Řadič

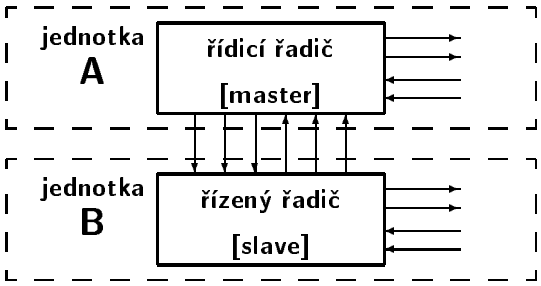
- v užším slova smyslu: **řídicí jednotka počítače** (viz schéma počítače von Neumannova typu) **[control unit]**
- v širším slova smyslu: **řídicí jednotka vůbec** (např. řadič tiskárny, řadič aritmetické jednotky, řadič počítače apod.) **[controller]**

sekvenční obvod — **výstupy: řídicí signály**
vstupy: stavové signály

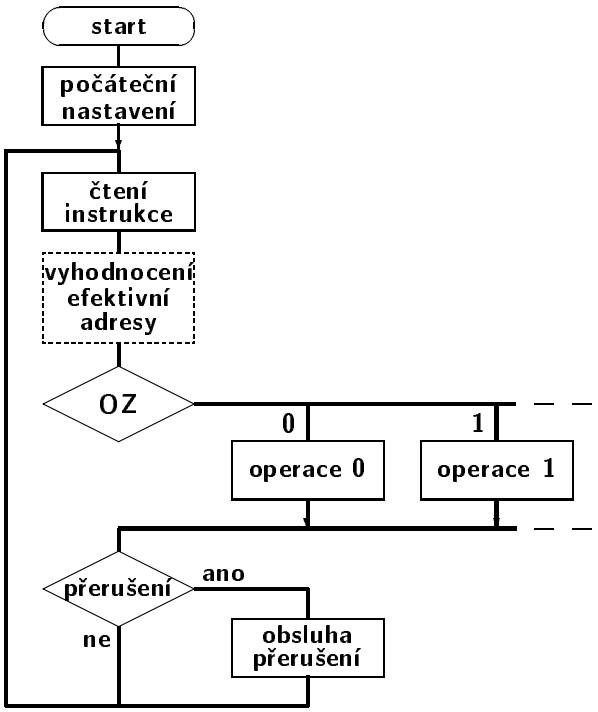
řídicí a stavové signály:
- **samostatné vodiče**
- **řídicí sběrnice**

---

hierarchie řadičů:



Řadič počítače: základní cyklus (instrukční cyklus)

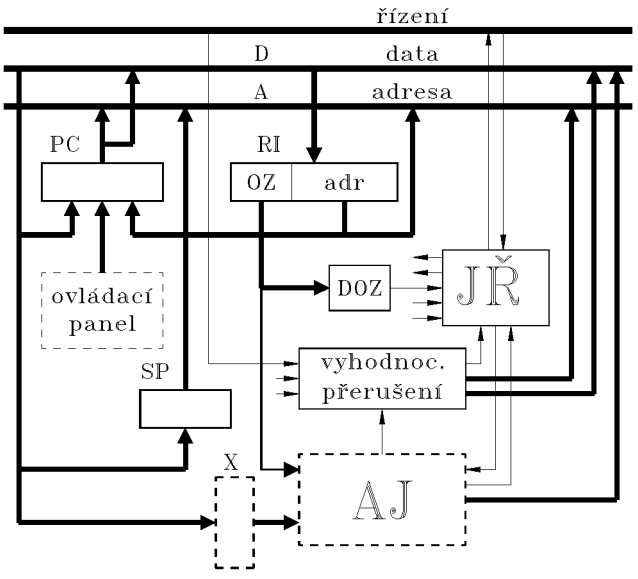


Poznámky k vývojovému diagramu:

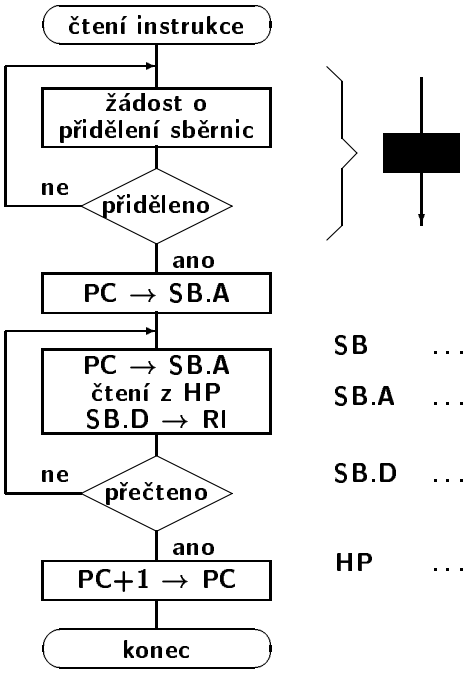
- **Vývojový diagram je třeba modifikovat tak, aby vyhovoval požadovanému chování konkrétního procesoru a co nejvíce vyhovoval jeho realizaci.**
- **V rámci realizace konkrétních operací se provádí také příp. čtení operandů a uložení výsledku; součástí těchto akcí může být i příp. vyhodnocení efektivních adres.**
- **Operační znak bývá dekódován postupně,** např.:
  - **skupina aritmetických a logických operací se dekóduje společně;**
  - **přečtou se operandy;**
  - **dekóduje se a provede se konkrétní operace.**
- **Někdy je třeba přejít na obsluhu přerušení dříve než je naznačeno ve vývojovém diagramu — chybí-li např. potřebná virtuální stránka v hlavní paměti, nelze instrukci přečíst** (natož dekódovat).
- **Test požadavku na přerušení a jeho obsluha bývají někdy zařazeny na začátek základního cyklu.**

Řadič počítače — příklad:

instrukce = 1 slovo (pojaté obecně, např. 32 b)
šířka datové sběrnice: 1 slovo



PC ...programový čítač, RI ...registr instrukcí, DOZ ...dekodér operačního znaku, JŘ ...jádro řadiče, SP ...ukazatel zásobníku, AJ ...aritm. j.



SB ... sběrnice

SB.A ... adresová sběrnice

SB.D ... datová sběrnice

HP ... hlavní paměť

$\chi \rightarrow$ SB.ψ $\Rightarrow$ vysílat signál „sběrnice obsazeny“

Některé aritmetické a logické operace a přesun:

operace typu

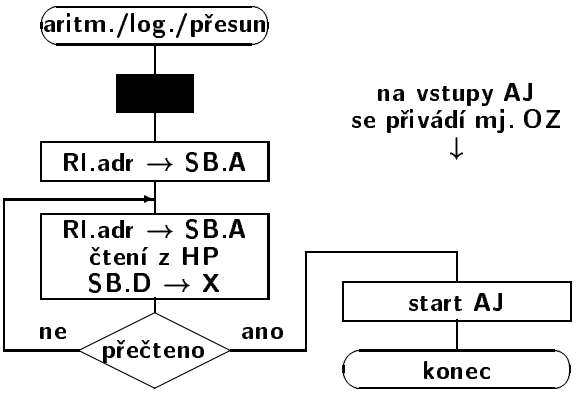
S $\odot$ <adr> → S

0 + <adr> → S (přesun)

