

Y36SAP - 13

procesor - control unit
obvodový a mikroprogramový řadič
RISC a jiné ...

28.5.2008

Y36SAP-control unit

1

Von Neumannova architektura (SAP1)

- Instrukce a data jsou uloženy v téže paměti.
- Paměť je organizována lineárně (tzn. jednorozměrně) a je rozdělena na stejně velké buňky, které se adresují celými čísly (zprav. 0, 1, 2, 3, . . .).
- Data ani instrukce nejsou explicitně označeny.
- Explicitně nejsou označeny ani různé datové typy.
- Pro reprezentaci dat i instrukcí se používají dvojkové signály.
- V instrukci zpravidla není uváděna hodnota operandu, ale jeho adresa.
- Instrukce se provádějí jednotlivě, a to v pořadí, v němž jsou zapsány v paměti, pokud není toto pořadí změněno speciálními instrukcemi (nazývanými skoky).

28.5.2008

Y36SAP-control unit

2

Von Neumannova architektura (SAP1)

- **Důsledek** - podle výpisu paměti nelze poznat, zda jde o instrukce nebo o data (ani o jaká data) – je třeba znát kontext
- Počítač tvoří:
 - hlavní paměť (main memory)
 - procesor:
 - datová část
 - ALU – aritmeticko-logická jednotka
 - Registry
 - řídicí část
 - Řadič – control unit, controller
 - vstupní/výstupní zařízení

28.5.2008

Y36SAP-control unit

3

Hardwarová architektura počítače (1)

Hlavní komponenty
počítačového
systému

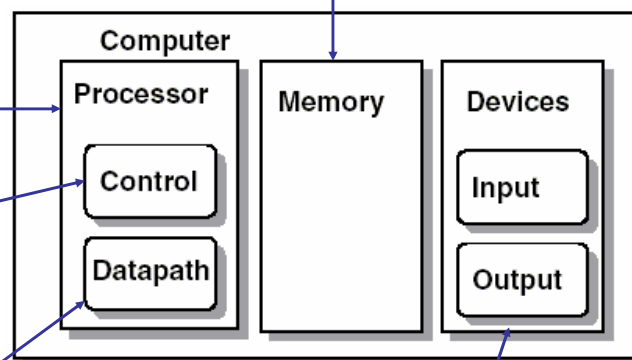
Procesor

Řídicí část
Řadič - Controller

Datová část
Aritmeticko-logická jednotka - ALU

Paměť

Vstupní/Výstupní zařízení



28.5.2008

Y36SAP-control unit

4

Řadič procesoru

- Pracuje podle instrukčního cyklu
- Řídí činnost všech výkonných jednotek počítače podle instrukcí a jejich kódu, podle **instrukčního cyklu**
- Je to sekvenční obvod – závisí na sekvenci vstupních (**stavových**) signálů, které generují výkonné jednotky (ALU, HP - instrukce) a vysílá jim **řídící** signály
- Pracuje v nekonečném cyklu – řídí zpracování instrukcí
- Navrhne se podle instrukčního cyklu a výběru ISA – z grafu přechodů – vývojového digramu
- Podle způsobu jeho realizace existuje tzv. **obvodový** (klasický) řadič a **mikroprogramový** řadič

28.5.2008

Y36SAP-control unit

5

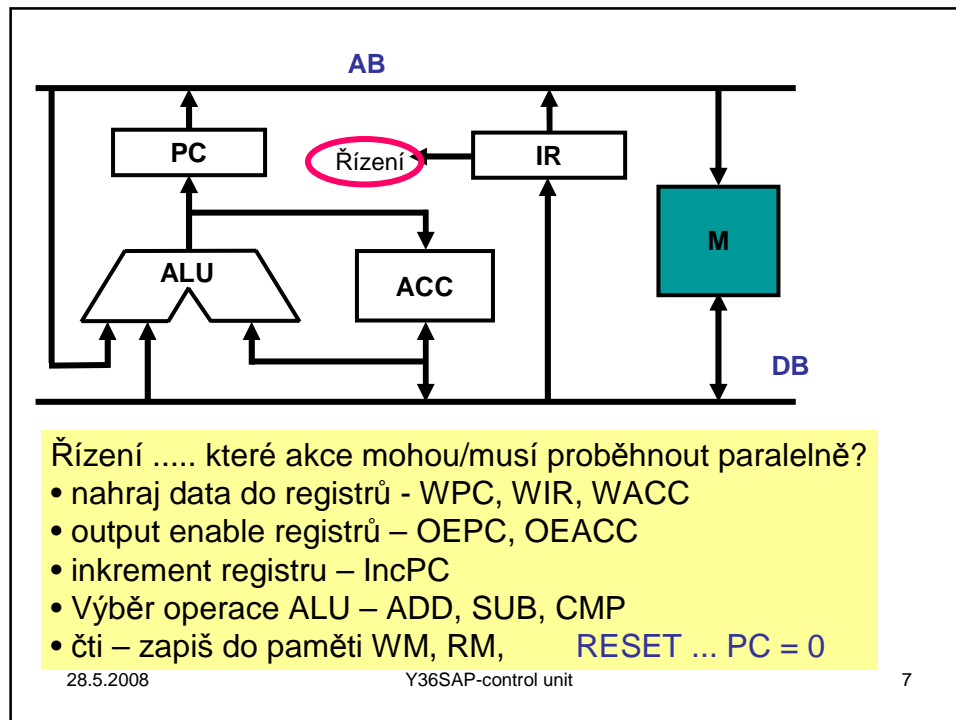
Instrukční cyklus



28.5.2008

Y36SAP-control unit

6



Návrh procesoru ADOP

GPR architektura

Registry ... 16 registrů dostupných programátorovi:

R0 – R11 universálních (datových) registrů

SP – ukazatel zásobníku

PC – programový čítač

PSW – stavový registr,

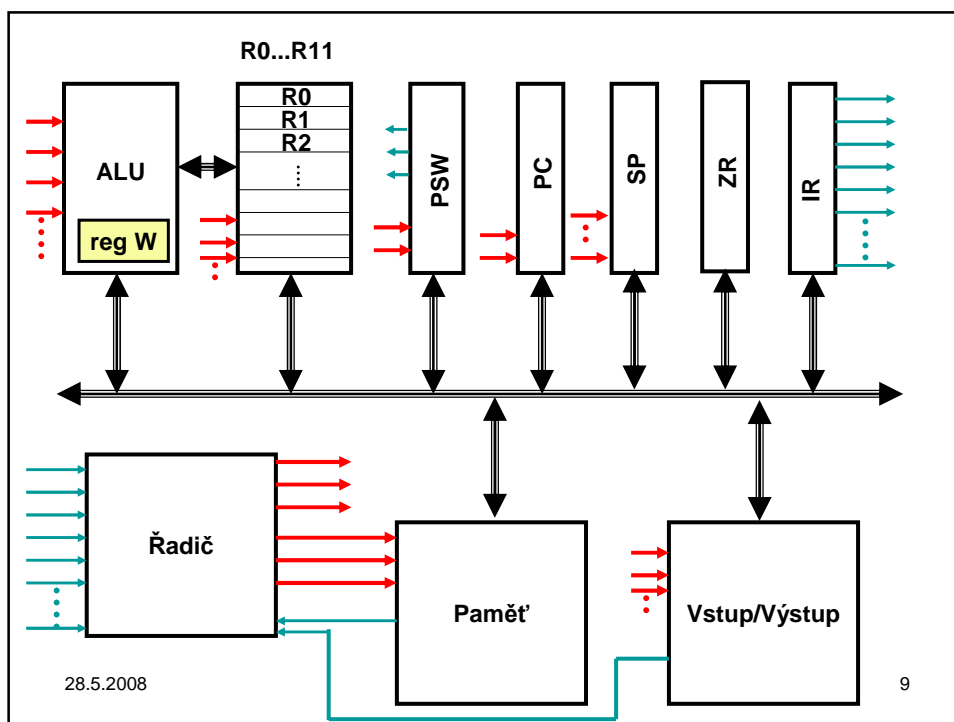
Příznaky Z ... zero, C ... carry, S ... sign, O ... overflow,
ES ... extended sign (znaménko 2. operandu v
binárních operacích)

ZR – obsahuje konstantní nulu

28.5.2008

Y36SAP-control unit

8



Kódování instrukcí – obsah IR

- 2 až 4 bytové instrukce
- Operační znak 2 slabiky, operand až 2 slabiky

3.půlbyte (15.-12.bit)	2.půlbyte (11.-8.bit)	1.půlbyte (7.-4.bit)	0.půlbyte (3.-0.bit)
---------------------------	--------------------------	-------------------------	-------------------------

Třetí půlbyte (nejvýznamnější bity operačního kódu) určuje skupinu instrukcí.

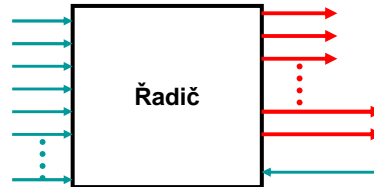
Nejvýznamnější tj. šestnáctý bit určuje délku instrukce:

- 0 – dvoubytové instrukce
- 1 – čtyřbytové instrukce

Jména strojových instrukcí vyjadřují význam jednotlivých půlbyťů operačního kódu.

Realizace řadiče

- jde o sekvenční obvod
- mnoho vstupů
- mnoho výstupů



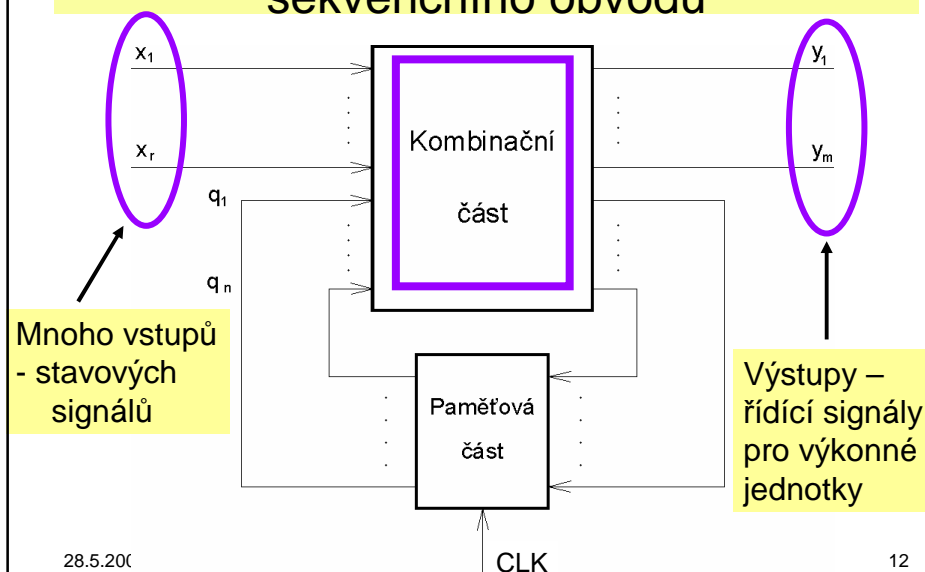
- ❑ ale v daném taktu (stavu)
se uplatní jen málo z nich, tzn. jeden nebo žádný
většinou se pokračuje následujícím stavem
- ❑ jde o popis algoritmu - činností v instrukčním
cyklu, lze použít vývojový diagram

