

```
=====
= ADVANCED MICROPROCESSORS AND =
= HIGH-LEVEL LANGUAGE COMPUTER ARCHITECTURE =
=====
```

Knjiga z naslovom **ADVANCED MICROPROCESSORS AND HIGH-LEVEL LANGUAGE COMPUTER ARCHITECTURE** je zbirka najpomembnejših člankov, poročil in ekspertiz, ki jih je zbral in uredil avtor Veljko Milutinović za potrebe poučevanja predmeta z enakim naslovom na Purdue University v ZDA.

Knjiga je učbenik za študente ter hrati tudi dober pripomoček načrtovalcem novih računalniških sistemov in vodjem razvoja, ki potrebujetejo znanje o sodobnih trendih v računalniških arhitekturah. Knjiga sestoji iz 39 člankov, ki so razdeljeni v 7 delov in 14 poglavij ter ima skupno skoraj 600 strani.

Knjiga govori o arhitekturah, ki temeljijo na visokoprogramskih jezikih, to je o HLL (High Level Language) računalniških arhitekturah. Avtor deli HLL računalniške arhitekture na dve osnovni skupini:

- arhitekture z indirektnim izvajanjem ter
- arhitekture z direktnim izvajanjem.

Pri arhitekturah z indirektnim izvajanjem je za izvajanje programa potreben izvorni kod prevesti v izvajalni kod. Pri arhitekturah z direktnim izvajanjem pa računalnik lahko direktno izvaja izvorni kod.

Nadalje deli Milutinović arhitekture z indirektnim izvajanjem v dva podrazreda:

- reducirane arhitekture in
- kompleksne arhitekture.

Kompleksne arhitekture se nadalje delijo v

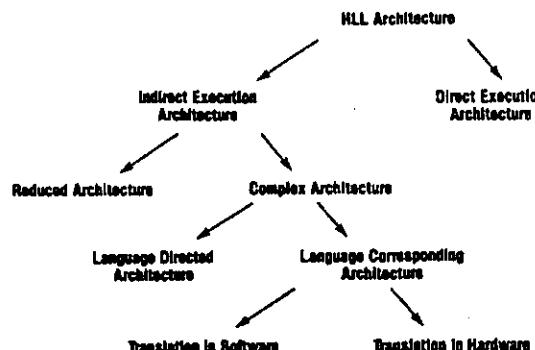
- jezikovno vodene arhitektura in
- jezikovno odvisne arhitekture.

Jezikovno vodene arhitekture lahko definiramo kot arhitekture, kjer se proti konstrukcije strojnega jezika na relativno visokem nivoju toda ne na nivoju HLL sintakse. Pri jezikovno odvisnih arhitekturah pa imamo direktna enotljivo odvisnost med HLL konstrukcijo in konstrukcijo strojnega jezika.

Na koncu so razdeljene jezikovno odvisne arhitekture še v:

- arhitekture s programskimi prevajalniki ter
- arhitekture s strojnimi prevajalniki.

Slika 1 kaže razdelitev HLL arhitektur.



Slika 1: Razdelitev HLL arhitektur.

Vtič o knjigi si najhitreje ustvarimo s pregledom kazala knjige, ki podaja poleg naslova vseh člankov tudi strukturo knjige in razdelitev vsebine v posamezne dele in poglavja.

#### Table of Contents

Preface.....	iii
Acknowledgements.....	v
<b>Part I -- Introduction</b>	
Chapter 1:Essential Issues.....	2
Directions and Issues in Architecture and Languages.....	3
M.J.Flynn(Computer,October 1980,pages 5-22)	
Compilers and Computer Architecture.....	20
W.A.Hulf(Computer,July 1981,pages 41-47)	
Requisites for Improved Architectures.....	27
G.J.Myers(Advances in Computer Architecture, 1982,pages 58-100)	
Retrospective on High-Level Language Computer Architecture.....	63
D.R.Ditzel and D.A.Patterson(Proceedings of the 7th International Conference on Computer Architecture,May 1980,pages 97-104)	
Chapter 2:Impacts of the VLSI Technology.....	71
VLSI Processor Architecture.....	73
J.L.Hennessy(IEEE Transactions on Computers, December 1984,pages 1221-1246)	
An Introduction to VLSI Microprocessor Architecture for GaAs.....	97
V.Milutinovic,D.Fura, and W.Helbig	
Chapter 3:A Survey of Advanced Microprocessors and High-Level Language Architecture.....	117
A Survey of Advanced Microprocessors and High-Level Language Computer Architecture.....	118
A.Silbey,V.Milutinovic, and V.Mendoza-Grado	
<b>Part II -- Reduced Architectures</b>	
Chapter 4:The RISC Approach.....	144
A VLSI RISC.....	145
D.A.Patterson and C.H.Sequin(Computer,September 1982,pages 8-21)	
Architecture of a VLSI Instruction Cache for a RISC.....	158
D.A.Patterson,P.Garrison,M.Hill,D.Liboupolis, C.Nyberg,T.Sippel, and K.Van Dyke(Proceedings of the 10th Conference on Computer Architecture June 1983,pages 108-116)	
Strategies for Managing the Register File in RISC.....	167
Y.Tamir and C.H.Sequin(IEEE Transactions on Computers,November 1983,pages 977-989)	
Architecture of SOAR:Smalltalk on a RISC....	180
D.Ungar,R.Blaau,P.Foley,D.Samples, and D.Patterson(Proceedings of the 11th International Conference on Computer Architecture,June 1984,pages 188-197)	
Chapter 5:The MIPS Approach.....	191
Hardware/Software Tradeoffs for Increased Performance.....	192
J.Hennessy,N.Jouppi,F.Baskett,T.Gross, and J.Gill(Proceedings of the ACM Symposium on Architectural Support for Programming Languages and Operating Systems, March 1982,pages 2-11)	

