PROGRAMSKA PODPORA ZA NAČRTOVANJE VEZIJ V ULA TEHNOLOGIJI

Andrej Dobnikar, Veselko Gustin, Mira Trebar, Damjan Podbregar, Pavle Ilija, Pavle Stajdohar

KLJUČNE BESEDE: logično vezje, integrirano vezje, ULA tehnologija, računalniško načrtovanje vezij, osebni računalnik IBM, programski sistem, grupiranje, razmeščanje,povezovanje.

POVZETEK: v prispevku smo opisali programsko podporo za načrtovanje integriranih komponent v ULA tehnologiji. Pokazali smo pomembnost podpore za zmanjšanje časa načrtovanja in povečanje zanesljivosti izdelka. Natančneje smo opisali osnovne procedure sistema, kot so grupiranje, razmeščanje in povezovanje. Poudarili smo prednost sistema, možnost vračanja in ponavljanja programa z novimi vrednostmi parametrov, kakor tudi interaktivno delo s sistemom. Programski sistem smo napisali za osebne računalnike IBM in kompatibilne, kar bo nedvomno razširilo krog uporabnikov na raziskovalce v digitalnih laboratorijih pa tudi drugod.

PROGRAMMING SUPPORT FOR IC DESIGN IN ULA TECHNOLOGY

KEY WORDS: logic circuits, integrated circuits, ULA technology, circuits CAD, personnal computer IBM, programming tools, grouping, placement, routing

ABSTRACT: The Programming tool for IC in ULA technology is described. It is shown that it represents a great support by significant decreasing the time to develop IC element and by increasing the reliability of product. Basic produres of the system, namely grouping, placement and routing are detailed. Its important feature lies in possibilities for backtracking and interactive handling of the system. As it is writen for PC - IBM compatible users, we expect its wide usage by researshers in digital laboratories and by other PC users.

UVOD

Logična vezja ULA (Uncomitted Logic Array) so poseben primer polja vrat (Gate Array, GA) z v naprej določenim področjem za logične sklope in povezave med njimi. Potrebujemo le dodaten sloj metalizacije za dokončno obliko integriranega vezja (IC). S tem, ko nanesemo metalizacijo na osnovno univerzalno polprevodniško plast, smo določili logično celico in njene ustrezne povezave. Prav ta preprostost nas privlači pri vezjih v tehnologiji ULA, saj za načrtovanje potrebujemo manj časa pa tudi manj denarja. Računalniško načrtovanje brez dvoma ne pripomore samo k hitrejši poti do integriranega vezja, pač pa tudi k večji zanesljivosti proizvoda.

V prispevku nameravamo opisati programsko podporo, ki predstavlja pomembno orodje za načrtovanje vezij v ULA tehnologiji in podkrepiti slednje z nekaj primeri. Poglavje 1 vsebuje nekaj značilnosti tehnologije ULA. Opisali smo programski sistem,poudarili funkcionalne zmožnosti in prilagodljivost sistema. V poglavju 2 smo natančneje opisali posamezne module, nanizanih je tudi nekaj ilustrativnih primerov. V poglavje 3 smo uvrstili statistični prikaz zmogljivosti sistema, kjer smo uporabili različne možne parametre znotraj programskega sistema. V zaključku so navedeni nekateri tehnični podatki o uporabnosti programskega sistema.

V zaključku so navedeni nekateri tehnični podatki o uporabnosti programskega sistema.

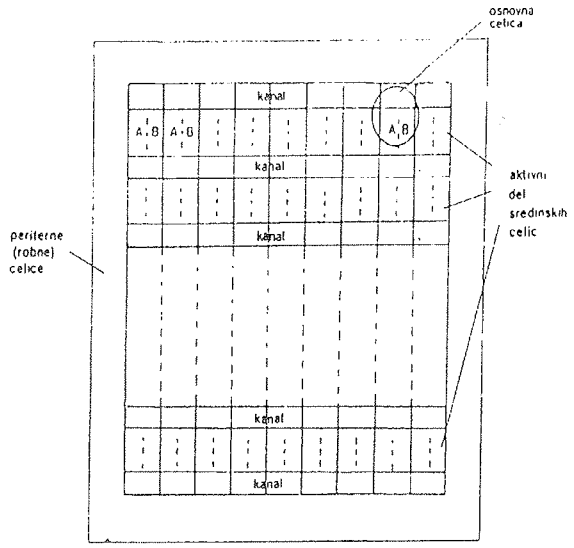
1. PREGLED PROGRAMSKEGA SISTEMA ZA NAČRTOVANJE INTEGRIRANIH VEZIJ V TEHNOLOGIJI ULA (PS ULA)

1.1 Tehnologija ULA

Slika 1 prikazuje arhitekturo integriranega vezja ULA. Vsaka osnovna celica je sestavljena iz 10 tranzistorjev, ki jih lahko povežemo v tri - (A) ali dvo-(B) vhodna vrata.