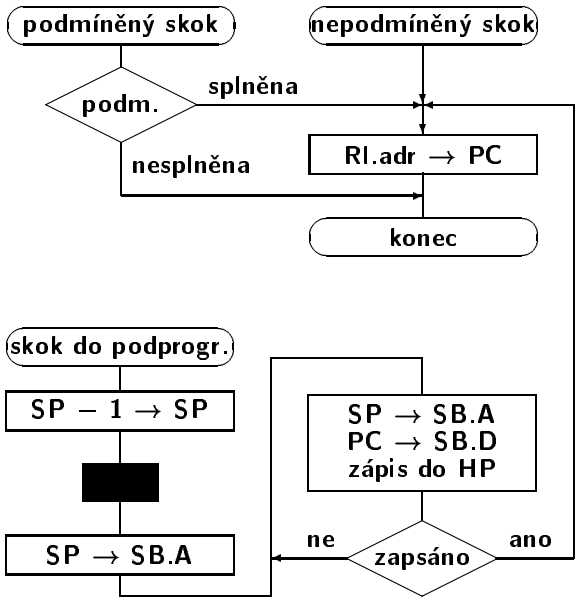
S ... střadač — obsažen v AJ

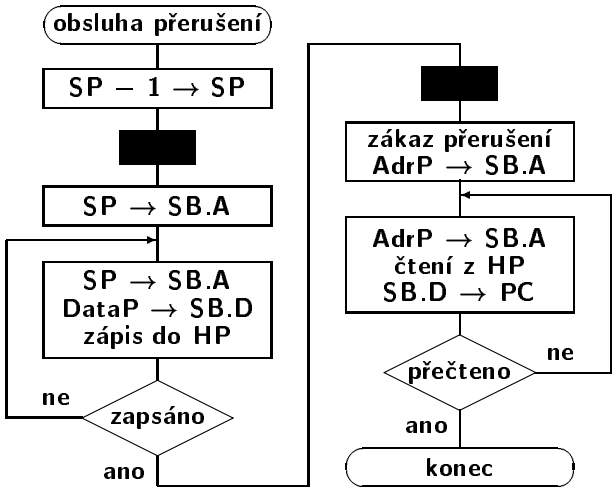
$\odot$ ... +, −, *and*, *or*, *xor* apod.

Tyto operace lze provést v 1 taktu.



Skoky:





DataP Data Přerušení (kontext): 1. PC
⋮
Umožňují přesnou identifikaci příčiny přerušení a znovuspuštění přerušeného programu (jeho pokračování).

AdrP Adresa, na níž je uložena Přerušovací adresa pro příslušnou příčinu přerušení.

Lze provést velmi rozmanité modifikace, např.:

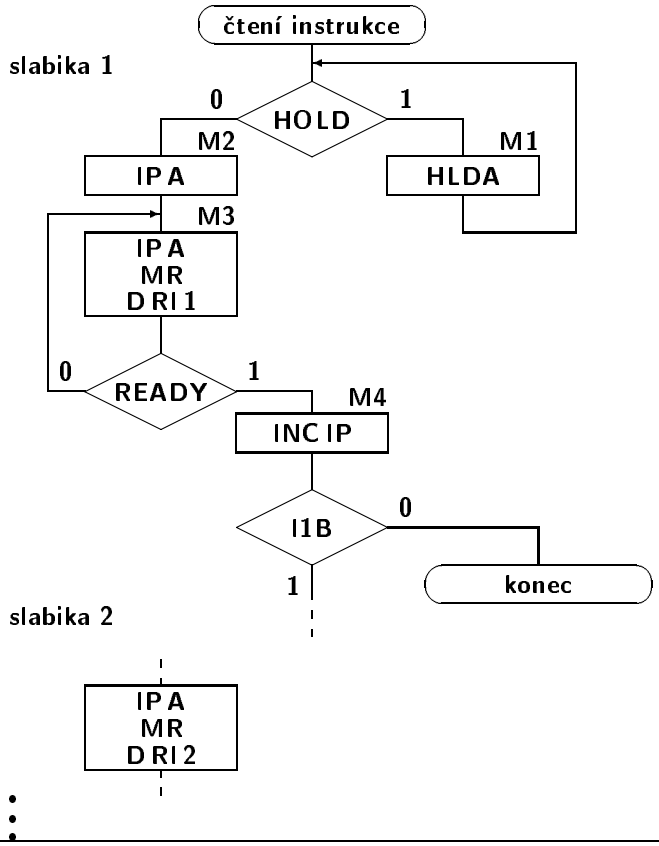
I. principiální modifikace:

1. **délka instrukce — proměnná** (1 nebo více slabik)
2. **šířka datové sběrnice: 8b = 1B** (1 slabika)
3. **funkci přidělovače sběrnic zastává procesor**
4. **reg. RI rozdělen na „podregistry" RI 1, RI 2, ...**

II. čistě formální modifikace:

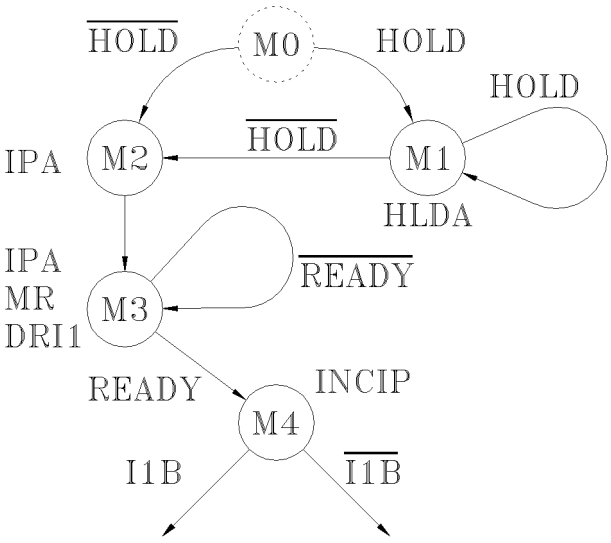
1. **IP** [Instruction Pointer] **místo PC**
2. **Místo dílčích operací (tzv. mikrooperací) i místo podmínek budou uváděny příslušné řídicí a stavové signály, např.:**