<b>Organization and VLSI Implementation of MIPS...</b>	202	<b>Part IV -- HLL Architectures -- Type A</b>
S.A.Przybylski,T.R.Gross,J.L.Hennessey,N.P.Jouppi, and C.Rowen(Stanford University Technical Report No.84-259,April 1984).		<b>Chapter 9:Some Early Experiments.....</b> 460
<b>Floating-Point Arithmetic on a Reduced-Instruction-Set Processor.....</b> 241		<b>Implementation of a High-Level Language Machine.....</b> 461
T.Gross(Proceedings of the 7th Symposium on Computer Arithmetic,June 1985,pages 86-92)		A.Hassitt,J.W.Lageschulte, and L.E.Lyon(Communications of the ACM,April 1973,pages 199-212)
<b>Postpass Code Optimization of Pipeline Constraints.....</b> 248		<b>Design of an Aerospace Computer for Direct HOL Execution.....</b> 475
J.L.Hennessey and T.R.Gross(ACM Transactions on Programming Languages and Systems,July 1983 pages 422-448)		W.C.Nielsen(Proceedings of the Symposium on High-Level Language Computer Architecture, June 1973,pages 34-42)
<b>Chapter 6:Miscellaneous Reduced-Instruction-Set Processors.....</b> 275		<b>Chapter 10:Some Current Research.....</b> 485
<b>The 801 Minicomputer.....</b> 276		<b>Scheme-79 --Lisp on a Chip.....</b> 486
G.Radin(IBM Journal of Research and Development,May 1983,pages 273-246)		G.J.Sussman,J.Holloway,G.L.Stein,Jr.,and A.Bell(Computer,July 1981,pages 10-21)
<b>Ridge 32 Architecture - A RISC Variation...<b>286</b></b>		<b>The Scheme-81 Architecture--System and Chip by the Designers.....</b> 497
E.Basart and D.Folger(Proceedings of ICCD'83, October 1983,pages 315-318)		J.Batali,E.Goodhue,C.Hanson,H.Shrobe,R.M.Stallman, and G.J.Sussman(Proceedings of the 1982 MIT Conference on Advanced Research in VLSI, January 1981,pages 69-77)
<b>Reduced-Instruction-Set Multi-Microcomputer System.....</b> 290		 
L.Foti,D.English,R.P.Hopkins,D.J.Kinniment, P.C.Traileaven, and H.L.Wang(Proceedings of the NCC,July 1984,pages 69-75)		<b>Part V -- HLL Architectures--Type B</b>
<b>Applying RISC Theory to a Large Computer...<b>297</b></b>		<b>Chapter 11:Two Interesting Experiments.....</b> 508
R.Ragan-Kelley and R.Clark(Computer Design, November 1983)		<b>Reflections on the High-Level Language Symbol Computer System.....</b> 509
 		D.R.Ditzel(Computer,July 1981,pages 55-66)
<b>Part III -- Language-Directed Architectures</b>		<b>High-Level Language Oriented Hardware and the Post-von Neumann Era.....</b> 521
 		H.J.Burkle,A.Frick, and C.Schlier(Proceedings of the 5th Annual Symposium on Computer Architecture,June 1978,pages 60-65)
<b>Chapter 7:Stack Machines.....</b> 304		 
<b>Stack Computers:An Introduction.....</b> 305		<b>Chapter 12:The DEL Approach.....</b> 527
D.M.Bulman(Computer,May 1977,pages 18-28)		<b>Ideal Directly Executed Languages:An Analytical Argument for Emulation.....</b> 528
<b>Exploring a Stack Architecture.....</b> 315		L.W.Hoewel(IEEE Transactions on Computers, August 1974,pages 759-767)
R.P.Blake(Computer,May 1977,pages 30-39)		<b>Execution Architecture:The DELtran Experiment..</b> 537
<b>Twenty Years of Burroughs High-Level Language Machines.....</b> 325		M.J.Flynn and L.W.Hoewel(IEEE Transactions on Computers,February 1983,pages 156-174)
E.D.Earnest(Proceedings of the International Workshop on High-Level Language Computer Architecture,June 1980,pages 64-71)		 
<b>Implications of Structured Programming for Machine Architecture.....</b> 333		<b>Part VI -- Direct Execution Architectures</b>
A.S.Tanenbaum(Communications of the ACM,March 1978,pages 237-246)		 
 		<b>Chapter 13:The University of Maryland Approach.</b> 558
<b>Chapter 8:Advanced Complex-Instruction-Set Microprocessors.....</b> 343		 
<b>An Architectural Comparison of 32-Bit Microprocessors.....</b> 344		<b>Interactive High-Level Language Direct-Execution on Microprocessor System.....</b> 559
A.Gupta and H.D.Toong(IEEE Micro,February 1983 pages 9-22)		Y.Chu and E.R.Cannon(IEEE Transactions on Software Engineering,June 1976,pages 126-134)
<b>Introduction to the iAPX 432 Architecture...<b>358</b></b>		<b>Programming Languages and Direct-Execution Computer Architecture.....</b> 568
Intel Corporation(Intel Corporation Manual Order Number 171821-001,1981)		Y.Chu and M.Abrams(Computer,July 1981,pages 22-32)
<b>A 32-Bit VLSI CPU Chip.....</b> 422		 
J.W.Beyers,L.J.Dohse,J.P.Fucetola,R.L.Kochis, C.G.Lob,G.L.Taylor, and E.R.Zeller(IEEE Journal of Solid-State Circuits,October 1981,pages 210-214)		<b>Part VII -- International efforts</b>
<b>The Motorola MC68020.....</b> 429		 
D.MacGregor,D.Mothersole, and B.Moyer(IEEE MICRO, August 1984,pages 101-118)		<b>Chapter 14:International Efforts.....</b> 580
<b>System Considerations in the NS32032 Design...<b>447</b></b>		 
R.Mateosian(Proceedings of the NCC,July 1984, pages 77-81)		<b>A Survey of High-Level Language Machines in Japan.....</b> 581
<b>An Inside Look at the Z80,000 CPU:Zilog's New 32-Bit Microprocessor.....</b> 451		M.Yamamoto(Computer,July 1981,pages 68-77)
A.Patel(Proceedings of the NCC,July 1984,pages 83-91)		<b>Selected European Contributions in the Area of High-Level Language Computer Architecture...<b>590</b></b>
		A.Silbey and V.Milutinovic

