Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

ŠTÁTNY PEDAGOGICKÝ ÚSTAV
NATIONAL INSTITUTE FOR EDUCATION

Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Počítačové systémy 3

Predmet: Počítačové systémy

Línia: Vlastný odborový kontext informatiky a informatickej výchovy

Operačný program VZDELÁVANIE

EURÓPSKA ÚNIA
Európsky sociálny fond

Európska únia
Európsky sociálny fond

Moderné vzdelávanie pre vedomostnú spoločnosť/Projekt je spolufinancovaný zo zdrojov ES

Počítačové systémy 3

Identifikácia modulu

Aktivita projektu:	1.2 Vzdelávanie nekvalifikovaných učiteľov informatiky na 2. stupni ZŠ a na SŠ
Línia aktivity:	Vlastný odborový kontext informatiky a informatickej výchovy
Predmet:	Počítačové systémy

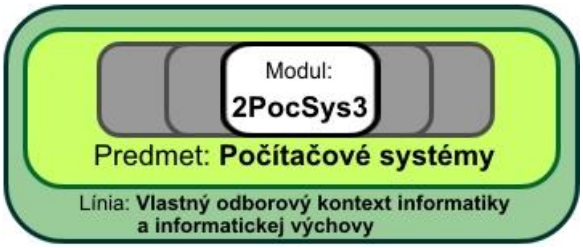
Garant predmetu:

RNDr. Peter Gurský, PhD.
ÚINF PF UPJŠ, Košice
peter.gursky@upjs.sk

Autori:

RNDr. Jozef Jirásek, PhD.
ÚINF PF UPJŠ, Košice
RNDr. Richard Ostertág, PhD.
KI FMFI UK, Bratislava

Zaradenie modulu



Študijný materiál modulu nadväzuje na prvú a druhú časť predmetu Počítačové systémy. Spolu pokrývajú obsah základného vysokoškolského kurzu Princípy a architektúry počítačov. Využíva poznatky z predmetov Programovanie, Operačné systémy, Internet a rozširuje vedomosti, získané v časti Základy hardvérového a softvérového vybavenia počítača predmetu Digitálna gramotnosť učiteľa.

Abstrakt modulu

Nadviažeme na opis činnosti procesora z druhého modulu predmetu Počítačové systémy a vysvetlíme funkciu zbernice ako komunikačného prepojenia medzi jednotlivými súčasťami počítača. Opíšeme jej štruktúru a podrobnejšie sa budeme zaoberať komunikáciou procesora s operačnou pamäťou, komunikáciou procesora so vstupno-výstupnými zariadeniami, funkciou radičov zariadení, komunikáciou prostredníctvom portov, mechanizmu prerušenia a priameho prístupu do pamäte. Predstavíme pamäťovú architektúru počítačového systému, pričom opíšeme realizáciu operačnej pamäte vrátane konštrukčných princípov dynamických RAM pamäťových buniek. Doplníme opis techník zvyšujúcich rýchlosť a efektívnosť využívania operačnej pamäte – konkrétne spôsoby využitia rýchlych vyrovnávacích pamätí (cache) a virtuálnej pamäte. Ukážeme funkcie ovládačov zariadení a ich začlenenie do operačného systému. V ďalšej časti urobíme historický prehľad vývoja počítačových systémov a poukážeme na zásadné myšlienky a udalosti, ktoré ho ovplyvnili. Z historického prehľadu prejdeme do prehľadu súčasných architektúr počítačových systémov, uvedieme špecifiká paralelných a distribuovaných systémov. Spomenieme aj počítačové systémy, ktoré nie sú riadené programom, ale tokom údajov a predstavíme možné architektúry počítačových systémov budúcnosti.

Obsah

Úvod

Predmet Počítačové systémy sa zameriava na základné konštrukčné princípy a funkčné mechanizmy počítačových systémov. Cieľom tejto časti je priblížiť spôsoby komunikácie medzi procesorom, operačnou pamäťou a vstupno-výstupnými zariadeniami.

K predstave o realizácii a funkciách procesora z predchádzajúcich častí bude podaný bližší opis realizácie operačnej pamäte a komunikačných rozhraní k vstupno-výstupným zariadeniam, mechanizmu prístupu procesora do pamäte, komunikácie s radičmi periférnych zariadení prostredníctvom portov, mechanizmu prerušenia a priameho prístupu periférnych zariadení do pamäte.

Absolvent by mal dostať konkrétnu predstavu o realizácii uvedených mechanizmov na zjednodušenom modeli a prehľad o princípoch ich realizácie v aktuálnych zariadeniach. Cieľom je tiež získať orientáciu v základných odborných pojmoch a odlíšiť ich od slangových výrazov.