Integrirano vezje ULA je sestavljeno iz dveh delov. Prvi, obrobni del je namenjen perifernim celicam, le-te povezujejo notranjost vezja ULA z zunajim svetom. Drugi, sredinski del sestavljajo kanali in aktivni deli celic. Sredino obravnavamo kot polje osnovnih (globalnih) celic, kjer je vsaka celica sestavljena iz A,B parov. Vsakemu paru pripada ustrezni del aktivnega in kanalnega dela celice.

Na razpolago imamo knjižnico nekaterih logičnih vrat, kot na primer: AND, OR, NAND, NOR, NOT, EX_OR pa do različnih flip- flopov, kot na primer: D, RS, T, JK, ipd.



Slika 1: **Arhitektura IC ULA**

Za vsako logično celico dobimo vse potrebne natančne podatke o povezavah znotraj ene ali več osnovnih celic v knjižnici. Prav tako dobimo tudi podatke o različnih perifernih celicah, kot so vhodni/izhodni krmilnik, oscilator ipd. Vseh različnih celic v knjižnici, ki smo jo uporabljali, je okoli 150.

Poleg standardnih celic je možno v knjižnico vgraditi tudi makro celice, ki so sestavljene iz nabora osnovnih celic (ali pa tudi ne). V tem primeru makro obravnavamo kot novo celico.

1.2. Diagram PS ULA

PS ULA je sestavljen iz naslednjih glavnih komponent, ki jih lahko vidimo na sliki 2, le-te so:

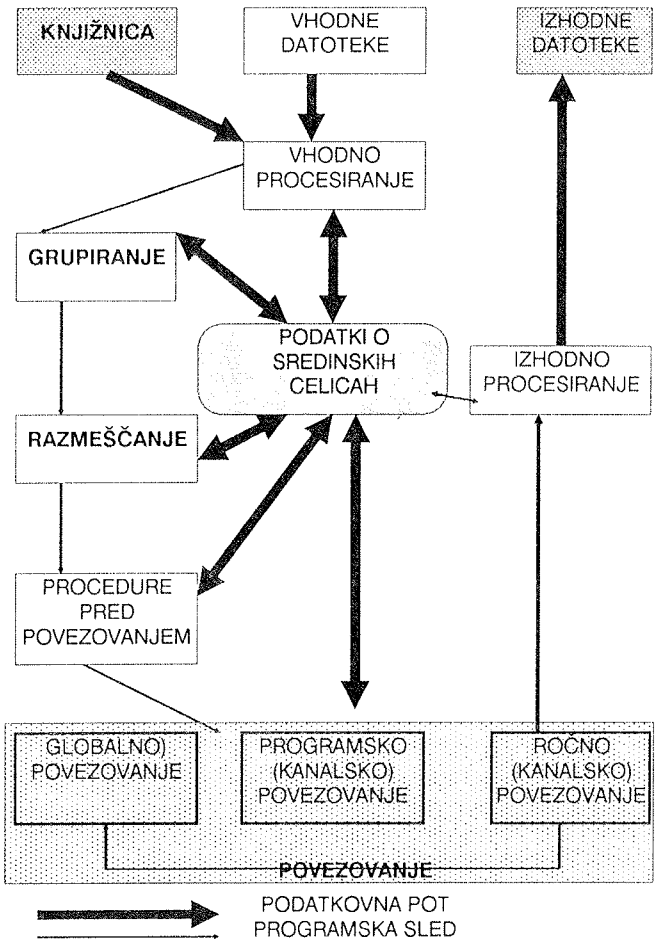
1. Vhodno procesiranje
2. Grupiranje
3. Avtomatsko razmeščanje
4. Predpovezovanje
5. Globalno povezovanje
6. Povezovanje v kanalu (avtomatsko in interaktivno)
7. Izhodno procesiranje

PS ULA smo zasnovali tako, da smo si za cilj izbrali blizu 100% povezavo signalov in to samo s pomočjo programskega povezovanja. Za nepovezane dele signalov smo predvideli interaktivno povezovanje s programskim paketom zunaj PS ULA.

Vhodne podatke za PS ULA predstavlja vhodna datoteka, ki jo lahko uporabimo tudi za simulacijo. Izhodišče za ULA vezje je lista povezav logičnih funkcij. Z modulom vhodnega procesiranja je formirana notranja baza podatkov, ki jo predstavlja lista povezav skupaj s knjižnico samo tistih logičnih celic, ki so uporabljene v listi povezav.

Logične celice združujemo v skupine, glede na medsebojne povezave. Makro celice lahko uporabimo kot kriterij za združevanje namesto, da jih damo v knjižnico. Podobno uporabimo tudi posebnosti, kot so stikanje, oz. natikanje ali dodatne zahteve operaterja.