V prvem poglavju knjige so 4 uvodni članki svetovno poznanih avtorjev (Flynn, Wulf, Myers,

Ditzel), ki obravnavajo odnos med programskim jezikom in arhitekturo računalnika. Hkrati predstavljajo izbrani članki pregled področja ter omogočajo bralcu, da na podlagi bibliografije razširi svoje znanje v posameznih specialnostih.

Drugo poglavje vsebuje 2 članka o vlogi VLSI tehnologije na računalniške arhitekture. Prvi članek (Hennessy) obravnava VLSI tehnologijo, ki temelji na klasični silicijevi tehnologiji, drugi članek (Milutinović) pa obravnava novo in hitro prodirajočo tehnologijo verzij na osnovi galijevega arzenida GaAs. Pristop pri načrtovanju računalniške arhitekture v mnogodem zavisi od izbrane tehnologije. Tako silicijeva kot tudi GaAs tehnologija imata vsaka svoje značilnosti, ki postavljajo specifične zahteve pri načrtovanju optimalne radikalniške arhitekture. Poglavlje obravnava reducirane arhitekture in sicer RISC računalnik iz Berkeley, MIPS iz Stanforda in ostale.

Prvi članek drugega poglavja, katerega avtor je Hennessy, primerja dva osnovna pristopa v arhitekturi, to je arhitekturo, ki zajema kompleksen nabor ukazov in arhitekturo z reduciranim naborom ukazov, kjer je obakrat procesor realiziran v VLSI tehnologiji na osnovi silicija. Članek analizira učinkovitost izvajanja predenega koda visokoprogramskega jezika.

Drugi članek je napisal Milutinovič in predstavlja GaAs kot možno osnovo za implementacijo VLSI procesorjev. Pri tem pa nam nova tehnologija narekuje novo arhitekturo vezij in režime, ki so bile razvite za silicijevlo okolje na splošno niso uporabne pri vezjih, ki temeljijo na GaAs. Kaze, da je pri vezjih, ki temeljijo na GaAs učinkovita samo arhitektura z reduciranim naborom ukazov.

Poglavja 3, 4 in 5 so posvečena kompleksnim arhitekturam, to je jezikovna odvisnim in jezikovno vodenim arhitekturam. V tretjem poglavju je narejen pregled različnih pristopov pri zasnovi in implementaciji novih mikroprocesorjev in HLL arhitektur ter analiza performans različnih sistemov.

Cetrtog poglavje je posveđeno RISC (Reduced Instruction Set Computer) arhitekturam. Ukvareja se z izborom reduciranega nabora ukazov in nekaterimi specifičnimi implementacijami, ki temeljijo na reduciranem naboru ukazov. Posebej je v prvem članku (Petterson) obdelan projekt University of California at Berkeley UCB-RISC, ki omogoča hitrejše izvajanje programa napisanega v visokoprogramskejem jeziku. Bitstvo rešitve problema je v metodi kako zasnovači arhitekturo, da bo čim bližje tistim konstrukcijam strojnega jezika, ki se pri visokoprogramskih jezikih najpogosteje uporabljajo. Nekrat naj bo arhitektura računalnika zasnovana tako, da bo zmanjšena potreba po komunikaciji z ostalimi vezji. Tako ima RISC I arhitekturi le 31 ukazov od katerih večina opravlja le preproste ALU in pomicne (shift) operacije na registrih.

Drugi članek v tem poglavju (Petterson) obravnava implementacijo VLSI verzja za cache ukaze za UCB-RISC računalnik. Ta VLSI verzija je neobčutljivo na napake (fault tolerant), vsebuje poseben programski stevec, izvaja komprimiranje kode in omogoča razširljivost. Kondni cilj skupine v Berkeleyu je združiti ukazni cache pomnilnik ter podatkovni cache pomnilnik s centralno procesorsko enoto v eno vezjo (chip).