Cieľom druhej časti je získať prehľad o historických medzníkoch vývoja architektúr počítačových systémov, ich súčasný stav a perspektívy ďalšieho rozvoja.

Vzhľadom na obmedzený rozsah študijného materiálu nie je možné prezentovať všetky vymedzené oblasti do patričnej hĺbky, vhodnej pre absolvovanie základného vysokoškolského kurzu odboru informatika. Materiál je možné použiť skôr na orientáciu a podnety pre štúdium. Pre vlastné štúdium odporúčame štandardnú vysokoškolskú učebnicu [1], prípadne alternatívne učebnice [2,3,4]. Slovenské (ani české) verzie zatiaľ neexistujú. Funkcia hardvérových rozhraní je veľmi dobre (až nad rámec štandardov) vysvetlená v českom preklade odborného textu [5]. História počítačov je v súčasnosti veľmi dobre spracovaná na Wikipédii Internetu, odporúčame aj českú publikáciu [16].

Materiál nepokrýva všetky fakty a pojmy vymedzenej problematiky. Sústreďuje sa skôr na priblíženie vybraných konkrétnych problémov a ich riešení na modelových zjednodušených príkladoch. Obsahuje tiež prvky, ktoré (po didaktickej úprave) je možné použiť aj na nižších stupňoch vzdelávania, prípadne v záujmových krúžkoch.

Vstupné vedomosti

Požadované prerekvizity

Predpokladáme absolvovanie predchádzajúcich modulov Počítačové systémy 1 a Počítačové systémy 2, pre prvotnú orientáciu aj Základy hardvérového a softvérového vybavenia počítača predmetu Digitálna gramotnosť učiteľa.

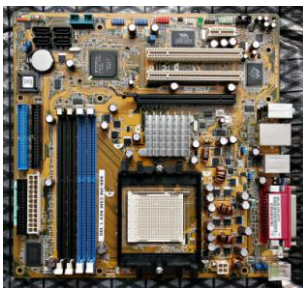
Predpokladané vstupné vedomosti, skúsenosti a zručnosti

Účastník má predstavu o spôsobe realizácie výpočtu v procesore pomocou logických obvodov, rozumie mechanizmu spracovania strojovej inštrukcie v jeho jednotlivých štádiách. Má tiež predstavu o realizácii jednoduchej (statickej) pamäte s priamym prístupom a spôsobe jej využitia v procese spracovania inštrukcií pre uchovanie údajov aj pre uchovanie postupností kódov inštrukcií.

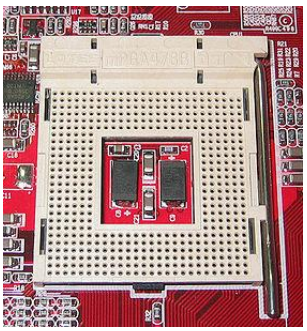
Preverenie vstupných vedomostí

Funkcia procesora - základné pojmy: register, aritmeticko-logická jednotka, riadenie údajového toku, pamäť inštrukcií, pamäť údajov, adresácia, strojový cyklus, inštrukčný cyklus, typy strojových inštrukcií a spôsob ich realizácie.

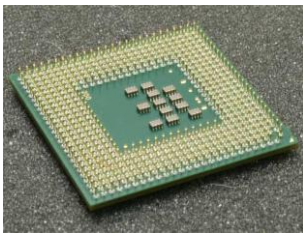
Vstupným testom by mohlo byť opísanie jednotlivých fáz spracovania niektorej konkrétnej strojovej inštrukcie (napr. MOV AX,[100] alebo CALL 500).



Základná doska



Zásuvka (478 pólová) na pripojenie procesora typu PGA (Pin Grid Array)



Procesor s vývodmi na pripojenie k systémovej zbernici

V súčasnosti sa častejšie používa pripojenie procesorov pomocou kontaktných plôch LGA (Land Grid Array). Na obrázku je zásuvka LGA 1366 a sú viditeľné aj prepojenia systémovej zbernice.

