de operație se stochează pe grila lui T_1 . T_2 este deschis deoarece φ_2 este pe nivel ridicat. Ieșirea circuitului NOR va fi la nivel coborât iar supertamponul inversor va preîncărca linia de comandă la V_{DD} . La apariția lui φ_2 , T_2 se va bloca iar sarcina stocată pe grila lui T_1 va controla intrarea supertamponului inversor. Linia de comandă va rămâne la nivel ridicat sau va coborî la masă în funcție de codul stocat pe grila lui T_1 .

6.2.2. Registrele UAL

Schema generală care prezintă amplasarea registrelor generale este prezentată în Figura 6.9.

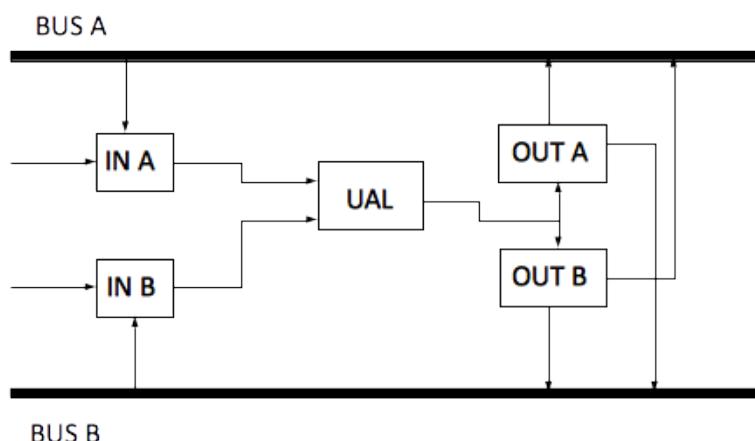


Figura 6.9 Registre de intrare: IN A, IN B. Registre de ieșire OUT A, OUT B

Registrele de intrare trebuie să stocheze datele pe durata lui φ_1 . Datorită faptului că în φ_2 are loc reîmprospătarea informației din registrele de intrare, barele de comandă pentru selectarea surselor de semnal nu pot fi preîncărcate pe durata lui φ_2 .

Semnalul de comandă pentru selectarea sursei se aduce în circuit pe durata lui φ_2 și va fi stocat pe grila lui T_1 (Figura 6.10). În acest timp $\overline{\varphi_1}$ va fi la nivel coborât și T_2 va fi tot la nivel coborât, astfel că ieșirea circuitului NOR va fi la nivel coborât. În acest fel se va realiza produsul $Ld_1\varphi_1$. În φ_1 următor se selectează una din surse prin forțarea uneia din barele Ld_1 , Ld_2 , Ld_3 la 1.

6. Proiectarea unui microp procesor Multichip

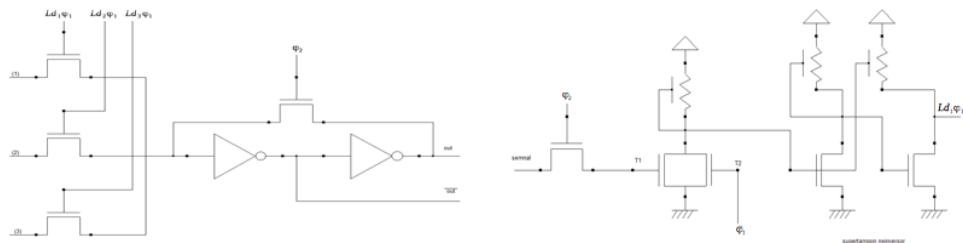


Figura 6.10 Realizarea produsului $Ld_1\varphi_1$

6.2.3. Registrul de ieșire OUT A, OUT B

Schema registrului de ieșire este prezentată în Figura 6.11. Pentru generarea lui $\varphi_2 SEL$ se va putea folosi exact circuitul driver pentru P, K, R.

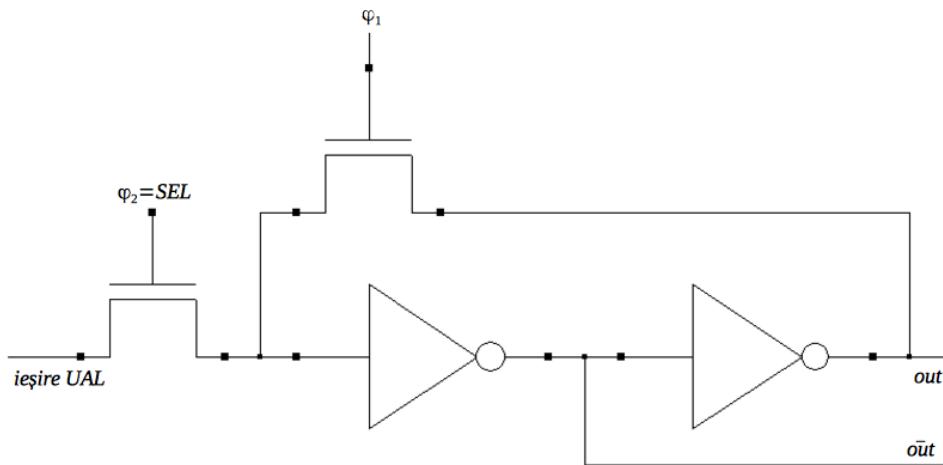


Figura 6.11 Registrul de ieșire și supertamponul neinversor

În φ_1 are loc reîmprospătarea datelor. Tot în φ_1 se aduce și codul de operație care selectează OUT A sau OUT B. El se va stoca tot pe grila lui T_1 iar în φ_2 următor se va controla ieșirea supertamponului neinversor pentru selecția registrului corespunzător de ieșire.

6.2.4. Magistralele (BUS A, BUS B)

Schema generală este prezentată în Figura 6.12. Semnalul MAG se preîncarcă în φ_2 și transferul de date are loc în φ_1 . Traseele de polisiliciu

CMOS VLSI

care alcătuiesc magistrala vor fi preîncărcate în φ_2 la V_{DD} . Ele pot fi controlate de mai multe surse. Fiecare sursă are asociată o pereche de tranzistoare T_1 , T_2 care sunt conectate în serie și plasate între linia magistrală și masă. T_2 are grila controlată în φ_1 de un semnal selecție sursă iar la grila tranzistorului T_1 se va conecta semnalul negat furnizat de sursă. Astfel magistrala va rămâne la nivelul 1 dacă T_1 nu conduce și va fi adusă la masă dacă T_1 conduce.

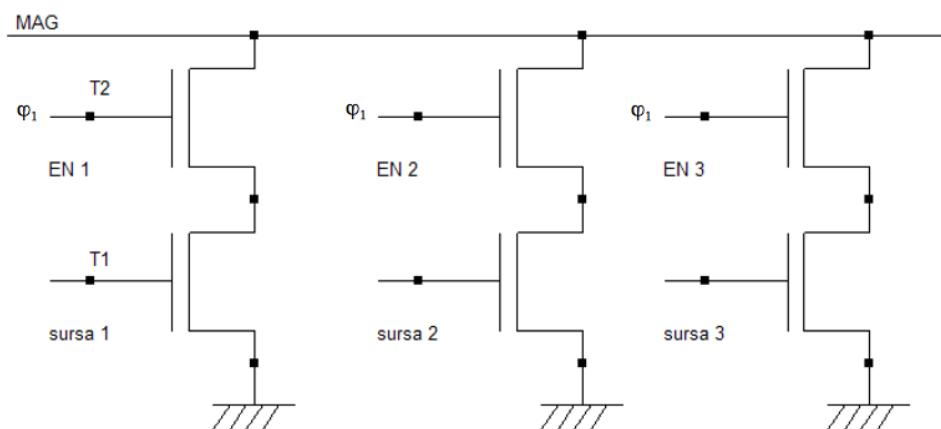


Figura 6.12 BUS A, BUS B

6.2.5. Tristate (TS)

Implementarea acestui modul este prezentată în Figura 6.13.

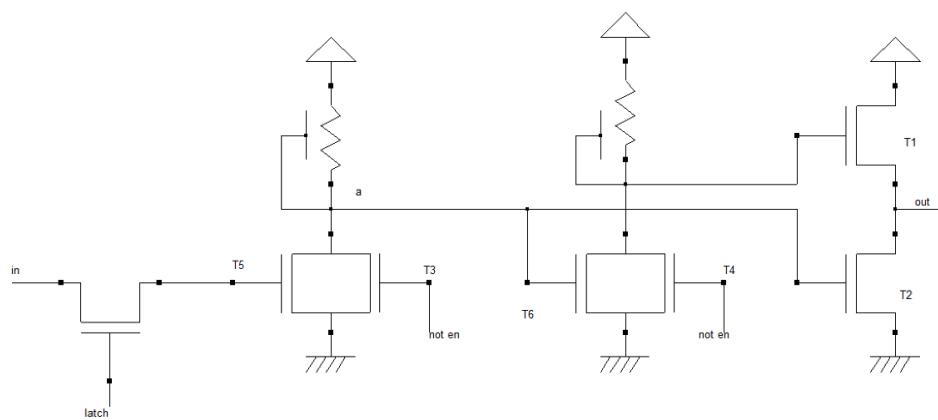


Figura 6.13 Tristate

6. Proiectarea unui micropresor Multichip

Tranzistoarele T_3 și T_4 sunt blocate, a și b operează în antifază și funcție de valoarea *in* care a fost sau nu stocată pe grila lui T_5 , fie T_1 va conecta semnalul *aut* la V_{DD} , fie T_2 va conecta semnalul *aut* la masă.

6.2.6. Circuitul de deplasare circulară

Circuitul de deplasare cu un bit folosește multiplexoare la intrarea celulelor de memorie.

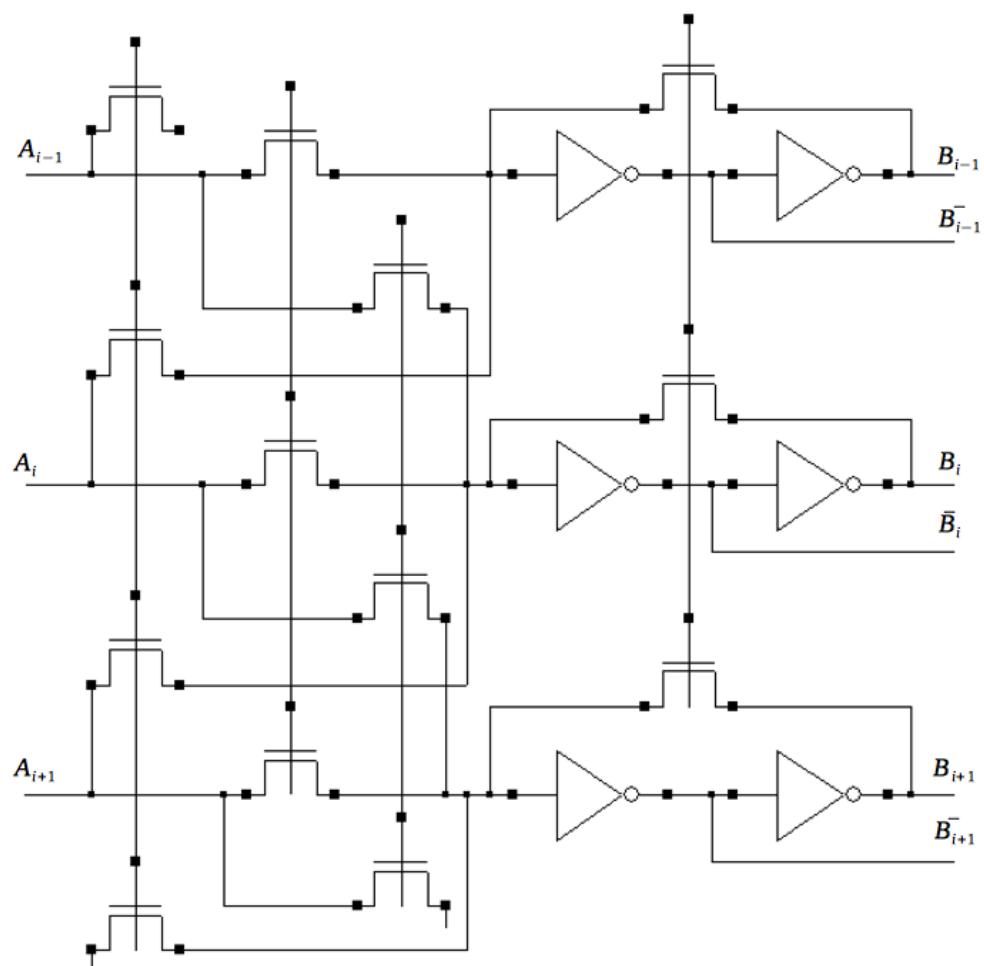


Figura 6.14 Circuitul de deplasare circulară

CMOS VLSI

Pentru deplasări multiple sunt necesare mai multe perioade de ceas.
Se va folosi rețeaua "cross-bar".

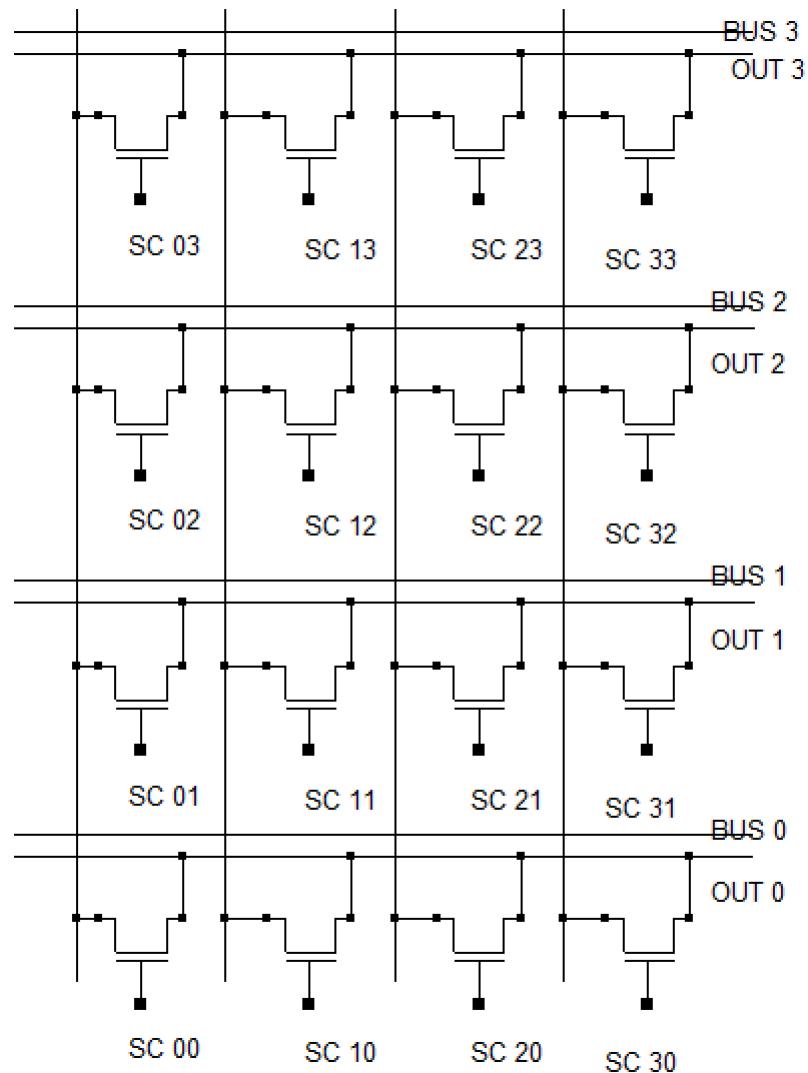


Figura 6.15 Rețeaua cross-bar

6. Proiectarea unui micropresor Multichip

6.2.7. Unitatea de execuție

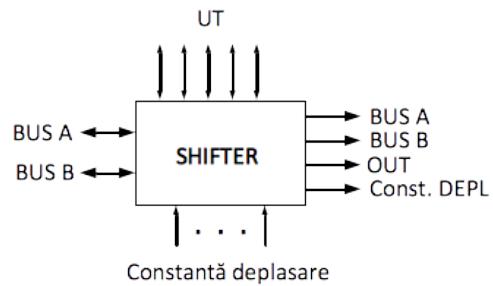


Figura 6.16 Unitatea de execuție

Schema circuitului de deplasare circulară este prezentată în Figura 6.17.

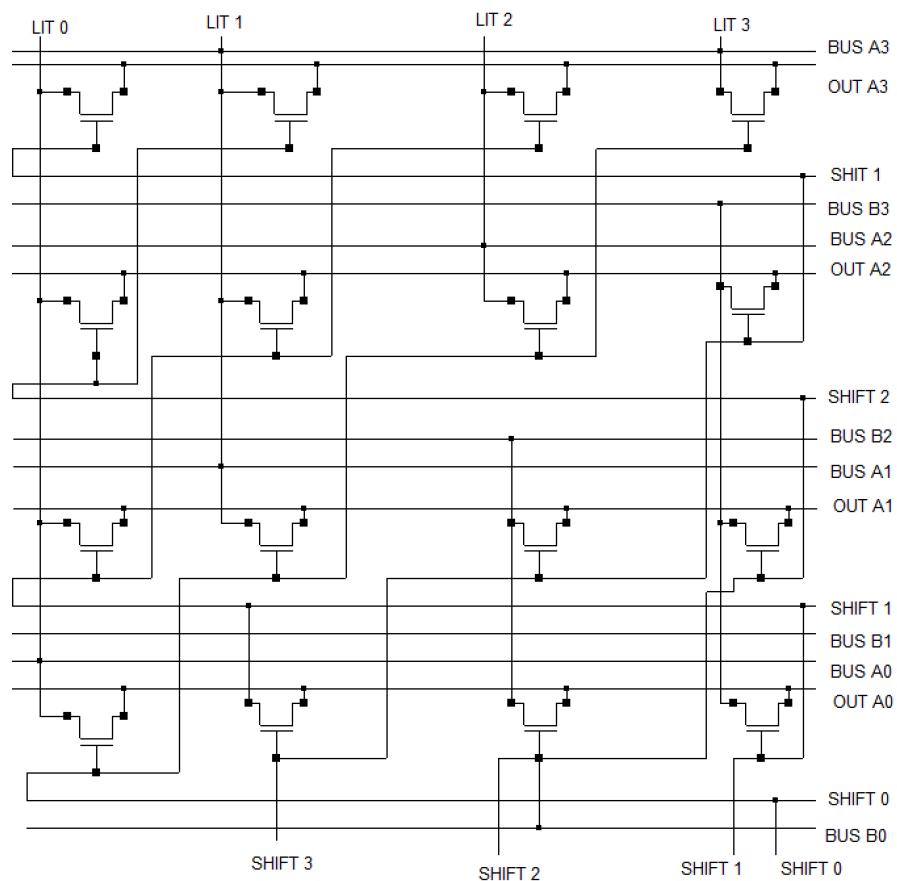


Figura 6.17 Schema circuitului de deplasare circulară

Având în vedere toate implementările realizate până în prezent, Figura 6.3 se poate detalia aşa cum se prezintă în.

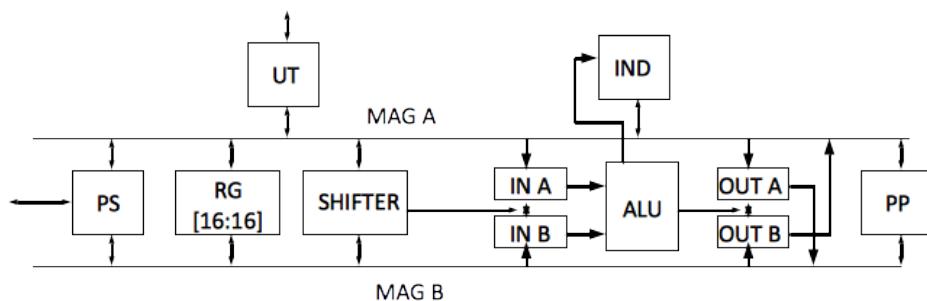


Figura 6.18 Detaliere implementare

Ca ieșire va fi constanta de deplasare shift-ată. Structura are conexiuni la părțile inferioare și superioare cu MAG A și MAG B precum și cu constanta de deplasare decodificată (shift) și cu biții portului literal (LIT) care sunt conectați la liniile MAG A. Prin portul literal, conținutul magistralei MAG A poate fi extras din unitatea de execuție sau poate fi forțat din exterior.

6.2.8. Conexiunea magistralei MAG A cu portul literal

Implementarea magistralei MAG A cu portul literal este prezentată în Figura 6.19.

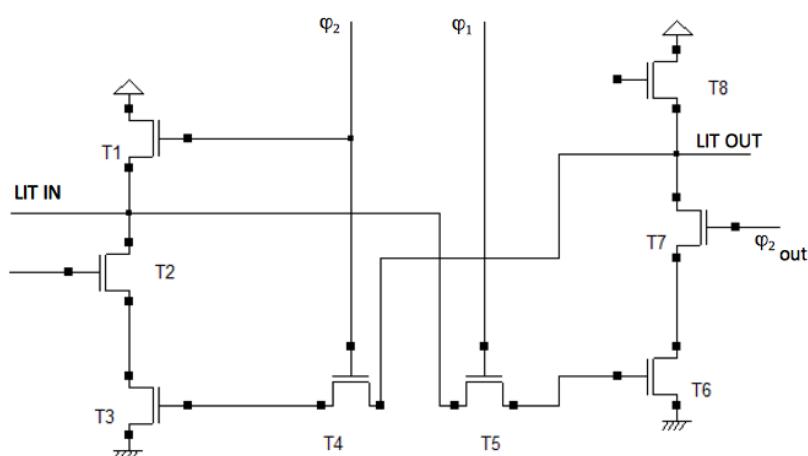


Figura 6.19 Magistrala MAG A

6. Proiectarea unui micropresor Multichip

Pentru ca magistrala să comunice în exterior cu portul literal, se folosește un circuit tampon. El este constituit din tranzistorii $T_1 - T_8$. Semnalul LIT IN are barele conectate direct la MAG A (MAG A este preîncărcată în φ_2 și comandată în φ_1). Pe durata lui φ_2 , T_1 este deschis și LIT IN e conectat la V_{DD} ; T_4 este deschis ceea ce implică că pe grila lui T_3 se stochează conținutul lui LIT OUT.

În φ_1 următor, dacă există o operație de intrare, φ_{1in} este activ ceea ce implică faptul că T_2 este deschis. În funcție de sarcina stocată pe T_3 , LIT IN va fi menținut pe nivelul ridicat sau descărcat rapid la masă.

φ_{1in} este semnalul de selecție al sursei portului literal pentru MAG A. În cazul în care se dorește ca, conținutul MAG A să fie transferat spre exterior (pe LIT OUT) va trebui să se îndeplinească condițiile:

- MAG A poartă semnal în φ_1 ;
- În φ_1 se stochează pe grila lui T_6 conținutul MAG A;
- T_8 este deschis și LIT OUT este preîncărcat la V_{DD} .

În φ_2 următor dacă φ_{2out} este activ, tranzistorul T_7 este deschis, tranzistoarele T_5 și T_8 sunt blocate iar LIT OUT va fi controlat de sarcina stocată în φ_1 anterior pe grila lui T_6 .

6.2.9. Decodificatorul

Rolul acestui circuit este acela de a ajuta la minimizarea semnalelor ce ajung la pinii circuitului. O implementare a acestui circuit folosind un circuit NAND este prezentat în Figura 6.20

CMOS VLSI

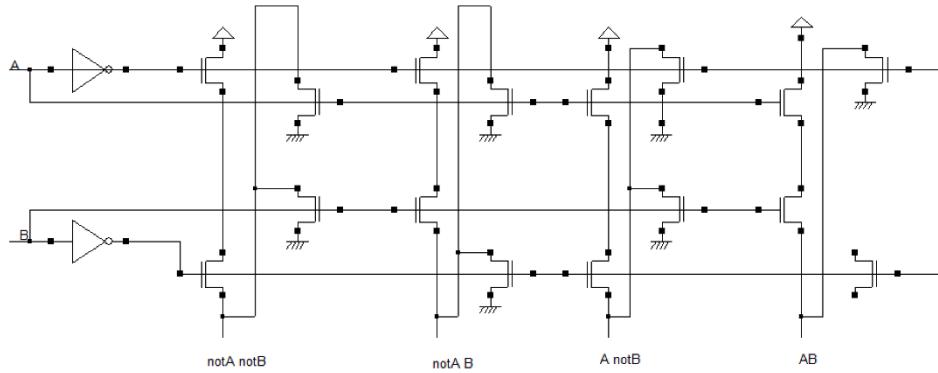


Figura 6.20 Exemplu de implementare a decodificatorului folosind circuite NAND

6.2.10. Tabloul de registre generale

Registrele generale sunt constituite din celule de memorare formate din două inversoare în serie și cu o reacție printr-un tranzistor de trecere deschis în φ_2 .

Implementarea unui registru pe 1 bit este prezentat în Figura 6.21.

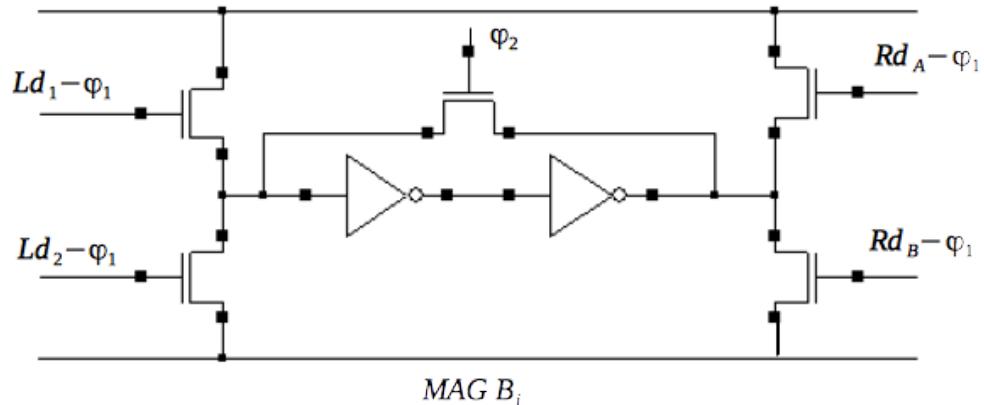


Figura 6.21 Implementarea unui registru pe 1 bit

6.2.11. Comunicația cu mediul înconjurător. Circuitele de comandă pentru ploturile de I/E

Cerințele pentru realizarea circuitului de comandă sunt:

6. Proiectarea unui micropresor Multichip

- Să comande sarcini capacitive mari
- Să opereze în regim TRISTATE
- Să memoreze pe timp limitat informația de ieșire

Implementarea circuitului de comandă este prezentată în Figura 6.22.