Naslednji članek v četrtem poglavju (Tamic) se ukvarja z različnimi strategijami in metodami, ki so povezane z vodenjem večokenskih register-skih datotek v računalnikih kot je na primer UCB-RISC. Pri visokoprogramskih jezikih je

klicanje procedure časovno najbolj zamudna operacija. V splošnem imajo lahko RISC programi še ved klicev procedur kot običajni računalniki saj so kompleksni ukazi, ki jih najdemo v CISC arhitekturah realizirani kot podprogrami v RISC arhitekturi. Zato morajo biti klici procedur pri RISC računalnikih kar se da hitri. To je realizirano z registerskimi okni. Večokenska registerska datoteka je eden od tistih pristopov, ki bistveno pripomore k zmanjšanju potrebu po komunikaciji z drugimi vezji. Ideja je v tem, da v verzje vgradimo dolodeno število delno prekrivajočih se registerskih oken, pri čemer je ob vsakem trenutku dostopno prevajalniku ali programerju v zbirnem jeziku samo posamezno okno. Ostala okna vsebujejo spremenljivke za ostale procese. Eden od problemov pri tem pristopu je kako ravnati, če pride pri registerski datoteki do presekka (overflow). Avtorja članka ugotavljata, da je optimalno število registrrov, ki shranjujejo presek ūnako na.

Zadnji članek v četrtem poglavju (Ungar) opisuje zasnovno in implementacijo mikroprocesorja, ki je namenjen za jezik Smalltalk. Ta procesor temelji na reducirani naboru ukazov, a se bistveno razlikuje od UCB-RISC procesorjev. To nam potrjuje domnevo, da se vsaka reducirana arhitektura navezuje na dolodeno aplikacijo zato različno aplikativno okolje vodi do različnih konkretnih rešitev RISC procesorja.

Peto poglavje vsebuje štiri članke in je posvečeno MIPS pristopu. MIPS je okrajšava za Microprocessor without Interlocked Pipeline Storage. Ta koncept so razvili na Stanford University. MIPS pristop je primeren za verzijo, ki temeljijo na tehnologiji silicija, kot tudi za verzijo ki temeljijo na tehnologiji GaAs. Če primerjamo MIPS arhitekturo z arhitekturo UC8-RISC vidimo, da je za MIPS arhitekturo izabilna nizka kompleksnost VLSI vezij, zahteva pa kompleksnejši nadir programiranja in prevajanja. Eno osnovnih prednosti MIPS arhitekture je dejstvo, da je cevna sinhronizacija izvedena s programskim pristopom in ne vedno v materialni računalniški opremi. To je ugodno predvsem zaradi lažje implementacije VLSI vezij in tudi hitrosti procesorja. Seveda pa mora prevajalnik sedaj generirati kod, ki je brez cevnih konfliktov. Generator kode zadosti tem zahtevam tako, da vstavi ukaze NO-OP povsed tam, kjer je to potrebno. Optimizator koda poskuša nato zamenjati čim več ukazov NO-OP s kodo od kjerkoli druge pod pogojem, da na ta način ne vpliva na odvisnost podatkov. Tu vidimo potrebo po kompleksnejši tehnologiji prevajalnikov.

## Instruction 1

Fetch	Decode	Operand Decode	Store or Executes	Operand Fetch

Instructions i + 1	Fetch	Decode	Operand Decode	Store or Execute
--------------------	-------	--------	----------------	------------------

### (a) Plastic Structure

<b>A := B + C</b>	<b>Load B, R1</b>	<b>Load B, R1</b>
	<b>Load C, R2</b>	<b>Load C, R2</b>
	<b>Add R1, R2</b>	<b>NO-OP</b>
	<b>Store R2, A</b>	<b>Add R1, R2</b>
		<b>Store R2, A</b>

(b) HLL Statement    (c) Output of the Code Generator    (d) Output of the Code Register

Slika 2.1 Primer snemanja z MIPS arhitekturo, ki kaže izhod generatorja koda in reorganizatorja koda.

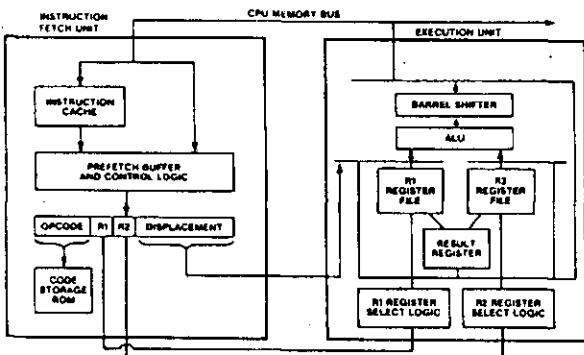
Naslednji članek (Przybylski) je tehnično poročilo Stanford University, ki opisuje VLSI implementacijo MIPS. Ta članek obravnava področje kot so nabor ukazov, cevna organizacija, ukrepanje pri izjemah, podpora za virtualni pomnilnik, podpora operacijskega sistema in visokoprogramskega jezika itd. Članek vsebuje tudi podatke o merjenju performans sistema.

Naslednji članek (Gross) se osredotoča na probleme, ki so povezani s programsko implementacijo aritmetike s plavajočo vejico za MIPS mikroprocesorje. Pri izbiri reduciranega nabora ukazov se pojavlja vprašanje, kako je tak reducirani nabor primeren za aritmetiko plavajoče vejice in za aritmetiko na sploš. Nabor ukazov za MIPS arhitekturo vsebuje ukaz, ki ustreza množenju po Booth algoritmu, kar je dalo relativno dobre performance.