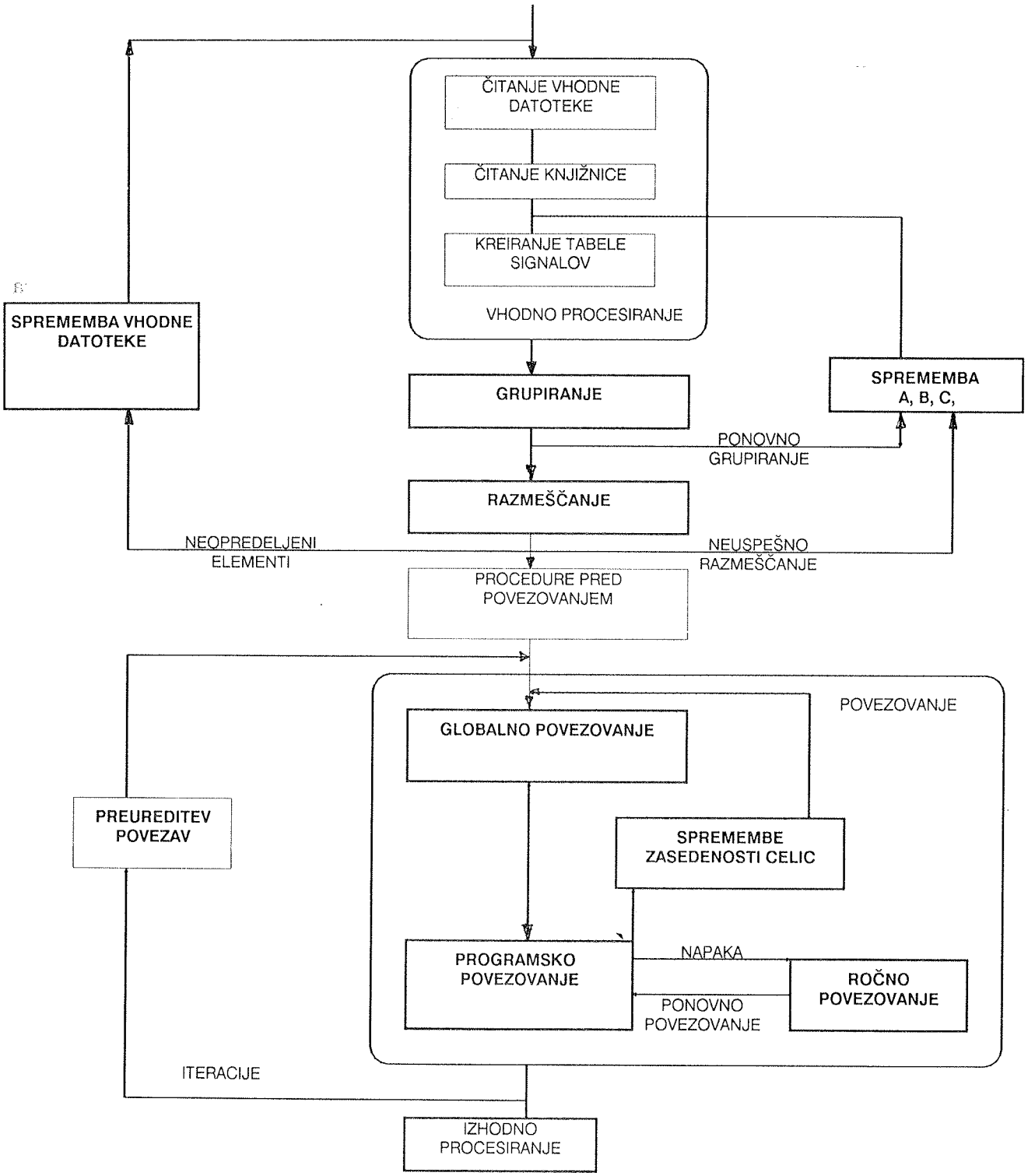
Razmeščanje v okviru PS ULA opravi polaganje grup v jedru glede na njihovo velikost, odvisno od vnaprej določenih robnih (perifernih) celic in medsebojne povezanosti. Vnovično razmeščanje celic znotraj posamezne skupine omogoča krajše povezave, tako v sami grupi (notranje povezave), kot tudi med grupami (zunanje povezave). Sledi procedura globalnega povezovanja in sicer priprava signalov za povezovanje. Signale lahko razvrstimo na več načinov, po najkrajši, po najdaljši ali po naključni dolžini povezav signala. Za vsako povezavo poiščemo najbližji vozlišči, ki ju skušamo povezati na optimalni način z uporabo Steinerjevega ali modificiranega Leejevega algoritma. Kriterij optimalne povezave pomeni enakomerno zasedenost osnovnih sredinskih celic. Po vsaki izračunani poti pokličemo program za povezovanje v kanalu, ki po uspešnem zaključku povezovanja, ustrezno zmanjša kapaciteto pravkar uporabljenega kanala ter označi sled povezave. Če neke povezave ne moremo realizirati, si pomagamo z novo globalno potjo ali z ročnim premeščanjem povezav.