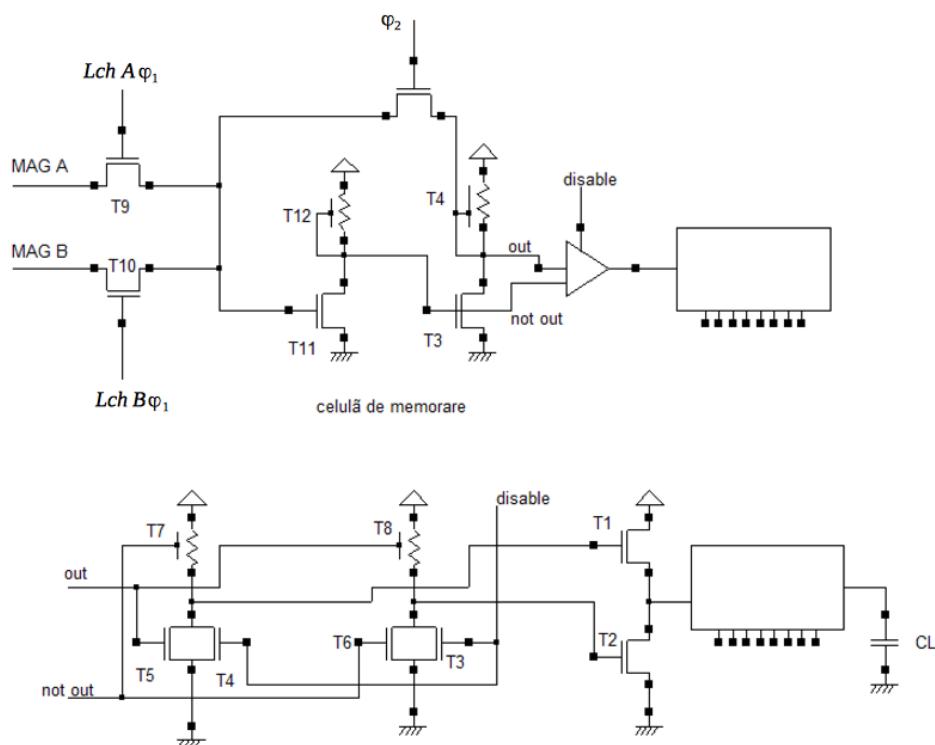


Figura 6.22 Circuitul de comandă

Tranzistoarele T_9 și T_{10} pe durata lui φ_1 pot aduce informația pe grila lui T_{11} dacă semnalele $Lch\ A$ și $Lch\ B$ sunt active. $T_{11} - T_{12}$ și $T_{13} - T_{14}$ formează o celulă de memorare pentru semnalul stocat pe grila lui T_{11} . Celula de memorare furnizează ieșirile out și \overline{out} corespunzătoare semnalului stocat direct și negat. Semnalele out și \overline{out} atacă intrarea unui circuit de comandă realizat cu ajutorul a două circuite NOR cu câte două intrări. Intrările T_4 și T_3 ale circuitului NOR sunt controlate de semnalul $disable$ care face ca ieșirile celor două circuite NOR să devină simultan 0 dacă $disable$ va avea valoarea 1.

În acest fel T₁ și T₂ sunt dezactivare iar plotul văzut din exterior prezintă o impedanță foarte mare. Când T₄ și T₃ sunt la nivel coborât, cele două circuite NOR se comportă ca supratampoane deoarece grilele tranzistoarelor P₇ și P₈ sunt conectate la semnale care sunt în antifază cu semnalele aplicate pe grilele tranzistoarelor trage-jos T₅ și T₆.

Dacă *disable* este inactiv tranzistoarele T₁ și T₂ sunt comandate în antifază și pot comanda plotul respectiv sarcina capacativă fie la V_{DD} fie la masă.

6.2.12. Circuitul de intrare

Implementarea circuitului de intrare este prezentată în Figura 6.23.

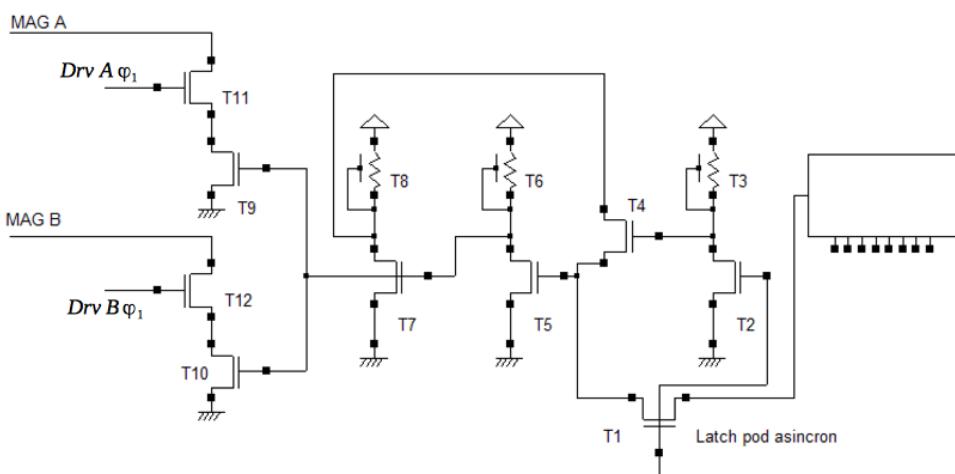


Figura 6.23 Implementarea circuitului de intrare

Informația forțată la plot este însotită de un semnal de control furnizat de sursa Latch pod aplicat pe grila lui T₁, T₅, T₆, T₇, T₈ împreună cu T₄ formează o celulă de memorie a informației care se atochează pe grila lui T₅. Datorită inversorului T₂, T₃ la apariția semnalului Latch pod, T₄ va fi blocat iar T₁ se va deschide ceea ce permite stocarea noii informații pe grila lui T₅.

După dispariția semnalului Latch pod, informația se menține stocată în celula respectivă deoarece T₄ va fi deschis. Informația stocată se aplică pe

6. Proiectarea unui micropresor Multichip

grilele lui T_9 și T_{10} care sunt conectate în serie cu T_{11} și T_{12} între MAG A respectiv MAG B și masă.

Dacă se dorește controlul magistralei MAG A, se aplică semnalul Drv A în φ_1 și astfel semnalul stocat se va regăsi în φ_1 următor, fie pe MAG A, fie pe MAG B.

6.2.13. Estimarea perioadei minime a ceasului plecând de la valorile parametrilor electrici pentru diverse straturi

Pentru φ_1 este de așteptat să se obțină un rezultat conform estimărilor de 50τ . Pentru φ_2 perioada este condiționată de timpul pentru propagarea transportului. Timpul de întârziere pentru propagarea transportului poate fi estimat plecând de la dimensiunile relative ale zonelor sau ariei de metal și difuzie din lanțul de propagare.

Metalul și difuzia acoperă suprafețe de aproximativ 15 și respectiv 8 ori mai mari decât aria tranzistorului de trecere folosit în lanțul de propagare. Capacitățile metalului și respectiv a difuziei pe unitatea de arie sunt de 0,1 și respectiv 0,25 ori mai mari decât capacitatea porții pe unitatea de arie. În concluzie, capacitatea totală a fiecărui etaj din lanțul de propagare este de 4 – 5 ori mai mare decât capacitatea tranzistorului de trecere.

Întârzierea efectivă prin n etaje ale unui lanț de propagare este τn^2 . Astfel pentru $n=4$ vom avea o întârziere de $4,5 * 16\tau = 72\tau$. La această întârziere trebuie să se adauge întârzierea în dublul buffer-ului inversor. Această întârziere este $(1 + k) * \text{timpul de tranzit}$ pentru tranzistorul trage-jos din inversor. În cazul nostru $k = 8$.

Capacitațile parazite trebuie luate în considerare iar întârzierile sunt de aproximativ 30τ . Înțând seamna de faptul că unitatea de execuție este constituită din patru secțiuni de patru biți, ajungându-se la o durată a lui φ_2 de 400τ , aşa cum se poate observa în

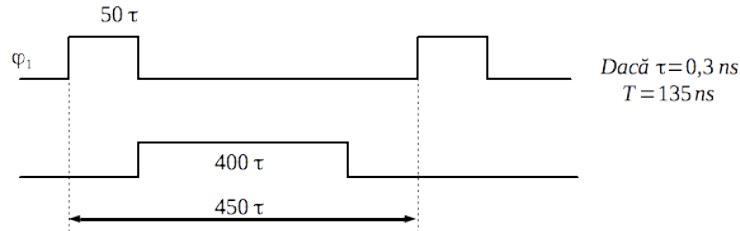


Figura 6.24 Durata unei perioade

6.2.14. Micropogramarea unității de execuție

Sistemul funcționează sub controlul unui ceas bifazic. În faza φ_1 , informațiile sunt transferate pe MAG A și MAG B de la surse spre destinație. În φ_1 un registru nu poate fi și sursă și destinație în același timp. Aceste transferuri sunt controlate de microinstructiuni specifice aduse în φ_2 anterior.

În φ_1 au loc operații aritmetico-logice în UAL cu stocarea rezultatului în registrele de ieșire $out A$ și $out B$. În φ_2 se preîncarcă magistralele MAG A și MAG B la V_{DD} . Pentru controlul acestor operații se folosește un alt tip de instrucțiune activă în φ_2 și citită în φ_1 anterior. Porturile PS și PD pot primi informații în φ_1 curent și le pot furniza în exterior în același timp sau pot primi datele în φ_1 și le pot livra în exterior în φ_2 . Cu unele modificări ar putea fi posibilă livrarea în exterior a informațiilor la cererea sau la solicitarea unui alt subsistem.

Intrarea în PS și PD se face asincron de la pod-ul informațional, fiind preluate asincron pe durata lui φ_1 .

6.2.15. Codificarea operațiilor în Unitatea de Execuție

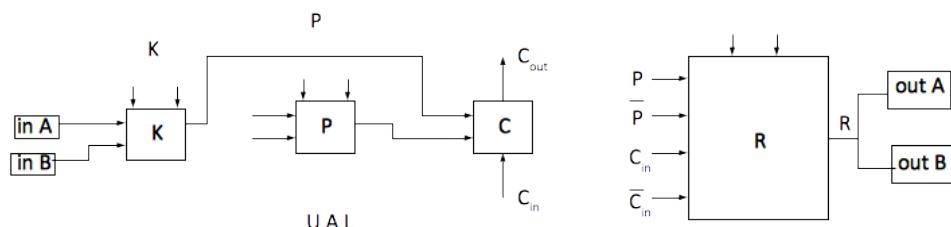


Figura 6.25 Unitatea de execuție

6. Proiectarea unui micropresor Multichip

Adunarea: K = 1000 P = 0110 R = 0110

RIND[16]

biți 0 – 4	neutilizați
bitul 5	bitul indicator anterior
bitul 6	valoarea inversată a transportului în poziția cea mai semnificativă
bitul 7	indicatorul \leq
bitul 8	valoarea negată a relației $>$
bitul 9	valoarea negată a relației $<$
bitul 10	cel puțin semnificativ bit al rezultatului
bitul 11	valoarea negată a indicatorului rezultat = 0
bitul 12	bitul cel mai semnificativ al rezultatului
bitul 13	overflow
bitul 14	valoarea negată C_{out}
bitul 15	bitul indicator curent

6.2.16. Micropogramarea unității de execuție

- I. În φ_1 are loc transferul pe magistrală a informației de la surse la destinații. Formatele sunt următoarele:
1. Transferuri de la sursă la destinație în cadrul unității de execuție
 2. Transferul de la unitatea de execuție la portul literal
 - a. Transferuri unitatea de execuție \rightarrow portul literal
 - b. Transferuri portul literal \rightarrow unitatea de execuție

Observație: Actualmente microinstructiunea se citește în φ_2 anterior

- II. În φ_2 au loc operații aritmetico-logice

I. microinstructiuni active în φ_1

Tipul 1.

2	5	6	5	5
00	B	Dest B	Sursa A	Dest A

Sursa A		Destinație A	
0 n n n n	Registru n	0 n n n n	Registru n
1 0 0 0 0	Terminale PD _n	1 0 0 0 0	PS comandă indicatori
1 0 0 0 1	Latch PD _n	1 0 0 0 1	PS comandă în φ_2
1 0 0 1 0	Terminale PS _t	1 0 0 1 x	PS fără comandă
1 0 0 1 1	Latch PS _t	1 0 1 0 0	PD comandă indicatori
1 0 1 0 0	UAL out Latch A	1 0 1 0 1	PD comandă în φ_2
1 0 1 0 1	UAL out Latch B	1 0 1 1 x	PD fără comandă
1 0 1 1 0	Registru indicatori	1 1 0 0 0	UAL In A
		1 1 0 0 1	UAL In B

Sursa B		Destinația B	
0 n n n n	Registru n	0 0 n n n	Registru n
1 0 0 0 0	Terminal PD	0 1 0 0 0	PS comandă imediată
1 0 0 0 1	Latch PD	0 1 0 0 1	PS comandă în φ_2
1 0 0 1 0	Terminal PS	0 1 0 0 1 x	PS fără comandă
1 0 0 1 1	Latch PS	0 1 0 1 0	PD comandă imediată
1 0 1 0 0	UAL out A	0 1 0 1 0 1	PD comandă în φ_2
1 0 1 0 1	UAL out B	0 1 0 1 1 x	PD comandă imediată
		0 1 1 0 x x	UAL în B
		1 0 n n n	În B se încarcă cu shiftout
		1 1 n n n	În B se încarcă cu constanta de deplasare

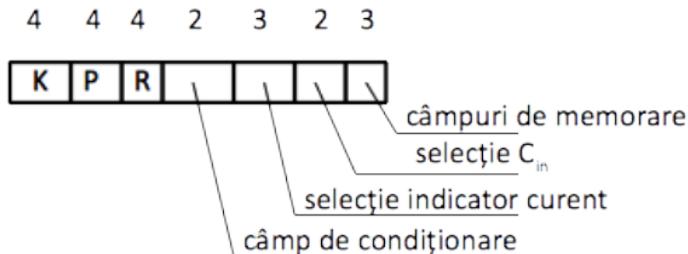
Tipul 2.

16 5

1 x	Literal – constantă	Dest în reg A
-----	---------------------	---------------

II. microinstructiuni care operează în φ_2

6. Proiectarea unui micropresor Multichip



Operație	K	P	R	C_{in}	Condiționarea
A + B	1	6	6	0	0
A + B + C_{in}	1	6	6	1	0
A - B	2	9	6	2	0
B - A	4	9	6	2	0
A - B - C_{in}	2	9	6	1	0
B - A - C_{in}	4	9	6	1	0
- A	12	3	6	2	0
- B	10	5	6	2	0
A + 1	3	12	6	2	0
B + 1	5	10	6	2	0
A - 1	12	3	9	2	0
B - 1	10	5	9	2	0
A \wedge B	0	8	12	0	0
A \vee B	0	14	12	0	0
A \oplus B	0	6	12	0	0
not A	0	3	12	0	0
not B	0	5	12	0	0
A	0	12	12	0	0
B	0	10	12	0	0
Pasul de înmulțire	1	14	14	0	1
Pasul de impărțire	3	15	15	0	2
81,SAU condiționarea	0	14	12	0	3

6.2.17. Câmpul de condiționare

Câmp condițional	Bit indicator	K	P	R	Observații
0 (0, 0)	0	-----	-----	-----	Câmpurile nu se modifică
	1	-----	-----	-----	

CMOS VLSI

1 (0, 1)	0	--- 0	- - 0 -	- - 0 -	Pas de înmulțire
	1	----	0 ---	0 ---	
2 (1, 0)	0	--- 0	- 0 0 -	- 0 0 -	Pas de împărțire
	1	- 0 0 -	0 - - 0	0 - - 0	
3 (1, 1)	0	----	----	----	AND / OR condițional
	1	----	- 0 0 -	----	

6.2.18. Câmpul de memorare

x x x x în urma efectuării operațiilor în UAL se pot încărca următoarele destinații

1 x x x se încarcă registrele indicatorilor

x 1 x x se încarcă latch-ul out A cu un rezultat

x x 1 x se încarcă latch-ul out B cu un rezultat

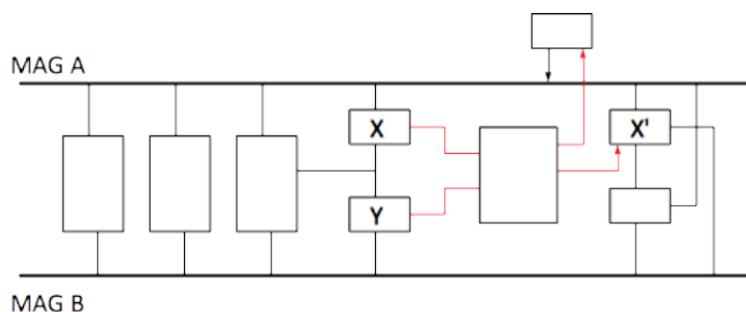
x x x 1 câmpul literal este încărcat cu conținutul magistralei A

0 0 0 0 nu există nici un efect

1 1 x x se încarcă și registrele indicatorilor și latch-ul out A

Exemplu – Înmulțirea a doi operanzi: $Z = X * Y$

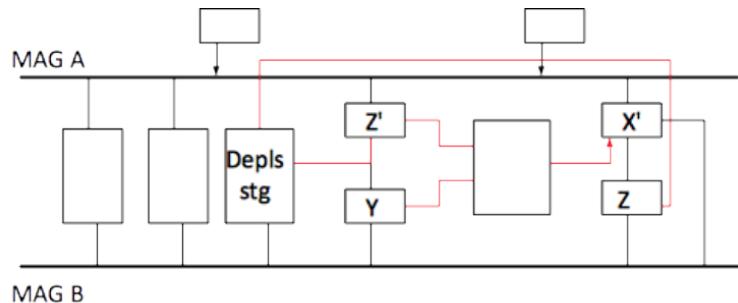
Pasul 1.



Se introduc valorile X, Y în $In A$ și $In B$. Pentru efectuarea înmulțirii, X este deplasat la stânga. În φ_2 vom avea $UAL \text{ outA} \leftarrow (UAL \text{ shift_stg}) \leftarrow UAL \text{ In A}$

Pasul 2.

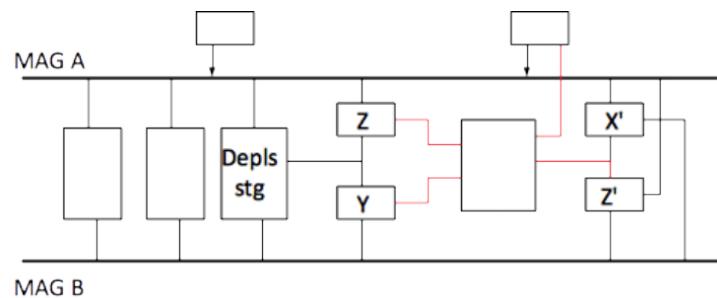
6. Proiectarea unui micropresor Multichip



UAL In A \leftarrow Out shifter \leftarrow UAL Out B

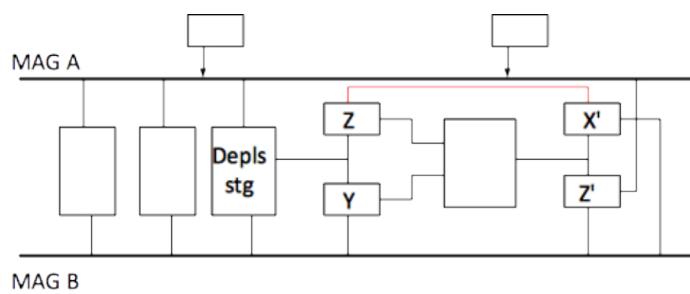
Pentru a da o constantă de deplasare cu valoarea 1, se fac următoarele operații la nivelul registrelor: R[1] \leftarrow MAG B \leftarrow R[0].

Pasul 3.



UAL Out B \leftarrow UAL pas_înmulțire \leftarrow adunare condiționată sau transfer

Pasul 4.



În φ_1 , UAL In A \leftarrow MAG A \leftarrow UAL Out A

7. IMPLEMENTAREA ÎN VLSI A SISTEMELOR DE CONDUCERE

7.1. Controlere digitale

Ingineria controlului este dedicată implementării unor sisteme de conducere care să permită obținerea unor performanțe impuse atât în ceea ce privește calitatea evoluției în timp a procesului cât și satisfacerea unor criterii particulare de tipul: energie consumată minimă, timp de evoluție minim etc. Strategia conducerii și tehnicele de realizare depind în mare măsură de tipul procesului condus, procese clasice de tip chimic, mecanic sau electric sau sisteme moderne bazate pe structuri robotice avansate, sisteme complexe de conducere a traficului, sisteme neuro-cibernetice etc. În majoritatea acestor cazuri, mărimile ce intervin în procesul de conducere sunt de tip analogic și procesarea lor în formă digitală impune introducerea unor convertoare analog-numerice și numeric-analogice care să permită implementarea unor algoritmi de tip numeric. Propriu-zis, tehnicele de implementare se bazează pe utilizarea unor controlere digitale a căror configurație asigură performanțele dorite. În foarte multe aplicații, implementarea acestor controlere este bazată pe utilizarea unor microcontrolere a căror resurse și tempi de comutare satisfac cerințele impuse. De exemplu, microcontrolerele familei Intel sau Motorola pe 8 sau 16 biți oferă un suport adecvat pentru multe aplicații. Totuși, în conducerea unor procese moderne complexe, cerințele legate de complexitatea algoritmilor de conducere și restricțiile impuse de timpii de comutare impun condiții extrem de severe care în mare măsură nu sunt satisfăcute de performanțele microprocesoarelor clasice. De exemplu conducerea sistemelor robotice controlate video impun bucle de conducere multiple a căror implementare prin microcontrolerele convenționale este extrem de dificilă sub raportul performanțelor, vitezei de procesare și fiabilității [10].

7. Implementarea în VLSI a sistemelor de conducere

În aceste condiții, utilizarea unor tehnici VLSI, ce oferă ca suport tehnic un singur chip, cu o arhitectură și aritmetică dedicată, îmbunătățește calitatea implementării datorită vitezei mari de procesare de tip paralel a operațiilor aritmétice, fiabilității sporite, reducerii efectelor perturbațiilor determinate de zgomot, sensibilității mai mari și prețului de cost mai mic. Toate aceste elemente determină o creștere substanțială a ponderii controlerelor digitale implementate prin tehnologii VLSI.

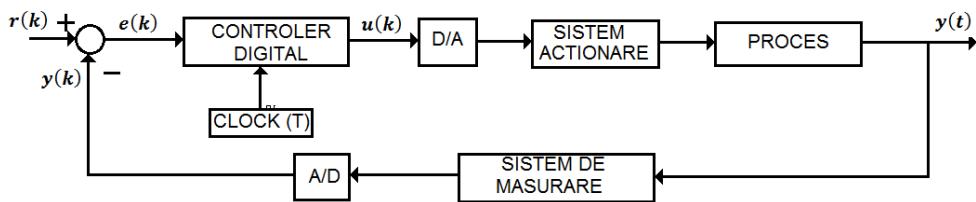


Figura 7.1 Conducerea numerică a unui proces

În Figura 7.1 este prezentată schema bloc generală a conducerii numerice a unui proces. Procesul evoluează continuu prin variabila de timp t cu ieșire controlată $y(t)$, iar condescerea este realizată discret la momente de timp $t_k = kT$, unde T este perioada de eşantionare a semnalului digital. Se va nota prin $y(t_k) = y(kT) = y(k)$ valorile obținute la momentele de eşantionare, $k = 1, 2, \dots$. Controlerul digital implementează legea de condescere $u(k)$ pe baza semnalului de eroare $e(k)$,

$$e(k) = r(k) - y(k) \quad 7.1$$

unde $y(k)$ generat de traductorul de măsurare convertit digital, iar $r(k)$ este mărimea de referință impusă.

Implementarea controlerului digital poate fi realizată în diferite soluții, aceste bazându-se pe utilizarea unor componente standard: elemente de întârziere, elemente de adunare și multiplicare. Există diverse metode de implementare cu aceeași soluție calitativă în condițiile în care se presupune că precizia operațiilor este infinită. În condițiile unei precizii finite, modul de selectare al arhitecturilor de implementare devine extrem de important.

Forma generală a funcției de transfer a unui controler digital este:

$$Y_C(z) = \frac{U(z)}{E(z)} = \frac{b_0 z^n + b_1 z^{n-1} + \dots + b_{n-1} z + b_n}{z^n + a_1 z^{n-1} + \dots + a_{n-1} z + a_n} \quad 7.2$$

unde a_i, b_i sunt coeficienți reali (unii pot fi zero), iar z este variabila asociată transformării \mathbf{z} . Este bine cunoscut în literatura de specialitate că orice funcție de transfer al unui controler poate fi rescrisă în forma discretă de mai sus [7]. Relația (7.2) poate fi rescrisă în forma:

$$Y_C(z) = \frac{U(z)}{E(z)} = \frac{b_0 + b_1 z^{-1} + \dots + b_{n-1} z^{-(n-1)} + b_n z^{-n}}{1 + a_1 z^{-1} + \dots + a_{n-1} z^{-(n-1)} + a_n z^{-n}} \quad 7.3$$

Pentru implementarea acestei funcții de transfer se utilizează proceduri nerecursive și recursive. În cazul recursiv, legea de conducere poate fi exprimată sub forma:

$$u(k) = -a_1 u(k-1) - a_2 u(k-2) - \dots - a_n u(k-n) \\ + b_0 e(k) + b_1 e(k-1) + \dots + b_n e(k-n) \quad 7.4$$

controlul actual $u(k)$ fiind exprimat de valorile actuale și trecute ale erorii precum și de valorile trecute ale controlului. Într-o procedură nerecursivă, legea de control are forma:

$$u(k) = b_0 e(k) + b_1 e(k-1) + \dots + b_n e(k-n) \quad 7.5$$

ceea ce indică faptul că controlul la momentul k este determinat numai din valorile trecute ale erorii. În general, implementarea nerecursivă se obține din (7.3) prin împărțirea numărătorului la numitor și trunchierea seriei obținute în z^{-k} . În acest caz, obținerea unei precizii suficiente de bune impune creșterea lui k la valorile maxim permise.

În cele ce urmează vor fi descrise câteva tehnici de implementare pentru ambele proceduri.

7.1.1. Implementarea recursivă directă

Această procedură derivă din (7.3) care poate fi rescrisă sub forma:

$$\begin{aligned} Y_C(z) &= \frac{U(z)}{E(z)} \\ &= \frac{(b_0 + b_1 z^{-1} + \dots + b_{n-1} z^{-(n-1)} + b_n z^{-n})X(z)}{(1 + a_1 z^{-1} + \dots + a_{n-1} z^{-(n-1)} + a_n z^{-n})X(z)} \end{aligned} \quad 7.6$$

sau

$$U(z) = (b_0 + b_1 z^{-1} + \dots + b_{n-1} z^{-(n-1)} + b_n z^{-n})X(z) \quad 7.7$$

7. Implementarea în VLSI a sistemelor de conducere

$$E(z) = (1 + a_1 z^{-1} + \dots + a_{n-1} z^{-(n-1)} + a_n z^{-n}) X(z) \quad 7.8$$

Exprimarea variabilelor u, e în funcție de variabila intermediară x permite implementarea obținută în Figura 7.2.

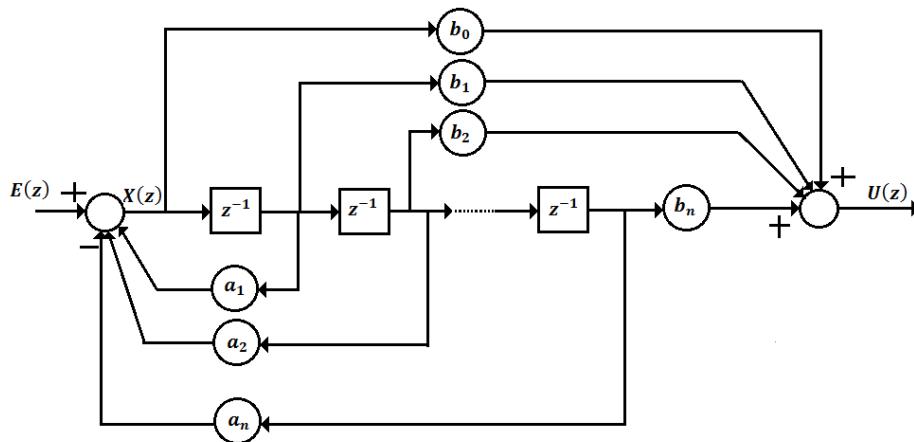


Figura 7.2 Schema bloc într-o implementare recursivă directă

Această procedură este sensibilă la erori determinate limitelor impuse de implementarea coeficientilor numerici ai funcției de transfer, erori ce cresc odată cu utilizarea unor funcții de ordin superior.