Zadnji članek v petem poglavju (Hennessey) analizira številne činitelje, ki so povezani z optimizacijo koda v MIPS okolju. Članek definira tip cevnih konfliktov do katerih lahko pride v MIPS mikroprocesorju. Govori o razlikah pri optimizaciji koda, ki se izvaja po alokaciji registerov ali sočasno z alokacijo. MIPS optimizator koda uporablja prvi pristop. Podana je teorija MIPS kodnega optimizatorja in podatki o oceni performans.

6. poglavje je posvečeno arhitekturam za direktno izvajanje. Vsebuje štiri članke, ki obravnavajo najzanimivejše procesorce z reduciranim naborom ukazov. Trije taki procesorji izhajajo iz industrije, to so: IBM (IBM 801), Ridge (Ridge-32) in Pyramid (Pyramid 90X). Tretji procesor pa je bil razvit na University of Reading v Angliji. Angleški procesor je narejen v VLSI tehnologiji (RIMMS) ostali trije pa temeljijo na hitri SSI/MSI tehnologiji. Intenzivne raziskave in razvoj procesorjev z reduciranim naborom ukazov tečajo tako v industrijskem okolju (Fairchild, Hewlett-Packard, RCA, TRW, Inmos) kot tudi na univerzah (Caltech, Purdue, ULCA, Wisconsin in druge).

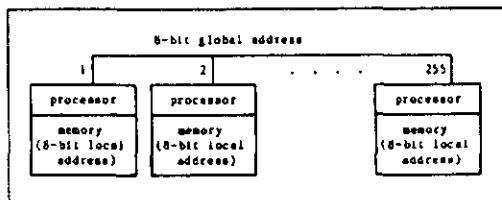
Nekateri tovrstni projekti so se pričeli že sredi sedemdesetih let. Tak je na primer projekt za miniračunalnik IBM 801. Članek, katerega avtor je Radin, govori o tem projektu in podaja nekatere značilnosti samega procesorja in tudi sistema kot celote. Opisuje na primer nabor ukazov, ki je realiziran popolnoma v materialni računalniški opremi, hierarhičnost pomnilnika, organizacijo V/I komunikacije, ki omogoča centralnemu procesorju izvajanje ukaza v skoraj vsakem ciklu in drugo. Projekt je bil poleg tega usmerjen k realizaciji sposobnejših prevajalnikov.



Slika 3.1: Zgradba Ridge-32 procesorja.

Basart in Folger v svojem članku predstavljata zasnovno in implementacijo superminiračunalnika Ridge-32, ki je verjetno prvi komercialno dobljiv RISC računalnik. Članek se ukvarja z osnovnimi cilji arhitekture, z izborom nabora ukazov, z implementacijo sistema in oceno performans. Ta računalnik je narejen za hitro izvajanje zahtevnih grafičnih aplikacij. Slika 3 kaže strukturo Ridge-32 procesorja, ki je implementiran s komercialno Schottky bipolarno logiko. Procesor ima ločeno dostavno (fetch) enoto in izvajano enoto. Dostavna enota je narejena za hitro izvajanje ukazov, ki se bo izvedel. Dostavna enota in izvajalna enota tvorita štiristopenjsko cevno arhitekturo, ki omogoča prekrivanje izvajanja ukazov. Ridge-32 vsebuje tudi zaleden pomnilnik (cache), kjer se hranijo vsi pred kratkim izvedeni ukazi.

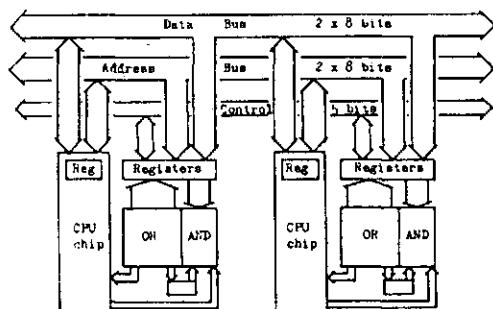
Foti sodelavci v četrtem članku šestega poglavja opisuje projekt RIMMS mikračunalnika (Reduced-Instruction set Multi-Microprocessor System project). RIMMS sestoji iz linearnega polja 255 mikračunalnikov, ki komunicirajo preko skupnega vodila kot kaže slika 4.



Slika 4.1: Zgradba RIMMS mikračunalnika z 255 procesorji.

Vsek mikračunalnik ima preprost procesor in 256 zlogov lokalnega pomnilnika.

Ta mikroprocesor ima skrajno reducirani nabor ukazov in je namenjen uporabi v multimikroprocesorskem okolju. Zato so posebno pozornost posvetili strojni računalniški podpori za medprocesorsko komunikacijo. Vodilo sestoji iz 16 bitnega naslova, 16 bitnega podatka in dodatnih bitov za delo s pomnilnikom, kar omogoča izvajanje NO-OP posega po pomnilniku. Slika 5 kaže organizacijo in kontrolo vodila ter pomnilnika za RIMMS mikračunalnik.



Slika 5.1: Nadzor vodila in pomnilnika za RIMMS mikračunalnik.

Med najzanimivejše lastnosti tega mikroprocesorja sodijo metode, ki jih uporabljajo za podporo programskih konstrukcij kot so FORK, JOIN, REMOTE LOAD in REMOTE STORE. Razvili so jezik za parallelno procesiranje imenovan BASAL, ki podpira osnovno idejo mikračunalnika z

reduciranim naborom ukazov za multimikroprocesorsko okolje.