28.5.2008

Y36SAP-control unit

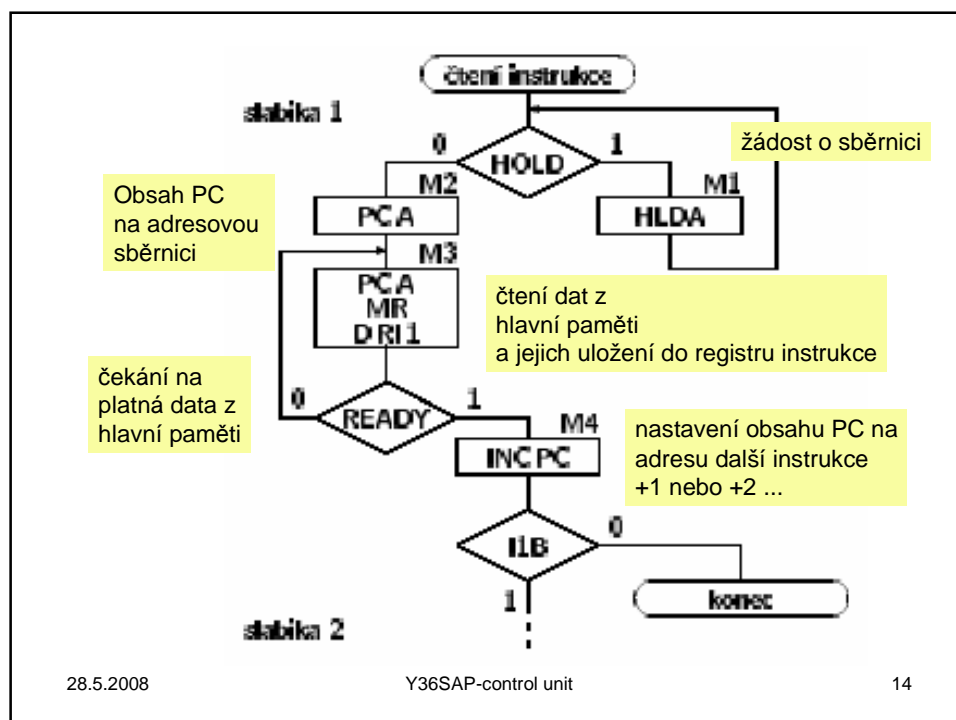
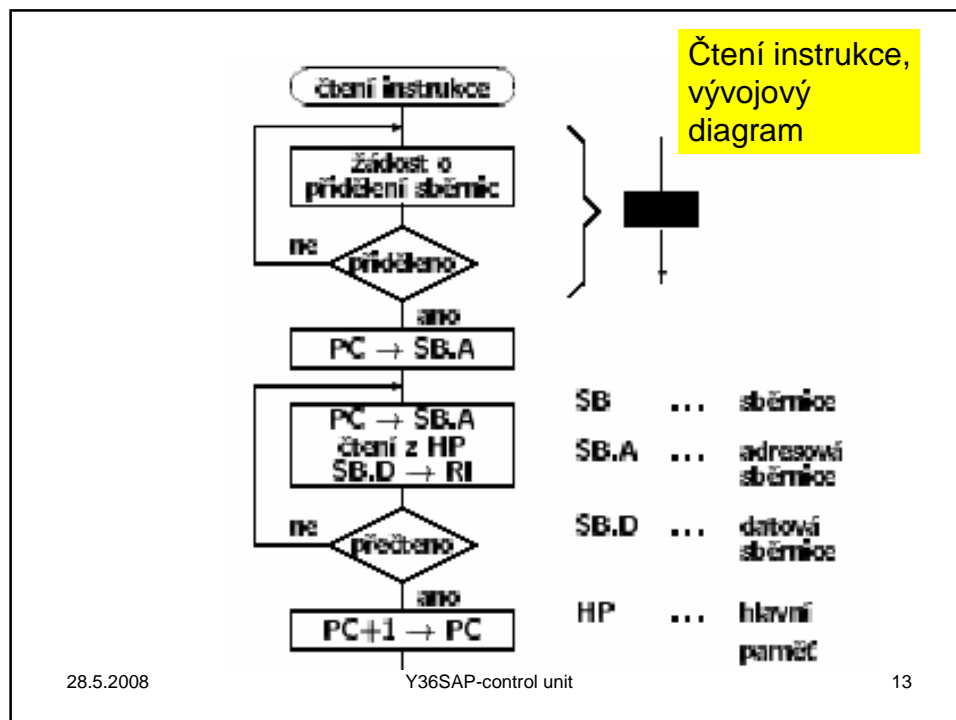
11

Obecný (Huffmannův) model sekvenčního obvodu

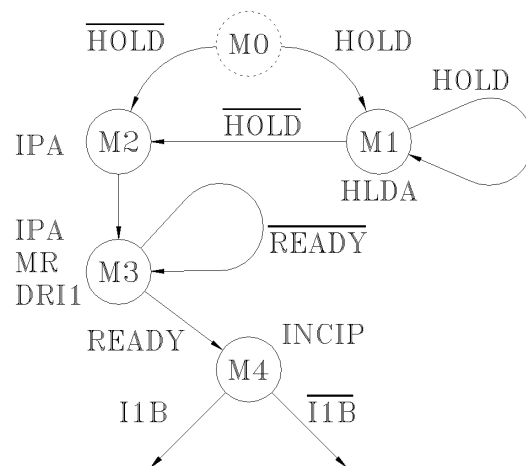


28.5.2008

12



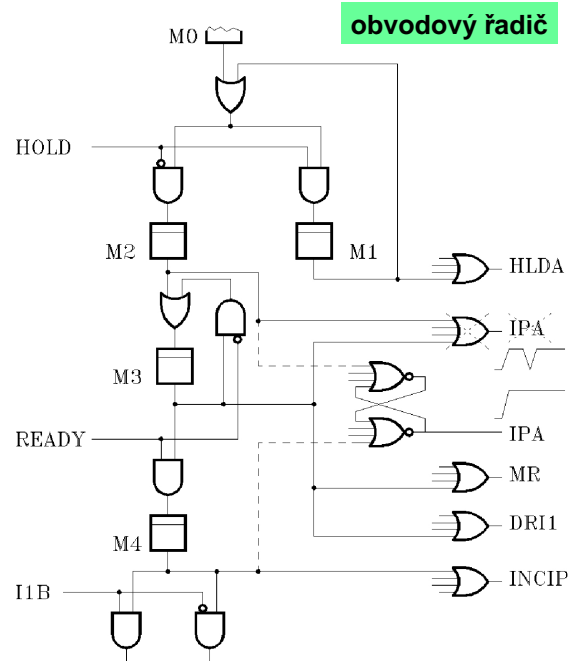
totéž jako graf přechodů



28.5.2008

15

totéž
realizované
jako
obvodový
řadič,
v kódu 1zN,
tzn. každému
stavu M_i
odpovídá
jeden D-KO



28.5.2008

Realizace řadiče

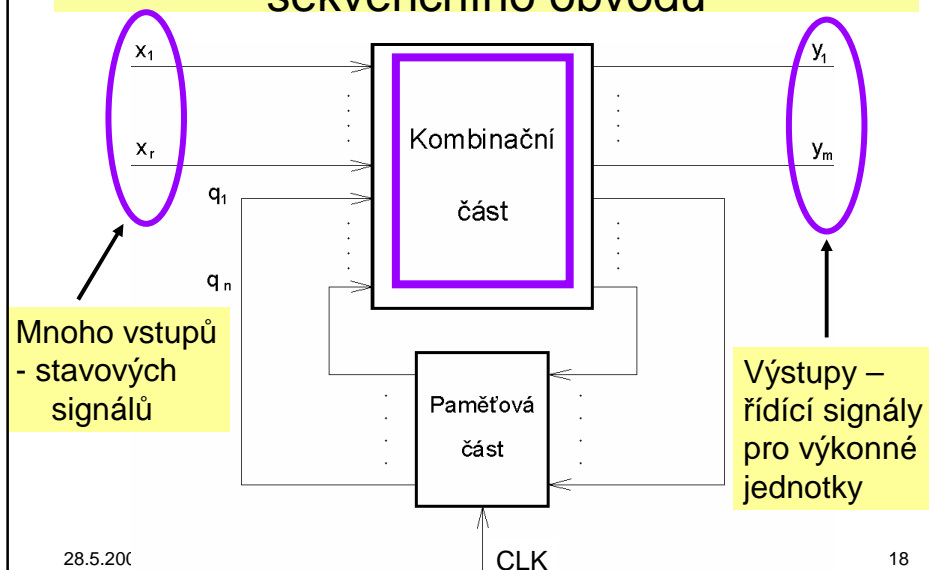
- Podle způsobu jeho realizace existuje tzv. **obvodový** (klasický) řadič a **mikroprogramový** řadič
- **Obvodový řadič** – návrh klasického sekvenčního obvodu ... usnadnění ... kód vnitřních stavů 1 z n – pak lze návrh provést z vývojových diagramů popisující činnost procesoru při provádění instrukcí podle instrukčního cyklu
- **Mikroprogramový řadič** –
 - Sekvenční obvod s kombinační částí realizovanou pamětí (nazývá se řídicí paměť, paměť mikroprogramů, control memory)
 - Jednotlivé dílčí operace, které se provádějí při zpracování instrukcí jsou uloženy v této paměti a říká se jim **mikroinstrukce**
 - Soubor mikroinstrukcí tvoří mikroprogram, soubor všech mikroprogramů je mikroprogramové vybavení - **firmware**