7.1.2. Implementarea recursivă în cascadă

Această procedură permite utilizarea unor modele de ordin inferior legate în cascadă ceea ce duce la o micșorare a sensibilității la erorile de implementare numerică [10],

$$Y_c(z) = Y_{c1}(z) \cdot Y_{c2}(z) \dots Y_{cm}(z) \quad 7.9$$

unde $Y_{ci}(z)$ sunt alese ca funcții de transfer de ordinul 1

$$Y_{ci}(z) = \prod \frac{(1+b_i z^{-1})}{(1+a_i z^{-1})} \quad 7.10$$

sau de ordinul 2

$$Y_{cj}(z) = \prod \frac{(1+b_j z^{-1} + c_j z^{-2})}{(1+a_j z^{-1} + d_j z^{-2})} \quad 7.11$$

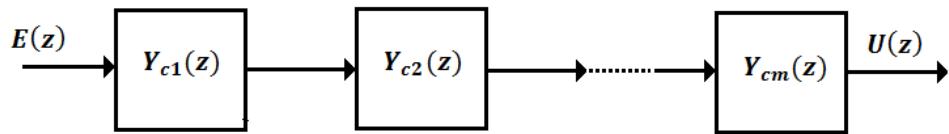


Figura 7.3 Schema bloc a unei reprezentări recursive în cascadă

Funcțiile de transfer (7.10) și (7.11) se pot ușor implementa pe baza tehnicii discutate anterior (Figura 7.4, respectiv Figura 7.5).

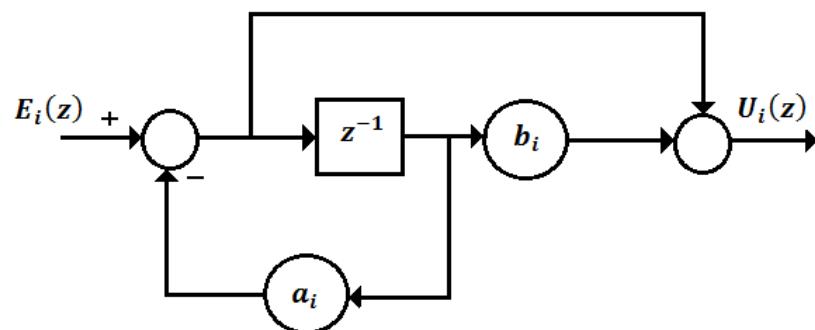


Figura 7.4 Reprezentarea recursivă a unei funcții de ordinul 1

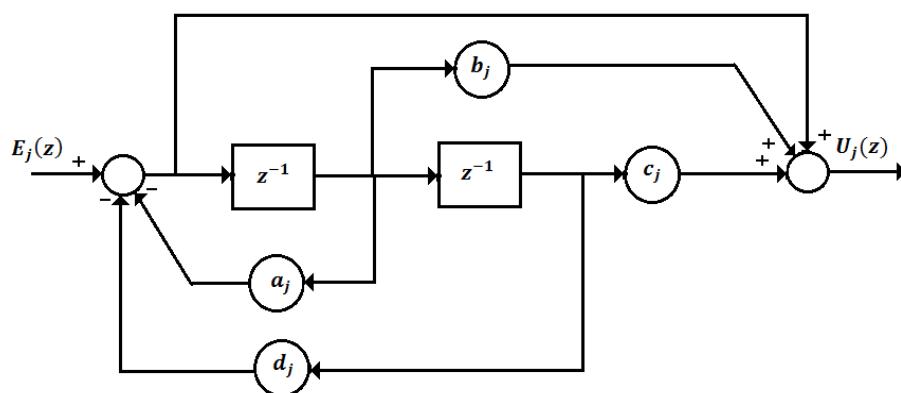


Figura 7.5 Reprezentarea recursivă a unei funcții de ordinul 2

7. Implementarea în VLSI a sistemelor de conducere

7.1.3. Implementarea recursivă în paralel

Această procedură se bazează pe descompunerea funcției de transfer sub forma [7,10]

$$Y_c(z) = K + Y_{c1}(z) + Y_{c2}(z) + \dots + Y_{cp}(z) \quad 7.12$$

unde K este o constantă iar $Y_{ci}(z)$ sunt funcții de transfer de ordin inferior,

$$Y_{ci}(z) = \frac{b_i}{1 + a_i z^{-1}} \quad 7.13$$

$$Y_{cj}(z) = \frac{b_j + c_i z^{-1}}{1 + a_j z^{-1} + d_j z^{-2}} \quad 7.14$$

Schema globală a unei astfel de implementări este prezentată în Figura 7.6.

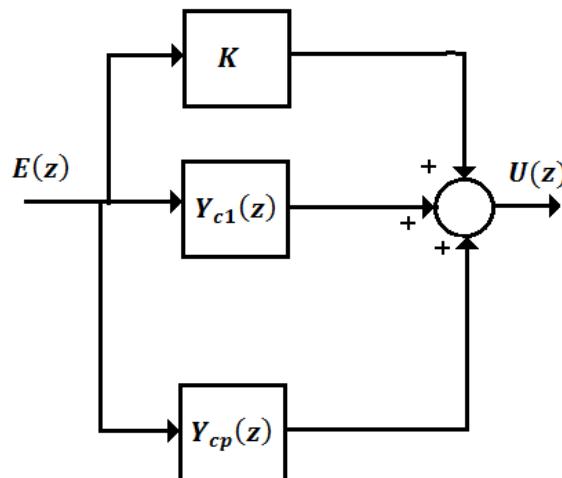


Figura 7.6 Reprezentarea recursiv paralelă

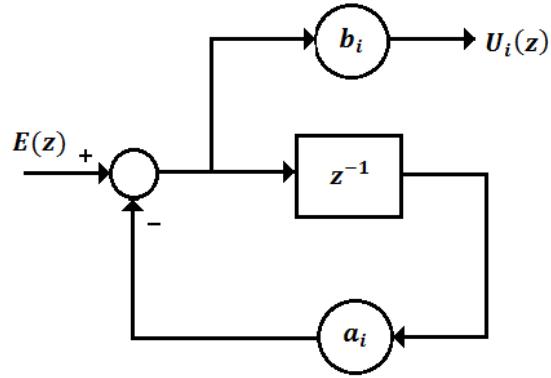


Figura 7.7 Reprezentarea recursivă a funcției de transfer (7.13)

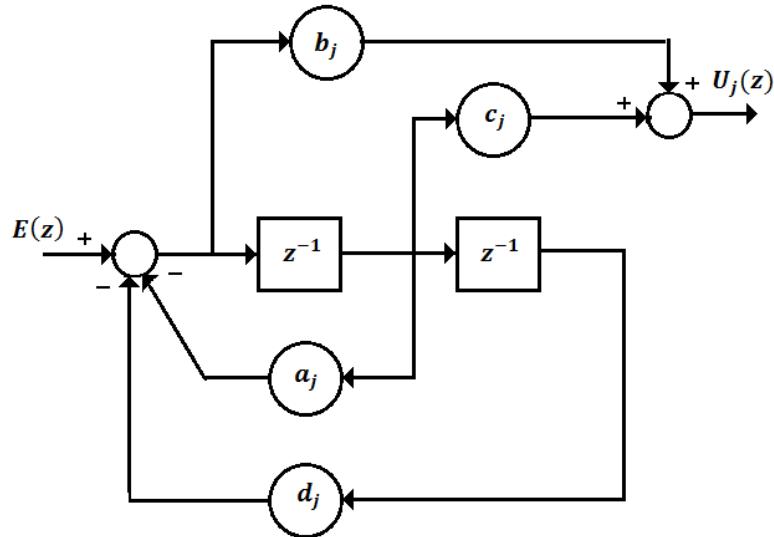


Figura 7.8 Reprezentarea recursivă a funcției de transfer (7.14)

7.1.4. Implementarea nerecursivă

Această procedură se bazează pe exprimarea legii de conducere (7.5) care poate fi rescrisă sub forma:

$$U(z) = (b_0 + b_1 z^{-1} + \dots + b_{n-1} z^{-(n-1)} + b_n z^{-n})E(z) \quad 7.15$$

și a cărei implementare poate fi urmărită în Figura 7.9.

7. Implementarea în VLSI a sistemelor de conducere

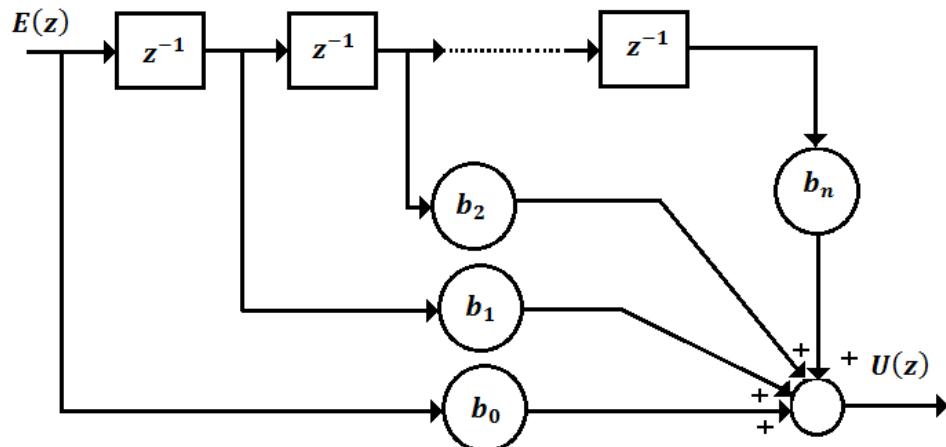


Figura 7.9 Schema bloc a procedurii nerecursive

7.2. Implementarea controlerelor

Algoritmii de conducere sunt definiți prin ecuațiile discutate în secțiunea anterioară, asociați cu diferite arhitecturi de implementare. În aceste ecuații, se va considera că perioada de eșantionare și de operare este T , valoarea erorii la momentul curent, starea actuală, este $e(k) = e(kT)$, ieșirea generată la momentul actual este $u(k) = u(kT)$, iar variabilele $e(k-1), e(k-2), \dots, e(k-n)$, $u(k-1), u(k-2), \dots, u(k-n)$ reprezintă valorile precedente ale erorii și respectiv ieșirii la momentele, $T, 2T, \dots$

7.2.1. Controler recursiv direct

Controlerul este descris prin ecuațiile (7.7), (7.8) ce pot fi rescrise sub forma:

$$u(k) = b_0x(k) + b_1x(k-1) + \dots + b_nx(k-n) \quad 7.16$$

$$x(k) = e(k) - a_1x(k-1) - a_2x(k-2) - \dots - a_nx(k-n) \quad 7.17$$

și pot fi transpusă în flowchart-ul din Figura 7.10.

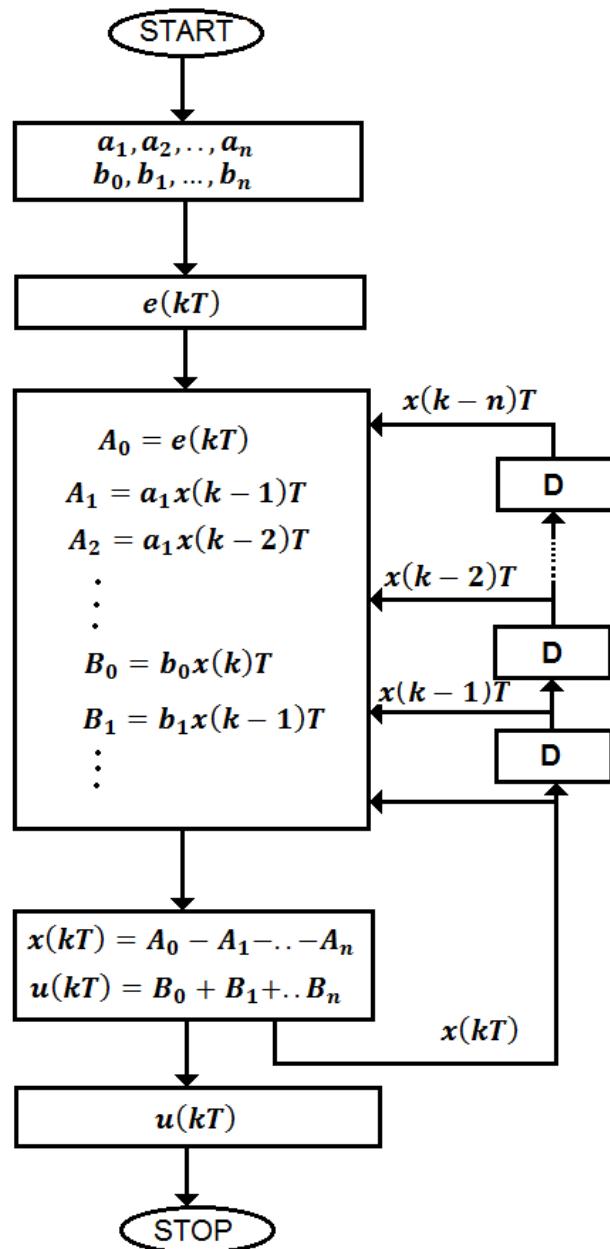


Figura 7.10 Flowchart-ul unui controler recursiv direct

7. Implementarea în VLSI a sistemelor de conducere

În Figura 7.11 este prezentată implementarea VLSI a controlerului. Semnalele de intrare și ieșire precum și coeficienții algoritmului sunt implementați pe registre de 8 biți. Pentru o precizie mai bună pot fi utilizate registre de 16 biți. Pentru operațiile aritmetice se utilizează aritmetică cu virgulă fixă.

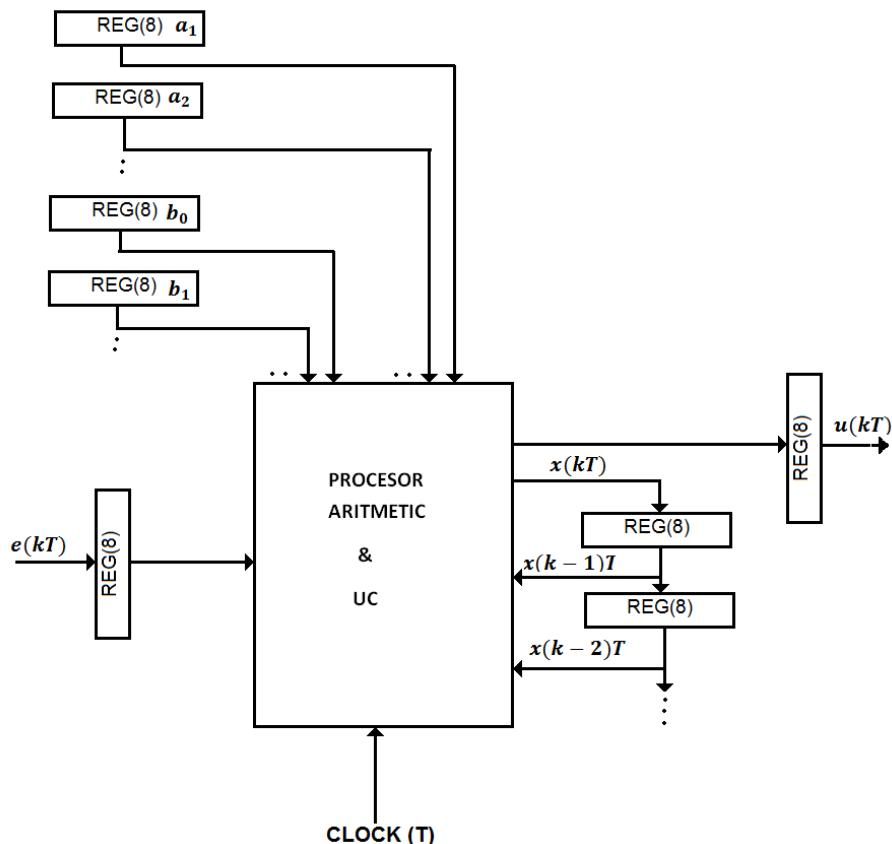


Figura 7.11 Implementarea VLSI a controlerului recursiv direct

Variabilele de intrare și ieșire $e(kT), u(kT)$ sunt memorate în registre pe 8 biți. De asemenea, registre de 8 biți sunt utilizate ca elemente de întârziere pentru memorarea valorilor $x(k - 1)T, x(k - 2)T \dots$. Operațiile de multiplicare și adunare sunt realizate cu procesoare separate. O unitate de control UC supervizează desfășurarea secvențială a operațiilor.

O tehnică similară de implementare poate fi utilizată pentru implementarea celorlalte arhitecturi de controlere recursive.

7.2.2. Controler nerecursiv

Implementarea controlerului nerecursiv este realizată pe baza relației (7.15) rescrisă sub forma:

$$u(k) = b_0 e(k) + b_1 e(k - 1) + \dots + b_n e(k - n) \quad 7.18$$

Flowchart-ul asociat este prezentat în Figura 7.12, iar implementarea controlerului este dată în Figura 7.13.

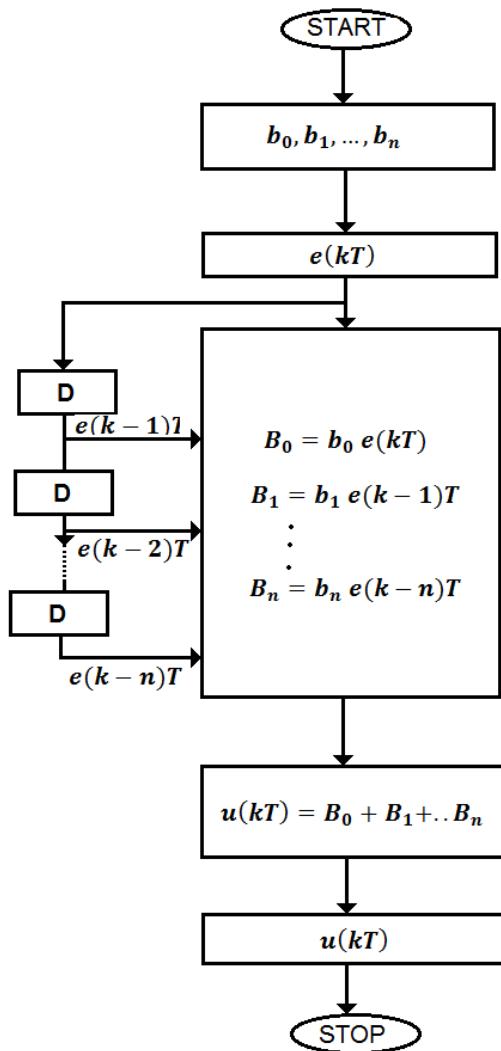


Figura 7.12 Flowchart-ul unui controler nerecursiv

7. Implementarea în VLSI a sistemelor de conducere

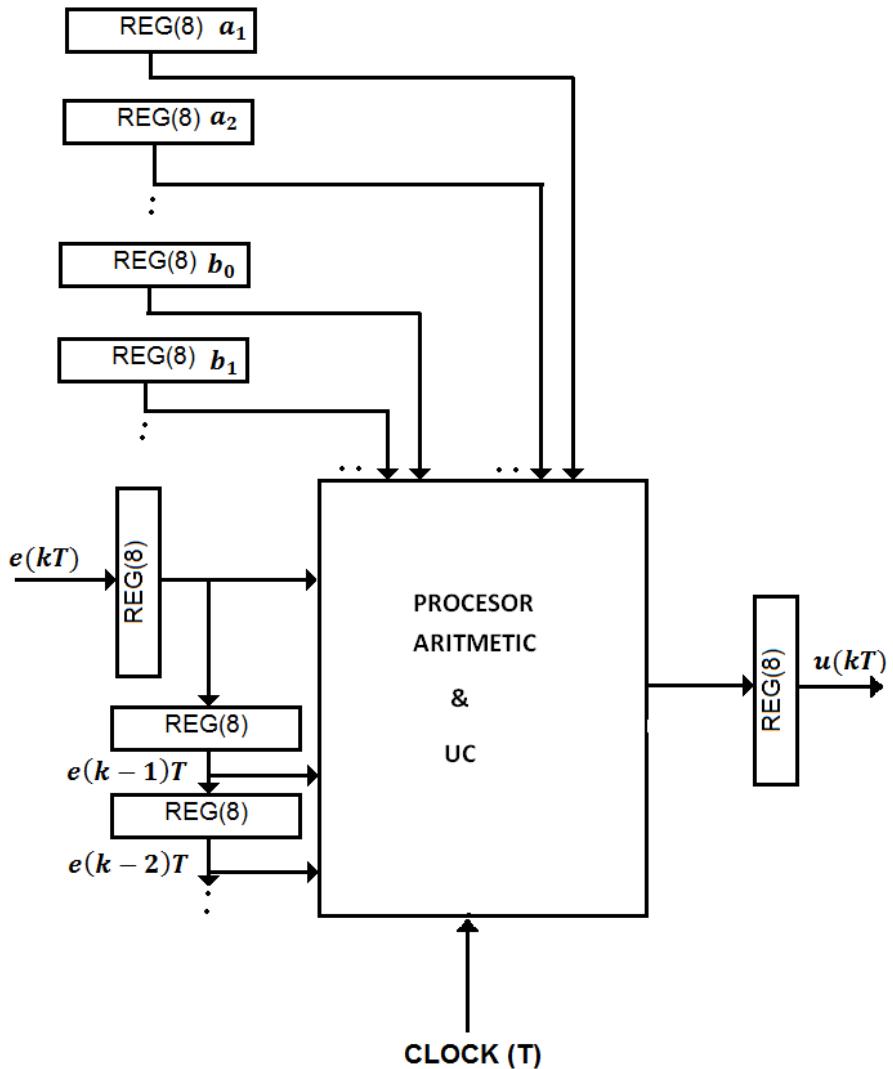


Figura 7.13 Implementarea VLSI a controlerului nerecursiv

7.3. Conducerea unui robot mobil

Roboții mobili cu roți sunt arhitecturi mecanice la care funcția de mișcare este obținută prin roți dispuse în diferite configurații. În exemplul analizat va fi discutat un robot mobil cu două roți la care controlul direcției

de mișcare este obținut prin modificarea vitezelor de rotație ale roților (Figura 7.14). Poziția curentă a robotului este definită prin:

$$P = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad 7.19$$

unde (x, y) reprezintă coordonatele de poziție ale centrului de greutate al robotului în raport cu un sistem de referință impus. iar θ este unghiul ce definește orientarea mișcării. Considerând că ω_s, ω_d reprezintă vitezele unghiulare ale roților, vitezele tangențiale vor fi:

$$\begin{aligned} v_s &= \omega_s r \\ v_d &= \omega_d r \end{aligned} \quad 7.20$$

unde r este raza roții.

Modelul cinematic este definit prin relațiile:

$$\begin{aligned} \dot{x} &= v \cos \theta \\ \dot{y} &= v \sin \theta \\ \dot{\theta} &= \omega \end{aligned} \quad 7.21$$

unde v, ω reprezintă vitezele liniare și unghiulare ale centrului de greutate.

Controlul vitezei de deplasare și orientarea mișcării sunt controlate prin controlul vitezelor unghiulare ale celor două roți ω_s, ω_d ,

$$\begin{aligned} v &= r \frac{\omega_s + \omega_d}{2} \\ \omega &= r \frac{\omega_s - \omega_d}{a} \end{aligned} \quad 7.22$$

Sistemul de conducere al mișcării este realizat sub forma unei legi de conducere cu controler PD (Figura 7.14),

$$u_{\omega_s}(t) = k_p e_\theta(t) + k_d \dot{e}_\theta(t) \quad 7.23$$

$$u_{\omega_d}(t) = -k_p e_\theta(t) - k_d \dot{e}_\theta(t) \quad 7.24$$

7. Implementarea în VLSI a sistemelor de conducere

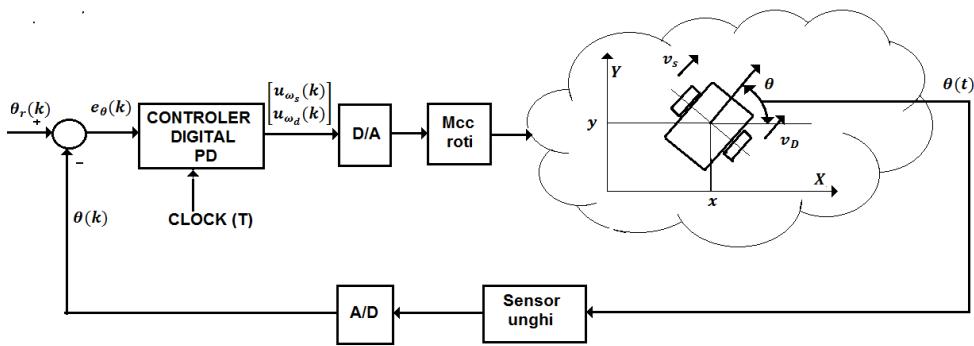


Figura 7.14 Schema bloc a conducerii unui robot mobil

unde $e_\theta(t)$ este eroarea de poziție în raport cu o referință prescrisă $\theta_r(t)$,

$$e_\theta(t) = \theta_r(t) - \theta(t) \quad 7.25$$

iar $u_{\omega_s}, u_{\omega_d}$ sunt variabilele de control ale motoarelor de c.c. ce acționează fiecare roată.

Transpusă în forma discretă, cele două legi de conducere (7.23), (7.24) devin

$$u_{\omega_s}(k) = k_p e_\theta(k) + \frac{k_d}{T} (e_\theta(k) - e_\theta(k-1)) \quad 7.26$$

$$u_{\omega_d}(k) = -k_p e_\theta(k) - \frac{k_d}{T} (e_\theta(k) - e_\theta(k-1)) \quad 7.27$$

sau

$$u_{\omega_s}(k) = (k_p + \frac{k_d}{T}) e_\theta(k) - \frac{k_d}{T} e_\theta(k-1) \quad 7.28$$

$$u_{\omega_d}(k) = (-k_p - \frac{k_d}{T}) e_\theta(k) + \frac{k_d}{T} e_\theta(k-1) \quad 7.29$$

Flowchart-ul asociat este similar cu cel prezentat în Fig 7.12, iar implementarea controlerului este dată în Fig 7.15.

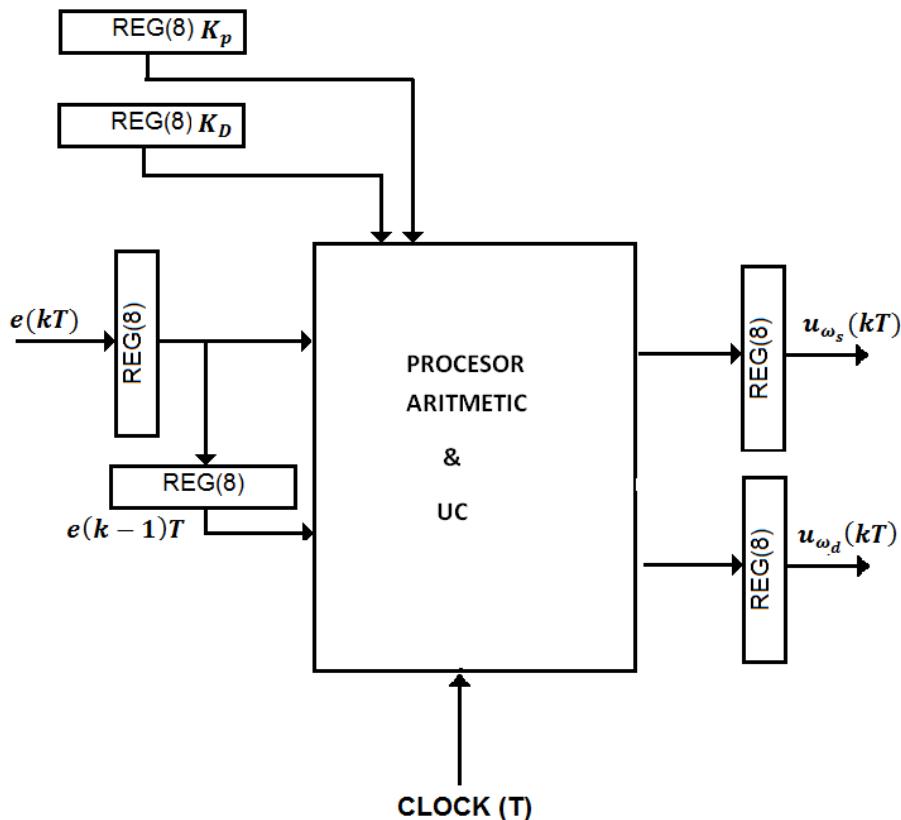


Figura 7.15 Implementarea VLSI a controlerului PD pentru conducerea unui robot mobil

7.4. Conducerea unui robot industrial (controler PID)

O procedură larg utilizată în conducerea roboților industriali este bazată pe utilizarea controlerelor PID datorită performanțelor deosebite, calității procesului de conducere și robusteștii față de perturbațiile spațiului de operare (Fig 7.16).