Ragan-Kelley opisuje razvoj superminiračunalnika pri Pyramid Technology Corporation, ki prav tako temelji na reducirarem naboru ukazov. Cilj projekta je sposoben računalnik, ki podpira UNIX operacijski sistem in visokoprogramski jezike kot so C in Pascal. Računalnik je namenjen večuporabiškem okolju. Na razvoj tega projekta je verjetno vplival projekt RISC iz Berkeleyja, kljub temu pa vsebuje ta projekt številne nove zamisli.

Sedmo poglavje se ukvarja s skladovnimi (stack) računalniki. Sklad omogoča učinkovito organizacijo preklapljanja konteksta kot tudi aritmetiko. S skladovno arhitekturo se posebej ukvarjajo Burroughs, Hewlett-Packard, Microdata, Intel, Xerox in Zilog. Prvi članek v tem poglavju opisuje osnovno idejo skladovne arhitekture, še posebej povzročanje in kontrolo podprogramov ter evaluacijo izrazov. Podana je primerjava skladovnih računalnikov in tradicionalnih računalnikov, ki temeljijo na splošnih registrih.

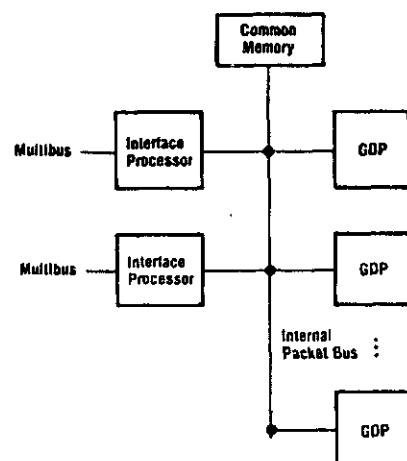
Blake je avtor članka, ki je uvod v optimalno zasnovo skladovnega računalnika in podaja pristop firme Hewlett-Packard. Članek analizira potrebe in značilnosti programov pisanih v visokoprogramskih jezikih in njihov vpliv na sklad. Posebna pozornost je posvečena strojni opremi skladovnih računalnikov, naboru ukazov, naslovnem prostoru za procese, spremljjanju procesov itd.

Earnest v tretjem članku sedmoga poglavja obravnava ved skladovnih računalnikov, ki so jih razvili pri Burroughs v zadnjih 20 letih. Eden glavnih razlogov, da se nekateri proizvajalci osredotočajo na skladovne računalnike je dejstvo, da je relativno lahko pisati prevajalnike za skladovne računalnike in, da je ta koda ponavadi precej kompaktna.

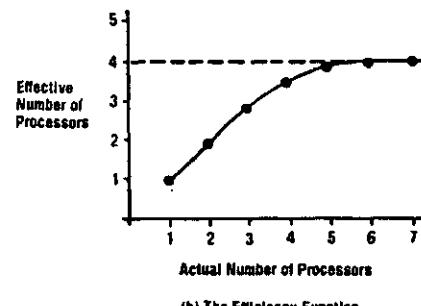
Osmo poglavje se ukvarja s sodobnimi mikroprocesorji za kompleksen nabor ukazov, ki jih pogosto imenujemo CISC (Complex-Instruction-Sets Computers). Poglavje vsebuje 8 člankov, ki govorijo o 32 bitnih procesorjih, ki so namejeni delu z visokoprogramskimi jeziki. Imajo bogat nabor ukazov, od katerih nekateri zelo spominjajo na konstrukte, ki so značilni za moderne visokoprogramski jeziki. Raziskave in razvoj na tem področju hitro napredujejo in novi ter populnejši 32 bitni mikroprocesorji se stalno pojavljajo v ZDA, na Japonskem in v Evropi. Uveljavljeno je prepričanje, da so CISC računalniki bolj primerni za podporo programske zanesljivosti kot pa RISC računalniki. Prav tako imajo močnejše sposobnosti obdelave aritmetičnih izrazov. Sveda pa so CISC računalniki v primerjavi z RISC računalniki podcenjenji pri izvajanju prevedenega programa, ki je pisani v visokoprogramskem jeziku. Ker so CISC procesorji kompleksnejši, zahtevajo daljši čas za razvoj procesorja. Dajstvo pa je, da so novi CISC procesorji zelo sposobni, uporabljajo visoko stopnjo cevnega parallelizma in vsebujejo že na čipu cache pomnilnik ter mehanizme za delo s pomnilnikom. Konečna odločitev in tekma med RISC in CISC mikroprocesorji je še pred nami in ni jasno kdo bo zmagoval.

Gupta v svojem članku primerja 4 zgodnje 32 bitne mikroprocesorje. Tehnično poročilo firme Intel podaja bistvene značilnosti procesorja iAPX 432. Dobra lastnost procesorja iAPX 432 je razširjivost sistema. Vedno sposobnost sistema dosegamo brez spremnjenja programske opreme, tako, da povečamo število procesorjev v sistemu na največ 5 GPD (General Data Processors). Pet takih procesorjev si deli isto vodilo in imajo zato performance, ki so enake

vesoti treh nedvignih GPD. Razširitev iAPX v multiprocesorski sistem in performance iAPX 432 multiprocesorskoga sistema kaže slika 6.



(a) An IAPX 432 Multiprocessor System



(b) The Efficiency Function

Slika 6.: iAPX 432 multiprocesorski sistem in njegove performance kot funkcija števila procesorjev.