Slika 2: **Arhitektura sitema PS ULA**

Z modulom izhodnega procesiranja pripravimo podatke o razmeščenih in povezanih elementih za izdelavo vezja ULA in za preizkušanje. V primeru, ko nismo povezali vseh signalov, tedaj izhodne podatke uporabimo s programom za grafični prikaz vezja ULA ter ročno povezovanje. Tako poskušamo doseči 100% povezanost signalov. Načelno lahko poženemo program brez vračanja nazaj, imamo pa možnost vnovičnega procesiranja, če z rezulatati nismo zadovoljni. To opravimo s testiranjem podatkov v različnih programskih točkah. Možnost vnovičnega procesiranja nam dopušča programsko spreminjanje nekaterih parametrov. Slika 3. nam prikazuje vse možne povratne programske poti in ustrezno spreminjanje parametrov.

V celotnem postopku načrtovanja vezij v tehnologiji ULA je potrebnih nekaj dodatnih programskih paketov. Slika 4 kaže mesto PS ULA in ostale programske module.

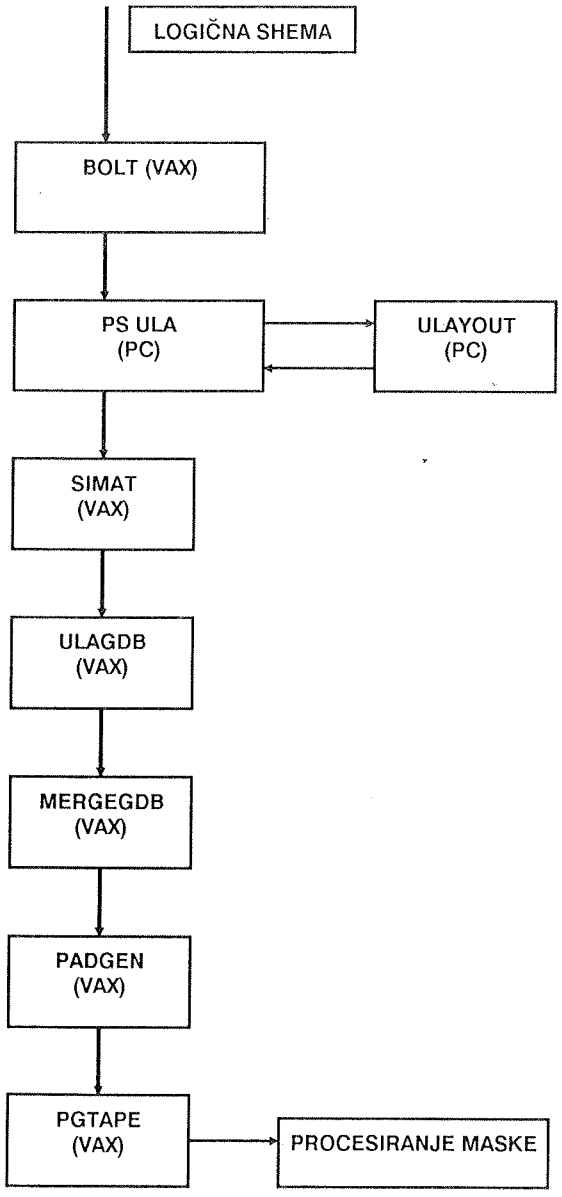


Slika 3: **Programske povratne zanke in parametri v PS ULA.**

Logično shemo opišemo s pomočjo programskega paketa BOLD, ki pripravi ustrezno listo povezav, vhodno datoteko za PS ULA.

ULAYOUT izvede vse tiste povezave, ki jih nismo uspeli povezati s PS ULA. To se odvija interaktivno, ročno z uporabo posebnih semigrafičnih procedur.

Sledi niz programskih paketov, ki tečejo na računalniku VAX 11/780. SIMAT opravi logično simulacijo izhodne datoteke PS ULA, ULAGDB metalno masko povezav, MERGEGDB doda preostale nivoje v strukturi ULA, medtem ko PADGEN in PGTAPE razdelita ULA vezje na pravokotnike ter pripravita trak za procesiranje maske.



Slika 4: Programska okolica PS ULA

2. OPIS PROGRAMSKIH MODULOV

2.1.Vhodno procesiranje

Poleg čitanja vhodne datoteke in knjižnice, izdelave interne knjižnice in tabele signalov, opravlja modul še testiranje vhodnih elementov, če so v knjižnici ali ne. Pri neopredeljenih elementih poznamo samo logično funkcijo, medtem ko strukturo določimo med procesiranjem. V primeru neopredeljenih imen elementov v vhodni datoteki znova kličemo modul za razmeščanje elementov in struktur, da se formirata knjižnica in vhodna datoteka. Slednje je potrebno pred postopkom povezovanja, ker je sicer interna knjižnica dvakrat večja pri neopredeljenih elementih, kot pa pri opredeljenih.

Pojavi se nam lahko še ena programska zanka v primeru, ko ne moremo razmestiti elementov na površino vezja ULA. V tem primeru poskušamo z novimi parametri v grupiranju, kar pomeni novo tabelo signalov. Če tudi po tretjem poskusu ne uspemo, tedaj je potreben poseg operaterja (večja ULA).