7. Implementarea în VLSI a sistemelor de conducere

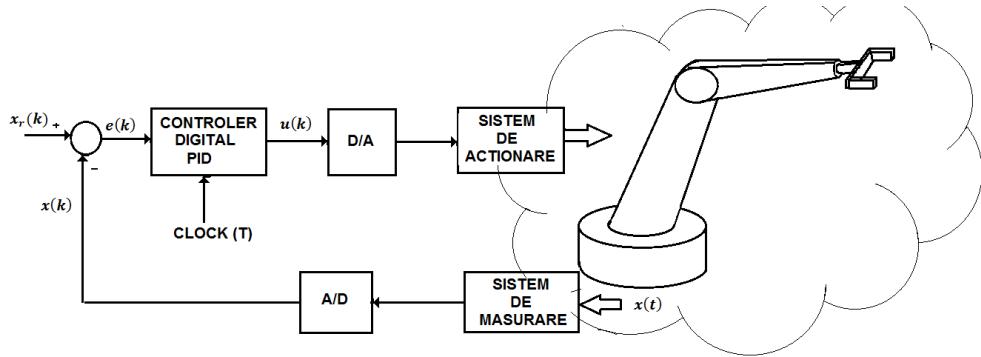


Figura 7.16 Schema bloc a conducerii unui robot cu controler PID digital

Acstea controlere sunt asociate câte unei axe de mișcare a robotului, operând independent. Considerând că traiectoria de referință a robotului este definită prin $x_r(t)$, unde x este variabila asociată axei de mișcare, eroarea de poziție va fi:

$$e(t) = x_r(t) - x(t) \quad 7.30$$

Controlerul PID va realiza legea de control:

$$u(t) = k_p e(t) + k_I \int_0^t e(t) dt + k_D \frac{de(t)}{dt} \quad 7.31$$

Într-o transpunere discretă, relația (7.4.2) se poate scrie sub forma:

$$u(k) = k_p e(k) + k_I T \sum_{i=1}^k e(i) + \frac{k_D}{T} (e(k) - e(k-1)) \quad 7.32$$

O tehnică des utilizată constă în determinarea predictivă a erorii $e(k)$ sub forma:

$$e(k) = e(k-1) + (e(k-1) - e(k-2)) \quad 7.33$$

Substituind această relație în (7.32) se obține:

$$u(k) = (2k_p + 2k_I T + \frac{k_D}{T}) e(k-1) - (k_p + k_I T + \frac{k_D}{T}) e(k-2) \quad 7.34$$

$$\frac{k_D}{T})e(k-2) + k_I T \sum_{i=1}^{k-1} e(i)$$

sau

$$u(k) = K_1 e(k-1) + K_2 e(k-2) + K_3 \sum_{i=1}^{k-1} e(i) \quad 7.35$$

unde

$$\begin{aligned} K_1 &= (2k_P + 2k_I T + \frac{k_D}{T}) \\ K_2 &= -(k_P + k_I T + \frac{k_D}{T}) \\ K_3 &= k_I T \end{aligned} \quad 7.36$$

Relația (7.35) permite o primă metodologie de implementare a controlerului utilizând valorile memorate ale erorii, $e(k-1), e(k-2), \dots, e(k-n)$, ceea ce impune prezența a n registre corespunzătoare. O simplificare a tehnologiei de implementare se poate obține prin memorarea ultimei secvențe de conducere:

$$u(k-1) = K_1 e(k-2) + K_2 e(k-3) + K_3 \sum_{i=1}^{k-2} e(i) \quad 7.37$$

Din (7.35), (7.37) se obține

$$\begin{aligned} u(k) &= u(k-1) + (K_1 + K_3)e(k-1) \\ &\quad + (K_2 - K_1)e(k-2) - K_2 e(k-3) \end{aligned} \quad 7.38$$

ceea ce minimizează efortul de implementare la patru registre, trei pentru eroare și unul pentru ultima secvență de control. În Figura 7.17 este implementată soluția dată de relația (7.35) unde un registru de 16 biti stochează $\sum_{i=1}^{k-1} e(i)$. În Figura 7.18 este prezentată soluția (7.38) unde s-au notat:

$$\begin{aligned} K_{-1} &= (K_1 + K_3) \\ K_{-2} &= (K_2 - K_1) \\ K_{-3} &= -K_2 \end{aligned} \quad 7.39$$

7. Implementarea în VLSI a sistemelor de conducere

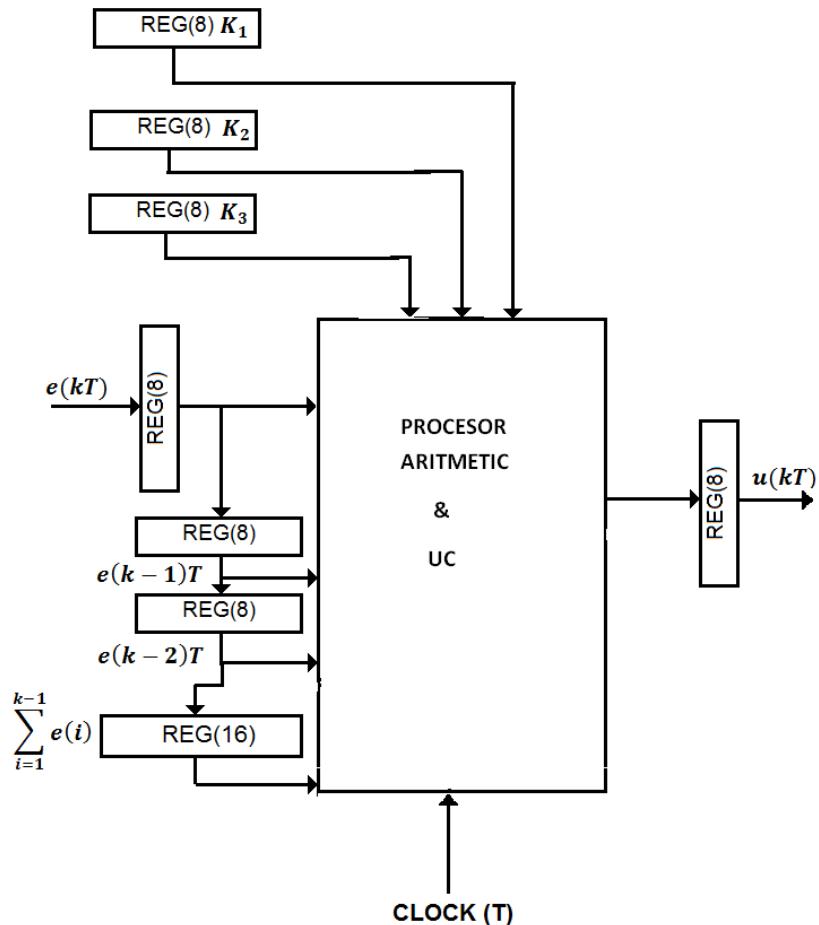


Figura 7.17 Implementarea controlerului PID după relația (7.35)

CMOS VLSI

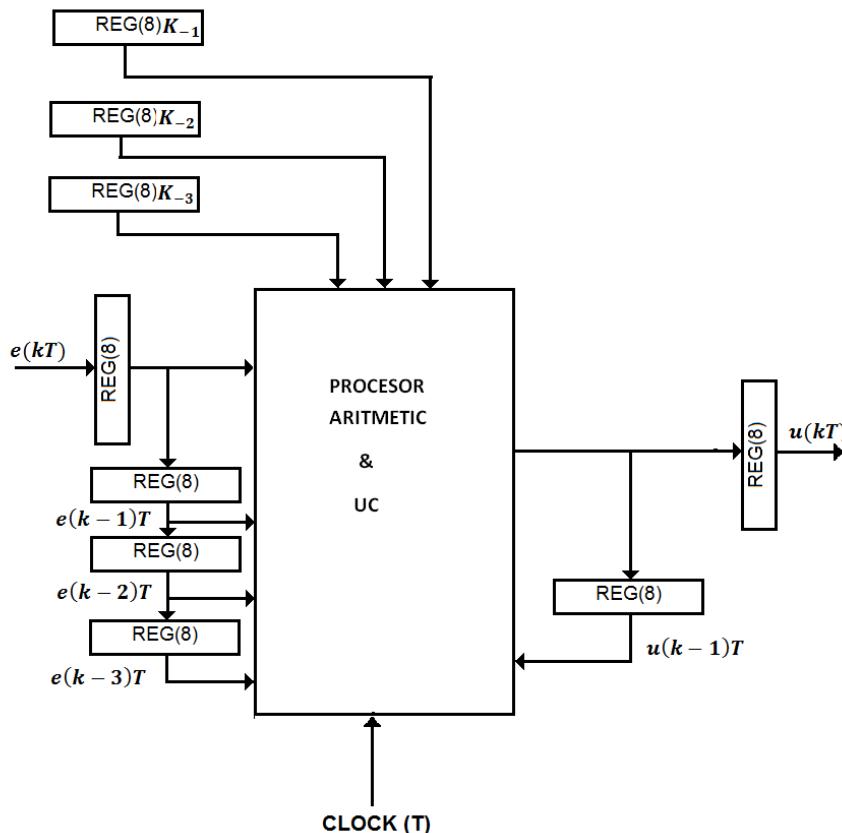


Figura 7.18 Implementarea controlerului PID după relația (7.38)

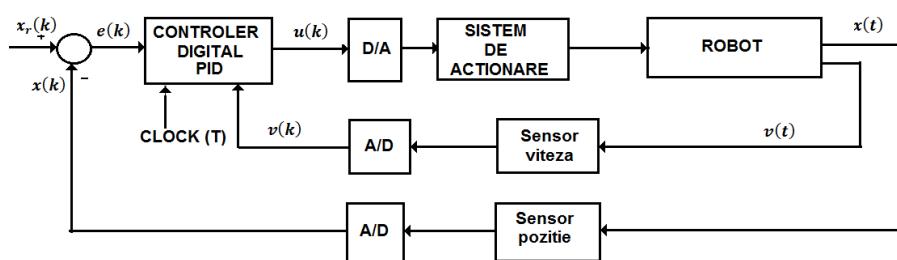


Figura 7.19 Schema bloc a conducerii unui robot după poziție și viteză

În unele cazuri, componenta derivativă, în construcția algoritmului de control, este obținută prin măsurarea vitezei, eliminându-se astfel erorile provocate de această componentă. Algoritmul de control va avea o formă similară cu cea prezentată în (7.31).

7. Implementarea în VLSI a sistemelor de conducere

$$u(t) = k_P e(t) + k_I \int_0^t e(t)dt + k_D v(t) \quad 7.40$$

sau, transpus în formă discretă va fi:

$$u(k) = u(k-1) + K_P e(k) + (K_I T - K_P)e(k-1) \\ + K_D(e_v(k) - e_v(k-1)) \quad 7.41$$

unde e_v este eroarea de viteză,

$$e_v(k) = -v(k) \quad 7.42$$

pentru un semnal de referință de viteză constantă.

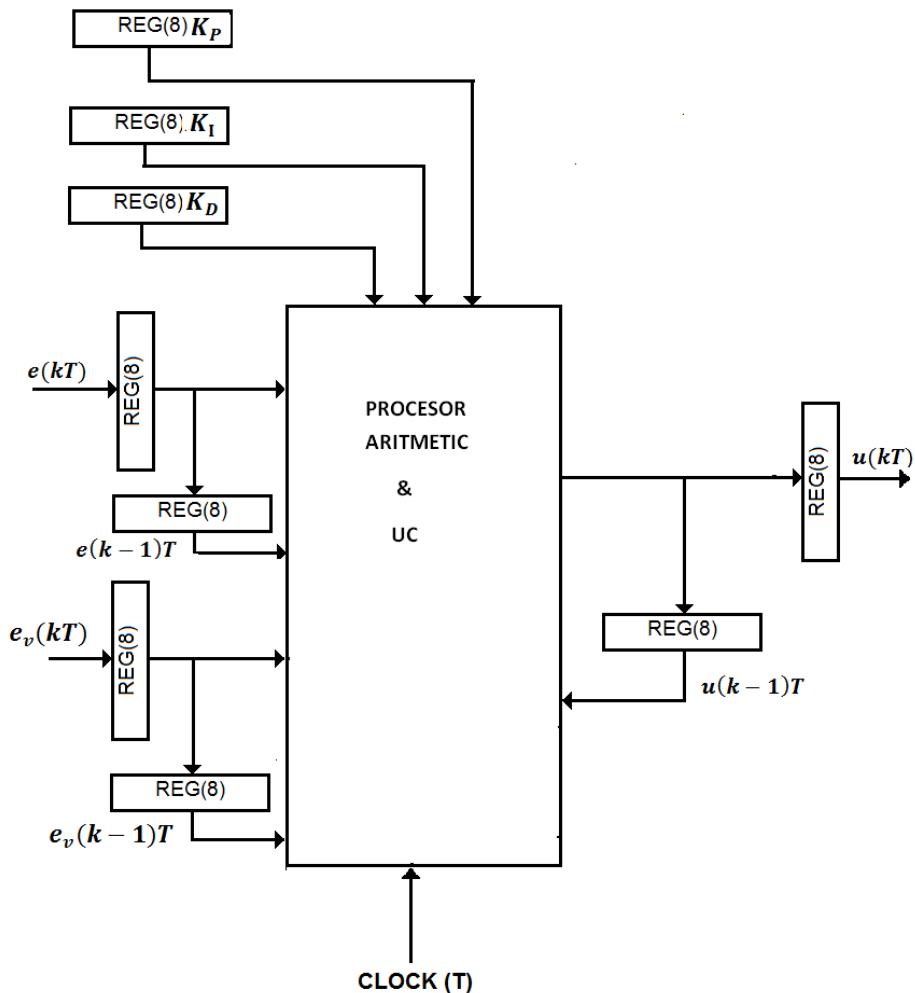


Figura 7.20 Implementarea controlerului PID (poziție și viteză)

7.5. Conducerea unui robot prin metoda variabilelor de stare

Se va analiza sistemul de conducere al articulației unui robot a cărui dinamică este descrisă prin variabile de stare. Se va considera că sistemul de acționare este realizat printr-un motor de c.c. a cărui funcționare este descrisă de ecuațiile:

$$k_b \frac{d\theta(t)}{dt} + R_a i_a(t) = u(t) \quad 7.43$$

$$J \frac{d^2\theta(t)}{dt^2} = k_T i_a(t) \quad 7.44$$

unde $\theta(t)$ este variabila unghiulară de poziție, $i_a(t)$, $u(t)$ sunt curentul, respectiv tensiunea de control, iar k_b , R_a , J , k_T sunt parametrii electrici și mecanici ai sistemului. Variabilele de stare ale sistemului se definesc prin:

$$x_1 = \theta \quad 7.45$$

$$x_2 = \frac{d\theta}{dt} \quad 7.46$$

Modelul dinamic (7.43) - (7.44) poate fi rescris sub formă discretă:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + Bu(k) \quad 7.47$$

unde matricile A, B au forma

$$A = \begin{bmatrix} 1 & a_{12} \\ 0 & a_{22} \end{bmatrix}, B = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad 7.48$$

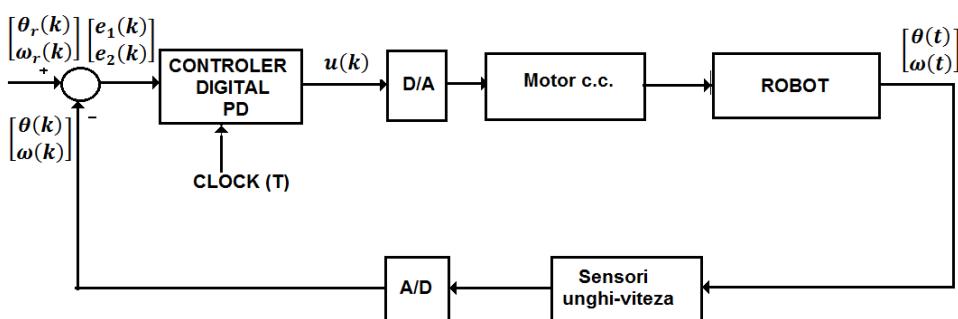


Figura 7.21 Schema bloc de conducere a unui robot prin variabile de stare

7. Implementarea în VLSI a sistemelor de conducere

Considerând valoarea de referință a poziției θ_r iar pentru viteza unghiulară o valoare constantă, eroarea de control a sistemului va fi:

$$e_1(k) = x_1(k) - \theta_r(k) \quad 7.49$$

$$e_2(k) = x_2(k) \quad 7.50$$

Se propune o lege de conducere de forma:

$$u(k) = -k_1 e_1(k) - k_2 e_2(k) \quad 7.51$$

În termenii erorii, ecuațiile (7.47), (7.51) devin:

$$\begin{bmatrix} e_1(k+1) \\ e_2(k+1) \end{bmatrix} = \begin{bmatrix} (1 - k_1 b_1) & (a_{12} - k_2 b_1) \\ -b_2 k_2 & (a_{12} - k_2 b_2) \end{bmatrix} \begin{bmatrix} e_1(k) \\ e_2(k) \end{bmatrix} \quad 7.52$$

Parametrii k_1, k_2 sunt selectați pentru a asigura performanțele dorite ale sistemului de conducere, factorul de amortizare ζ și pulsătia naturală ω_n [7].

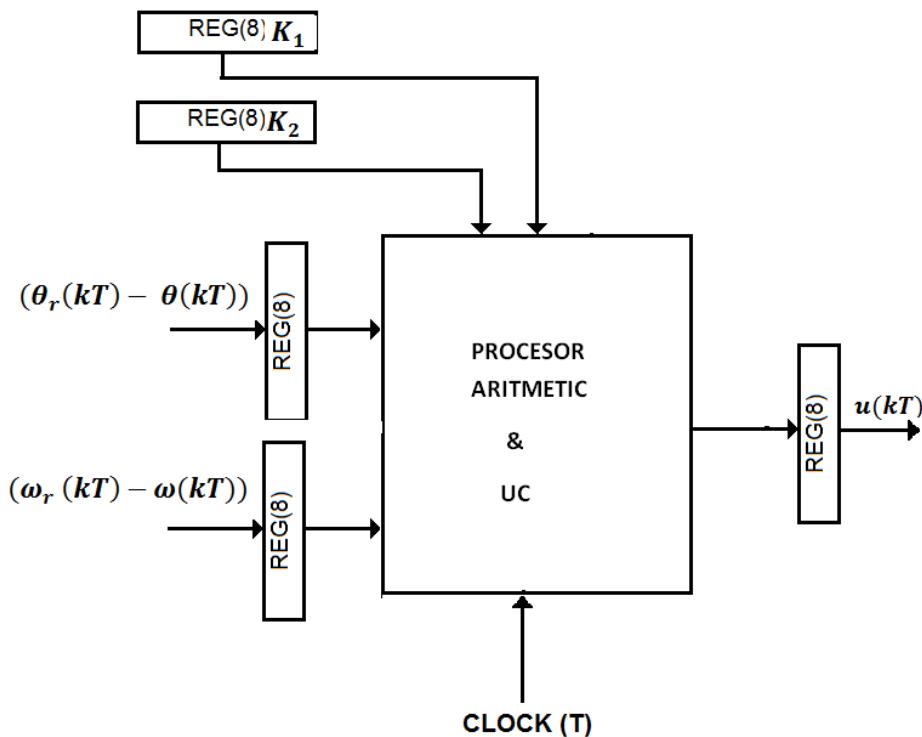


Figura 7.22 Implementarea VLSI a conducerii unui robot prin metoda variabilelor de stare

Arhitectura controlerului ce implementează algoritmul (7.52) este prezentată în Figura 7.22. În cazurile în care sistemul de conducere dispune numai de un senzor de variabilă unghiulară, estimarea stării vitezei unghiulare $\omega(k)$ se obține printr-un observer de stare.

Ecuația observerului pentru modelul descris de ecuațiile (7.5.5) se obține prin procedurile clasice [7]:

$$\hat{\omega}(k+1) = a_{22} \hat{\omega}(k) + b_2 u(k) + k_o (\theta(k) - \hat{\theta}(k)) \quad 7.53$$

unde k_o este factorul de amplificare al observerului ales astfel încât să asigure convergența spre zero a erorii $(\theta(k+1) - \hat{\theta}(k+1))$ iar $\hat{\omega}, \hat{\theta}$ sunt stările estimate. Schema bloc este prezentată în Figura 7.23, iar implementarea VLSI poate fi urmarită în Figura 7.24.

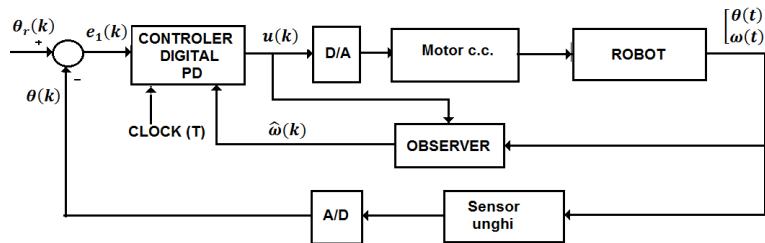


Figura 7.23 Schema bloc a conducerii unui robot cu observer

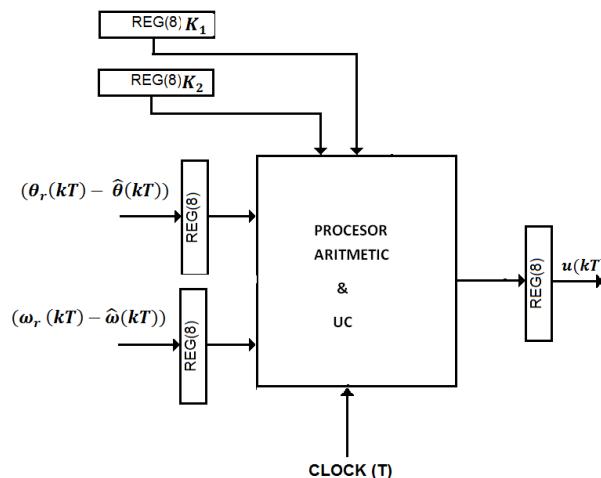


Figura 7.24 Implementarea VLSI a conducerii unui robot cu observer de stare

7. Implementarea în VLSI a sistemelor de conducere

7.6. Conducerea unui braț de robot prin metoda funcțiilor de transfer

Schema bloc generală asociată sistemului de conducere este similară cu cea prezentată în Figura 7.21. Ecuațiile ce descriu comportarea brațului în formă diferențială sunt descrise prin relațiile (7.43) și (7.44). Funcția de transfer asociată are forma:

$$Y_R(s) = \frac{\theta(s)}{u(s)} = \frac{\alpha}{s(\beta s + 1)} \quad 7.54$$

unde α, β sunt determinați de parametrii electrici și mecanici ai sistemului de acționare, iar s este variabila Laplace. Funcția de transfer a controlerului este selectată în forma:

$$Y_c(s) = \frac{u_c(s)}{e(s)} = K_c \frac{(\beta s + 1)}{s + \gamma} \quad 7.55$$

unde $(\beta s + 1)$ a fost selectată pentru eliminarea polului din expresia funcției de transfer iar K_c și γ sunt parametrii controlerului determinați pentru satisfacerea anumitor performanțe de conducere. Pentru obținerea formei digitale a controlerului se aplică transformata $-z$ [10], pentru o perioadă de eșantionare T :

$$s = \frac{2}{T} \frac{(z - 1)}{(z + 1)} \quad 7.56$$

ceea ce permite obținerea controlerului discret:

$$\frac{u_c(z)}{e(z)} = \frac{a - bz^{-1}}{1 - cz^{-1}} \quad 7.57$$

sau

$$u_c(k) = c u_c(k - 1) + a e(k) - b e(k - 1) \quad 7.58$$

Implementarea VLSI a controlerului este dată în Figura 7.25.

CMOS VLSI

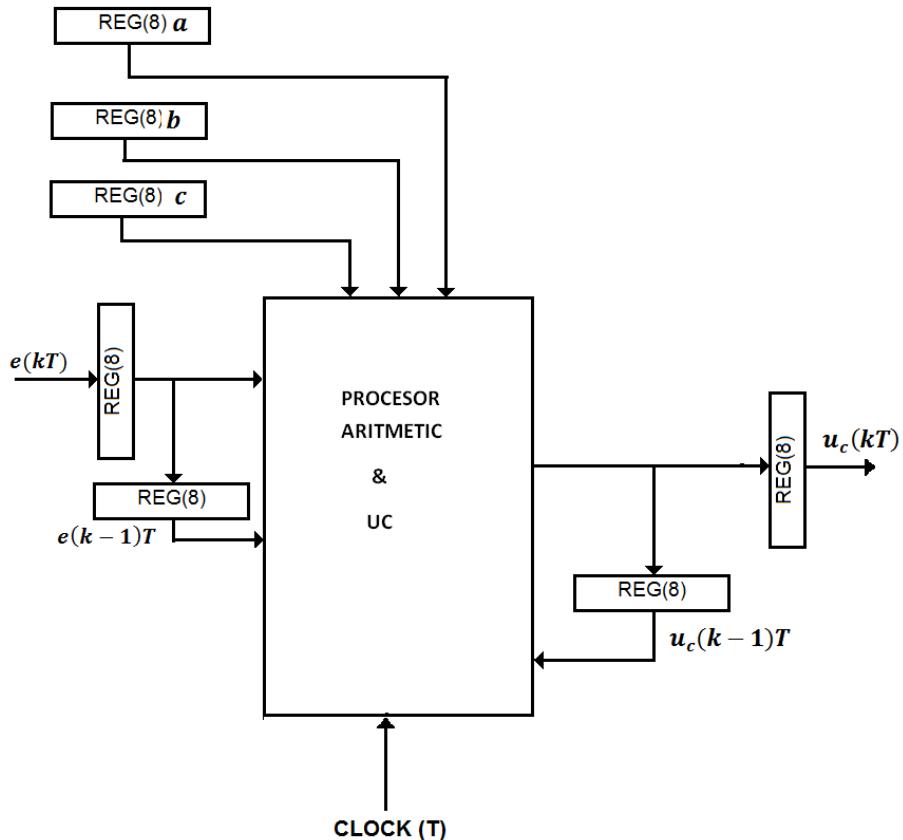


Figura 7.25 Implementarea VLSI a controlerului prin metoda funcțiilor de transfer

8. IMPLEMENTAREA ÎN VLSI A SISTEMELOR CU PREDICȚIE

8.1. Modele dinamice predictoare implementate VLSI

Implementarea unui model dinamic prin tehnologii VLSI se bazează pe transpunerea modelului dinamic în forma discretă și utilizarea componentelor aritmetice, operatori aritmetici și registre, pentru transpunerea ecuațiilor. În capitolul precedent, derivatele erau aproximate prin diferențe sau erau utilizate tehnici bazate pe transformata -z. În cadrul acestui capitol, modelul dinamic al unui proces se construiește prin modelul discret de conoluție sau modelul cu răspuns la impuls [6,8], (Fig 8.1).



Figura 8.1 Variabilele discrete ale unui proces

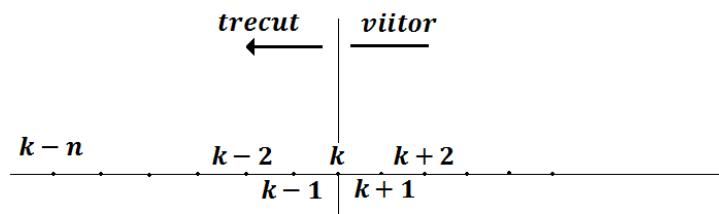


Figura 8.2 Orizontul de timp al unui model discret

$$y(k+1) = y(0) + \sum_{i=1}^n h_i u(k-i+1) \quad 8.1$$

unde $y(k)$, $u(k)$ reprezintă valorile variabilelor de intrare și ieșire, respectiv, la momentul de eşantionare k , iar h_i sunt răspunsurile la impuls ale sistemului la momentul i :

$$h_i = S_i - S_{i-1} \quad 8.2$$

unde S_i este răspunsul la semnalul treaptă corespunzător. Din relațiile (8.1) și (8.2) se obține modelul la răspuns treaptă:

$$y(k+1) = y(0) + \sum_{i=1}^{n-1} S_i \Delta u(k-i+1) + S_n u(k-n+1) \quad 8.3$$

unde

$$\Delta u(k) = u(k) - u(k-1) \quad 8.4$$

Pentru implementarea în VLSI, din relația (8.3) se construiește flowchart-ul din Figura 8.3. Arhitectura modelului VLSI este prezentată în Figura 8.4.