28.5.2008

Y36SAP-control unit

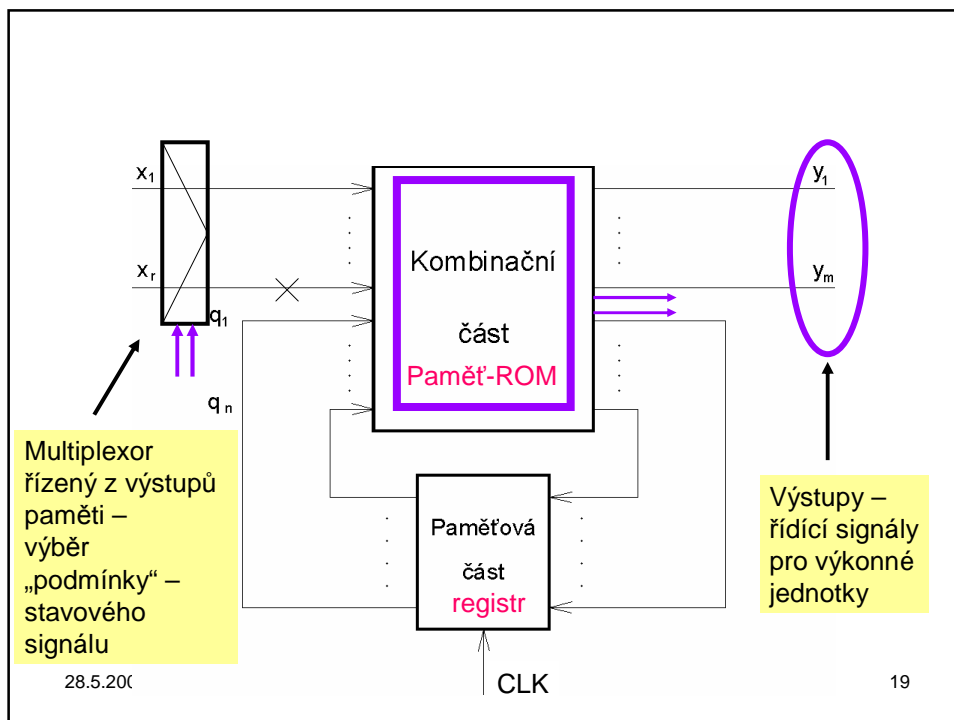
17

Obecný (Huffmannův) model sekvenčního obvodu



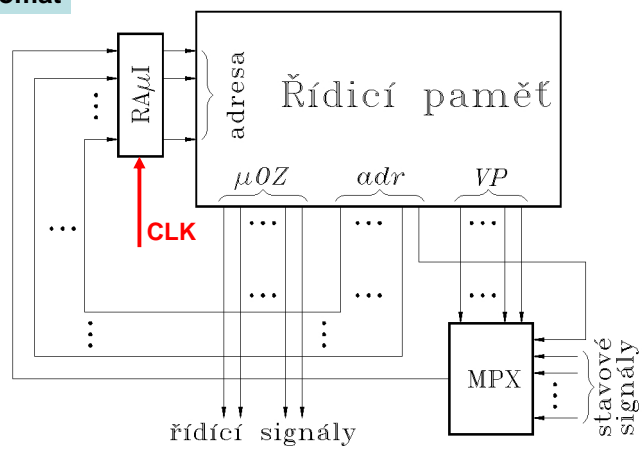
28.5.2008

18



Mikroprogramový řadič

Moorův automat



Mikroprogramový řadič

- jsou možné další úpravy a vylepšení podle souboru instrukcí a výkonných jednotek
- čítač adres mikroinstrukcí, protože většinou se pokračuje následující
- např. pro často se opakující části instrukcí zavést možnost podmikroprogramů a HW zásobník jako součást řadiče a čítač taktů s možností přednastavení pro počet opakování cyklů mikroinstrukcí

28.5.2008

Y36SAP-control unit

21

- Soubor všech mikroprogramů, tzn. popisu činností každé instrukce v taktech je mikroprogramové vybavení - **firmware**
- **Horizontální** mikroprogramování - viz Sl.17
 - dlouhé mikroinstrukce
 - jedna mikroinstrukce v jednom taktu
 - řídicí signály součástí mikroinstrukce
 - omezené větvení
- **Vertikální** mikroprogramování
 - krátké mikroinstrukce
 - čítač adres mikroinstrukcí
 - jedna mikroinstrukce ve více taktech
 - dekodér pro řídicí signály

28.5.2008

Y36SAP-control unit

22

Řadič

program se skládá z instrukcí

instrukce ... se provádí ve několika taktech a skládá se z mikroinstrukcí

mikroinstrukce ... okamžitý stav procesoru skládá se:

- z řídících signálů pro výkonné jednotky
- určení následného stavu (kde se bude pokračovat)
- volby vstupů, které jsou v příštím taktu významné

28.5.2008

Y36SAP-control unit

23

Optimalizace

- Jaký má být návrh procesoru, aby pracoval co nejefektivněji?
- Má mnoho instrukcí nebo minimálně?
- Jaký má mít řadič?
- Jak celou činnost zrychlit?
- Má mít hodně registrů?
-

28.5.2008

Y36SAP-control unit

24

Problémy k řešení při návrhu

- Specifikace – co chceme realizovat
- Hlavně aby to fungovalo
- Optimalizace z různých hledisek
 - Velikost
 - Rychlost
 - Příkon
 - Spolehlivost
 - Cena včetně návrhových prostředků
 - Rychlost návrhu
- Testovatelnost – DFT = design for testability

28.5.2008

Y36SAP-control unit

25

Statistika

<i>Rank</i>	<i>Instruction</i>	<i>Frequency</i>
1	load	22%
2	branch	20%
3	compare	16%
4	store	12%
5	add	8%
6	and	6%
7	sub	5%
8	register move	4%
9	call	1%
10	return	1%
Total		96%

10 instrukcí tvoří **96%** instrukcí v programech.
(výsledky z měření SPECInt95)

