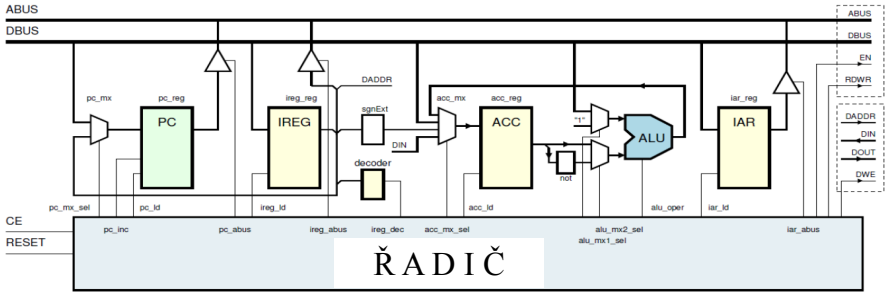
Řadiče

INP 2019
FIT VUT v Brně

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

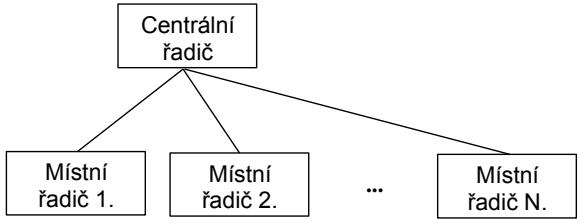


Co dělají a jak jsou realizovány řadiče „reálných“ procesorů?

Hlavní funkce řadiče

- interpretace instrukcí – dekódování a provedení
- krokování instrukcí – vytváření toku instrukcí
- řízení systémových procesů – přerušení, obsluha RVP (cache), řízení segmentace a stránkování, atd.

Řadič se obvykle realizuje jako **centrálně řízená soustava místních řadičů**, které mají svoje funkce rozdělené a specializované.



Soustava místních řadičů “reálného” počítače

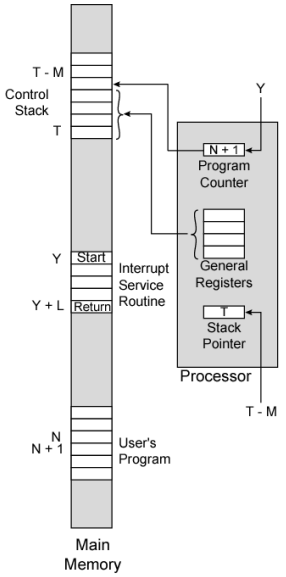
- řadič provedení instrukce ALU s pevnou řádovou čárkou včetně výpočtu efektivní adresy operandu, řízení algoritmu operace, vyhodnocení podmínky skoku apod.
- řadič provedení instrukce FPU s pohyblivou řádovou čárkou
- řadič přerušení
- řadič vstup-výstupních operací, který má u počítače s nezávislými I/O kanály podobu řadiče kanálu
- řadič paměti
- řadič kanálu přímého přístupu k paměti (DMA - Direct Memory Access)
- řadiče jednotlivých periferních zařízení
- řadič sběrnice
- atd.

Přerušení

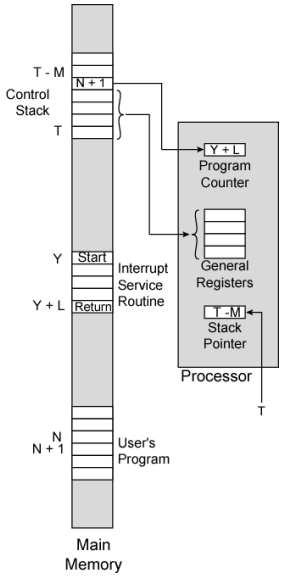
- Přerušení bylo u počítačů zavedeno pro ošetření nestandardních situací, jako
  - dělení nulou
  - chyba v datech přečtených z paměti
  - výpadek síťového napájení
  - připravenost diskové jednotky číst nebo zapsat blok dat
  - přeplnění disku síťového serveru
  - výpadek bloku dat RVP (cache miss)
  - atd.