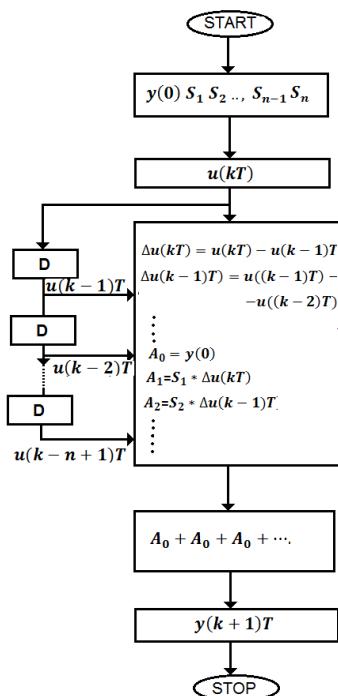


Figura 8.3 Flowchart-ul modelului cu predicție

8. Implementarea în VLSI a sistemelor cu predicție

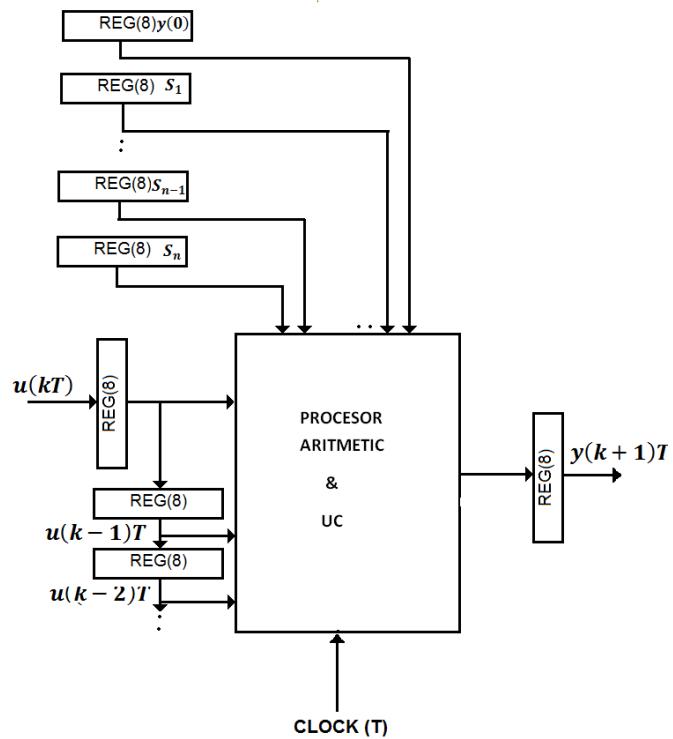


Figura 8.4 Implementarea VLSI a unui model cu predicție prin metoda răspunsului la semnalul treaptă

8.2. Conducerea cu model predictiv

Controlul cu model predictiv (CMP) reprezintă o abordare modernă a noilor tehnologii de conducere ce încearcă să satisfacă în tot mai mare măsură, performanțele din ce în ce mai ridicate impuse complexelor sisteme actuale de producție sau de gestionare economică, în concordanță cu suportul tehnologic din ce în ce mai avansat de care se dispune. Aceste modele se bazează pe evaluarea comportării viitoare a unui sistem, precizarea evoluției viitoare, pe baza analizei evoluției anterioare, a istoriei acestuia, pe segmente de timp mai lungi sau mai scurte. Avantajul implementării CMP îl reprezintă următoarele:

1. descriere exactă statică și dinamică a interacțiunilor între variabilele de intrare, ieșire și perturbații,
2. o abordare riguroasă matematică a restricțiilor impuse acestor variabile
3. impunerea unor proceduri de optimizare a efortului de calcul în stabilirea strategiilor de conducere.

Într-un CMP, obținerea legii de conducere se bazează pe modelul discret de convoluție discutat în paragraful anterior și pe identificarea orizontului de predicție care să permită evaluarea legii de conducere astfel încât variabila de ieșire (variabilele) să atingă o comportare dorită [7,8].

Pentru ilustrarea procedurii se va considera un model de convoluție discret, cu o singură intrare $u(k)$ și o singură ieșire $y(k)$, descris printr-o ecuație de forma (8.1.3). Procedura CMP determină o secvență de variabile de intrare u astfel încât comportarea viitoare a ieșirii, valoarea estimată, \hat{y} , să fie cât mai aproape (în raport cu un criteriu de optim) de valoarea dorită. Orizontul de timp impus în această procedură este prezentat în Figura 8.5.

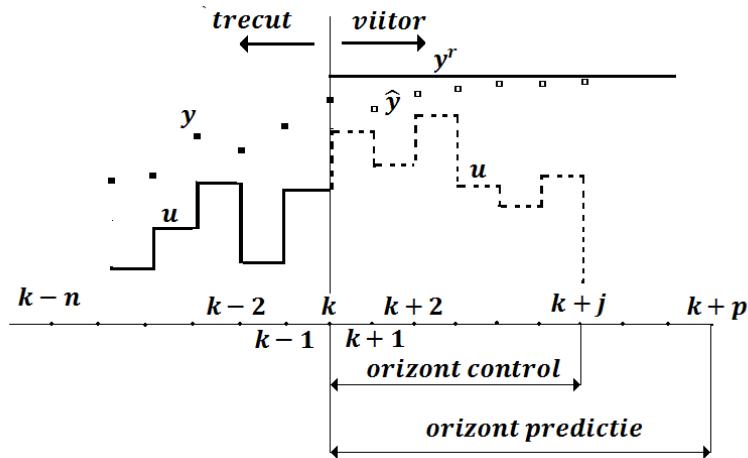


Figura 8.5 Orizontul de timp în predicție

Se va nota prin k momentul curent iar valorile memorate, trecute, ale controlului sunt $u(k - (i - 1)), i = 1, 2, \dots, (n - 1)$. Se va nota prin $\hat{y}(k + 1)$ predicția valorii de ieșire la momentul viitor ($k + 1$) ce se obține din (8.1.3), (unde s-a considerat $y(0) = 0$).

8. Implementarea în VLSI a sistemelor cu predicție

$$\hat{y}(k+1) = \sum_{i=1}^{n-1} S_i \Delta u(k-i+1) + S_n u(k-n+1) \quad 8.5$$

Ecuația (8.5) poate fi rescrisă sub forma:

$$\hat{y}(k+1) = S_1 \Delta u(k) + \sum_{i=2}^{n-1} S_i \Delta u(k-i+1) + S_n u(k-n+1) \quad 8.6$$

unde primul termen $S_1 \Delta u(k)$ reprezintă efectul acțiunii actuale, iar ultimii doi termeni corespund efectului acțiunilor trecute $u(i), i < k$. Procedând în aceeași manieră, din relația de mai sus se obține:

$$\begin{aligned} \hat{y}(k+2) = & S_1 \Delta u(k+1) + S_2 \Delta u(k) + \sum_{i=3}^{n-1} S_i \Delta u(k-i+2) \\ & + S_n u(k-n+2) \end{aligned} \quad 8.7$$

în care primii doi termeni sunt associați efectului acțiunii curente și ai momentului imediat următor. Extinzând acest rezultat pentru un moment de predicție viitor j , obținem:

$$\begin{aligned} \hat{y}(k+j) = & \sum_{i=1}^j S_i \Delta u(k+j-i) + \sum_{i=j+1}^{n-1} S_i \Delta u(k+j-i) \\ & + S_n u(k+j-n) \end{aligned} \quad 8.8$$

Ultimii doi termeni reprezintă ieșirea sistemului când nicio acțiune curentă nici viitoare până la momentul j nu au loc:

$$\sum_{i=1}^j S_i \Delta u(k+j-i) = 0 \quad 8.9$$

Dacă se notează prin:

$$\hat{y}^o(k+j) = \sum_{i=j+1}^{n-1} S_i \Delta u(k+j-i) + S_n u(k+j-n) \quad 8.10$$

relația (8.8) poate fi rescrisă sub forma:

$$\hat{y}(k+j) = \sum_{i=1}^j S_i \Delta u(k+j-i) + \hat{y}^o(k+j) \quad 8.11$$

Această relație poate fi simplificată în ipoteza admiterii unei singure predicții, la momentul j ceea ce înseamnă că:

$$\Delta u(k+j-i) = 0, i = 1, 2, \dots (j-1) \quad 8.12$$

În acest caz, expresia (8.11) devine:

$$\hat{y}(k+j) = S_j \Delta u(k) + \hat{y}^o(k+j) \quad 8.13$$

Această ultimă relație permite determinarea legii de conducere pentru atingerea unei stări de referință impusă y_r .

$$\Delta u(k) = \frac{1}{S_j} (y_r - \hat{y}^o(k+j)) \quad 8.14$$

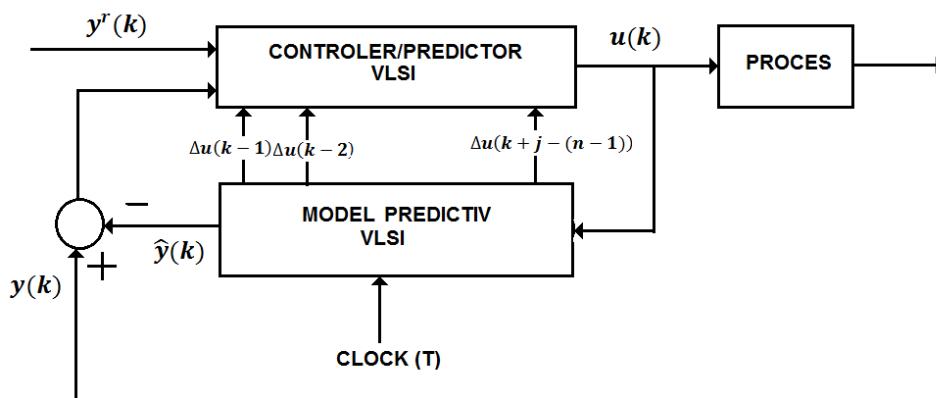


Figura 8.6 Schema bloc a sistemului de conducere cu model predictiv

Acest rezultat permite estimarea legii de conducere, practic în circuit deschis, fără a ține cont de comportarea reală a sistemului. O soluție calitativ superioară se obține dacă evaluarea lui $\hat{y}(k+j)$ în (8.2.8) este realizată cu un termen de corecție ($y(k) - \hat{y}(k)$) [8],

$$\hat{y}'(k+j) = \hat{y}(k+j) + (y(k) - \hat{y}(k)) \quad 8.15$$

Schema bloc a conducerii unui proces este prezentată în Figura 8.6. Implementarea VLSI este realizată prin două componente, modelul procesului implementat prin metoda convoluției discrete și controlerul ce determină valorile $u(k)$ ce conduc sistemul către o valoare de referință impusă, $y_r(k)$. Implementarea VLSI a modelului este descrisă în paragraful anterior și este prezentată în Figura 8.4. Pentru implementarea controlerului se va utiliza algoritmul dat de (8.14) și (8.15) în care o parte din variabile sunt generate de modelul sistemului. Flowchart-ul asociat este prezentat în Figura 8.7, iar structura controlerului este ilustrată în Figura 8.8.

8. Implementarea în VLSI a sistemelor cu predicție

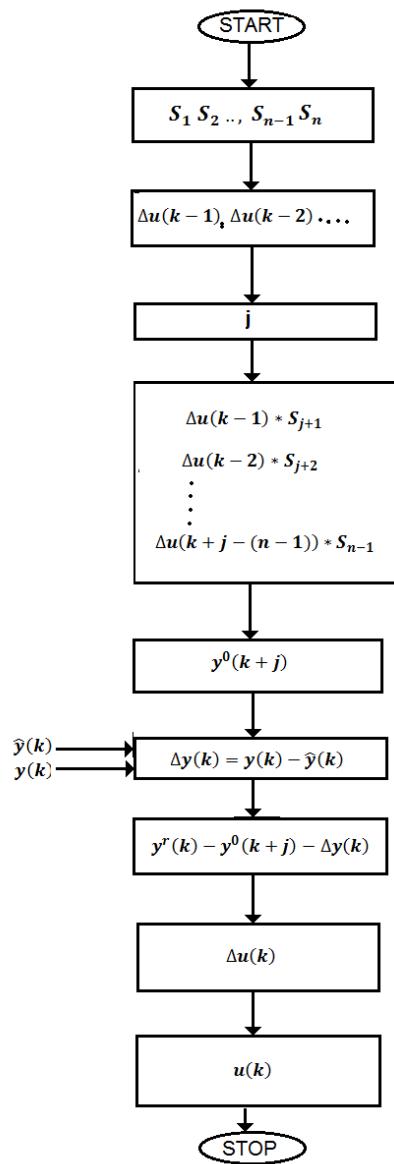


Figura 8.7 Flowchart-ul controlerului cu predicție

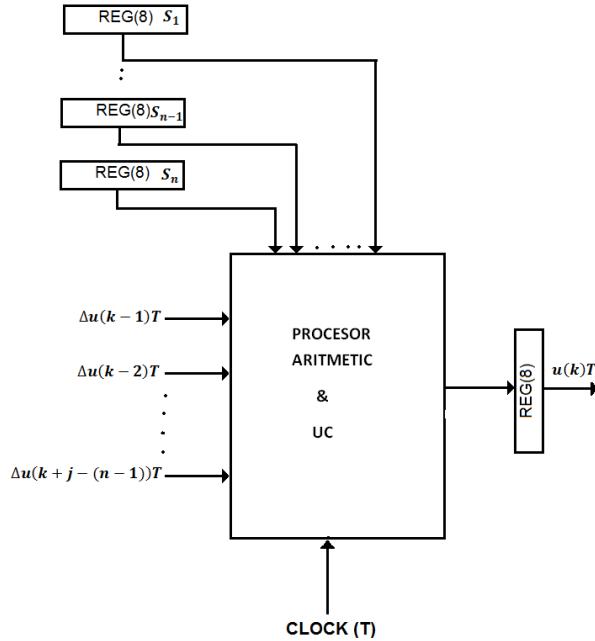


Figura 8.8 Implementarea VLSI a controlerului cu predicție

8.3. Conducerea cu model predictiv a unui robot industrial

Principiile stabilite în paragraful precedent vor fi aplicate pentru stabilirea legii de conducere pentru un robot industrial acționat prin sisteme cu motor de curent continuu. Modelul dinamic este descris prin ecuațiile (7.5.1) și (7.5.2) care pot fi rescrise sub forma:

$$J \ddot{q} + b \dot{q} + c q = \tau + H \quad 8.16$$

unde J, b, c reprezintă parametrii mecanici ai sistemului: momentul de inerție, coeficientul echivalent de frecare vâscoasă și cel de elasticitate, respectiv. Variabila de control este cuplul motor τ , iar H reprezintă variabila perturbatoare corespunzătoare efectului sarcinilor manipulate de robot și al componentelor gravitaționale ale brațului. Se va presupune că aceste perturbații sunt măsurabile. Luând în considerație liniaritatea modelului, variabila de ieșire, controlată, q , poate fi obținută sub forma a două

8. Implementarea în VLSI a sistemelor cu predicție

componente q_τ , q_H , ce determină contribuția variabilei de control și a perturbației, respectiv:

$$q_\tau(k+1) = \sum_{i=1}^{N-1} S_{\tau i} \Delta \tau(k-i+1) + S_{\tau N} \tau(k-N+1) \quad 8.17$$

$$q_H(k+1) = \sum_{i=1}^{N-1} S_{Hi} \Delta H(k-i+1) + S_{HN} H(k-N+1) \quad 8.18$$

unde $S_{\tau i}$, S_{Hi} sunt răspunsurile la semnalul treaptă în raport cu variabila de control și în raport cu perturbația. Pentru valori uzuale ale sistemului mecanic:

$J = 0.000254 \text{ kg m}^2$, $b = 0.002031 \text{ Nms/rad}$, $c = 2.45 \text{ Nm/rad}$, aceste răspunsuri sunt prezentate în Tabelul 8.1 și 8.2.

0.1150	0.1391	0.1640	0.1893	0.2147	0.2398	0.2644
0.2883	0.3111	0.3329	0.3534	0.3724	0.3900	0.4061
0.4205	0.4333	0.4446	0.4542	0.4623	0.4689	0.4741
0.4779	0.4805	0.4818	0.4821	0.4814	0.4798	0.4775
0.4744	0.4708	0.4667	0.4622	0.4575	0.4524	0.4473
0.4421	0.4369	0.4318	0.4267	0.4219	0.4172	0.4128
0.4087	0.4048	0.4013	0.3981	0.3952	0.3926	0.3904
0.3885	0.3869	0.3856	0.3846	0.3839	0.3834	0.3832
0.3832	0.3833	0.3837	0.3842	0.3849	0.3856	0.3865
0.3874	0.3884	0.3895	0.3905	0.3916	0.3927	0.3937
0.3947	0.3957	0.3967	0.3976	0.3984	0.3992	0.3999
0.4005	0.4011	0.4016	0.4021	0.4024	0.4028	0.4030
0.4032	0.4033	0.4034	0.4035	0.4035	0.4034	0.4033
0.4032	0.4031					

Tabel 8.1 Coeficientii $S_{\tau i}$

0.0000	0.0001	0.0001	0.0003	0.0004	0.0005	0.0007
0.0009	0.0010	0.0012	0.0014	0.0016	0.0018	0.0020
0.0022	0.0023	0.0025	0.0027	0.0028	0.0029	0.0030
0.0032	0.0032	0.0033	0.0034	0.0035	0.0035	0.0036
0.0036	0.0036	0.0036	0.0036	0.0036	0.0036	0.0036
0.0036	0.0035	0.0035	0.0035	0.0034	0.0034	0.0034
0.0031	0.0030	0.0030	0.0030	0.0030	0.0029	0.0029
0.0029	0.0029	0.0029	0.0029	0.0029	0.0029	0.0029
0.0029	0.0029	0.0029	0.0029	0.0029	0.0029	0.0029
0.0029	0.0029	0.0029	0.0029	0.0029	0.0029	0.0030

CMOS VLSI

0.0030	0.0030	0.0030	0.0030	0.0030	0.0030	0.0030
0.0030	0.0030	0.0030	0.0030	0.0030	0.0030	0.0030
0.0030	0.0030	0.0030	0.0030	0.0030	0.0030	0.0030
0.0030	0.0030					

Tabel 8.2 Coeficientii S_{Hi}

Modelul predictor al robotului va avea forma:

$$\begin{aligned}\hat{q}(k+1) = & S_{\tau 1} \Delta \tau(k) \\ & + \sum_{i=2}^{N-1} S_{\tau i} \Delta \tau(k-i+1) + S_{\tau N} \tau(k-N+1) \\ & + \sum_{i=1}^{N-1} S_{Hi} \Delta H(k-i+1) + S_{HN} H(k-N+1)\end{aligned}\quad (8.19)$$

Implementarea VLSI a acestui model este realizată după aceeași procedură ca cea prezentată în secțiunea precedentă și poate fi urmarită în Figura 8.9.

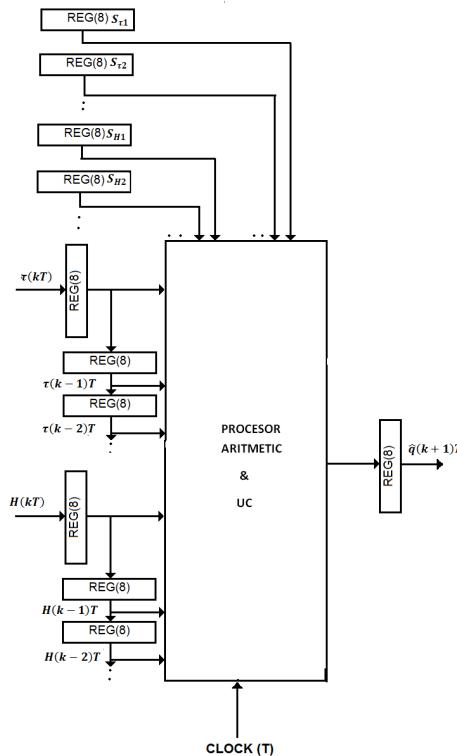


Figura 8.9 Modelul predictor al robotului

8. Implementarea în VLSI a sistemelor cu predicție

Extinzând acest rezultat pentru un moment de predicție viitor j , obținem:

$$\begin{aligned}\hat{q}(k+j) = & \sum_{i=1}^j S_i \Delta \tau(k+j-i) + \sum_{i=j+1}^{N-1} S_i \Delta \tau(k+j-i) \\ & + S_n u(k+j-n) \\ & + \sum_{i=1}^j S_{Hi} \Delta H(k+j-i) + \sum_{i=j+1}^{N-1} S_{Hi} \Delta H(k+j-i) \\ & + S_{HN} H(k+j-N)\end{aligned}\quad (8.20)$$

Dacă se notează prin:

$$\hat{q}^o(k+j) = \sum_{i=j+1}^{n-1} S_i \Delta \tau(k+j-i) + S_n \tau(k+j-n) \quad (8.21)$$

$$\hat{q}^P(k+j) = \sum_{i=j+1}^{n-1} S_{Hi} \Delta H(k+j-i) + S_n H(k+j-n) \quad (8.22)$$

relația (8.20) poate fi rescrisă sub forma:

$$\begin{aligned}\hat{q}(k+j) = & \sum_{i=1}^j S_i \Delta \tau(k+j-i) + \hat{y}^o(k+j) \\ & + \sum_{i=1}^j S_{Hi} \Delta H(k+j-i) + \hat{y}^P(k+j)\end{aligned}\quad (8.23)$$

Această relație poate fi simplificată în ipoteza admiterii unei singure predicții, la momentul j ceea ce înseamnă că:

$$\Delta \tau(k+j-i) = 0, i = 1, 2, \dots, (j-1) \quad (8.24)$$

Considerând, de asemenea că perturbația verifică condiția:

$$H(k+j-i) = H(k+j), i = 1, 2, \dots, (j-1) \quad (8.25)$$

expresia (8.23) devine:

$$\hat{q}(k+j) = S_j \Delta \tau(k) + \hat{q}^o(k+j) + \hat{q}^P(k+j) \quad (8.26)$$

Această ultimă relație permite determinarea legii de conducere pentru atingerea unei stări de referință impusă y_r .

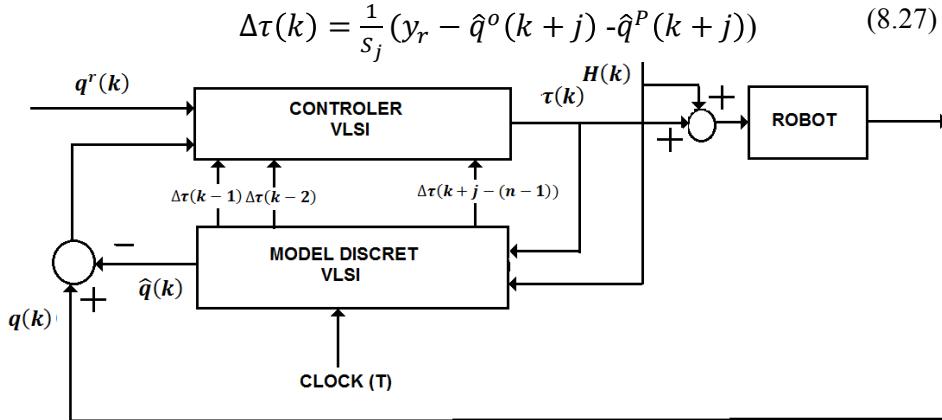


Figura 8.10 Schema bloc a conducerii unui robot cu model predictor

Relația (8.27) reprezintă algoritmul de conducere. Structura globală a schemei de conducere poate fi urmarită în Figura 8.10. Implementarea controlerului VLSI aferent se realizează după aceeași procedură ca cea prezentată în secțiunea precedentă (Figura 8.8).

8.4. Estimarea parametrilor unui robot prin tehnici cu model predictor

În secțiunea anterioară, legea de conducere pleacă de la premiza existenței unor perturbații măsurabile. În realitate, perturbațiile determinate de sarcinile robotului sau de modificarea componentei gravitaționale sunt extrem de dificil de măsurat. În această secțiune va fi dezvoltat un algoritm cu predicție care să permită evaluarea efectului perturbațiilor. Algoritmul cuprinde două etape, în prima este determinată viteza de mișcare a brațului robotului, iar în a doua este estimată variabila perturbatoare.

Variabila de viteză se determină din valoarea măsurată a poziției (unghi). Pentru a evita erorile introduse prin tehnici uzuale de diferențiere, se propune un observer definit prin:

$$\hat{q} = v + k_0 e_q \quad (8.28)$$

$$\dot{v} = k_1 \operatorname{sgn}(e_q) + k_2 e_q \quad (8.29)$$

8. Implementarea în VLSI a sistemelor cu predicție

unde e_q este eroarea de poziție:

$$e_q = q - \hat{q} \quad (8.30)$$

\hat{q} este valoarea estimată a poziției, iar k_0, k_1, k_2 sunt constante pozitive ce definesc parametrii observerului [7]. În termeni CMP, predicția vitezei poate fi obținută prin:

$$\begin{aligned} \dot{\hat{q}}(k+1) &= \sum_{i=1}^{N-1} S_{\dot{q}i}(q(k-i+1) - \hat{q}(k-i+1)) + S_{\dot{q}N}(q(k-N \\ &\quad + 1) - \hat{q}(k-N+1)) \end{aligned} \quad (8.31)$$

unde $S_{\dot{q}i}$ sunt coeficienții de răspuns la intrarea treaptă a modelului (8.28) și (8.29) în raport cu variațiile $(q(k-i+1) - \hat{q}(k-i+1))$. Valorile acestor coeficienți, pentru parametrii electrici și mecanici specificați în paragraful anterior și $N = 100$, sunt prezentate în Tabelul 8.3, iar implementarea VLSI a estimatorului de viteză prin CMP este prezentată în Figura 8.11. În Figura 8.12 sunt prezentate, prin simulare, valorile estimate ale vitezei în raport cu viteză măsurată.