MR	**čtení z HP** [Memory Read]
IPA	**IP** $\rightarrow$ **SB.A**
INC IP	**inkrementace IP** (IP+1 $\rightarrow$ IP)
DRI 1	**SB.D** $\rightarrow$ **RI 1**
DRI 2	**SB.D** $\rightarrow$ **RI 2**
HLDA	**sběrnice volná** [HoLD Ackowledgement]
READY	**hotovo** (přečteno/zapsáno)
HOLD	**uvolnit sběrnice (žádost)**
I1B	**délka instrukce > 1B**
I2B	**délka instrukce > 2B**



Návrh řadiče (popř. jádra řadiče)

vývojový diagram — jiná forma grafu přechodů:

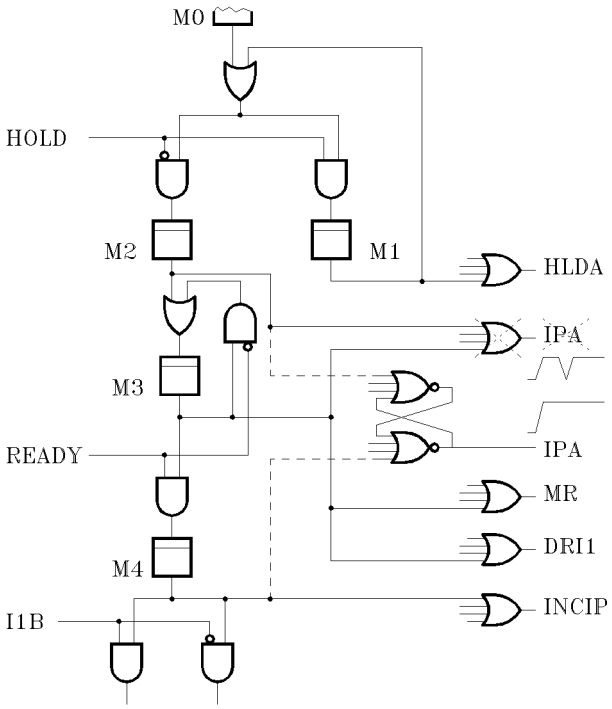


graf přechodů → sekvenční obvod — tzv.

„klasický" (nebo ***obvodový) řadič***

Řadič s řídicími řetězci

(jedna z možností realizace klasického řadiče)



mikroprogramovaný řadič

Provedení 1 operace (např. provedení instrukce, včetně přečtení, dekódování atd. — u řadiče počítače) **sestává z provedení řady dílčích operací — tzv. mikrooperací. Příkazy k provedení mikrooperací lze zapsat ve formě připomínající instrukce — tzv. mikroinstrukce.**
Ty tvoří tzv. mikroprogram, který lze uložit do tzv. řídicí paměti.
Lze navrhnout „klasický" řadič (nebo i jednodušší obvod)**, který bude zajišťovat postupné provádění mikroinstrukcí. „Počítač" takto řízený bude fungovat jako tzv. mikroprogramovaný řadič.**

mikroprogram — „nekonečný cyklus"
(viz základní cyklus počítače)

řadič počítače (jádro řadiče):
program
↓
instrukce ... **mikroprogram (1 průchod cyklem)**
↓
mikroinstrukce

soubor všech mikroprogramů daného procesoru:
mikroprogramové vybavení **[firmware]**

- **vertikální mikroprogramování:**
  - **krátké mikroinstrukce (např. 16b) obdobné instrukcím → čítač adres mikroinstrukcí**
  - **1 mikroinstrukce ... několik taktů**
- **horizontální mikroprogramování:**
  - **dlouhé mikroinstrukce (64 a víc bitů)**
  - **1 mikroinstrukce ... 1 takt**
  - **v mikroinstrukci přímo určeny řídicí signály**
  - **omezená volba adresy násl. mikroinstrukce**