2.2. Grupiranje

Procedura grupiranje izvaja združevanje logičnih elementov iz tabele povezav (vhodne datoteke) v grupee. Namen grupiranja je pripraviti podatke programu za razmeščanje tako, da se združujejo tesno povezani elementi, povezovanje med njimi pa zasede kar se da malo kanalskega protora. Z nekaterimi parametri imamo možnost vplivati na potek izvajanja programa grupiranja. Parameter A ločuje načine, kako razbijamo grupe. Če je A=1, tedaj ostanejo skupni elementi v večji grupi, sicer (A=2) postanejo del manjše grupe. Prva možnost v splošnem pomeni večje grupe medtem ko nam druga možnost daje več grup z manj elementi. Parameter B določa največje število elementov v grupi. Najboljše rezultate smo dosegli, ko smo izbrali B okrog 3/4 vrednosti širine izbrane ULA celice. Parameter C omogoča dodatno združevanje podobnih grup ali grup, ki so povezane največ s C signali. S povečanjem parametra C onemogočimo dodatno grupiranje, kar se odraža v povečanju števila grup.

2.3. Razmeščanje

Procedura razmeščanja ima dve fazi. Prva faza nam izvede razmeščanje skupin, naslednja faza pa razmeščanje posameznih elementov znotraj skupine. Obsežnost skupine je odvisna od velikosti elementov v knjižnici, ki je del vhodnih podatkov. Samo sredinske celice uporabimo za izvedbo posamezne grupe. Položaj perifernih celic je definiran v fazi priprave vhodnih podatkov.

V programu za računalniško razmeščanje v glavnem uporabljamo metode začetnega razmeščanja, ki sta jih

podala N. Hanan in J. Kurtzberg v /10/. Metoda je zasnovana na združevanju grup elementov, ki ga določa število povezav med posameznimi grupami. Nameščanje začnemo s skupino, ki ima največ povezav z drugimi skupinami. Postavimo jo v sredino vezja ULA, k njej pa dodajamo grupe, ki so z njo najtesneje povezane.

Prvo fazo smo zaključili, ko smo programsko namestili vse skupine, nakar sledi druga faza razmeščanja znotraj grupe. Najprej razvrstimo elemente glede na število povezav, ki jih združujejo in upoštevamo natikanje elementov, kadar je to možno. To pomeni, da se elementi povezujejo iz izhoda na vhod le z dotikanjem (natikanjem). Vse ostale elemente razmeščamo optimalno glede na najmanjšo dolžino metalizacije v povezovalnem kanalu.

2.4 Postopki pred povezovanjem.

Modul aktiviramo po uspešni razmestitvi in pred postopkom povezovanja. Poskrbi za:

* dopolnitev tabele signalov z dejanskimi koordinatami elementov (iz modula razmeščanja),
* izračune globalnih koordinat vozlišč, ki se dajo povezati z natikanjem in ne uporabljajo kanala za povezovanje,
* razvrščanje povezav glede na naraščajoči x ali y,
* inicializacijo začetnih zmogljivosti kanalov, ki so zasedeni zaradi razmeščenih elementov v osredju,
* nekaj manjših poslov.

2.5. Globalno iskanje povezav

* Algoritmi za globalno povezovanje so največkrat definirani za dvo-dimenzionalni problem, torej za povezovanje v eni ravnini, ki jo obravnavamo kot pravokotno mrežo. Navedimo nekaj značilnosti kanalskega povezovanja pri ULA tehnologiji:
* elemente moramo povezovati s horizontalnimi in vertikalnimi deli, zato da zagotovimo enakomerno zasedenost v obeh smereh,
* povezovanje po polprevodniški plasti naj bo najmanjše,
* dolžina poti bodi minimalna.

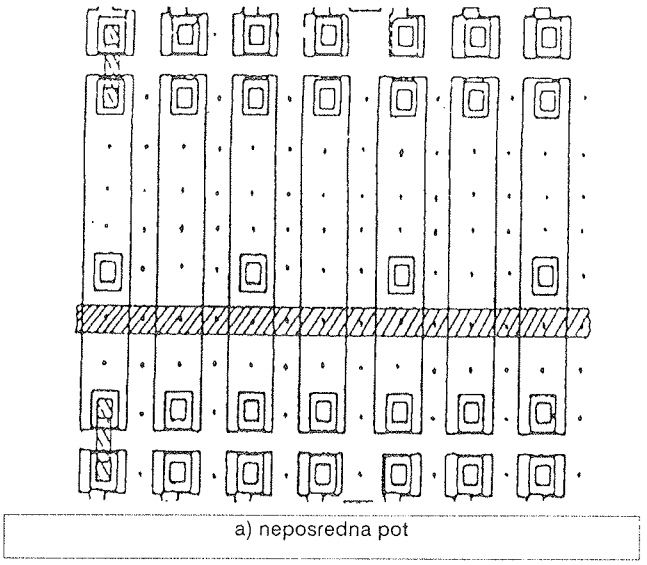
Da bi zadostili omenjenim zahtevam, smo uporabili dva algoritma: Steinerjevo drevo /1/ in modificiran Leejev algoritem /4/.

Steinerjevo drevo je zelo ustrezno za povezovanje dveh točk po enem od možnih robov ustrezajočega pravokotnika. Metoda nas hitro vodi k cilju in zahteva malo računalniškega časa.