750.1000	750.2000	750.3000	750.4000	750.5000	750.6000	750.7000
750.8000	750.9000	751.0000	751.1000	751.2000	751.3000	751.4000
751.5000	751.6000	751.7000	751.8000	751.9000	752.0000	752.1000
752.2000	752.3000	752.4000	752.5000	752.6000	752.7000	752.8000
752.9000	753.0000	753.1000	753.2000	753.3000	753.4000	753.5000
753.6000	753.7000	753.8000	753.9000	754.0000	754.1000	754.2000
754.3000	754.4000	754.5000	754.6000	754.7000	754.8000	754.9000
755.0000	755.1000	755.2000	755.3000	755.4000	755.5000	755.6000
755.7000	755.8000	755.9000	756.0000	756.1000	756.2000	756.3000
756.4000	756.5000	756.6000	756.7000	756.8000	756.9000	757.0000
757.1000	757.2000	757.3000	757.4000	757.5000	757.6000	757.7000
757.8000	757.9000	758.0000	758.1000	758.2000	758.3000	758.4000
758.5000	758.6000	758.7000	758.8000	758.9000	759.0000	759.1000
759.2000	759.3000	759.4000	759.5000	759.6000	759.7000	759.8000
759.9000	760.0000					

Tabel 8.3 Coeficienții $S_{\dot{q}i}$

CMOS VLSI

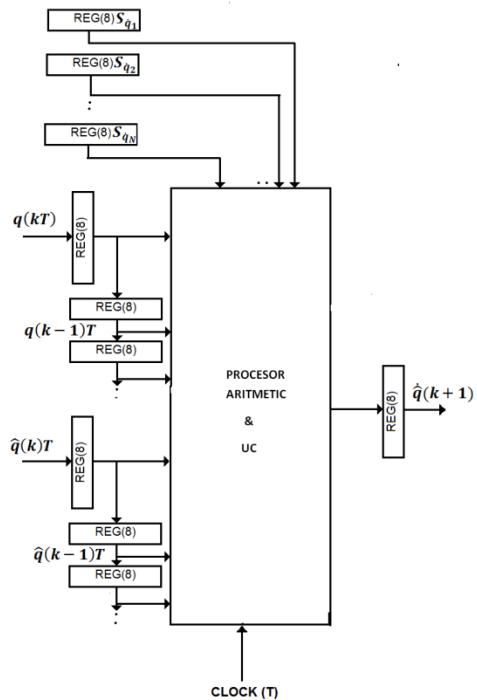


Figura 8.11 Implementarea VLSI a estimatorului de viteza cu predictie

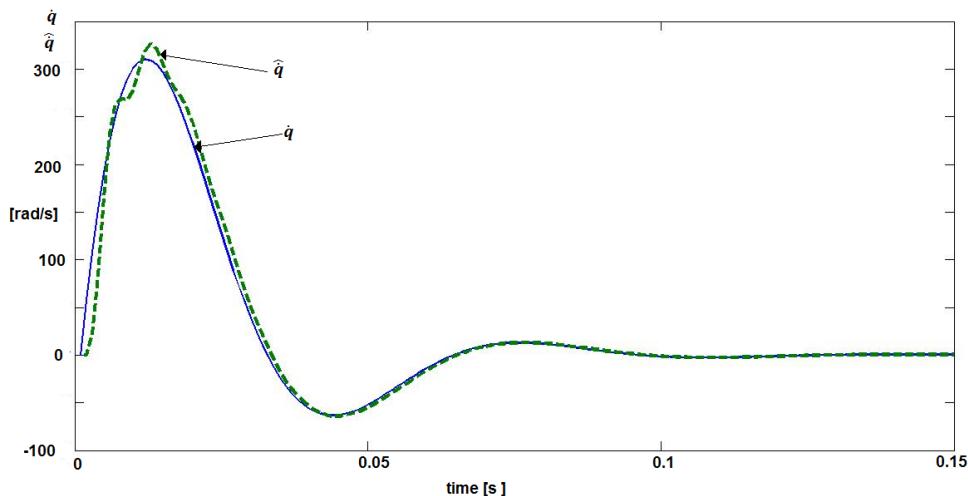


Figura 8.12 Valorile estimate ale vitezei

8. Implementarea în VLSI a sistemelor cu predicție

Pentru estimarea cuplului perturbator, sunt utilizate valorile unghiulare de poziție și de viteză. În acest scop se construiește un observer de forță sub forma:

$$\hat{H} = w + \alpha \hat{q} \quad (8.32)$$

$$\dot{w} = -w + \frac{\alpha}{J}(-\alpha + b)\hat{q} + c \frac{\alpha}{J}q + \frac{\alpha}{J}g(q) - \frac{\alpha}{J}\tau \quad (8.33)$$

unde α este parametrul observerului, $\alpha > 0$. Convergența observerului se demonstrează prin evaluarea erorii,

$$e = H - \hat{H} \quad (8.34)$$

$$\dot{e} = \dot{H} - \dot{\hat{H}} \quad (8.35)$$

în care se poate accepta o variație relativ lentă a perturbației în raport cu timpii de evoluție ai observerului, $\dot{H} = 0$. Din (8.32) și (8.33) rezultă

$$\dot{e} = -\frac{\alpha}{J}e \quad (8.36)$$

deci observerul este global asymptotic stabil.

În termeni CMP, estimarea cuplului perturbator se obține sub forma:

$$\begin{aligned} \hat{w}(k+1) = & \frac{\alpha}{J} \left(\sum_{i=1}^{N-1} S_{wi} (\dot{q}(k-i+1) - \hat{q}(k-i+1)) \right. \\ & + S_{wi} (\dot{q}(k-i+1) - \hat{q}(k-i+1)) (-\alpha + b) \\ & + \frac{\alpha}{J} c \sum_{i=1}^{N-1} S_{wi} (q(k-i+1) - \hat{q}(k-i+1)) \end{aligned} \quad (8.37)$$

$$\begin{aligned} & + S_{wi} (q(k-i+1) - \hat{q}(k-i+1)) \\ & - \frac{\alpha}{J} \sum_{i=1}^{N-1} S_{wi} (\tau(k-i+1) - \hat{\tau}(k-i+1)) \\ & + S_{wi} (\tau(k-i+1) - \hat{\tau}(k-i+1)) \end{aligned} \quad (8.38)$$

$$\hat{H}(k+1) = \hat{w}(k+1) + \alpha \hat{q}(k+1)$$

unde S_{wi} sunt coeficienții de răspuns la semnalul treaptă ai modelului (tabelul 8.4)

0.0010	0.0019	0.0028	0.0036	0.0044	0.0052	0.0059
0.0066	0.0072	0.0079	0.0085	0.0090	0.0096	0.0101

CMOS VLSI

0.0106	0.0110	0.0115	0.0119	0.0123	0.0126	0.0130
0.0133	0.0137	0.0140	0.0143	0.0145	0.0148	0.0151
0.0153	0.0155	0.0158	0.0160	0.0162	0.0163	0.0165
0.0167	0.0169	0.0170	0.0172	0.0173	0.0174	0.0176
0.0177	0.0178	0.0179	0.0180	0.0181	0.0182	0.0183
0.0184	0.0184	0.0185	0.0186	0.0187	0.0187	0.0188
0.0188	0.0189	0.0190	0.0190	0.0191	0.0191	0.0191
0.0192	0.0192	0.0193	0.0193	0.0193	0.0194	0.0194
0.0194	0.0195	0.0195	0.0195	0.0195	0.0196	0.0196
0.0196	0.0196	0.0196	0.0197	0.0197	0.0197	0.0197
0.0197	0.0197	0.0197	0.0198	0.0198	0.0198	0.0198
0.0198	0.0198	0.0198	0.0198	0.0198	0.0198	0.0199
0.0199	0.0199					

Tabel 8.4 Coeficienții S_{wi}

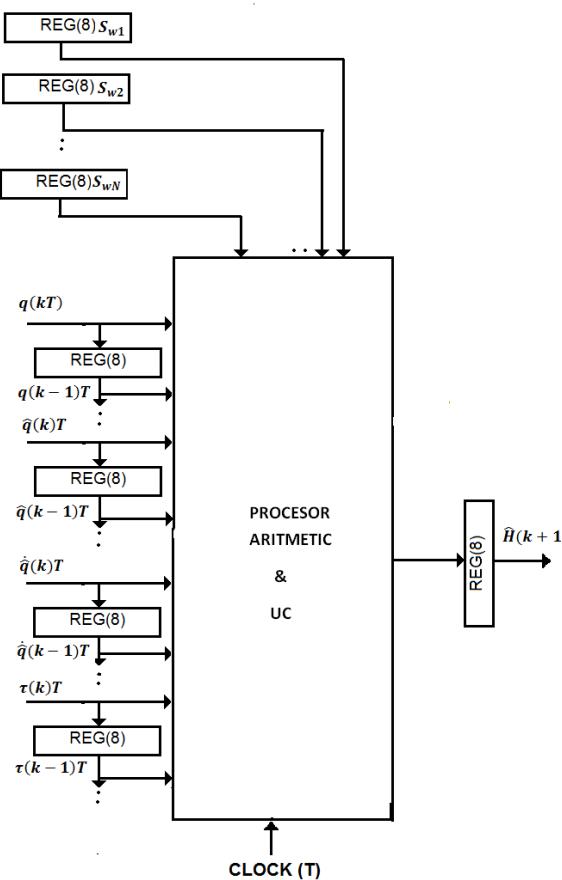


Figura 8.13 Implementarea VLSI a observerului de cuplu perturbator

8. Implementarea în VLSI a sistemelor cu predicție

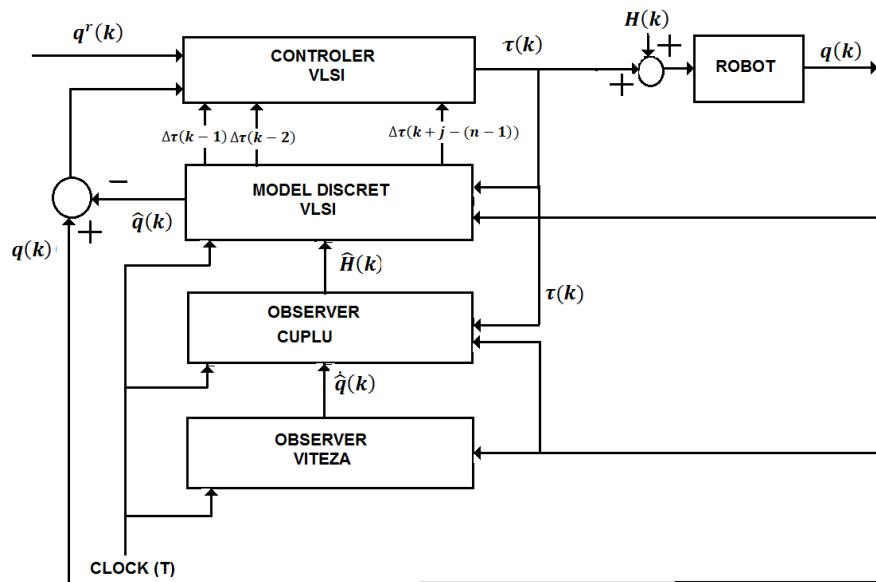


Figura 8.14 Schema de control cu observeri de viteză și cuplu perturbator

Relațiile (8.37) și (8.38) permit implementarea prin predicție a observului de cuplu perturbator (Figura 8.13). Schema generală de control a robotului se poate urmări în Figura 8.14.

9. IMPLEMENTAREA ÎN VLSI A REȚELELOR NEURONALE

9.1. Modelul unei rețele neuronale

Neuronul artificial modelează trei procese ce au loc într-un neuron biologic [10,12]:

1. evaluează mărimele (variabilele) de intrare;
2. calculează o valoare ponderată a combinațiilor variabilelor de ieșire;
3. generează ieșirea în funcție de un anumit prag realizat în faza 2.

Unui neuron artificial îi vor fi asociate: un număr de variabile de intrare x_1, x_2, \dots, x_n , o singură variabilă de ieșire, y , și un set de coeficienți de ponderare w_1, w_2, \dots, w_n , (Figura 9.1)

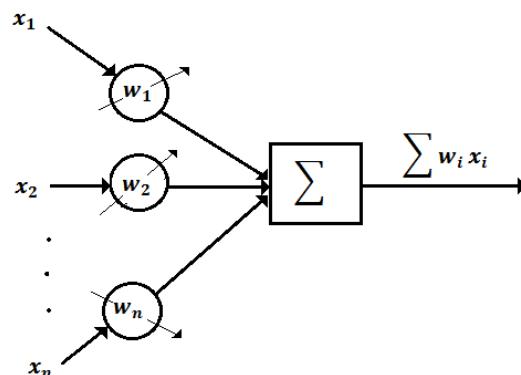


Figura 9.1 Modelul unui neuron

Încercând să redăm funcția unui neuron biologic, ieșirea este generată când o anumită condiție „de activare a stării” este atinsă. Această condiție este introdusă printr-un circuit de prag și o variabilă de „polarizare” b .

9. Implementarea în VLSI a rețelelor neuronale

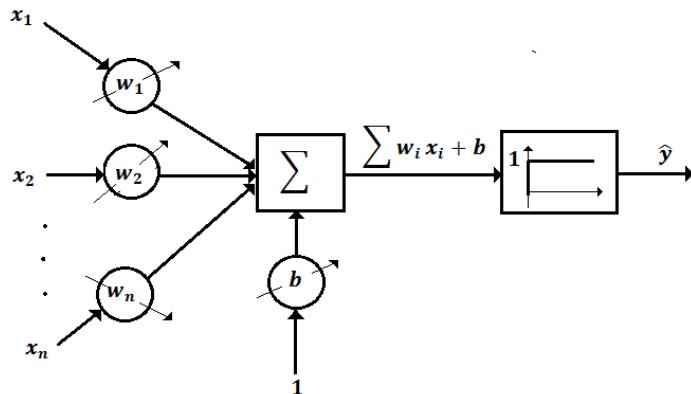


Figura 9.2 Modelul unui neuron cu polarizare și circuit de prag

Într-o formă matematică compactă, modelul unui neuron poate fi scris sub forma:

$$\hat{y} = \sigma(\sum w_i x_i + w_0) \quad (9.1)$$

unde \$w_0\$ este valoarea sintetică a polarizării, iar \$\sigma\$ definește operatorul de activare, operatorul de prag. O rețea neuronală (RN) este realizată dintr-un număr de neuroni interconectați, fiecare neuron procesând la un nivel specificat informația de intrare. Neuronii asociați unui anumit nivel de procesare formează un „layer”. Dacă informația de intrare este complet și independent procesată la nivelul unui layer și rezultatul procesării este transmis layer-ului următor, rețeaua se numește „cu propagare înainte (feedforward network)”. Numărul de intrări și de ieșiri dintr-un layer depinde de complexitatea procesului modelat prin rețea și impune o structură de rețea cu un număr specificat de neuroni. De exemplu, o rețea cu un singur layer (un singur nivel de procesare) cu \$n\$ intrări și \$q\$ neuroni este definită prin relațiile:

$$\hat{y}_j = \sigma(\sum w_{ji} x_i + w_{0j}) \quad j = 1, 2, \dots, q \quad (9.2)$$

Considerând că într-un layer cu un singur nivel, valorile generate \$\hat{y}_j\$ reprezintă chiar ieșirile din sistem \$z_j\$, relațiile (9.1.2) pot fi rescrise în forma matriceală [11,13]:

$$z = \Gamma(Wx + w_0) \quad (9.3)$$

unde

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \quad w_0 = \begin{bmatrix} w_{01} \\ w_{02} \\ \vdots \\ w_{0n} \end{bmatrix} \quad (9.4)$$

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{q1} & w_{q2} & \dots & w_{qn} \end{bmatrix} \quad (9.5)$$

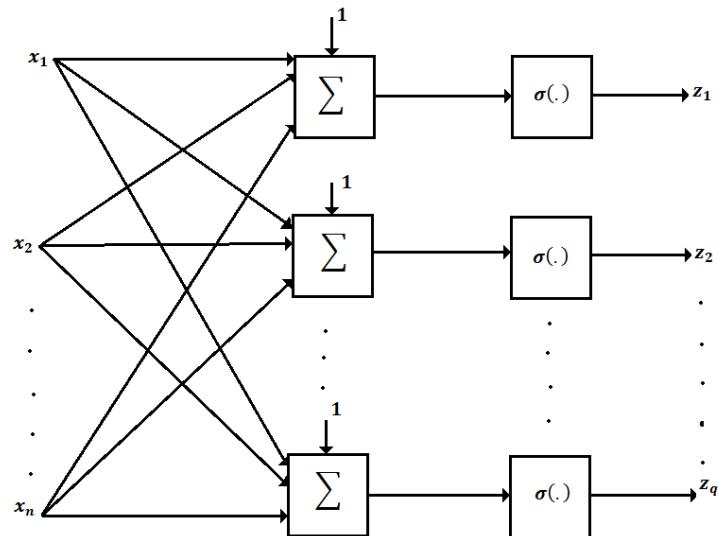


Figura 9.3 Rețea neuronală cu un layer

$$\Gamma(\cdot) = \begin{bmatrix} \sigma(\cdot) & 0 & 0 & \dots & 0 \\ 0 & \sigma(\cdot) & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \dots & \sigma(\cdot) \end{bmatrix} \quad (9.6)$$

Arhitectura unei astfel de rețele este prezentată în Figura 9.3. Funcția de activare $\sigma(\cdot)$ poate avea diferite forme. În literatura de specialitate sunt utilizate câteva funcții [11,12,14]:

a) *funcție liniară*

$$f(\sigma) = \sigma \quad (9.7)$$

9. Implementarea în VLSI a rețelelor neuronale

b) funcție prag

$$\begin{aligned} f(\sigma) &= 1 \text{ dacă } \sigma > 0 \\ f(\sigma) &= 0 \text{ dacă } \sigma \leq 0 \end{aligned} \quad (9.8)$$

c) funcție sigmoid

$$f(\sigma) = \frac{1}{1 + e^{-\sigma}} \quad (9.9)$$

d) tangenta hiperbolică

$$f(\sigma) = \frac{1 - e^{-\sigma}}{1 + e^{-\sigma}} \quad (9.10)$$

e) perceptron

$$\begin{aligned} f(\sigma) &= \sigma \text{ dacă } \sigma > 0 \\ f(\sigma) &= 0 \text{ dacă } \sigma \leq 0 \end{aligned} \quad (9.11)$$

Rețeaua neuronală cu un singur strat (layer) poate fi extinsă la rețele multi-strat (multi-layer). În Figura 9.4 este prezentată structura unei rețele multi-strat feedforward în care funcțiile de însumare ponderată și de activare sunt reprezentate simbolic.

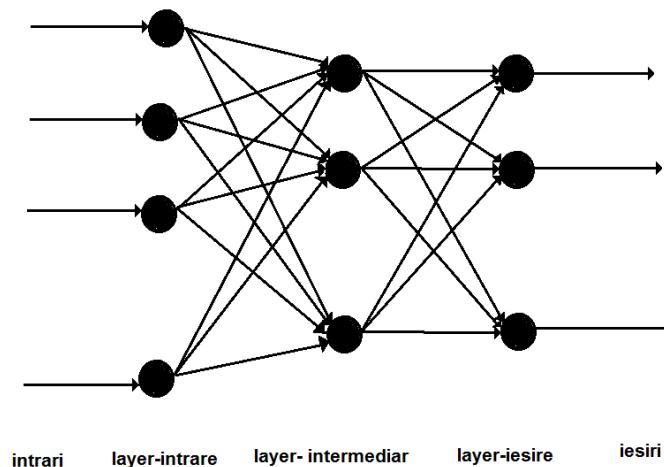


Figura 9.4 Rețea neuronală multi-layer tip feed-forward

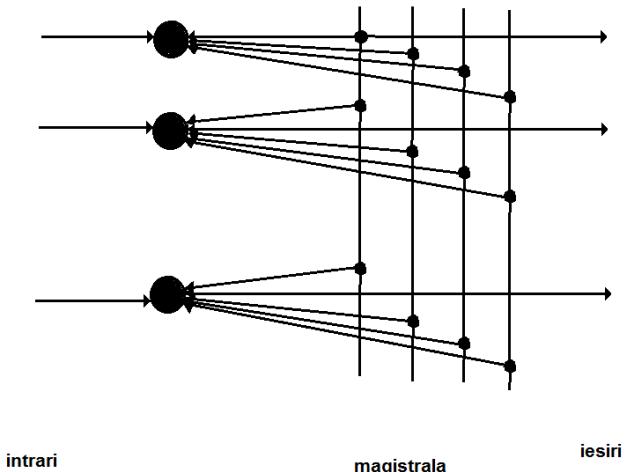


Figura 9.5 Rețea neuronală cu reacție tip Hopfield

În Figura 9.5 este prezentată o rețea (Hopfield) în care ieșirile fiecărui strat sunt captate de o magistrală de date.

Relațiile de tip (9.3) pot fi extinse în cazul, de exemplu, al unei rețele cu două nivale layer:

$$z = \Gamma(V(\Gamma(Wx + w_0) + v_0)) \quad (9.12)$$

unde V este matricea ponderilor la nivelul doi, iar v_0 este vectorul depolarizare corespunzător.

$$V = \begin{bmatrix} v_{11} & v_{12} \dots v_{1n} \\ v_{21} & v_{22} \dots v_{2n} \\ \dots & \dots \dots \\ v_{q1} & v_{q2} \dots v_{qn} \end{bmatrix}, \quad v_0 = \begin{bmatrix} v_{01} \\ v_{02} \\ \vdots \\ v_{0n} \end{bmatrix} \quad (9.13)$$

Un element extrem de important în studiul și aplicațiile RN se referă la posibilitatea aproximării oricărei funcții $f(x)$ printr-o arhitectură de rețea conținând cel puțin două nivele layer și cu un număr suficient de neuroni [14,15]:

$$f(x) = \Gamma(V(\Gamma(Wx + w_0) + v_0) + \varepsilon \quad (9.14)$$

unde ε este eroarea de aproximare. Aproximarea este obținută prin ajustarea corespunzătoare a coeficienților de ponderare w_{ij} . Mecanismul de ajustare este cunoscut în literatura [11] ca procedură de „instruire” sau de „învățare”

9. Implementarea în VLSI a rețelelor neuronale

a rețelei. Pentru ilustrarea procedurii, să considerăm că, pe un anumit nivel, y^d_j este valoarea dorită a variabilei \hat{y}_j și notăm prin e_j eroarea corespunzătoare:

$$e_j(k) = y^d_j - \hat{y}_j(k) \quad (9.15)$$

unde k este secvența de instruire.

Eroarea totală este:

$$E(k) = \frac{1}{2} \sum_{j=1}^p e_j^2(k) = \frac{1}{2} \sum_{j=1}^p (y^d_j(k) - \hat{y}_j(k))^2 \quad (9.16)$$

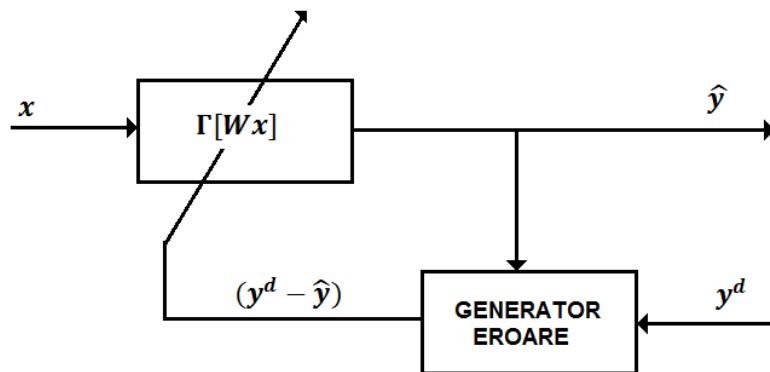


Figura 9.6 Mecanismul de instruire

Tehnicile de ajustare a parametrilor w_{ij} sunt tehnici de tip gradient descendenter, bazate pe calculul derivatei erorii (9.16).

$$\frac{\partial E(k)}{\partial w_{ij}} = \frac{\partial E(k)}{\partial \hat{y}_j(k)} \frac{\partial \hat{y}_j(k)}{\partial w_{ij}} = -e_j(k)x_i \frac{d\sigma(s_j)}{ds_j} \quad (9.17)$$

unde

$$s_j = \sum w_{ji} x_i + w_{0j} \quad (9.18)$$

Pentru o funcție de activare de tip sigmoid (9.1.9), $\frac{d\sigma}{ds_j}$ are forma:

$$\frac{d\sigma(s_j)}{ds_j} = \sigma(s_j)(1 - \sigma(s_j)) \quad (9.19)$$

Din aceste relații, algoritmul de ajustare al parametrilor w_{ij} devine:

$$w_{ij}(k+1) = w_{ij}(k) + \gamma e_j(k) \hat{y}_j(k)(1 - \hat{y}_j(k))x_i \quad (9.20)$$

O relație similară se poate stabili pentru variabila de polarizare:

$$w_0(k+1) = w_0(k) + \gamma e_j(k) \hat{y}_j(k)(1 - \hat{y}_j(k))x_i \quad (9.21)$$

9.2. Implementarea unei rețele neuronale

Implementarea unei rețele neuronale într-o tehnologie VLSI constă în integrarea într-un chip a componentelor de bază ce apar în configurația rețelei: componenta de adunare/multiplicare și componenta de activare a rețelei. Prima componentă realizează operația de însumare ponderată, $\sum w_i x_i$ iar a doua implementează una din funcțiile de activare (9.7) - (9.11). În literatură [10,13] sunt propuse și studiate mai multe soluții tehnologice de implementare.

9.2.1. Coeficienții de ponderare

Structura de bază a rețelei este formată dintr-un etaj inversor cu tranzistoare C-MOS, (Figura 9.7).

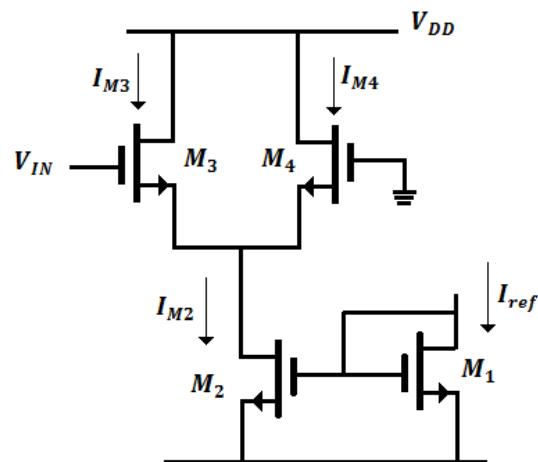


Figura 9.7 Circuitul de bază

9. Implementarea în VLSI a rețelelor neuronale

Curenții de ieșire din circuit sunt [14,15]

$$I_{M3} = \frac{K}{2}(V_{IN} - V_X - V_{TN})^2 \quad (9.22)$$

$$I_{M4} = \frac{K}{2}(-V_X - V_{TN})^2 \quad (9.23)$$

$$I_{M2} = I_{M3} + I_{M4} \quad (9.24)$$

unde K este determinat de parametrii geometrici și tehnologici ai implementării, V_X este potențialul de sursă al lui M_3 și M_4 iar V_{TN} este tensiunea de prag a tranzistorului MOS. Din relațiile de mai sus se deduce caracteristica de transfer a etajului de tip sigmoid (9.9) [14], alura curbei și factorii de comutare depinzând de parametrii W, L , lățimea și respectiv lungimea ariei de implementare (Figura 9.8).

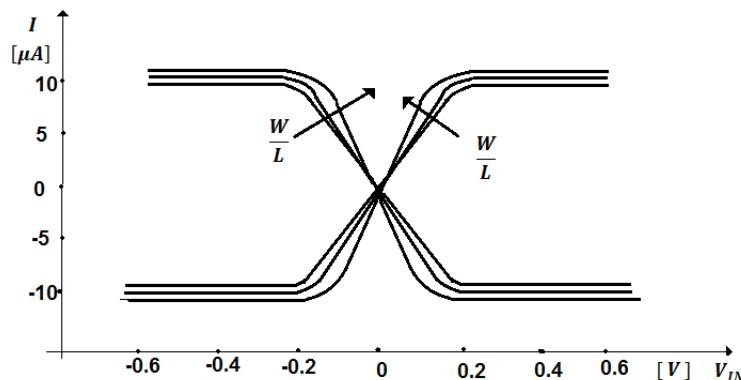


Figura 9.8 Caracteristica de transfer

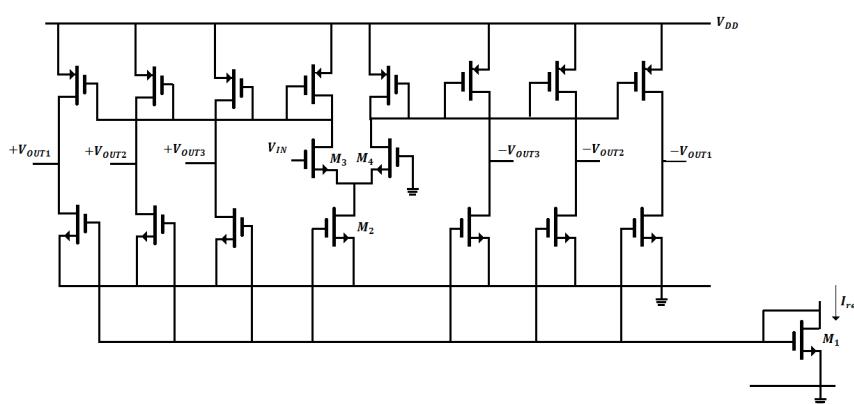


Figura 9.9 Implementarea unei celule neuronale

Prin utilizarea acestei configurații se poate implementa o celulă neuronală ca cea prezentată în Figura 9.9. Celula are o intrare determinată de potențialul V_{IN} și câte trei ieșiri ponderate, ponderile fiind determinate de raportul W/L de implementare. Ieșirile pot fi obținute cu ponderi pozitive sau negative pe ieșirile etajelor respective.