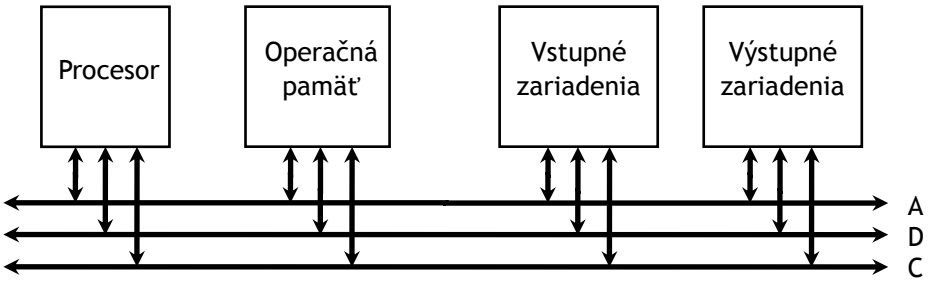
1 Súčasti počítača a komunikácia medzi nimi

Predchádzajúci modul nám predstavil základnú jednotku počítača, určenú pre spracovanie inštrukcií a automatizovaný výpočet – **procesor** (stretnete sa aj s označením **CPU** – Central Processing Unit – centrálna procesorová jednotka). Efektívne využitie samostatného procesora však nie je možné. Na prezentáciu výsledkov výpočtu, resp. na zadávanie vstupných parametrov výpočtu a jeho programu potrebujeme **periférne** (alebo tiež **vstupno-výstupné - V/V) zariadenia.** Na uchovanie údajov, ktoré presiahnu kapacitu registrov procesora, resp. jeho údajovú a inštrukčnú (vyrovnávaciu) pamäť, je určená **operačná (hlavná) pamäť** (*main memory*). Tvorí ju indexované pole 8-bitových záznamov – **bajtov** s priamym prístupom podľa indexu – **adresy** (už sme sa s ňou stretli tiež v predchádzajúcom module). Zjednodušene možno povedať, že počítač tvorí procesor, operačná pamäť, periférne zariadenia a prepojenia medzi nimi.

Prepojenia medzi súčasťami počítača sú v súčasnosti realizované prevažne prostredníctvom základnej (matičnej) dosky (*mainboard, motherboard*). **Základná doska** pozostáva z množstva (plošných) vodivých spojov, zásuviek a konektorov, do ktorých sú pripojené okrem procesora aj ostatné súčasti počítača. Procesor ich riadi, zisťuje ich stav, posiela aj prijíma údaje pomocou elektrických signálov cez vodivé spoje základnej dosky. Jeho komunikačná schopnosť je však obmedzená počtom jeho vývodov. Priame prepojenie procesora s každým zariadením zvlášť by vyžadovalo veľký počet vývodov, komplikovalo by jeho návrh, zdražilo výrobu, obmedzilo rozširovanie. Efektívna komunikácia pomocou relatívne malého počtu vývodov sa však dá uskutočniť prepojením zariadení zbernicovým spôsobom.

Systémová zbernica a jej pripojenie k procesoru

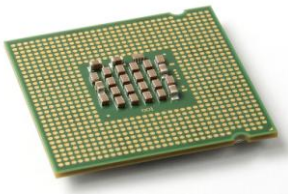
Zbernica (*bus*) je tvorená niekoľkými paralelnými vodičmi (resp. prepojeniami na základnej doske), umožňujúcimi prenos elektrických signálov, ku ktorým je pripojených viacero (prípadne aj všetky) súčastí počítača. Oproti priamej komunikácii (samostatnými komunikačnými prepojeniami – komunikačnými kanálmi - medzi dvojicami súčastí počítača) tak vzniká možnosť posielať signál naraz viacerým príjemcom. Ušetríme na počte prepojení a získame možnosť pružne pridávať a odoberať ďalšie zariadenia. Nevýhodou zbernicového prístupu je nižšia priepustnosť komunikácie a zložitejšia organizácia práce – stále musí byť určené, kto môže na zdieľanom vodiči vysielať.



Zbernica, ku ktorej je pripojený a ktorú riadi procesor, sa nazýva **systémová zbernica.** Obyčajne sa skladá z niekoľkých vyhradených vodičov na paralelný prenos adresy – **adresová zbernica** (*address bus*)**,** niekoľkých vodičov na paralelný prenos údajov - **údajová zbernica** (*data bus*) a ďalších vodičov na riadenie komunikácie a zisťovanie stavu ostatných zariadení - **riadiaca zbernica** (*control bus*).

Adresovou zbernicou posiela procesor adresy. Každý vodič prenesie (signálom) informáciu o jednom bite adresy. Počet vodičov adresovej zbernice (tiež **šírka zbernice**) takto určuje rozsah adries, ktoré môže procesor použiť. Napríklad systém s 32 vodičmi adresovej zbernice môže preniesť 32-bitovú adresu, a teda vie vybrať ktorékoľvek z 2^{32} (4 294 967 296) adresových miest. Ak ňou adresujeme operačnú pamäť, vytvára adresovateľný priestor 4 GiB. Šírka údajovej zbernice zase určuje, koľko bitov údajov možno súčasne preniesť v jednom takte procesora.