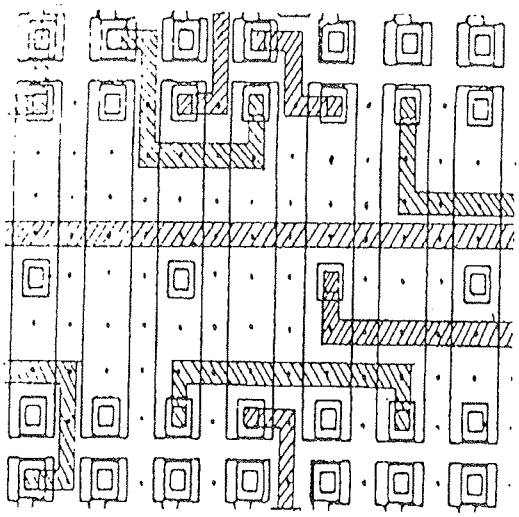
Leejev algoritem se uporablja v primeru, ko povezovanje dveh točk s Steinerjevo motodo ne gre. Z uporabo algoritma si vselej zagotovimo minimalno povezovalno pot.

2.6. Povezovanje v kanalu

Kanalsko povezovanje nam služi za iskanje povezav v kanalskem delu globalne celice. S programom upoštevamo podatke, ki mu jih o signalni poti da globalno povezovanje. Po uspešnem povezovanju določimo nove zasedenosti kanalov. Če poti ne najdemo neposredno skozi kanale, tedaj uporabimo Lee-Moorov algoritem /7/ za tiste globalne celice, kjer je to potrebno. Osnova strategije je v tem, da poskušamo vertikalne povezave peljati kar se da po polprevodniški plasti ter dodajamo horizontalne povezave v drugem nivoju. Le to nam omogoča križanje linij, stikanje in zavijanje poti (slika 5). Poleg teh povezav so možne tudi povezave v aktivnem delu globalnih celic. V takem primeru je potrebno upoštevati neuporabljene vertikale, saj ima vsaka logična celica svoje proste vertikale in horizontale znotraj aktivnega dela.