9.2.2. Implementarea prin amplificatoare Gilbert

O soluție constructivă mult abordată în literatura de specialitate se bazează pe utilizarea unor amplificatoare diferențiale Gilbert [14] care permit realizarea funcțiilor de multiplicare ponderată $w_i x_i$. Circuitul de bază al unei celule de acest tip este prezentat în Figura 9.10.

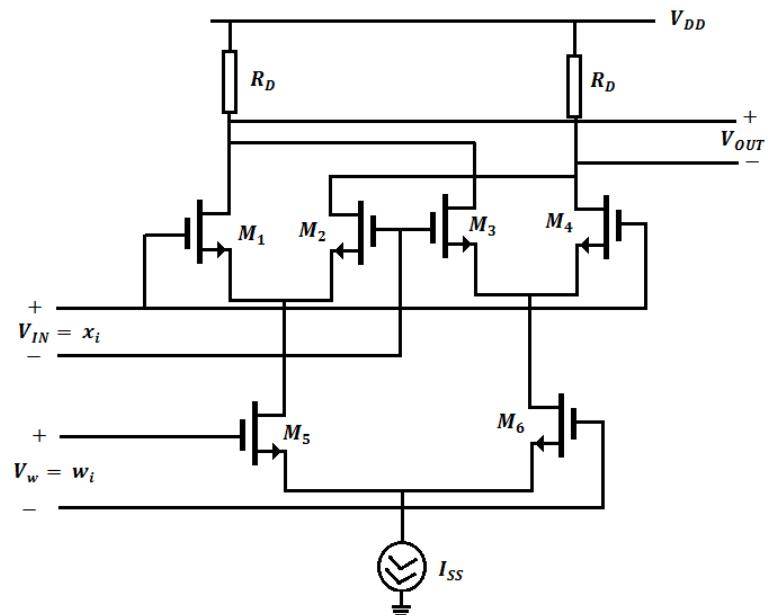


Figura 9.10 Amplificator diferențial Gilbert

Se demonstrează [14,15] că semnalul de ieșire V_{OUT} este obținut ca produsul celor două semnale, de intrare V_{IN} și de control V_w ,

$$V_{OUT} = \propto V_{IN} V_w \quad (9.25)$$

sau

9. Implementarea în VLSI a rețelelor neuronale

$$V_{OUT} = \propto x_i w_i \quad (9.26)$$

unde \propto este dat de parametrii de implementare tehnologici. Circuitul din Figura 9.9 rezolvă problema blocurilor de multiplicare, iar operația de adunare $\sum w_i x_i$ se obține prin simpla conectare la un nod comun, ca o însumare a curentilor generați de amplificatoarele diferențiale. Pentru realizarea funcției de activare se utilizează un amplificator diferențial (Figura 9.11) care generează la ieșire un curent de forma:

$$I_{OUT} = \gamma \tanh (\beta V_{IN}) \quad (9.27)$$

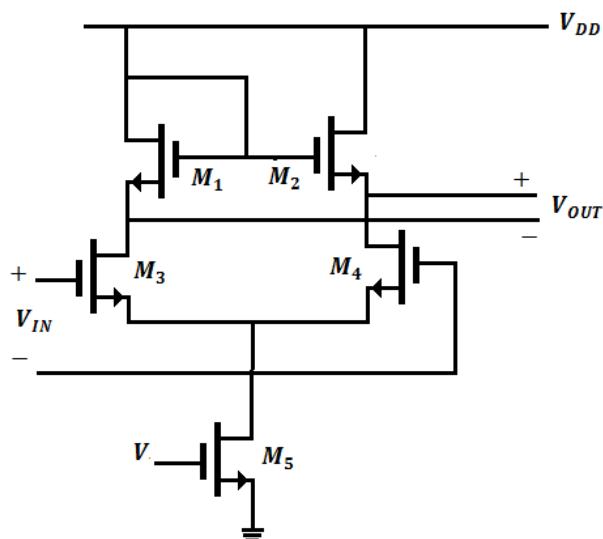


Figura 9.11 Circuitul funcției de activare

ceea ce permite obținerea unei funcții de activare tanh-sigmoid al cărui prag este controlat prin polarizarea V . Pentru exemplificare se va considera o rețea neuronală cu două nivele, conform Figura 9.12.

CMOS VLSI

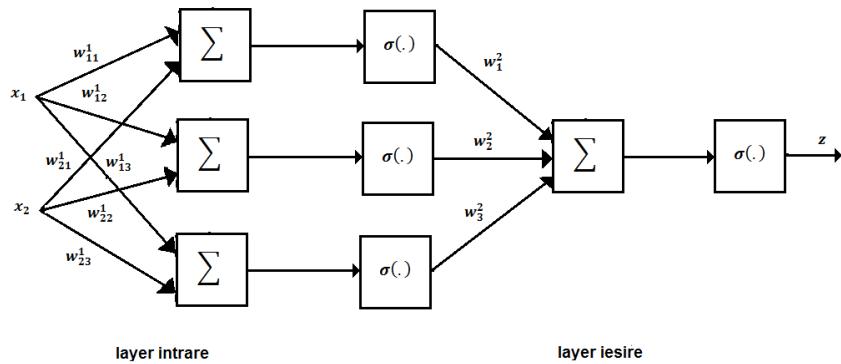


Figura 9.12 Rețea neuronală cu două nivele

Primul nivel de procesare realizează operațiile:

$$z_j = \sigma(\sum w_{ji}^1 x_i), j = 1, 2, 3 \quad (9.28)$$

în şase amplificatoare Gilbert pentru care operaţia de adunare se obţine prin conexiune directă a circuitelor de ieşire, urmate de circuitele de activare de tip tanh-sigmoid. În al doilea nivel de procesare se calculează ieşirea finală:

$$z = \sigma(\sum_j w_j z_j) \quad (9.29)$$

utilizând configurații de circuit similare (Figura 9.13)

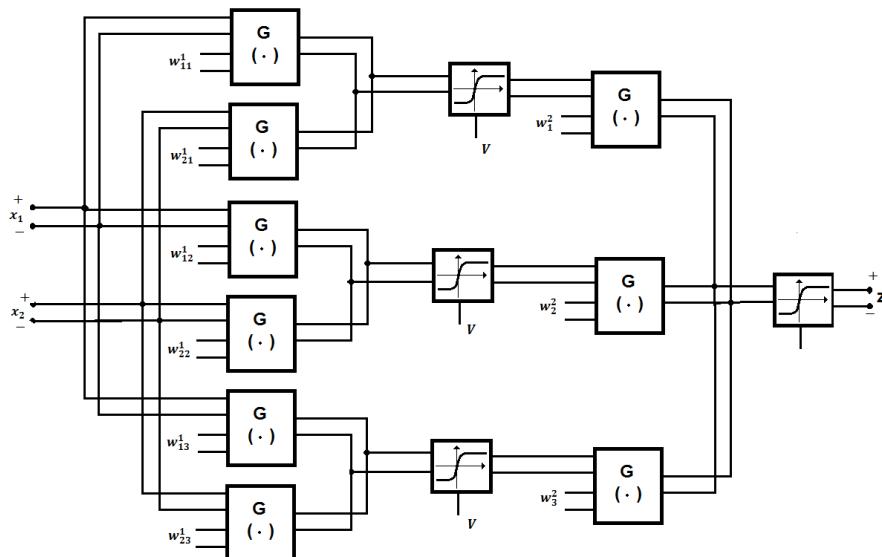


Figura 9.13 Implementarea VLSI a rețelei neuronale

9. Implementarea în VLSI a rețelelor neuronale

9.3. Conducerea unui robot prin rețelele neuronale folosind modelul invers

Utilizarea modelului invers al unui sistem reprezintă o procedură des întâlnită în arhitectura sistemelor de conducere [9] datorită obținerii, cel puțin teoretic, a unor performanțe de conducere superioare. Dificultatea majoră în abordarea acestor tehnici de conducere este determinată de calitatea implementării sistemului ce generează acest model invers. În acest context, utilizarea rețelelor neuronale pare să reprezinte o soluție care să asigure acuratețea implementării:

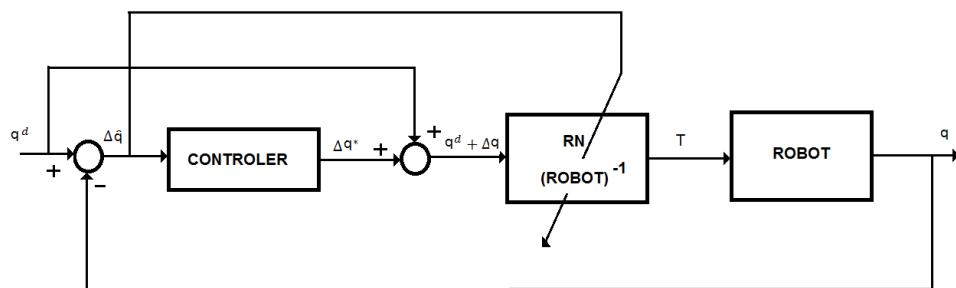


Figura 9.14 Conducerea prin RN a unui robot prin model invers

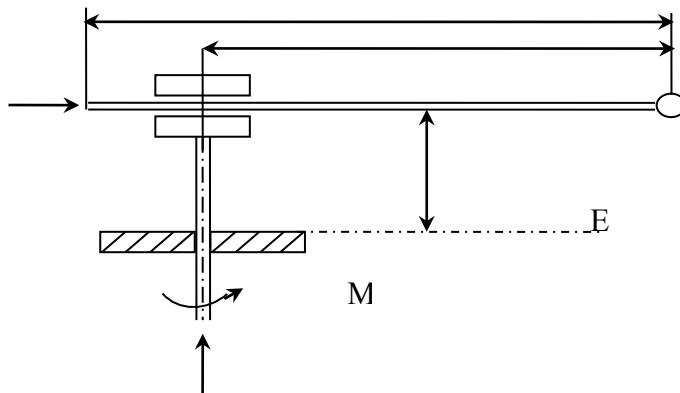


Figura 9.15 Configurația unui robot în coordonate cilindrice

Pentru exemplificarea procedurii se va considera un robot în coordonate cilindrice al cărui model dinamic are forma [9,10]:

$$J'_1(d_3)\ddot{\varphi}_1 + B_1(d_3, \dot{d}_3, \dot{\varphi}_1) = M_1 \quad (9.30)$$

$$m'\ddot{d}_2 + m'g = F_2 \quad (9.31)$$

$$(m_3 + m_n)\ddot{d}_3 + B_2(d_3, \dot{\varphi}_1) = F_3 \quad (9.32)$$

Introducându-se vectorul de stare și variațiile asociate:

$$q = \begin{bmatrix} \varphi_1 \\ d_2 \\ d_3 \end{bmatrix}; \quad \delta q = \begin{bmatrix} \delta\varphi_1 \\ \delta d_2 \\ \delta d_3 \end{bmatrix}; \quad \Delta q = \begin{bmatrix} \Delta\varphi_1 \\ \Delta d_2 \\ \Delta d_3 \end{bmatrix} \quad (9.33)$$

Pentru sistemul în buclă închisă, valorile prescrise sunt:

$$q_d^*(t) = \begin{bmatrix} q_d(t) \\ \dot{q}_d(t) \\ \ddot{q}_d(t) \end{bmatrix} \quad (9.34)$$

Iar compensarea erorii este realizată prin controler:

$$\begin{aligned} \delta q &= K_1 \Delta q \\ \delta \dot{q} &= K_2 \Delta \dot{q} \\ \delta \ddot{q} &= K_3 \Delta \ddot{q} \end{aligned} \quad (9.35)$$

Se demonstrează [9] că prin alegerea corespunzătoare a matricilor de control K_1, K_2, K_3 stabilitatea conducerii este asigurată dacă

$$\begin{bmatrix} \Delta \dot{q} \\ \Delta \ddot{q} \end{bmatrix} = \begin{bmatrix} 0 & I \\ -LP_1(I - K_1) & -LP_2(I - K_2) \end{bmatrix} \begin{bmatrix} \Delta q \\ \Delta \dot{q} \end{bmatrix} \quad (9.36)$$

sistemul (9.36) este asimptotic stabil. Pentru implementarea modelului invers al robotului se utilizează configurația RN din Figura 9.16 cu o rețea cu un singur layer cu trei neuroni ce generează ieșirile. Ponderile modelului sunt ajustate prin tehnici de gradient pe baza erorii vectorului de stare Δq . În scopul simplificării circuitului, anumite componente neliniare sunt generate direct, $\dot{\varphi}_1^2$.

9. Implementarea în VLSI a rețelelor neuronale

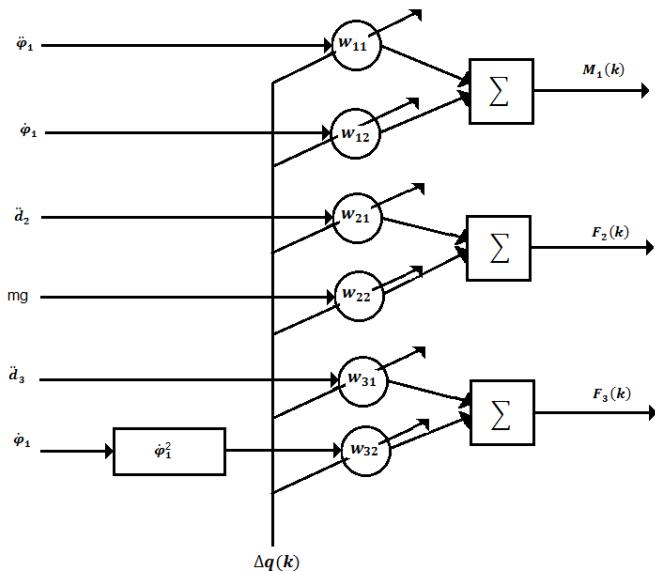


Figura 9.16 RN pentru modelul invers al robotului

9.4. Conducerea unui robot prin rețelele neuronale folosind tehnica decuplării neliniare

Tehnica decuplării neliniare dezvoltată în [9,10] reprezintă o soluție practică de implementare a unor structuri de conducere performante pentru modele robotice cu neliniarități deosebit de pronunțate. Se va considera modelul dinamic al unui robot de forma:

$$\dot{x} = A(x) + B(x)u \quad (9.37)$$

$$y = C(x) \quad (9.38)$$

Se va considera o lege de conducere de forma:

$$u = -D^*(x)^{-1}(C^*(x) + M^*(x)) + D^*(x)^{-1}\Lambda w \quad (9.39)$$

unde

$$C^*(x) = [C^*_1(x), C^*_2(x), \dots, C^*_n(x)]^T \quad (9.40)$$

$$D^*(x) = [D^*_1(x), D^*_2(x), \dots, D^*_n(x)]^T \quad (9.41)$$

$$C^*_i(x) = \frac{\partial}{\partial x} \left(\frac{\partial C_i(x)}{\partial x} A(x) \right) A(x) \quad (9.42)$$

$$D^*_i(x) = \frac{\partial}{\partial x} \left(\frac{\partial C_i(x)}{\partial x} A(x) \right) B(x) \quad (9.43)$$

$$M^*(x) = \begin{bmatrix} \alpha_{01}C_1(x) + \alpha_{11} \frac{\partial C_1(x)}{\partial x} A(x) \\ \alpha_{02}C_2(x) + \alpha_{12} \frac{\partial C_2(x)}{\partial x} A(x) \\ \dots \\ \alpha_{0i}C_i(x) + \alpha_{1i} \frac{\partial C_i(x)}{\partial x} A(x) \\ \dots \\ \alpha_{0n}C_n(x) + \alpha_{1n} \frac{\partial C_n(x)}{\partial x} A(x) \end{bmatrix} \quad (9.44)$$

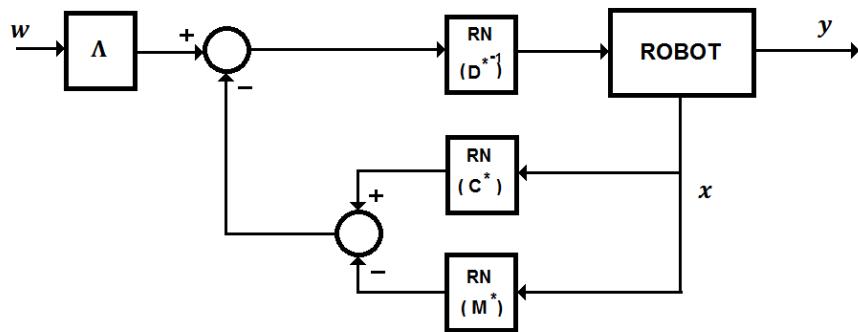


Figura 9.17 Conducerea unui robot cu RN prin tehnica decuplării neliniare

Componentele neliniare ale legii de conducere (9.39) se implementează prin rețele neuronale conform circuitului din Figura 9.16. Aplicarea legii de conducere (9.39) determină o comportare generală a sistemului (9.37) și (9.38) sub forma unui model liniar descris prin ecuațiile []:

$$\ddot{y} + \alpha_{1i}\dot{y}_i + \alpha_{0i}y_i = \lambda_i w_i, i = 1, 2, \dots \quad (9.45)$$

unde parametrii $\alpha_{1i}, \alpha_{0i}, \lambda_i$ sunt astfel selectați încât să permită obținerea performanțelor dorite. Pentru implementarea termenilor neliniari $C^*(x), D^*(x), M^*(x)$ prin RN se utilizează aceleasi proceduri ca cele descrise în secțiunea precedentă.

10. IMPLEMENTAREA ÎN VLSI A SISTEMELOR FUZZY

10.1. Principiul logicii fuzzy

Logica fuzzy poate fi interpretată ca un concept de implementare a logicii umane în problemele de tip ingineresc. În limbajul uman există diferite tipuri de incertitudini numite frecvent „incertitudini lexicale”, care identifică imprecizia în evaluarea unor concepte sau în stabilirea unor concluzii. Această imprecizie în evaluarea unui element dintr-o mulțime X este evaluată prin funcția de apartenență asociată, $\tilde{\mu}(x)$ al elementului x/X . Funcția de apartenență transformă X , universul de discurs, într-un spațiu M al funcției de apartenență $\tilde{\mu}(x)$. Dacă M conține numai două puncte 0 și 1, atunci caracterul incert dispare și se poate discuta de o mulțime convențională [15-18]. O mulțime clasică (mulțime *crisp*) este definită ca o colecție de elemente sau obiecte x/X care poate fi finită, numărabilă sau nu. Dacă considerăm acum o submulțime A , A/X , un element x poate să „aparțină lui A ” sau să „nu aparțină lui A ”. Dacă x/A atunci declarația „aparține lui A ” este adevărată, în caz contrar fiind falsă.

O mulțime fuzzy \tilde{A} este o mulțime de perechi ordonate

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) / x \in X\} \quad (10.1)$$

Considerând X un univers de discurs și o submulțime A a lui X , atunci submulțimii fuzzy \tilde{A} i se asociază funcția caracteristică, funcția de apartenență

$$\mu_{\tilde{A}} : X \rightarrow [0,1] \quad (10.2)$$

prin care este specificat gradul prin care x este un membru al submulțimii A . Formele cele mai des întâlnite ale funcțiilor de apartenență în aplicații industriale sunt cele triunghiulare și trapezoidale (Figura 10.1), [17], [18].

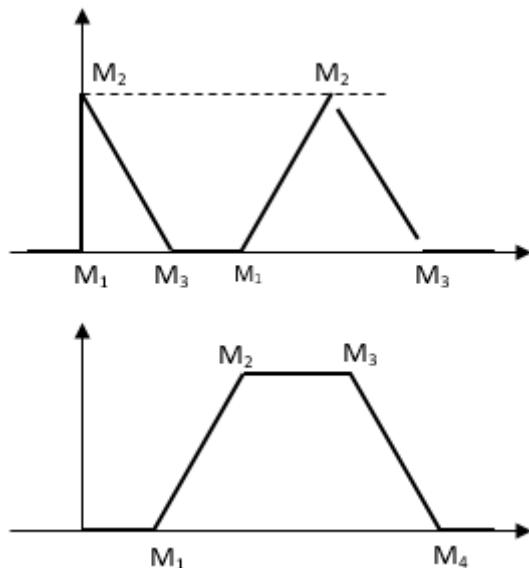


Figura 10.1 Funcții de apartenență triunghiulare și trapezoidale

Gradul de apartenență este exprimat, în aceste cazuri, prin relații liniare. În mod normal, aceste funcții sunt definite prin punctele de referință \$M_1, M_2, M_3\$ sau \$M_1, M_2, M_3, M_4\$, respectiv.

Sunt utilizate de asemenea și alte forme [??], de exemplu funcții de apartenență de tip "clopot" (Figura 10.2), definite prin relațiile:

$$\mu(x) = \frac{1}{1 + \left(\frac{x - x_0}{a}\right)^2} ; -\infty < x < +\infty \quad (10.3)$$

Respectiv

$$\mu(x) = \frac{1}{2} \left[1 + \cos \frac{\pi(x - x_0)}{2a} \right] ; x_0 - 2a \leq x \leq x_0 + 2a \quad (10.4)$$

unde \$x_0\$ definește poziția vârfului pentru \$\mu = 1\$ iar \$a\$ reprezintă lărgimea domeniului.

10. Implementarea în VLSI a sistemelor fuzzy

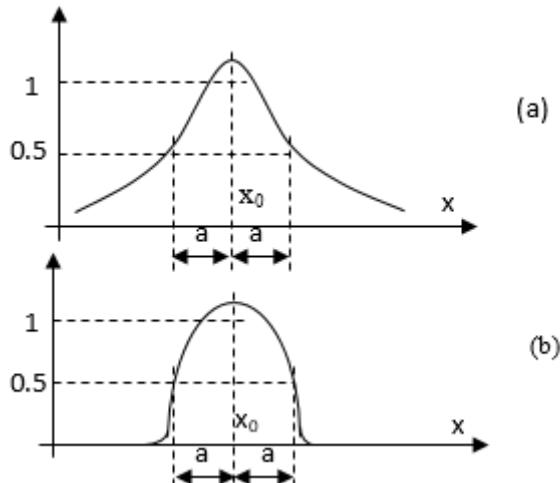


Figura 10.2 Forma generală a funcțiilor de apartenență

Sistemele convenționale de conducere fuzzy se încadrează în configurația clasică a sistemelor de reglare (Figura 10.3). Într-o astfel de structură, valoare dorită a variabilei de poziție (de referință) q_d este comparată cu valoarea reală măsurată de traductor (q_T). Eroarea rezultată este procesată într-un regulator care generează mărimea de comandă a sistemului de acționare u și, prin aceasta, este determinată mișcarea elementului mecanic. Regulatoarele uzuale utilizate în aceste sisteme sunt de tipul P, PI și PID, ele asigurând, în condiții de alegere corespunzătoare a parametrilor regulatorului, performanțele de control dorite. Un sistem de conducere în logică fuzzy (SCLF) respectă configurația generală a oricărui sistem de conducere, rolul regulatorului convențional fiind preluat de un Controller în Logică Fuzzy (CLF) (Figura 10.4). Un CLF va implementa un algoritm de conducere care, prin procesarea erorii e , va determina performanțele calitative dorite pentru întregul sistem de conducere, în mod curent obținerea unei erori staționare zero. În contrast cu regulatorul standard, CLF nu implementează o relație matematică bine definită (algoritmul de reglare), ci utilizează inferențe cu mai multe reguli bazate pe variabile lingvistice și care sunt tratate prin operatori în logică fuzzy. Configurația generală a unui CLF cuprinde trei părți (Figura 10.5) [16-19]:

1. fuzificare;
2. inferențe (baza de reguli);
- defuzificare

CMOS VLSI

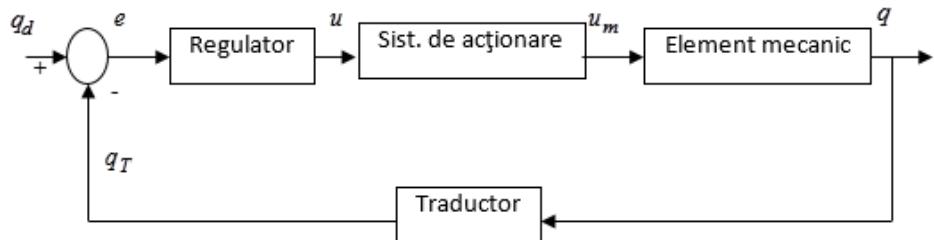


Figura 10.3 Sistem convențional de reglare

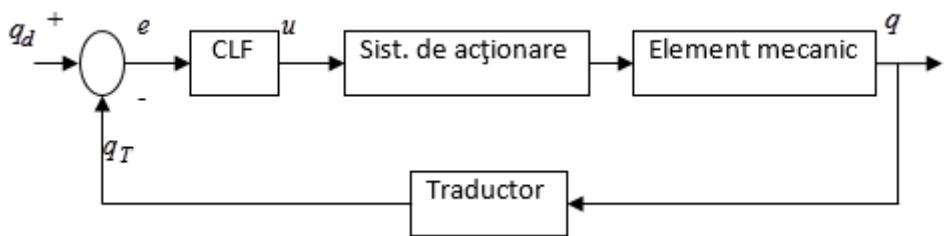


Figura 10.4 Sistem de reglare fuzzy

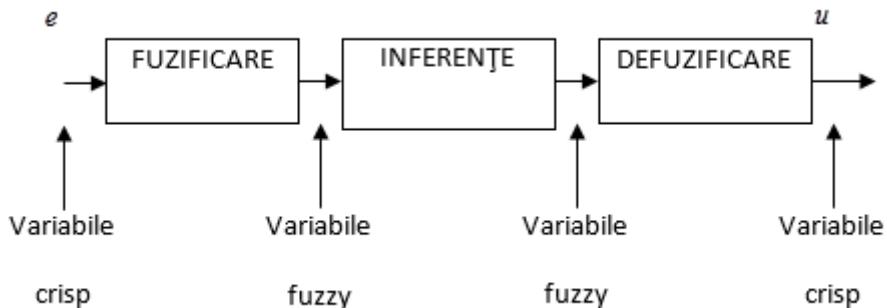


Figura 10.5 Părțile componente ale unui CLF

Blocul de fuzificare transformă datele de intrare, eroarea sistemului, în mărimi fuzzy, atribuind acestor mărimi variabile lingvistice și asociindu-le funcții de apartenență corespunzătoare. Blocul de inferențe reprezintă partea principală a unui CLF. Acest bloc generează legea de control sub forma unei familii de reguli logice de tipul IF...THEN, ce descriu relațiile între intrarea e (eroare) și ieșirea u a controlerului. Adaptând un model direct, aceste reguli implementează de fapt o lege de formă generală

10. Implementarea în VLSI a sistemelor fuzzy

$$u(k) = F(e(k), e(k-1), \dots, e(k-v), u(k-1), u(k-2), \dots, u(k-v)) \quad (10.5)$$

Din considerente practice, implementarea unei astfel de legi pentru $v > 1$ este extrem de complexă și dificil de aplicat. În mod curent, legea de control de mai sus capătă forma simplificată

$$u(k) = F(e(k), e(k-1), u(k-1)) \quad (10.6)$$

sau

$$u(k) = F(e(k), e(k-1)) \quad (10.7)$$

Legea de control este generată prin baza de reguli IF...THEN sub forma

$$\Delta u(k) = u(k) - u(k-1) \quad (10.8)$$

și ieșirea actuală se obține sub forma:

$$u(k) = u(k-1) + \Delta u(k) \quad (10.9)$$

Un astfel de CLF este numit CLF-Mamdani, după numele cercetătorilor Mamdani și Assilian care l-au propus pentru prima dată în anul 1975 (Fig 10.6), [20-23,27].