28.5.2008

Y36SAP-control unit

26

Adresní módy

Adresní mód	Syntaxe	Sémantika
Registrový	ADD R4,R3	$R4 \leftarrow R4 + R3$
Přímý operand	ADD R4,#3	$R4 \leftarrow R4 + 3$
Bázovaný s offsetem	ADD R4,100(R1)	$R4 \leftarrow R4 + \text{Mem}[100 + R1]$
Registrový nepřímý	ADD R4,(R1)	$R4 \leftarrow R4 + \text{Mem}[R1]$
Bázovaný/indexovaný	ADD R3,(R1+R2)	$R3 \leftarrow R3 + \text{Mem}[R1 + R2]$
Přímá/absolutní adresa	ADD R1,(1001)	$R1 \leftarrow R1 + \text{Mem}[1001]$
Paměťový nepřímý	ADD R1,@(R3)	$R1 \leftarrow R1 + \text{Mem}[\text{Mem}[R3]]$
S post-inkrementací	ADD R1,(R2)+	$R1 \leftarrow R1 + \text{Mem}[R2]; R2 \leftarrow R2 + d$
S pre-dekrementací	ADD R1,-(R2)	$R2 \leftarrow R2 - d; R1 \leftarrow R1 + \text{Mem}[R2]$
Škálovaný	ADD R1,100(R2)[R3]	$R1 \leftarrow R1 + \text{Mem}[100 + R2 + R3 \cdot d]$



28.5.2008

Y36SAP-control unit

27

Statistika

SPECInt měření na počítači DEC VAX který podporuje téměř všechny adresní módy

- **Bázovaný s offsetem:** 42% prům. 32% - 55% 
- **Přímý operand:** 33% prům., 17% - 43% 
- **Registrový nepřímý:** 13% prům., 3% - 24%
- **Škálovaný:** 7% prům., 0% - 16%
- **Paměťový nepřímý:** 3% prům., 1% - 6%
- **Ostatní:** 2% prům., 0% - 3%

Velikost přímého operandu :

- 50% - 60% se vejde do **8 bitů**

- 75% - 80% se vejde do **16 bitů**

Velikost offsetu :

- pouze 1% offsetu se nevejde do **16 bitů**

28.5.2008

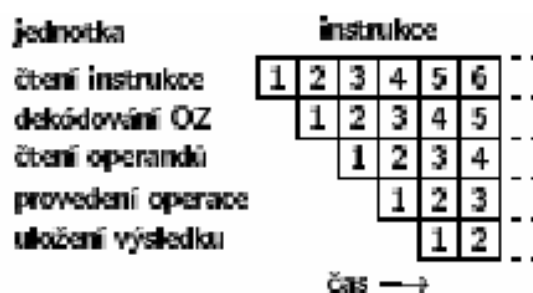
Y36SAP-control unit

28

Proudové zpracování instrukcí *pipelining*

princip výrobního pásu - zpracování instrukce po částech

- každá jednotka provede část operace
- jednotky pracují současně



28.5.2008

29

...pipelining

- triviální případ - předčítání instrukcí, jedna instrukce se čte, další dekóduje, provádí ...
- v ideálním případě je v každém taktu dokončena jedna instrukce

konflikty:

- datový - potřebná data dosud nejsou uložena
 - skokový - adresu skoku zatím nelze určit
- řešení ... počkat (to nejjednodušší, ale ne jediné),

....

28.5.2008

Y36SAP-control unit

30

Počítače typu RISC

Reduce Instruction Set Computers

- jak navrhnout rychlý procesor? Co nejvíce výkonných instrukcí?
- statistika .. co nejefektivnější mají být ty nejpoužívanější (přesuny, skoky, srovnání, ...), výkonné se používají málo

28.5.2008

Y36SAP-control unit

31

Kvantitativní studie existujících počítačů nepotvrzuje, že cesta směrem ke komplexním instrukcím je správná:

- Komplexní instrukce a adresní módy jsou málokdy využívány (komplexnější instrukce = méně používaná instrukce)
- **10 %** instrukcí je využíváno **90 %** času
=> **Amdahlův zákon** doporučuje soustředit se na těchto **10 %** instrukcí
- Spousty **chyb** ve složitých mikroprogramech implementujících komplexní instrukce (nutno distribuovat **záplaty mikrokódu**)
- Málo používané komplexní instrukce vyžadující **kódování instrukcí proměnlivé délky** zpomalující provádění používaných jednoduchých instrukcí
- **Paměťová hierarchie** (cache) umožní zkrátit dobu načítání instrukcí
- **Překladače ne hardware** uzavřou "**sémantickou mezeru**" !

Počítače s redukovanou složitostí instrukcí jsou nazývány RISC.
Tradičním architektuám je následně vymyšlen název CISC.

28.5.2008

Y36SAP-control unit

32

Počítače typu RISC - charakteristika

- malý počet jednoduchých instrukcí (<128)
- krátká doba provedení instrukce - dokončení v jednom taktu
- pipelining
- obvodový řadič
- malý počet formátů instrukcí (≤ 4)
- malý počet způsobů adresace (≤ 4)
- velký počet registrů (> 32)
- komunikace s pamětí pouze instrukcí "přesun"

protipól CISC - Complex Instruction Set Computers
počítače s rozsáhlým souborem instrukcí

28.5.2008

Y36SAP-control unit

33

RISC vs CISC

- Mikroprogramování není třeba, řídící paměť uvolní místo více registrům či skryté paměti (cache)
- Snadnější implementace proudového zpracování (*pipeliningu*)
- Více registrů umožní efektivnější využití překladačem
- Jednodušší návrh je snadno a dříve dokončen
- Jednodušší procesor může běžet na vyšší hodinové frekvenci

RISC vs CISC (zjednodušené srovnání výkonnosti)

Tradiční CISC mají průměrné **CPI** mezi **5 – 10** takty,

RISC mají průměrné **CPI** mezi **1.3 a 2** takty (díky *pipeliningu*).

Počet instrukcí (IC) programů pro RISC je pouze o málo vyšší než u CISC procesorů

=> Tento fakt dává RISC procesorům **výkonnostní náskok** oproti **tradičním CISC** procesorům.

28.5.2008

Y36SAP-control unit

34

Chronologie

Konec 70.let - počátek 80.let : „RISC movement“

- John L. Hennessy, David A. Patterson a další
- RISC I, RISC II, MIPS (projekty na UCB a Stanfordu)

80. léta – polovina 90.let: Komericializace RISC

- RISC architektury jsou výrazně výkonnější než CISC
- RISC ovládají trh pracovních stanic, prvních serverů a vestavné aplikace. Jedním z posledních CISC je **x86** ...
- Všechny nové počítače navrženy jako RISC

Konec 90. let a současnost : “Post-RISC era”

- **x86** dosahuje srovnatelnou i vyšší výkonnost než RISC při nižší ceně => vytlačuje RISC počítače z „low end“ trhu a tlačí se stále výše ...
- „PowerPC G5 má více instrukcí a je stejně složité jako Pentium 4“
- “Na ISA nezáleží.”, “RISC a CISC architektury konvergovaly. ”
- „Na obou přístupech je něco pravdy“ ?