- Má-li se přerušení správně obsloužit, musí se v první řadě správně identifikovat zdroj a typ přerušení, a pak spustit příslušná obslužná rutina.
- Klasifikace přerušení podle typu není zcela ustálená.

Princip přerušení



Přerušení programu po vykonání instrukce na adrese N.



Návrat z obsluhy přerušení

Př. Pokud byla stisknuta klávesa, procesor přeruší uživatelský program, uschová obsahy registrů a návratovou adresu na zásobník a obslouží klávesnici (rutina na adrese Y). Potom obnoví obsahy registrů a vrátí se do uživatelského programu.

Klasifikace přerušení

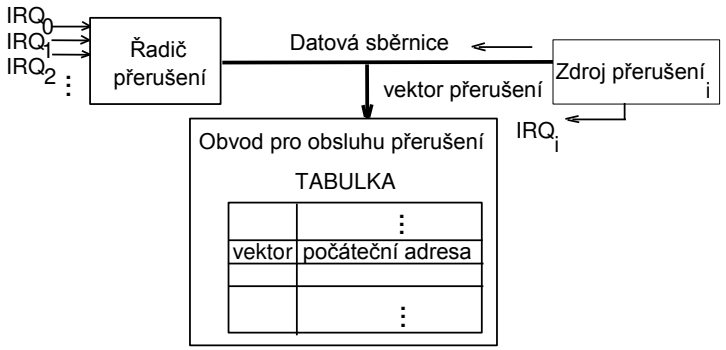
- Podle typu zdroje rozeznáváme přerušení
  - **programové**, vyvolané prováděním instrukcí, např. instrukcí INTn, přeplněním při sčítání, dělením nulou, neexistující adresou paměti apod.
  - **technické**, vyvolané poruchou některé jednotky, výpadkem napájení, překročením časového limitu (watchdog) atd.
  - **vstup-výstupní**, jako přerušení od časovače, připravenost V/V operace, zpráva z počítačové sítě aj.
- Přerušení můžeme rozdělit na
  - vnitřní a
  - vnější,
- nebo z hlediska jejich synchronismu s hodinovým taktem procesoru na
  - synchronní (trap), např. při krokování instrukcí
  - asynchronní (interrupt)
- nebo z hlediska naléhavosti rozlišují některé procesory přerušení
  - maskovatelná
  - nemaskovatelná.
- Zdrojům přerušení je přiřazena priorita – dle důležitosti.

Mechanismy přerušení

- Přerušení vyvolá zařízení žádající o obsluhu aktivací signálu IRQ (Interrupt Request). Žádosti vyhodnocuje řadič přerušení.
- Vlastní mechanismus přerušení může být založen na
  - vnucení adresy
  - vnucení instrukce
  - vnucení vektoru – nejobvyklejší způsob

Vnucení přerušovacího vektoru

- Každý zdroj přerušení generuje přerušovací číslo, zvané vektor přerušení (např. 8 bitů), které je ukazatelem do tabulky počátečních adres přerušovacích rutin.
- Obsahem tabulky mohou být např. 4 byty čísla segmentu a adresy uvnitř segmentu, nebo 8 bytů, které se vloží do tabulky přerušovacího deskriptoru (I 80 286 a další) a definují tak začátek obslužné rutiny.



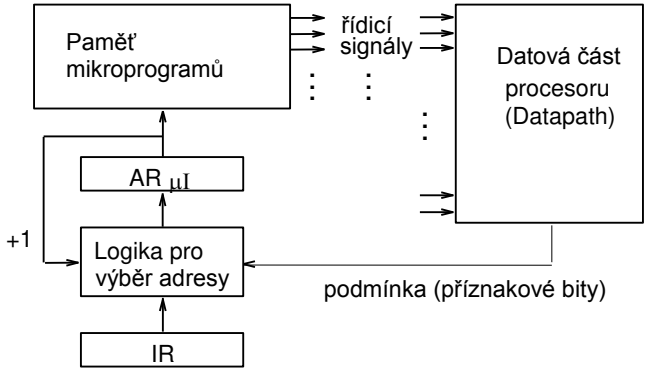
Implementace řadiče

- Obvodový řadič
  - Řadič je definován pomocí automatu a tento automat je implementován běžným způsobem.
  - viz první přednáška
- Mikroprogramový řadič
  - Namísto řadiče je vytvořen „malý procesor“, který vykonává mikroprogramy. Vykonáním jednoho mikroprogramu je provedena jedna instrukce procesoru.