Beyers v svojem članku podaja notranje organizacijo procesorja HP-FOCUS. McGregor sodeloval obravnava bistvene značilnosti Motorolla nega procesorja MC68020. Mateosian opisuje glavne značilnosti procesorja NS32032 proizvajalca National Semiconductor. Na koncu poglavja pa je Patel podal pregled Zilogovega procesorja Z80,000. Vsi ti članki so zgolj pregledni članki.

Deveto poglavje podaja nekaj zgodnjih poskusov na področju HLL arhitektur s programskim prevajalnikom. To so arhitekture z naborom strojnih ukazov, ki je v enolični (eden proti enemu) zvezi z ukazi značilnega visokoprogramskega jezika. Prevajanje je izvršeno programsko. Pogoste te arhitekture imenujemo jezikovno skladne arhitekture s programskim prevajanjem. Zanimalje za te arhitekture se je zadealo že leta 1960, ko se je pojavila potreba po računalniku, za katerega bi bilo lažje pisati sistemski in uporabniški kod. Glavni problem tega pristopa pa je kompleksnost materialne računalniške opreme.

To poglavje vsebuje 2 članka. Prvi, katerega avtor je Hassit et al., opisuje enega prvih tovornih eksperimentov pri IBM. Rezultat tega eksperimenta je bil računalnik, ki je bil namenjen za jezik APL. To je bil pravzaprav računalnik IBM 8/360 M/25, ki je bil mikropro-

gramirani in je generiral APL kod. Osnovni razlog za ta razvoj je bila želja razviti učinkovito arhitekturno podporo za jezike kot so APL ali SNOBOL. Ti jeziki ponavadi potrebujejo interpreter in konvencionalen pristop ima lahko za posledico znatno zmanjšanje hitrosti izvajanja programa.

Nielsen predstavlja v drugem članku devetega poglavja glavne rezultate študije, katero cilj je bil načrt računalniške arhitekture in programskega jezika za vesoljske aplikacije.

Oba članka iz tega poglavja podajata rezultate eksperimenta. Računalnika, ki sta pri tem nastala sta danes zastarela, omenjena pa sta v tej knjigi zato, ker sta s svojimi idejami in eksperimentalnimi rezultati vplivala na sodobne tokove pri raznovi novih računalniških arhitektur.

Deseto poglavje opisuje nekatere sodobne raziskave na MIT. Vsebuje dva članka, ki se navezujejo na jezikovno skladne arhitekture. Te so zanimive tako z gledišča VLSI načrtovanja vezij, kot tudi z gledišča mikroprogramiranja. Večina računalnikov, ki temelji na tem pristopu vsebuje mikroprogram za podporo konstrukciji visokoprogramskega jezika. Trenutno stanje razvoja VLSI lahko udinkovito podpira velike ROM pomnilnike za mikroprogram na čipu. Vemo pa, da je VLSI tehnologija premalo močna za podporo kompleksnih visokoprogramskega jezikov na popolni enolični korrespondenci. Torej lahko z VLSI tehnologijo podpiramo bodisi samo izbrano podmnožico visokoprogramskega jezika ali pa uporabimo tehnologijo, ki ni VLSI in podpiramo celotni visokoprogramski jezik.

Prvi članek, katerega avtor je Sussman sodelavci, opisuje zasnova in implementacijo mikroračunalnika na enem vezju, ki direktno interpretira Scheme-79, to je dialect jezika Lisp. Skupina na MIT je razvila interpreter v obliki mikrokoda, ki so ga realizirali v strojni računalniški opremi. Pri tem so uporabili dodatne in nekonvencionalne hardverske pripomočke ter s tem povedali udinkovitost računalnika.

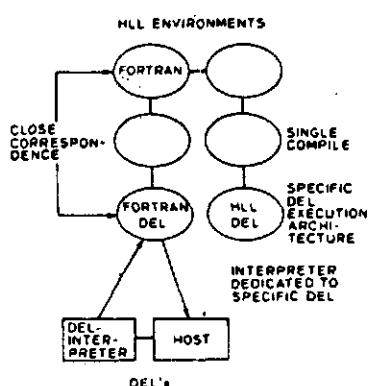
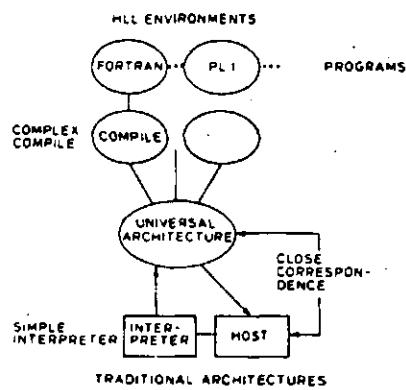
Drugi članek desetega poglavja, katerega avtor je Batali et al., opisuje vezje Scheme-81, ki je naslednik vezja Scheme-79. Scheme-81 je namenjen za okolja, kjer so računalniki specifikirani in kjer velike skupine računalnikov sodelujejo pri reševanju posamezne zahtevne naloge.

V 11. poglavju je govora o dveh zanimivih eksperimentih. Prvi izhaja iz šestdesetih let in opisuje Symbol računalniški sistem, to je računalnik, ki ima nekonvencionalno arhitekturo in o katerem je bilo v preteklosti veliko diskusij. Symbol je računalnik, ki ima splošen programski jezik in time-sharing operacijski sistem implementiran v materialni računalniški opremi. Vzpadbuda za tak eksperiment temelji na dejstvu, da so sčasoma postali prevajalniki zapleteni in jih je bilo težko napisati. Po drugi strani je cena materialne računalniške opreme padala, kompleksnost načrtovalnih orodij za razvoj vezij pa se je boljšala.