28.5.2008

Y36SAP-control unit

35

Nové oblasti aplikací (90.léta)

- multimédia, grafika – speciální aritmetické instrukce
- počítačová bezpečnost
- podpora virtualizace počítače (Java, .NET, více OS)

Podpora ISA pro větší výkonnost – datový paralelismus

“SIMD instrukce”

- jedna instrukce pracující s krátkými **datovými vektory**
- **128b = 16 x 8b, 8 x 16b, 4 x 32b, 2 x 64b**
- operace podporující multimediální aplikace (saturační aritmetika, **celočíslná i pohyblivá řád. čárka**)
- efektivnější využití registrů a datové cesty
- podpora knihoven a také překladačů

Komerční příklady:

MMX, 3DNow!™, SSE, SSE2, PowerPC AltiVec™, Sun VISTM

28.5.2008

Y36SAP-control unit

36

Podpora ISA pro větší výkonnost – instrukční paralelismus (ILP)

“VLIW instrukce”

- VLIW = **Very Long Instruction Word**
- jedna “dlouhá instrukce” = povel k více (4-16) operacím, které mohou být provedeny paralelně
- VLIW může potenciálně využít více **instrukčního paralelismu (ILP)** než stávající superskalární procesory s tradiční ISA
- VLIW by měl umožnit implementaci jednoduššího procesoru než je stávající superskalární RISC/CISC
- VLIW ISA se používají úspěšně v **DSP** a **Media procesorech** (vestavné aplikace)

Myšlenky VLIW v univerzálních počítačích – úspěch či neúspěch ?

- Transmeta Crusoe (VLIW s JIT překladačem x86)
- Intel IA64 (EPIC) – komplexní ISA kombinující VLIW s predikátováním a dalšími technikami softwarové spekulace

28.5.2008

Y36SAP-control unit

37

SoC, NoC, ASIC, FPGA, CPLD,

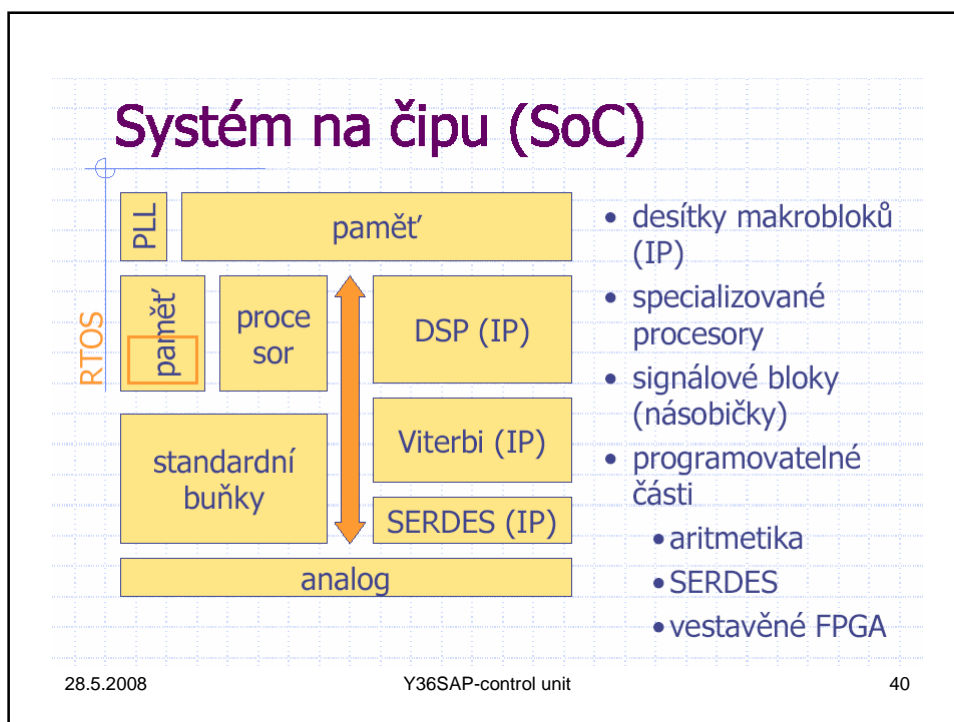
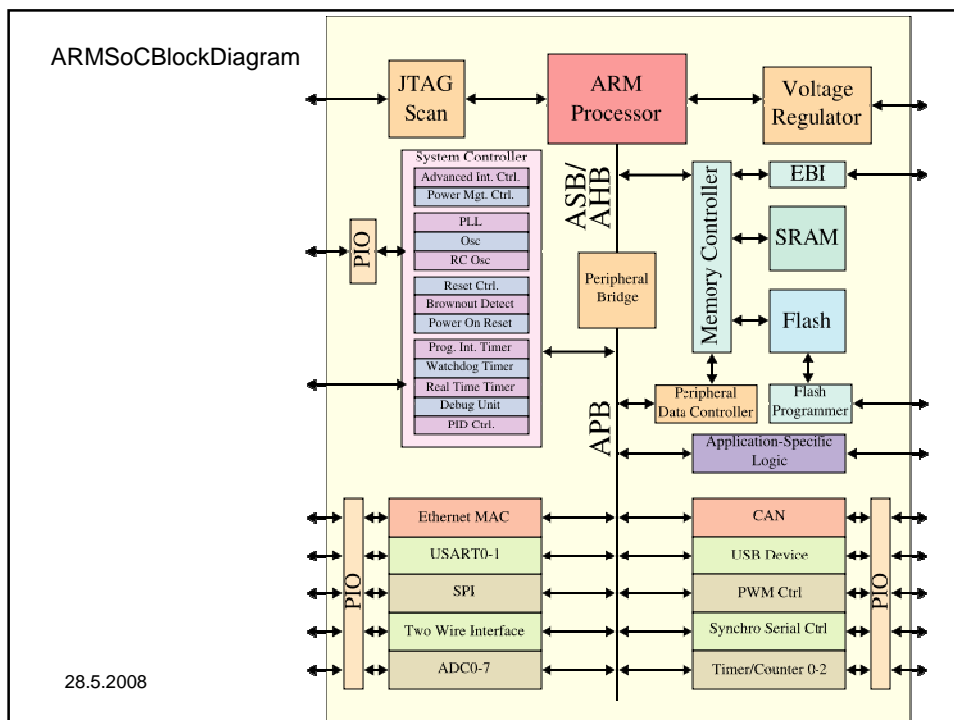
Trendy v návrhu čipů:

- **System-on-a-chip** or **system on chip (SoC or SOC)** integruje všechny komponenty počítače nebo jiného elektronického systému na jeden čip. Může obsahovat: číslicové, analogové i mixed-signal komponenty a často i radiový přenos. Typické aplikace – vestavné (embedded) systémy.
- **System in package (SiP)** více čipů v jednom pouzdře.