Mikroprogramový řadič

- Konstrukce mikroprogramového řadiče a podpůrná disciplina mikroprogramování jsou velmi rozšířené postupy, které se po mnoho let staly převažující metodikou návrhu řadičů.
- Mikroprogramování je systematický postup, který za pomoci překladače mikroprogramů napsaných v mikroassembleru vytvoří binární obsah permanentní řídicí paměti, která je základem mikroprogramového řadiče.
- Převažující část problému syntézy řadiče operací procesoru je tak přesunuta na programovou úroveň, která byla v historii vybavena komfortními podpůrnými prostředky dříve než disciplina obvodového návrhu.
- Mikroprogram slouží jako interpret instrukce v reálném čase.

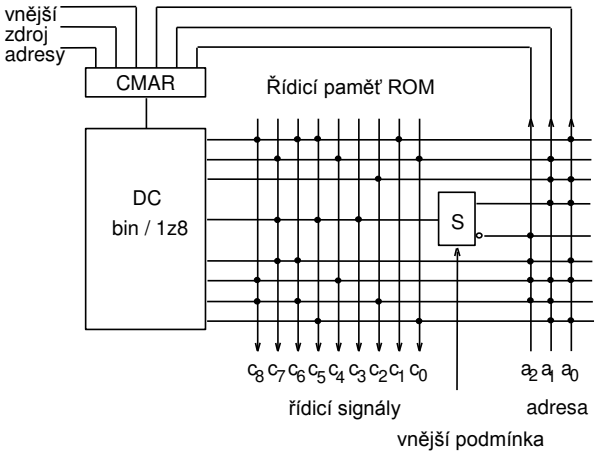
Základní schéma mikroprogramového řadiče



- Začátek mikroprogramu zajišťujícího provedení určité instrukce se určí logikou pro výběr adresy podle kódu operace v IR.
- Jednotlivé mikroinstrukce se čtou z paměti mikroprogramů podle obsahu adresového registru mikroinstrukce AR μI v nejjednodušším případě sekvenčně, tedy adresa příští mikroinstrukce se určí ze současné adresy inkrementací.
- Jednotlivé bity mikroinstrukce jsou řídicí signály pro datovou část procesoru, tj. ALU, a dále M a I/O.

Základní implementace mikrořadiče

(Wilkesovo schéma, 1951)



Registr CMAR je adresový registr řídicí paměti (Control Memory Address Register), přepínač S (demultiplexor) řízený vnější podmínkou určuje adresu příští mikroinstrukce; buď je to 011, nebo 100.

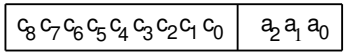
Paměť mikroprogramu a firmware

- Paměť mikroprogramů díky své pružnosti umožňuje poměrně rychlé a snadné modifikace, opravy a rozšiřování instrukčního souboru.
  - Dříve se jednalo o ROM, v poslední době s používají Flash paměti.
- Firmware = obsah této paměti (je to firemní záležitost).
- Další využití je pro diagnostické účely
  - Po startu systému se do paměti mikroprogramu uloží diagnostický mikroprogram, který velmi podrobně a rychle provedou diagnózu systému. Teprve po úspěšném testu se uloží do řídicí paměti mikroprogramy pro standardní instrukční soubor.