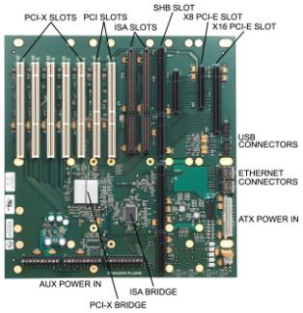
Riadiacou zbernicou riadi procesor prenos údajov. Jednotlivé vodiče riadiacej zbernice sú určené na signalizáciu špecifických udalostí. Väčšinou sa využívajú v definovanom smere – teda buď od procesora (výstupné signály), resp. k procesoru (vstupné signály). Súčasťou riadiacej zbernice je aj vstupný signál RESET, ktorým sa procesor nastaví do definovaného počiatočného stavu, signál od generátora systémových hodinových pulzov, vodiče s napájacím napätím a uzemnením.



Pentium 4 – kontaktné plochy namiesto "nožičiek"

Aktivita 1	Ukážka základnej dosky počítača, pripojenia procesora, pamätí a radičov vstupno-výstupných zariadení. Ukážka realizácie spojov na doske a možností pripojenia ďalších radičov pomocou štandardizovaných zásuviek a konektorov.

Procesor komunikuje s ostatnými súčasťami počítača len prostredníctvom signálov po systémovej zbernici. Na uchovanie kompatibility softvérového vybavenia pre jednotlivé architektúry (napr. x86, SPARC, PowerPC, ARM) sú špecifikované mechanizmy komunikácie, ktoré potom môže programátor využiť na úrovni strojových inštrukcií, prípadne sprostredkovane na vyšších úrovniach riadenia výpočtu. Mechanizmy sú navrhnuté tak, že sa ich hardvérová realizácia môže meniť a rozvíjať v súlade s technickým a technologickým pokrokom, ale softvérový pohľad ostáva nezmenený (vytvorené programové vybavenie ostáva funkčné).

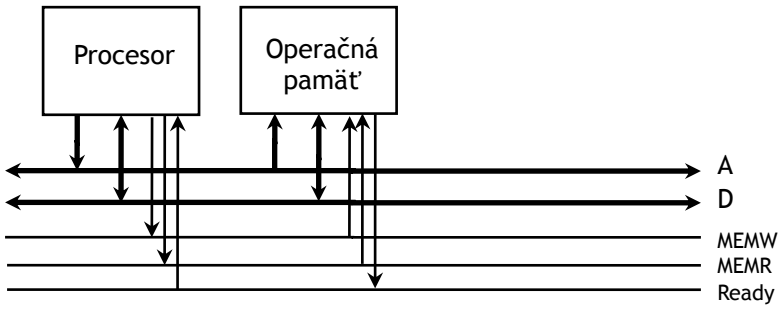


Základná doska so zásuvkami rôznych typov rozhraní

Radiče a adaptéry periférnych zariadení, pamäť a procesor nie je možne vymieňať pri zapnutom počítači. Je potrebné odpojiť aj napájaciu šnúru - inak ostáva základná doska aj pri vypnutom počítači pod napätím.

Zápis a čítanie z operačnej pamäte

Komunikácia s operačnou pamäťou je kľúčová pre zložitejšie výpočty procesora, ale aj pre prácu s rozsiahlejšími súbormi údajov, či uloženie dlhšieho programu. Procesor ju riadi signálmi **MEMW** (Memory Write) a **MEMR** (Memory Read) riadiacej zbernice. Nábehom príslušného signálu žiada pamäť, aby zareagovala a vykonala zvolenú operáciu. Pokiaľ pamäť stihne vykonať činnosť v dohodnutom počte taktov (hovoríme o synchrónnom prenose), procesor nevyžaduje potvrdenie o ukončení operácie. Ak ale rýchlosť vykonania operácie je premenlivá (napr. prístup k niektorým adresám je urýchlený vyrovnávacou pamäťou resp. pamäť je mapovaná do radiča V/V zariadenia), je možné použiť asynchrónny prenos. Vtedy pamäť signalizuje ukončenie operácie signálom **Ready** riadiacej zbernice.



Príklad zápisu obsahu registra AX do pamäte na adresu 80 (inštrukcia MOV [80],AX):

Procesor nastaví na adresovú zbernicu úrovne odpovedajúce hodnote 80.

Procesor nastaví na údajovú zbernicu úrovne podľa obsahu registra AX.