28.5.2008

Y36SAP-control unit

38



Vestavné systémy

- Řešení specifických úloh, ne „general-purpose“ počítač
- Real-time aplikace, bezpečnost, maximální zjednodušení – minimální cena
- Vestavěné (built-in) do zařízení, které řídí
- Software – obvykle zvaný firmware, uložený v ROM nebo Flash paměti a běžící na limitovaných HW zdrojích

28.5.2008

Y36SAP-control unit

41

Network-on-a-chip (NoC)

- Network-on-a-chip (**NoC**) – zahrnuje více domén s asynchronními hodinami. Pro návrh vzájemné komunikace se používají síťové metody již na čipu na rozdíl od konvenčních sběrniceových metod

28.5.2008

Y36SAP-control unit

42

ASIC

- Application Specific Integrated Circuit –
Obvod vyvíjený na zakázku pro jednu aplikaci jednoho zákazníka

28.5.2008

Y36SAP-control unit

43

Strukturovaný ASIC

- ◆ Obvod vyroben předem včetně spodních vrstev propojení
- ◆ Připravené bloky vyšší úrovně (procesor, násobička, paměť)
- ◆ Architektura obvodu dána propojením (vrchních vrstev)
- ◆ Verze s programovatelnou logikou (e-ASIC)
- ◆ Jednoduchý a rychlý návrh na vysoké úrovni

28.5.2008

Slides z předmětu X36ACS

44

Hradlové pole

připravené tranzistory pro logiku

jestliže není dedikovaný prostor pro propojení: architektura **sea of cells**

připravené tranzistory pro periferie

čip se vyrábí ve velkých sériích celý kromě propojovací sítě
funkce obvodu dána dodatečně propojením (personalizace)

28.5.2008 Y36SAP-control unit 45

Programovatelné obvody

- ◆ procesory
- ◆ ASIP
- ◆ konfigurovatelné bloky (UART, ...)
- ◆ FPGA, CPLD
- ◆ Platforma: prostředek k realizaci SoC (System on a Chip) s vhodně programovatelnými částmi

co je to programovatelnost?

programovatelný (OTP) keramický kondenzátor

28.5.2008 Y36SAP-control unit 46

Vlastnosti

- ◆ **Žádné** masky, velké logistické výhody
- ◆ O řád **větší** a **pomalejší** než ASIC stejné technologie
- ◆ Ale: ASIC běžně 180-250nm, FPGA 90nm

28.5.2008

Y36SAP-control unit

47

Dělení programovatelných obvodů

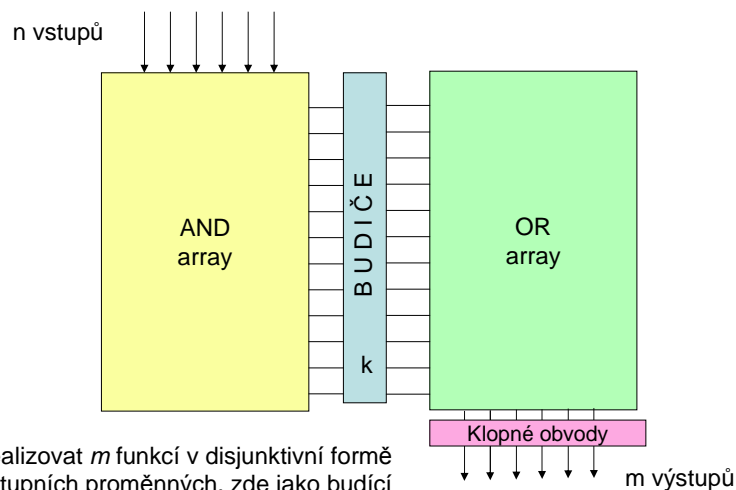
- **PLD (Programmable Logic Device)**
pevně daná struktura typu:
vstup - pole **AND** - pole **OR** – výstup (+někdy D-FF)
PROM (Programmable Read Only Memory)
PAL (Programmable Array Logic)
..... dnes **GAL** (Generic Array Logic)
PLA (Field Programmable Logic Array)
- **CPLD (Complex PLD)**
– složitější architektury vycházející z PLD (vrstevnaté, s propoj. maticí, ...)
- **FPGA (Field Programmable Gate Array)**
– pravidelná struktura programovatelných logických bloků s vodorovnými či svislými propojovacími linkami a propojovacími maticemi, <http://sweb.cz/fpga/>

28.5.2008

Y36SAP-control unit

48

PLD - struktura



28.5.2008

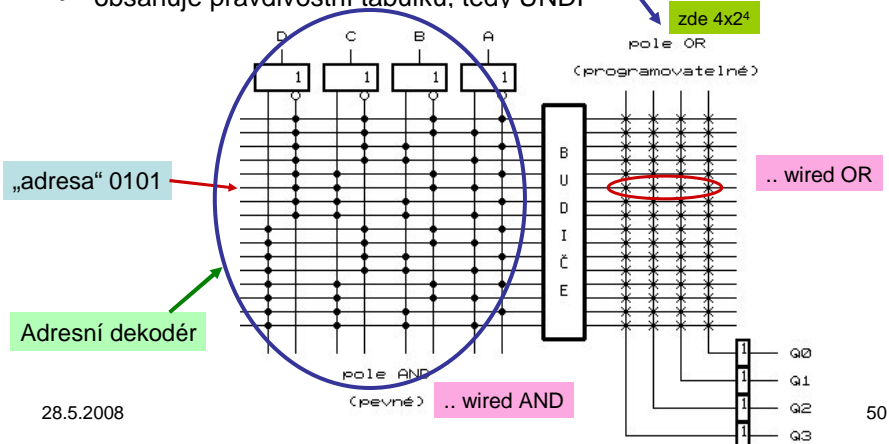
Y36SAP-control unit

49

Obvody PROM

- programovatelné pole OR
- počet programovatelných bodů: $N = m \cdot 2^n$
- EEPROM (Electrically Erasable PROM)
- obsahuje pravdivostní tabulku, tedy ÚNDF