Nevýhody mikroprogramování

- Mikroprogramový řadič je obvykle pomalejší než obvodový řadič.
  - O 10% u HP 2116 (obvodový řadič) vs HP 2100S (mikroprogramový řadič).
  - Tato ztráta se však dalším technologickým vývojem brzy nahradila a výhody mikroprogramového řadiče na dlouhou dobu jednoznačně převážily.
- Zásah do mikroprogramů vyžaduje vysokou kvalifikaci a není to v žádném případě běžná uživatelská záležitost.
  - Změna mikroprogramů oproti firemní verzi se uplatňovala pouze ve výjimečných situacích, jako při řešení závažného selhání systému vlivem poruchy a neměla trvalý charakter.
  - U současných procesorů je paměť mikroprogramů pro uživatele nepřístupná.
- Mikroprogramy nejsou přemístitelné a optimalizace mikroprogramů z hlediska činnosti celého systému je náročná a dlouhodobá záležitost.
  - Může totiž snadno dojít k potlačení či přibrzdění některého podsystému, což se může projevit poklesem výkonnosti I/O podsystému, paměti, přerušovacího systému atd.

Formát Wilkesovy mikroinstrukce



funkce: řízení operací krokování sledu mikroinstrukcí

Základní dvě funkce mikroinstrukčního řadiče jsou *řízení operací*, resp. mikrooperací a *krokování sledu* mikroinstrukcí.

Kromě těchto funkcí musí mikroprogramový řadič obecně provádět tyto činnosti: *přiřadit každé instrukci mikroprogram* a *provádět nepodmíněné i podmíněné skoky*. K těmto funkcím se postupně přidaly další.

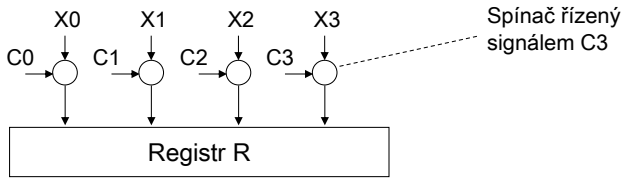
Obecný formát mikroinstrukce

zdroj 1	zdroj 2	ALU	cíl	M R/W	konst	různé	CC	adresa	modadr

Mikroinstrukce je tříadresová. Pole *zdroj 1* a *zdroj 2* určují vstupní operandy, pole *ALU* určuje operaci, *cíl* je adresa pro výsledek operace, polem *M R/W* se může spustit operace paměti M, *konst* je pole pro výběr konstant z permanentní paměti, zvláštní situace se mohou ošetřit v poli *různé*, *adresa* je pole udávající adresu příští mikroinstrukce, která se může dále modifikovat polem *modadr*. Tento formát je s jistými modifikacemi použit ve všech mikroprogramových řadičích.

Kódování mikroinstrukcí

Př. Zápis do registru R ze 4 nezávislých zdrojů X0 až X3



a) Nekódovaný formát (horizontální) – 4 bity

…	C0	C1	C2	C3	…	
	1	0	0	0		R ← X0
	0	1	0	0		R ← X1
	0	0	1	0		R ← X2
	0	0	0	1		R ← X3
	0	0	0	0		nop

nop – instrukce neprovádí zápis do R

b) Kódovaný formát (vertikální) – 3 bity

…	K0	K1	K2	…	
	0	0	1		R ← X0
	0	1	0		R ← X1
	0	1	1		R ← X2
	1	0	0		R ← X3
	0	0	0		nop

Výhoda: nižší počet bitů
Nevýhoda: nutnost dekódovat, tj. větší zpoždění

Poznámky

- Mikroprogramový řadič je často implementován jako řetězená jednotka
  - Musí se řešit stejné problémy jako u řetězeného zpracování na úrovni instrukcí.
- Mikroprogramová realizace instrukcí je typická pro procesory se složitým instrukčním souborem (CISC).
  - Pružnost této koncepce se přímo nabízí pro doplňování dalších složitějších funkcí. Typickou aplikací je realizace mikrodiagnostického systému, který se spustí po startu počítače a provede základní testy paměti M, procesoru, periferií apod.
- Často doplňovanými funkcemi jsou mikroprogramové realizace typických instrukčních posloupností generovaných překladači vyšších jazyků.
  - Tyto prostředky přinášely zrychlení provádění programů řádově v jednotkách až desítkách procent.