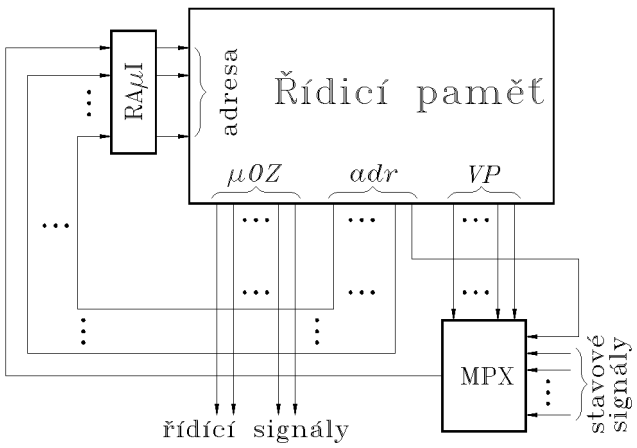
---

horizontální mikroprogramování

μOZ	adr	VP

- μOZ — **mikrooperační znak — hodnoty řídicích signálů**
- adr — **adresa následující mikroinstrukce**
- VP — **výběr podmínky**

struktura mikroprogramovaného řadiče — horizontální organizace



poslední bit adresy := hodnota vybrané podmínky

- **podmínka = některý stavový signál ⇒ větvení: adresa $\alpha\beta\cdots\gamma 0$ nebo $\alpha\beta\cdots\gamma 1$**
- **podmínka = poslední bit adr ⇒ nedochází k větvení**

Proudové zpracování
[Pipelining]

několik jednotek v kaskádě (srov. výrobní pás):
— každá jednotka provede část operace
— jednotky pracují současně

triviální případ — předčítání instrukcí:
jedna instrukce se provádí, další se čte a dekóduje

typická organizace:

jednotka	instrukce					
čtení instrukce	1	2	3	4	5	6
dekódování OZ		1	2	3	4	5
čtení operandů			1	2	3	4
provedení operace				1	2	3
uložení výsledku					1	2

čas →

ideální případ: každý takt dokončena 1 instrukce

konflikty:

- **datový: potřebná data dosud neuložena**
- **skokový: adresu skoku prozatím nelze určit**

nejjednodušší řešení: počkat

Počítače typu RISC

Jak navrhnout „rychlý“ procesor?

Řešení: Co „nejvýkonnější“ instrukce.

???

Statistika:

Př.: fa DEC: vývoj MicroVAX–32 ← VAX 11/780
98% instrukcí v typických úlohách (?) používá:
58% instrukcí zařazených do operačního kódu
15% mikroprogramového vybavení

Př.: fa Motorola (68000)

přesuny :	**cca 33%**	cca 33%
skoky a srovnávání :	**cca 35%**	cca 68%
$+ \; - \;$ *and* *or* :	**cca 11%**	cca 79%
posuvy :	**cca 3%**	cca 82%
násobení :	**cca 0,6%**	
dělení :	**cca 0,1%**	

Závěr: Velmi „výkonné“ instrukce se používají příliš málo!

Co s tím?

Co „nejtriviálnější“, ale co „nejrychlejší“ instrukce.

Počítače typu RISC

[Reduced Instruction Set Computers]

(Počítače s redukovaným souborem instrukcí)

charakteristické rysy:

- **Poměrně malý počet instrukcí** (≤ 128 ?), **a to „velmi jednoduchých“**
- **Velmi krátká doba provedení instrukce (1 instrukce zprav. dokončena v 1 taktu)**
- **„Klasický“** (tj. obvodově realizovaný) **řadič**
- **Proudové zpracování**
- **1 instrukce — 1 slovo**
- **Malý počet formátů instrukcí** (≤ 4)
- **Malý počet způsobů adresace** (≤ 4)
- **Velký počet registrů** (≥ 32)
- **Komunikace s hlavní pamětí: pouze instrukcemi „přesun“**
  (aritmetické/logické operace: pouze mezi registry)

„protipól“ počítačů typu RISC:

Počítače typu CISC

[Complex Instruction Set Computers]

(Počítače se „rozsáhlým“ souborem instrukcí)