Drugi članek enajstega poglavja (Burkis) predstavlja laboratorijski eksperiment, ki je rezultiral v računalnik z imenom Abacus. Abacus je nastal pod vplivom računalnika Symbol, ima pa vrsto originalnih novih rešitev kot na primer kontrolo poslov (job control), učinkovito delo s podatkovnimi tipi in drugo.

Ovanajstvo poglavje vsebuje dva članka, ki obravnavata DEL (Directly Executed Language) pristop to je arhitekturo za direktno izvajanje jezika. To so, po razdelitvi iz slike 1, HLL arhitekture s strojnimi prevajalniki. Pri teh

arhitekturah uporabljam za programiranje običajen visokoprogramska jezik. S predprocesiranjem izvornega koda se prevade izvorni program v DEL obliko, kar je optimalni vmesnik med posameznim visokoprogramskega jezikom in izvajalnim računalnikom (Slika 7).



Slika 7: Primerjava med DEL arhitekturo in tradicionalno arhitekturo.

Prvi članek v tem poglavju, katerega avtor je Hoevel, podaja pregled različnih metod, ki se uporabljajo za generiranje DEL arhitekture.

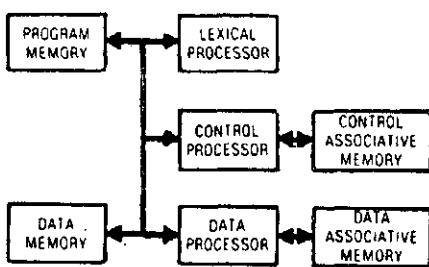
Flynn je avtor drugega članka in opisuje DELtran eksperiment, ki se navezuje na okolje programskega jezika Fortran II. Izvajalni računalnik pri tem eksperimentu je mikroprogramabilni računalnik EMHY z univerze Stanford. Članek podaja primerjavo v performansah, ko se uporablja DELtran ali pa druge konvencionalne arhitekture kot na primer IBM 370. Prihranek pri velikosti programa in naslavljanju pomnilnika je pri uporabi DELtran-a znaten.

Flynn in Hoevel sta s svojim delom močno vplivala na razvoj celotnih HLL arhitektur, vključno z RISC arhitekturami.

Trinajsto poglavje opisuje v dveh člankih arhitekturo računalnika za direktno izvajanje, ki so jo zasnovali na University of Maryland. Bistvo pristopa univerze Maryland je dejstvo, da arhitektura omogoča direktno interpretiranje izvornega koda programov. Na ta način je semantični prepad med programskega jezikom in arhitekturo popolnoma premogčen.

Prvi članek v trinajstem poglavju (Chu) se ukvarja s sistemskimi vprašanji interaktivnega visokoprogramskega jezika in z arhitekturo za njegovo direktno izvajanje. Članek obravnava razliko med sistemom z interaktivnim prevajanjem in sistemom z interaktivnim interpretiranjem.

Drugi članek trinajstega poglavja (Chu) analizira interakcijo med procesorjem in visokoprogramskega jezikom pri arhitekturi za direktno izvajanje (Slika 8). Procesor za to arhitekturo sestoji iz treh glavnih delov, to je slovarskega dela procesorja, nadzornega dela procesorja in podatkovnega dela procesorja. V članku je prikazana interakcija teh treh delov procesorja na primerih, ko uporabljamo visokoprogramskega jezik.



Slika 8: Organizacija računalnika za direktno izvajanje.

Knjiga se konča s 14 poglavji, ki obravnava preglede članke o razvoju HLL arhitektur v različnih razvojnih centrih po svetu. Poglavje vsebuje dva članka, ki predstavljata raziskovalne in razvojne dosežke na področju HLL računalniških arhitektur v Japonski in v Evropi. Yamamoto je v članku, ki daje pregled teh raziskav na Japonskem, prikazal strukturo PL/I procesorja, konfiguracijo Cobol računalnika, več različnih Lisp računalnikov, NEC-ov singgle-chip Pascal procesor in druge zanimivosti.

Drugi članek zadnjega poglavja (Sylbey) daje pregled raziskav s področja računalniških arhitektur v Evropi. Omenjeni so dosežki s področja reducirnih arhitektur z University of Kent, univerze v Bonnu in Gesellschaft für Mathematik und Datenverarbeitung. S podatkovno vodenimi arhitekturami se ukvarjajo na University of Manchester, Katholieke Universiteit Leuven in na univerzi v Dortmundu. Z reduciranimi arhitekturami se ukvarjajo na University of Reading. Jožikovno vodenje arhitekture razvijajo na Tehnični univerzi Berlin in na Univerzi Dortmund. HLL arhitekture razvijajo na univerzi Paul Sabatier v Toulouse ter na univerzah v Dortmundu, Frankfurtu ter Kaiserslautern. Arhitekture za direktno izvajanje razvijajo na gvedskem in v Franciji. Vsada je centrov, kjer raziskujejo in razvijajo HLL računalniške arhitekture v Evropi več mnogo ved. V tem članku je izbranih samo nekaj najbolj zanimivih. Na koncu je priložen še spisek evropskih revij in simpozijev s področja HLL računalniških arhitektur.

Pripravil: Saša Prešern