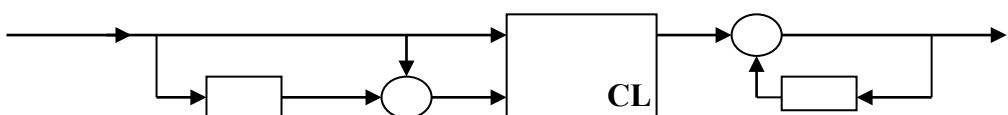


Figura 10.6 Modelul Mamdani

Fiecare din regulile bazei este caracterizată printr-o parte IF, numită „antecedent” și o parte THEN, numită „consecință”. Antecedentul unei reguli conține setul de condiții, consecința conține concluzia. Modul în care operează CLF-ul este următorul: dacă condițiile antecedentului sunt satisfăcute, atunci concluzia, consecința, este activă. CLF-ul poate fi privit ca un sistem care are ca intrări variabilele incluse în antecedentele regulilor și ca ieșire variabila inclusă în consecință. Așadar se va putea defini eroarea $e(k)$ și variația ei $\Delta e(k)$, ca intrări în sistem, iar ieșirea poate fi reprezentată de variația $\Delta u(k)$. Un exemplu de regulă poate fi următorul:

IF $e(k)$ este POZITIV AND $\Delta e(k)$ este ZERO

THEN $\Delta u(k)$ este POZITIV

...

Este evident că intrările și ieșirile unui CLF sunt stări sau substări ale întregului sistem controlat, deci CLF poate fi privit și ca un controler de variabilă de stare, guvernăt de o familie de reguli și de un mecanism de inferență fuzzy.

Ultimul bloc al unui CLF, blocul de defuzificare, asigură formarea unor semnale crisp, prin metode specifice, compatibile cu sistemele de acțiune utilizate în controlul roboților.

Interfațarea pe intrare a unui controler (CLF) este realizată prin blocul de fuzificare. Aceasta trebuie să asigure:

- conversia mărimilor de intrare (eroare) e într-o formă digitală compatibilă cu procesarea ulterioară în logică fuzzy. Dacă eroarea e este un semnal continuu, se impune o conversie analog-numerică corespunzătoare. De asemenea, dacă e este un semnal numeric, o procesare numerică suplimentară este necesară pentru asigurarea performanțelor dorite;
- formarea variabilelor ce vor determina condițiile antecedentului în baza de reguli a CLF. În general, aşa cum s-a arătat anterior, o primă variabilă de intrare este eroarea $e(k)$, dar impunerea unor condiții complexe necesită generarea și a altor variabile de intrare. Acestea pot fi definite prin mărimea de eroare evaluată la câteva momente de timp, $e(k-1), e(k-2), \dots, e(k-v)$ sau prin alte variabile de stare ale sistemului condus, cum ar fi viteza sau accelerarea brațului mecanic (pentru controlul unui sistem robot);
- exprimarea variabilelor de intrare ca mulțimi fuzzy. În acest sens, mărimile fizice li se atribuie variabile lingvistice, variabile fuzzy și funcții de apartenență. Se determină totodată universul de discurs pe care este definită fiecare variabilă.

Funcțiile de apartenență utilizate în sistemele de conducere au forme triunghiulare sau trapezoidale. De asemenea, universul de discurs este frecvent ales ca $[-1,1]$, iar variabilele de intrare astfel definite sunt

10. Implementarea în VLSI a sistemelor fuzzy

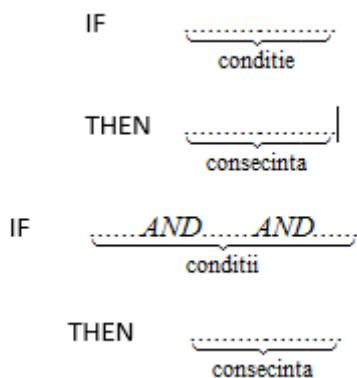
considerate normalize. Este evident că aceasta impune utilizarea unor factori de scală corespunzători pentru fiecare variabilă. Alegerea funcțiilor de apartenență nu poate să fie supusă unor reguli foarte precise. Acestea pot fi alese simetrice sau nesimetrice (în raport cu originea), triunghiulare sau trapezoidale, echidistante sau nu, pe universul de discurs normalizat sau nu. Totuși, indiferent de modalitățile de alegere, se impun categoric respectarea următoarelor condiții:

$$\mu_{A_1}(x) \cap \mu_{A_2} \cap \dots \cap \mu_{A_n} \neq \emptyset, \quad \forall x \in U \quad (10.10)$$

$$\mu_{A_1}(x) \cup \mu_{A_2} \cup \dots \cup \mu_{A_n} = 1, \quad \forall x \in U \quad (10.11)$$

Acste condiții arată că relația (10.10) impune ca pentru orice punct U din universul de discurs să fie definită cel puțin o funcție $\mu_{A_i}(x)$. A doua condiție este determinată de o anumită comoditate în calculele numerice ale CLF.

Partea principală a unui CLF, cea care implementează algoritmul de conducere al sistemului, este baza de reguli (blocul de inferențe) a sistemului. Așa cum am arătat anterior, fiecare regulă este formată din două părți: partea IF, numită și antecedent și partea THEN, numită consecință.



Antecedentul conține condiția sau condițiile (variabile de intrare) care activează regula. Dacă mai multe variabile formează condițiile de activare, acestea sunt legate prin AND. Consecința permite generarea variabilei sau a variabilelor de ieșire activate ca urmare a îndeplinirii condițiilor regulii.

Este evident că procesarea unor reguli într-o bază (mecanismul de inferență) impune determinarea, în primul rând, a condițiilor ce definesc

regula și a gradului lor de activare. Acest lucru este obținut prin tratarea condiției ca mulțime fuzzy și prin testarea gradului de activare al acesteia cu ajutorul funcțiilor de apartenență asociate. Într-o scriere sintetică, o regulă poate fi exprimată sub forma:

Regula j: IF CONDIȚIA C_1 AND CONDIȚIA C_2 AND... AND CONDIȚIA C_p

THEN DECIZIA D_j

Considerând \tilde{A}_1 , mulțimea fuzzy asociată condiției C_1 , gradul de activare al condiției pentru o valoare $x=x^*$ în universul de discurs este exprimat prin

$$g_{\tilde{A}_1} = \mu_{\tilde{A}_1}(x^*) \quad (10.12)$$

În figura 10.7 este ilustrat grafic modul de calcul al gradului de activare.

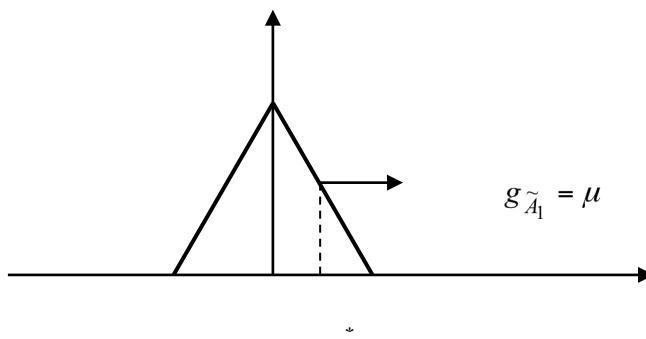


Figura 10.7 Ilustrarea gradului de activare

Pentru o regulă completă definită prin condițiile C_1, C_2, \dots, C_p cărora li se asociază mulțimile fuzzy $\tilde{A}_1, \tilde{A}_2, \dots, \tilde{A}_p$, definite pe universul de discurs x_1, x_2, \dots, x_p , se obține

$$g_{\tilde{A}_i} = \mu_{\tilde{A}_i}(x_i^*), i=1, 2, \dots, p \quad (10.13)$$

Condițiile C_i sunt interpretate prin operatorul AND

$$f_{D_j}(x_1^*, x_2^*, \dots, x_p^*) = AND(g_{\tilde{A}_1}(x_1^*), \dots, g_{\tilde{A}_p}(x_p^*)) \quad (10.14)$$

10. Implementarea în VLSI a sistemelor fuzzy

unde f_{D_j} va reprezenta funcția de activare a variabilei de ieșire, consecință, decizie, a regulii determinate de expresia THEN. Sinteză deciziei, la nivelul unei reguli, este obținută în general prin algoritmul Mandami [24,25] care este scris sub forma

$$\tilde{D}_j^* = \mu_{\tilde{D}_j}(u) \Big|_{\mu_{\tilde{D}_j}(u) \leq f_{D_j}, u \in U} \quad (10.15)$$

$$\tilde{D} = OR(\tilde{D}_1^*, \dots, \tilde{D}_r^*) = \max_u (\tilde{D}_1^*, \dots, \tilde{D}_r^*) \quad (10.16)$$

In implementarea unui CLF se realizeaza fuzificarea mărimilor de intrare și procesarea algoritmului de conducere sub forma unei baze de reguli. În final, mărimea de ieșire este generată sub forma unei multimi fuzzy. Este evident că se impune compatibilizarea acestei mărimi cu procesul condus și este necesară reconversia mărimii de ieșire într-o mărime crisp. Procedura aceasta este cunoscută sub numele de defuzificare.

Defuzificarea reprezintă o etapă importantă în structura generală a proiectării unui CLF și literatura de specialitate abundă în soluții [21-25]. În cele ce urmează se vor prezenta câteva din tehniciile de implementare cele mai eficiente.

Defuzificarea prin „centru de greutate” este cea mai răspândită procedură de reconversie a deciziei fuzzy de la ieșirea CLF într-o variabilă crisp. Această metodă se bazează pe evaluarea funcției de apartenență a deciziei fuzzy după aceleași reguli ca cele utilizate în calculul centrului de greutate din mecanica clasică. În acest mod se asociază funcției de apartenență a variabilei de ieșire o valoare crisp corespunzătoare. În cazul general, defuzificarea apelează la formula clasică a „centrului de greutate”

$$u^* = \frac{\int_0^1 u \mu^*(u) du}{\int_{-1}^1 \mu^*(u) du} \quad (10.17)$$

unde $\mu^*(u)$ este funcția de apartenență a deciziei finale, numitorul reprezintă aria delimitată de funcția de apartenență, iar numărătorul definește momentul ariei. În această relație s-a presupus că universul de discurs U este normalizat. Într-o formă mai generală, expresia (10.18) poate fi rescrisă ca

$$u^* = \frac{\int_U u \mu(u) du}{\int_U \mu(u) du} \quad (10.18)$$

În mod curent este utilizată o procedură de discretizare

$$u^* = \frac{\sum_{j=1}^n u_j \mu(u_j)}{\sum_{j=1}^n \mu(u_j)} \quad (10.19)$$

ceea ce permite, în cele mai multe cazuri, un calcul ușor de implementat.

Pentru anumite cazuri particulare ale funcției de apartenență, relațiile ((10.18) – ((10.20) pot căpăta anumite forme simplificate. De exemplu, dacă ieșirea este definită prin funcții singleton, expresia ((10.20) devine

$$u^* = \sum_{j=1}^r u_j w_j \quad (10.20)$$

unde

$$w_j = \frac{\mu_j}{\sum_{j=1}^r \mu_j} \quad (10.21)$$

O interpretare și mai facilă, deși evident destul de aproximativă, se poate obține dacă funcția de apartenență conține un maxim sau domenii maximale,

$$u^* = \mu_{\max} \quad (10.22)$$

10.2. Implementarea VLSI a componentelor fuzzy

Componentele principale ale unei arhitecturi ce implementează un sistem de conducere fuzzy sunt prezentate în Figura 10.8. Sistemul conține blocuri de fuzificare ale mărimilor de intrare, operatori MIN, MAX ce implementează regulile fuzzy și un bloc de defuzificare.

10. Implementarea în VLSI a sistemelor fuzzy

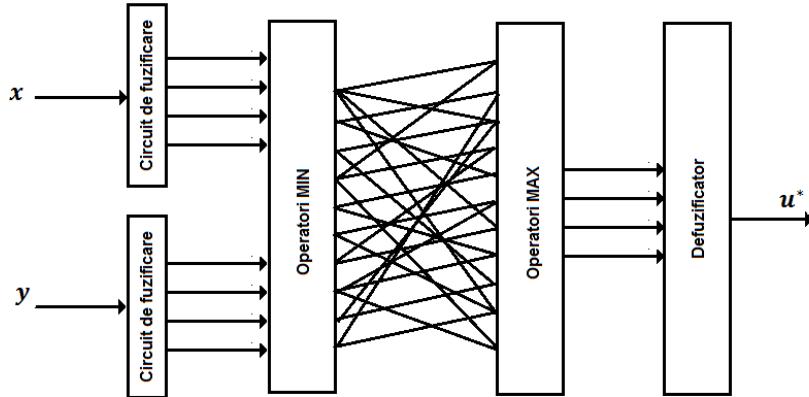


Figura 10.8 Componentele unui sistem fuzzy

10.2.1. Circuite de fuzificare

Circuitele de fuzificare convertesc mărimile analogice de tip crisp în variabile fuzzy. Conversia se realizează prin intermediul funcțiilor de apartenență implementate sub una din formele standard: triunghiular, trapezoidal sau Gausian. În Figura 10.9 este prezentat un astfel de circuit. Arhitectura circuitului se bazează pe utilizarea unei conexiuni în cascadă de circuite diferențiale cu polarizări de referință diferite. În etajul inferior este conectat un generator de curent I_{SS} . În Figura 10.10 este prezentată caracteristica de tip sigmoidal realizată de primul etaj. Pentru tensiuni de intrare V_{IN} mai mici decât potențialul de referință V_{ref1} tranzistorul M_1 este blocat și curentul generatorului I_{SS} este comutat numai pe tranzistorul M_4 ce atinge valoarea maximă I_{SS} . Când tensiunea de intrare atinge pragul de referință, începe conducția tranzistorului M_1 și comutarea treptată a curentului de drenă de la tranzistorul M_4 către tranzistorul M_1 . Ulterior, acesta se blochează. Obținerea unei forme de undă de tip gausian se realizează în același mod pe ieșirea tranzistorului M_5 . În acest caz, pentru tensiuni mai mici sau mai mari decât tensiunea V_{ref2} , curentul din drenă I_5 este practic zero, sursa I_{SS} fiind în întregime absorbită de tranzistorii M_1 și M_4 . Numai pentru valori în jurul tensiunii de referință V_{ref2} , tranzistorul M_5 conduce și asigură vârful curbei gausiene. În mod similar se obțin curbe de tip gauss în jurul valorii de referință V_{ref3} sau funcție de tip sigmoid cu pantă

CMOS VLSI

pozitivă. Alura acestor curbe, panta și fronturile, pot fi modificate prin ajustarea parametrilor tehnologici de implementare, w și L .

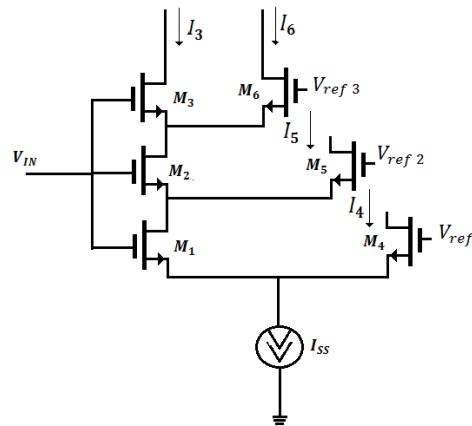


Figura 10.9 Circuit de fuzzyficare

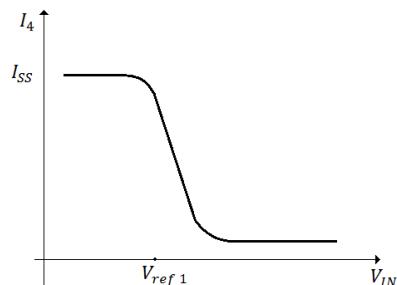


Figura 10.10 Obținerea unei funcții sigmoid

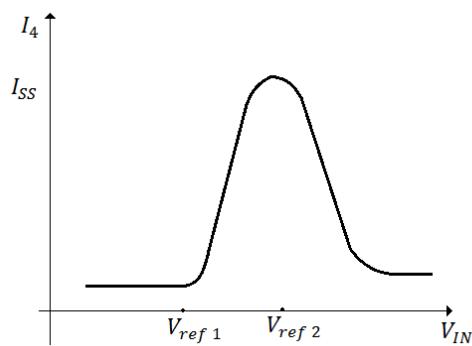


Figura 10.11 Obținerea unei funcții de tip Gauss

10. Implementarea în VLSI a sistemelor fuzzy

10.2.2. Circuit pentru realizarea funcției MAX

Configurația circuitului (Figura 10.12) se bazează pe principiul „winner – take - all” al lui Lazarro [26-28]. În acest caz, potențialul de intrare de nivel maxim determină un nivel maxim corespunzător pe grila tranzistorului M_a și blocarea tuturor celorlalte perechi de tranzistoare de intrare.

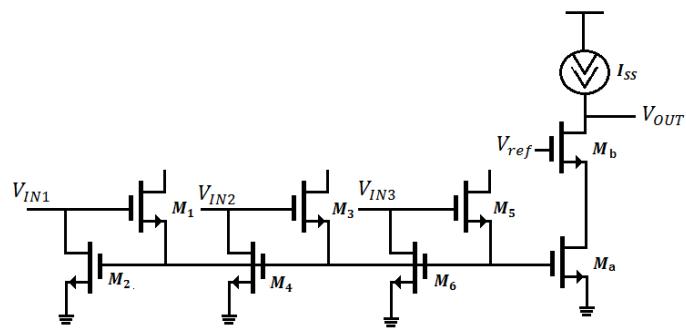


Figura 10.12 Circuit pentru realizarea funcției MAX

10.2.3. Circuit pentru realizarea funcției MIN

Implementarea funcției MIN se obține după un principiu similar cu tranzistoare CMOS (Figura 10.12). Semnalele de intrare aplicate pe grilele tranzistorilor M_1, M_2, \dots vor determina conducția cu prioritate a tranzistorului cu potențial minim și blocare a tuturor celorlalte tranzistoare de intrare. Acest potențial se va aplica pe grila tranzistorului de ieșire M_4 generând semnalul corespunzător pe ieșirea circuitului.

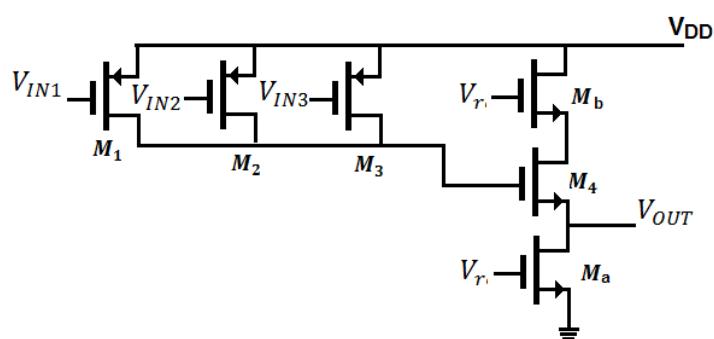


Figura 10.13 Circuit pentru realizarea funcției MIN

10.2.4. Circuit pentru realizarea funcției de defuzificare

Cea mai simplă soluție pentru realizarea circuitului de defuzificare se bazează pe cazul în care se utilizează funcții singleton (10.21),

$$u^* = \sum_{j=1}^r u_j w_j \quad (10.23)$$

Circuitul se bazează pe implementarea unor sumatoare ponderate realizate cu amplificatoare diferențiale Gilbert.

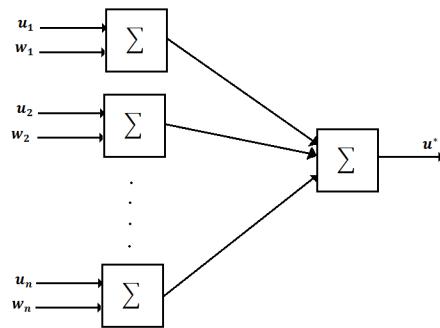


Figura 10.14 Circuit pentru realizarea funcției de defuzificare

10.3. Conducerea dinamică a unui robot mobil în câmp cu potențial artificial

Se consideră modelul dinamic al unui robot mobil definit sub forma generală prin:

$$m\ddot{x} + \mu mg - k_f \dot{x} = F_x \quad (10.24)$$

$$m\ddot{y} + \mu mg - k_f \dot{y} = F_y \quad (10.25)$$

unde $\mathbf{x} = (x, y)$ definesc coordonatele de mișcare, iar $F_x, F_y, k_f \dot{x}, k_f \dot{y}$ reprezintă forțele active sau de frecare dinamică ce acționează asupra robotului. Într-un câmp de potențial artificial Π , aceste forțe devin:

10. Implementarea în VLSI a sistemelor fuzzy

$$F_x = -k_1 \dot{x} - \frac{\partial \Pi}{\partial x} \quad (10.26)$$

$$F_y = -k_1 \dot{y} - \frac{\partial \Pi}{\partial y} \quad (10.27)$$

ceea ce determină obținerea unui model dinamic sub forma:

$$m\ddot{x} + \mu mg - k_f \dot{x} = -k_1 \dot{x} - \frac{\partial \Pi}{\partial x} \quad (10.28)$$

$$m\ddot{y} + \mu mg - k_f \dot{y} = -k_1 \dot{y} - \frac{\partial \Pi}{\partial y} \quad (10.29)$$

unde Π cuprinde componente de atracție și, respectiv, de repulsie:

$$\Pi = \Pi_A + \Pi_R \quad (10.30)$$

Eroarea între poziția dorită a țintei și poziția robotului va fi:

$$e = x_T - x \quad (10.31)$$

Se va considera o dreaptă de comutare de forma:

$$S = e + \sigma \dot{e} \quad (10.32)$$

ceea ce va permite implementarea unei legi de conducere *sliding* în jurul dreptei de comutare (Figura 10.15).

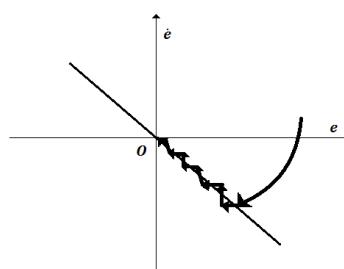


Figura 10.15 Evoluția de tip sliding

Implementarea acestui controler fuzzy impune evaluarea erorilor de poziție și de viteză pe cele două axe de mișcare și generarea forțelor de acționare prin ieșirile generate.

CMOS VLSI

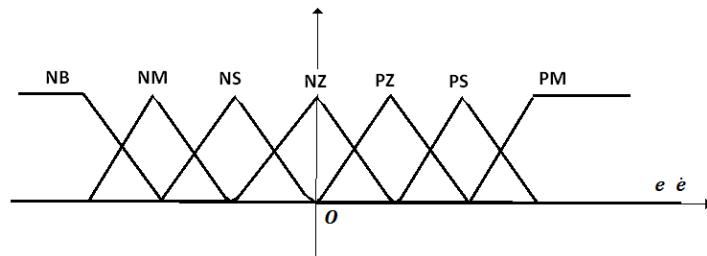


Figura 10.16 Funcțiile de apartenență ale variabilelor de intrare

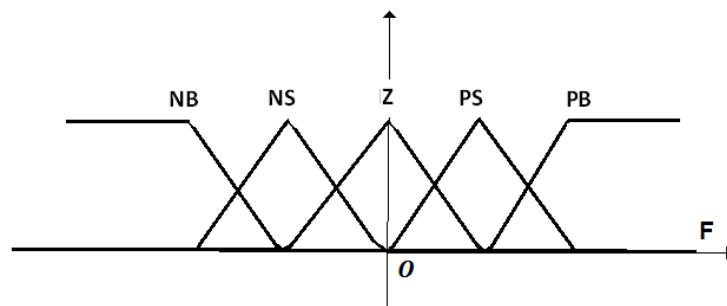


Figura 10.17 Funcțiile de apartenență al variabilei de ieșire

e/\dot{e}	NB	NM	NS	NZ	PZ	PS	PM	PB
PB	Z	NS	NS	NB	NB	NB	NB	NB
PM	PS	Z	NS	NS	NB	NB	NB	NB
PS	PS	PS	Z	NS	NS	NB	NB	NB
PZ	PB	PS	PS	Z	NS	NS	NB	NB
NZ	PB	PB	PS	PS	Z	NS	NS	NB
NS	PB	PB	PB	PS	PS	Z	NS	NS
NM	PB	PB	PB	PB	PS	PS	Z	NS
NB	PB	PB	PB	PB	PB	PS	PS	Z

Tabel 10.1 Reguli fuzzy pentru controlul robotului

Ansamblul de reguli fuzzy (Tabelul 10.1) asigură evoluția traiectoriei în planul (e, \dot{e}) în jurul dreptei de comutare și convergența către origine.

11. BIBLIOGRAFIE

- [1] http://en.wikipedia.org/wiki/Hydrofluoric_acid
- [2] Bezhad Razavi, Design of CMOS Integrated Circuits, Mc Graw – Hill Co., 2001.
- [3] R.S. Muller, T.I. Kamins, Device Electronics for Integrated Circuits, New York Wiley, 1986
- [4] Y.Tsividis, Operation and Modeling of the MOS Transistor, Mc Graw - Hill Co, 1999
- [5] Y. Taur, T.H. Ning, Fundamental of Modern VLSI Devices, New York Cambridge University Press, 1998
- [6] M. Gopal, Digital Control and State Variable System, McGraw - Hill Book Co., 2006
- [7] D. Seborg, T. Edgar, D. Mellichamp, Process Dynamics and Control, Willey Series in Chemical Engineering, 1989
- [8] K. Prasad, A. Raj, VLSI Implementation of Digital Compensators and Predictive PID Controllers, IEEE Xplore, August, 2009.
- [9] M. Ivanescu, Sisteme avansate de conducere in robotica, Ed Scrisul Romanesc, Craiova, 2004
- [10] M. Gopal, Digital Control and State Variable Systems, Mc Graw Publishing Co, 2003
- [11] Ben Kruse, P. van der Smagt, An Introduction to Neural Networks, Amsterdam University, 1996
- [12] M. Hajek, Neural Networks, Mc Graw Pub.Co, 2001
- [13] B.M. Wilamowski, J. Bin fet, M.O. Kaynac. VLSI Implementation of Neural Networks, Univ Report, Dec, 1998

CMOS VLSI

- [14] N. Chasta, S. Chouhan, Y. Kumar, Analog VLSI Implementation of Neural Network Architecture for Signal Processing, Int, J.of VLSI Design & Communication System, Vol 3 No 2, April, 2012, pp 243-249
- [15] C. Prasanna Raj, S.L. Pinjare, Design and Analog VLSI Implementation of Neural Network Architecture for Signal Processing, EuroJournals Publishing, Inc, Vol 27, No 2, pp199-216
- [16] L.D. Zadeh, Fuzzy sets, Information and Control, vol. 8, 1965, 338 – 353
- [17] M. Jamshidi, Fuzzy logic and Control, Prentice Hall, Englewood Cliffs, N.J. 07632, 1993.
- [18] R.R. Yager, D.P. Filev, Essentials of Fuzzy Modeling and Control, A Wiley – Interscience Publication, New york,1994.
- [19] J. Yan, M. Ryan, J. Power, Using Fuzzy Logic – Towards intelligent systems, Prentice Hall, New York, 1994.
- [20] S. Preitl, R.E. Precup, Introducere în conducerea fuzzy a proceselor, Ed. Tehnică, Bucureşti, 1997.
- [21] F. Van der Rhee, Knowledge Based Fuzzy Control System, IEEE Transactions an Automatic Control, Vol. 35, No. 2, February 1990.
- [22] J. Lee, On Methods for Improving Performance of PI – Type Fuzzy Logic Controllers, IEEE Transactions on Fuzzy System, Vol. 1, No. 4, November 1993.
- [23] R.K. Mudi, N.R. Pal, A Robust Self – Tuning Scheme for PI – and Pd – Type Fuzzy Controllers, IEEE Transactions on Fuzzy System, Vol. 7, No. 1, February 1993.
- [24] R. Fuller, Neural Fuzzy Systems, Donner Visiting professor Abo Akademi University, via Internet.
- [25] M.M. Gupta, R.K. Ragade, R.R. Yager, Advances in Fuzzy Set Theory and Applications, North – Holland, New York, 1979.
- [26] D. Dubois, Prade, Fuzzy Sets And Systems: Theory and Applications, Academic Press, New York, 1979.

11. Bibliografie

- [27] E.H. Mamdani, T.A. Folger,R.R. Gaines, Fuzzy Reasoning and Its Applications, Academic Press, London, 1981.
- [28] M. Sugeno, An introductory survey of fuzzy control, Inform. Sci., Vol. 36, 1985.