n adresních vstupů
 m datových výstupů
 realizuje m funkcí
 n proměnných

zde 4×2^4 

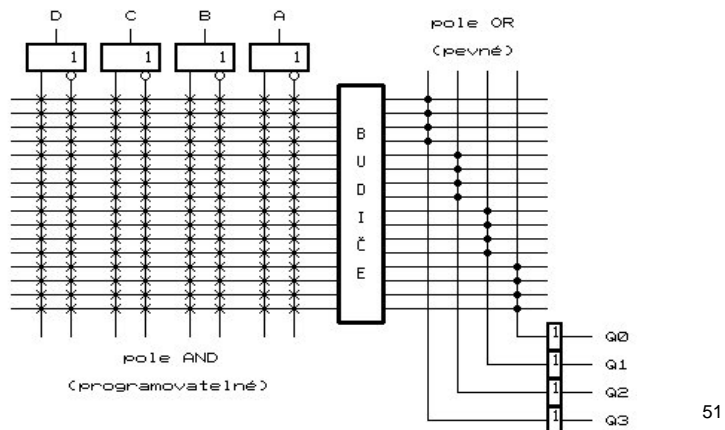
28.5.2008

50

Obvody PAL

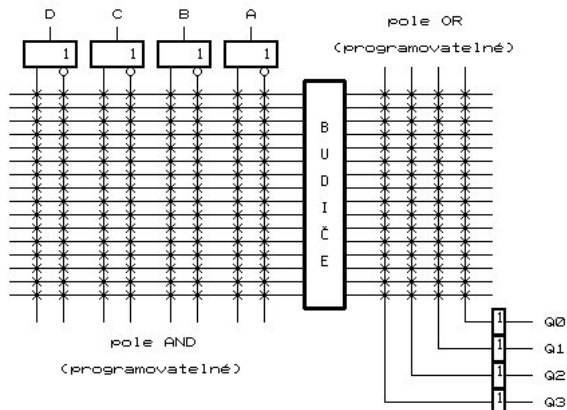
dnes – základ CPLD,
např. CoolRunner

- programovatelné pole AND, pevné pole OR
- počet programovatelných bodů: $N = 2m.k.n$
- omezený počet součinných termů k (pro každou výstupní funkci)
- realizuje MNDF pro každý výstup



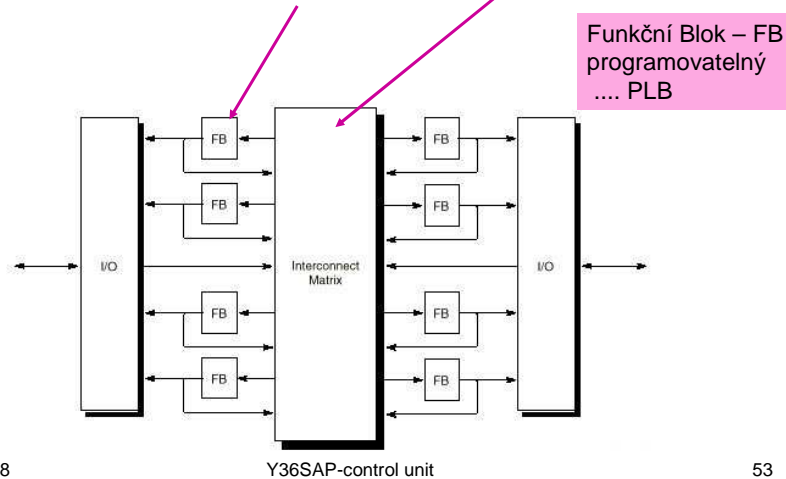
Obvody PLA

- programovatelné pole AND i OR
- počet programovatelných bodů: $N = m.k + 2k.n$
- umožňuje skupinovou minimalizaci
- v současnosti se už nepoužívají



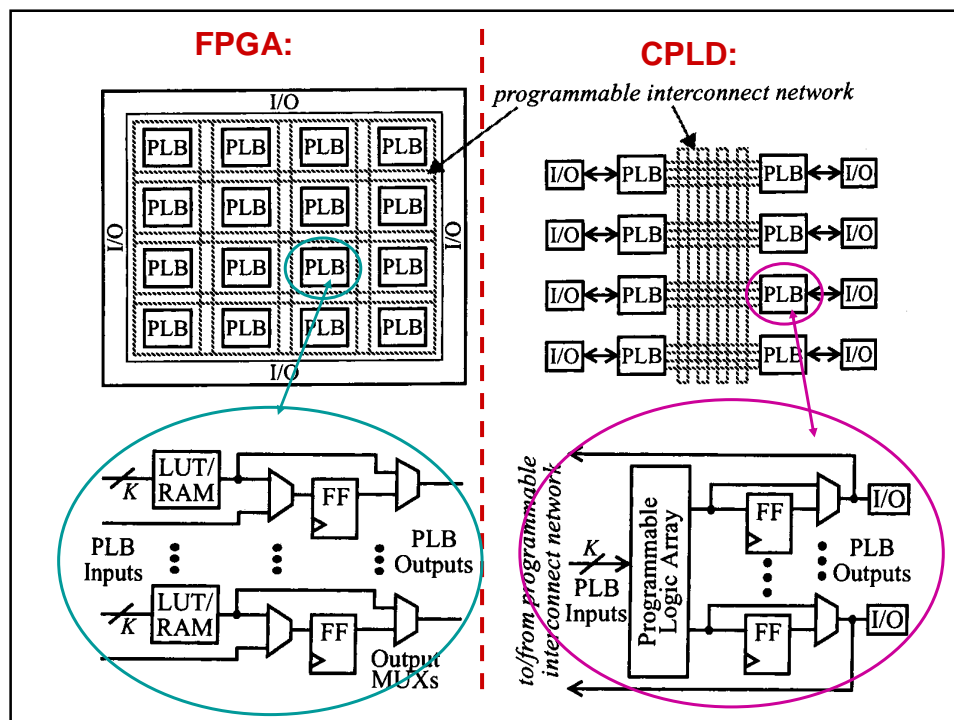
Obvody CPLD

- vychází z PAL, tj. programovatelná matice AND
- obsahují několik PALů (zde FB) a centrální propojovací matici



28.5.2008

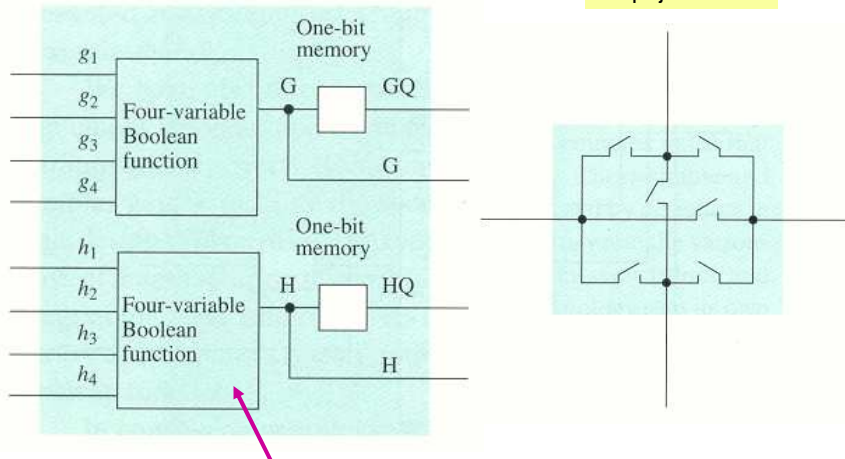
53



FPGA, příklad

Programmable logic block - PLB

Propojovací bod



4-input Look Up Table (4-LUT) paměť RAM, programuje se tabulkou

28.5.2008

Y36SAP-control unit

55

The End

Přejeme úspěch u zkoušky

28.5.2008

Y36SAP-control unit

56

Výzkum v oblasti DD na KP

DD = Digital Design, číslicový návrh

- BIST
- Fault-tolerant
- Kryptografický HW
- Logická minimalizace
- Heuristické algoritmy
- Formální modely

28.5.2008

Y36SAP-control unit

57