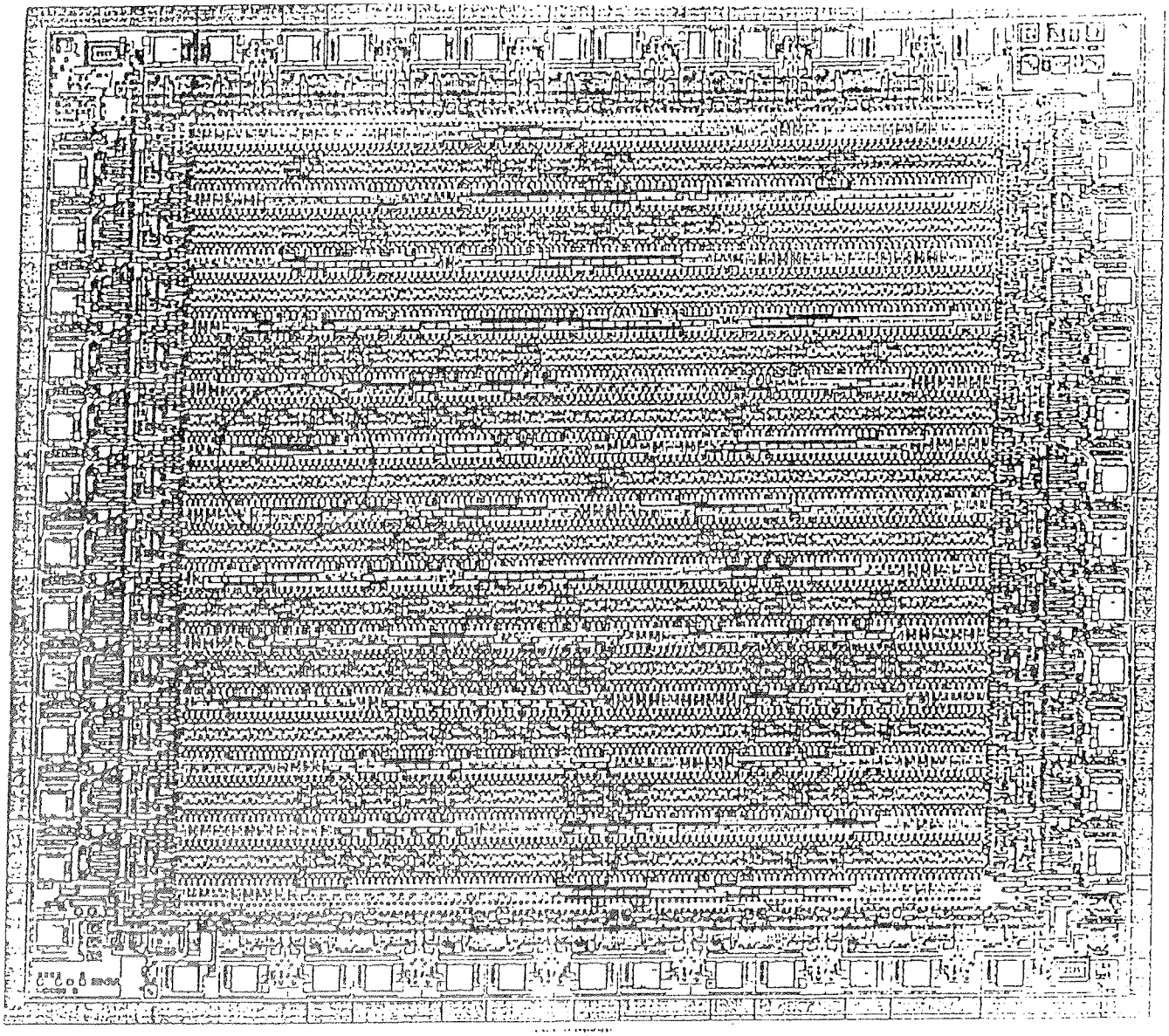


a) neposredna pot



b) izračunana pot

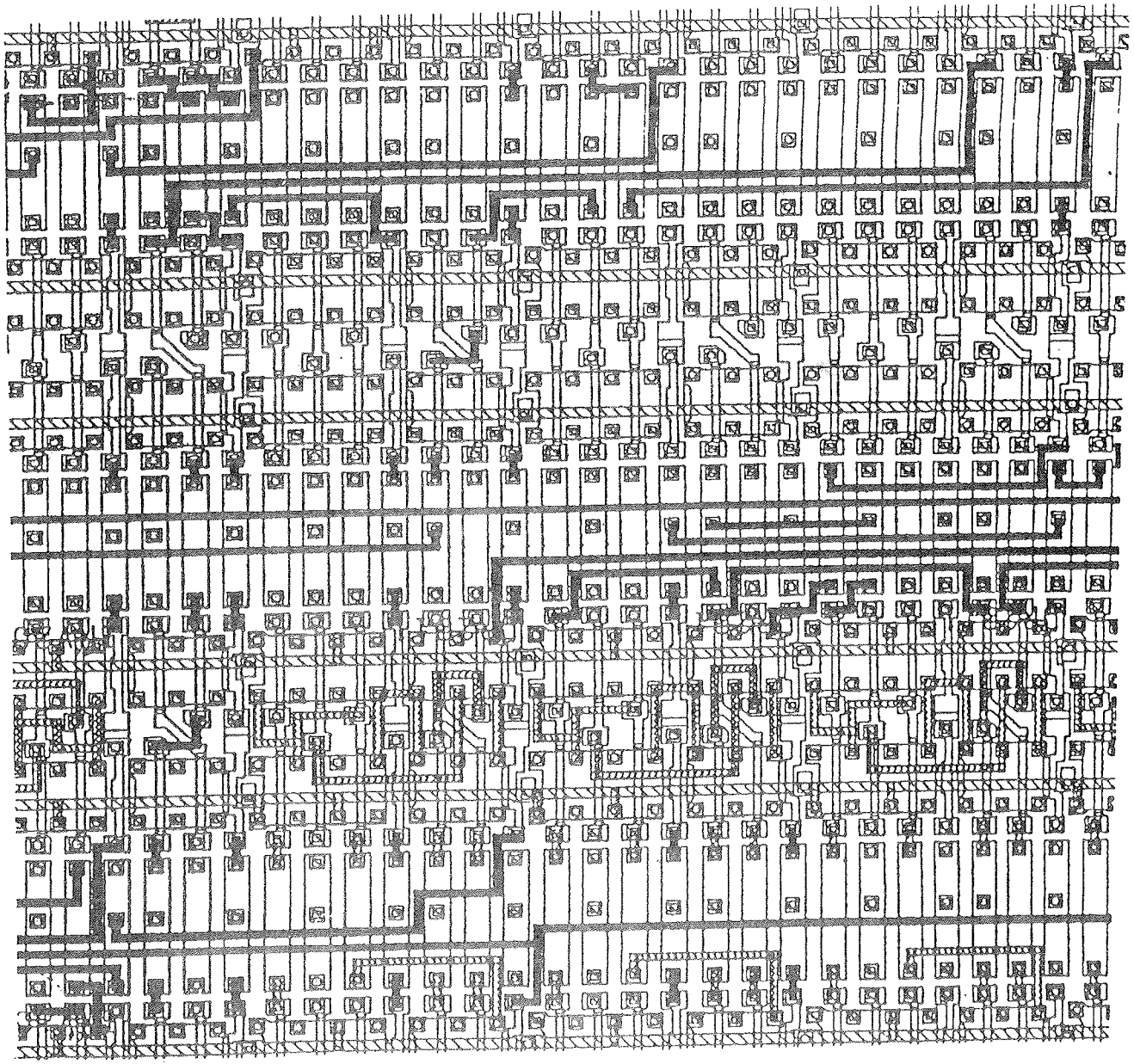
Slika 5: Primer določene poti skozi kanal:



Slika 6: a) Integrirano vezje Test 3

Tabela 1: Rezultati testiranja PS ULA.

primer	ULA tip	Št.signalov	Št. povezav	Št. elementov	Zased.	Nepov.
Test1	ULA 1	35	51	36	22%	0(0%)
Test2	ULA 3	187	293	129	92%	30(10%)
Test2	ULA 4	187	293	129	70%	14(5%)
Test3	ULA 2	133	216	108	82%	31(14%)
Test3	ULA 3	133	216	108	61%	36(12%)
Test4	ULA 3	231	347	216	93%	56(16%)
Test4	ULA 4	231	347	216	67%	37(11%)
Test4	ULA 5	231	347	216	51%	31 (9%)
Test4	ULA 6	231	347	216	40%	24(7%)



Slika 6: b) Obkroženi detajl za Test 3

2.7. Izhodno procesiranje

Izhodno procesiranje je zadnji modul, s katerim shranimo rezultate v takoimenovano datoteko DO. Le-ta vsebuje koordinate vseh razmeščenih elementov iz vhodne datoteke, (x,y) koordinate vozlišč izhodnih signalov in vmesnih povezav pripravljenih za metalizacijo.

3. STATISTIČNI REZULTATI

Tabela 1. prikazuje rezultate testiranja PS ULA na štirih različnih primerih. Ugotovili smo, da so rezultati v tolerančnih mejah, če zasedenost ne presega 70% celotne sredinske površine ULA. Število povezanih signalov se poveča, če vzamemo večje vezje ULA.

Primer izrisane integrirane komponente za Test3 vidimo na sliki 6.a in obkrožen detail na sliki 6.b:

Rezultati testiranja s tabele 1 so zadovoljivi za Test 1 in 3. V obeh primerih je nataknjenih signalov med 30 in 50%, kar pomeni manj zahtev po povezovanju v kanalu. Nasprotno vidimo pri Testu 4, kjer je samo 20% nataknjenih signalov, kar se pozna pri končnih 11% nepovezanih signalov. To vrednost lahko še vedno zmanjšamo, vendar na račun večjega vezja ULA.

ZAKLJUČEK

Opisali smo računalniško programsko podporo za načrtovanje integriranih komponent v tehnologiji ULA. Natančneje so opisani programski moduli razmeščanja in povezovanja v PS ULA. S primeri smo potrdili uporabnost programskega paketa za avtomatizirano načrtovanje integriranih komponent. PS ULA smo priredili za osebne računalnike IBM ali kompatibilne, kar bo nedvomno pripomoglo k široki uporabnosti programa, tako

med načrtovalci digitalnih vezij, kot drugimi koristniki programskega sistema.

PS ULA smo napisali v programskem jeziku Turbo Pascal 3.3 in teče pod operacijskim sistemom MS DOS 3.2. Zanj potrebujemo 1M zlogov pomnilnika (RAM), EGA grafični vmesnik in trdi disk z vsaj 1,5 M zlogov prostora.

LITERATURA

1. AD Friedman: Theory& Design of Switching Circuits, Computer Science Press, 1975.
2. A.A. Szepienice, R.H.S.M. Otten: The genealogical Approach to the Layout Problem, ACM IEEE 21th Design Automation Conference, June 1984, Albuquerque, New Mexico.
3. C.F. Shupe: Automatic Component Placement in an Interactive Minicomputer Environment, ACM IEEE 18th Design Automation Conference, 1981.
4. S.B. Akers: A Modification of Lee's Path Connection Algorithm, IEEE trans. on. Comp.,Feb.1967.
5. T. Yoshimura, E.S. Kuh: Efficient Algorithms for Channel Riuting, IEEE Trans. on Comp. Aided Design of IC Aids, Vol. CAD- 1, No. 1, Jan. 1982.
6. R.L. Rivest, CM. Fiduccia: A Greedy Channel Router, 19th Design Automation Conference, Las Vegas, 1982.
7. F. Rubin: The Lee Path Conection Algorithm, IEEE Trans. on Comp., Vol. C-23, No. 9, Sept. 1974.
8. L.I. Corrigan: A Placement Capability Based on Partitioning, ACM IEEE 16th Design Automation Conference, 1979.
9. E.P. Stabler, V.N. Kureichik: Placement Algorithm by Partitioning for Optimum Rectangular Placement, ACM IEEE 16 th Design Automation Conference, 1979.
10. M.Hanan, J.M. Kurtzberg: A Review of the Placement and Quadratic Assignment Problems, SIAM Review, vol. 14, No. 2, April, 1972.

Dr. Andrej Dobnikar, dipl. ing.
Dr. Veselko Gustin, dipl. ing
Mira Trebar, dipl. ing
Damjan Podbregar
Pavle Ilija
Univerza E. Kardelja, Ljubljana
Fakulteta za eletrotehniko
Tržaška 25
61000 Ljubljana
mag. Pavle Stajdohar, dipl. ing
Iskra Elektrooptika
Stegne 17
61000 Ljubljana

Prispelo: 25.01.1989 *Sprejeto: 26.02.1989*