Procesor nastaví signál MEMW Memory Write riadiacej zbernice do aktívnej úrovne.

Na prechod signálu MEMW do aktívnej úrovne reaguje operačná pamäť, ktorá dekóduje adresu z adresovej zbernice. Na dekódované miesto v pamäti privedie signály údajovej zbernice, čím zmení jeho aktuálny obsah na stav podľa zbernice – odpovedajúci pôvodne obsahu registra AX.

Po úspešnom zápise v prípade asynchrónneho prenosu pamäť signalizuje signálom Ready riadiacej zbernice, že prenos bol ukončený a obsah zberníc je možné zmeniť.

Procesor zruší signál MEMW.

Pamäť zruší signál Ready

Príklad čítania z pamäťového miesta 80 do registra AX: (inštrukcia MOV AX,[80]):

Procesor nastaví na adresovú zbernicu úrovne odpovedajúce hodnote 80.

Procesor nastaví signál MEMR Memory Read riadiacej zbernice do aktívnej úrovne.

Na prechod signálu MEMR do aktívnej úrovne reaguje operačná pamäť, ktorá dekóduje adresu z adresovej zbernice. K údajovej zbernici potom privedie operačná pamäť signály, nastavené podľa aktuálneho stavu dekódovaného miesta.

Pri asynchrónnom prenose signalizuje pamäť procesoru signálom Ready riadiacej zbernice, že požadovaný údaj je nastavený na údajovej zbernici.

Procesor zapíše stav údajovej zbernice do registra AX.

Procesor zruší signál MEMR.

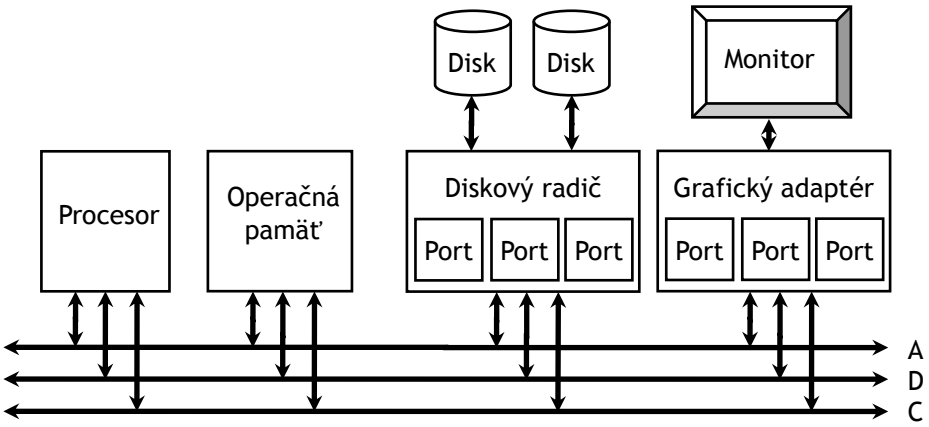
Pamäť zruší signál Ready.

V skutočnosti prenosy prebiehajú po väčších celkoch, pričom sa využívajú rôzne vyrovnávacie pamäte. Podrobnejšie sa nimi budeme zaoberať v ďalšej časti.

Radiče periférií, porty, komunikácia procesora s portami

Komunikácia s perifémymi zariadeniami je komplikovaná pestrosťou zariadení, ktoré chceme k procesoru pripojiť. Mechanizmus by mal byť natoľko univerzálny, aby umožnil komunikáciu aj so zariadeniami, ktoré budú vyrobené v budúcnosti. V súčasných počítačoch rieši prispôsobenie signálov procesora potrebám konkrétneho periférneho zariadenia jeho **radič** (controller), resp. **adaptér** signálov.

Radič komunikuje na jednej strane štandardným spôsobom prostredníctvom systémovej zbernice s procesorom a na druhej strane špeciálnymi signálmi s konkrétnym zariadením (resp. viacerými zariadeniami určitého typu).



Komunikácia radiča na strane systémovej zbernice je riešená pomocou portov**. Port** je v podstate register radiča, pripojený na údajovú zbernicu. Je možné do neho ukladať aktuálny stav údajovej zbernice, alebo naopak na základe jeho obsahu nastaviť úrovne signálov na údajovej zbernici.

Riadenie prístupu do portu vykonáva radič zariadenia na základe signálov riadiacej zbernice. Súčasne tiež reaguje na aktuálny stav portov vytvorením patričných signálov pre pripojené zariadenie, alebo naopak, na základe signálov od zariadenia ich stav aktualizuje.

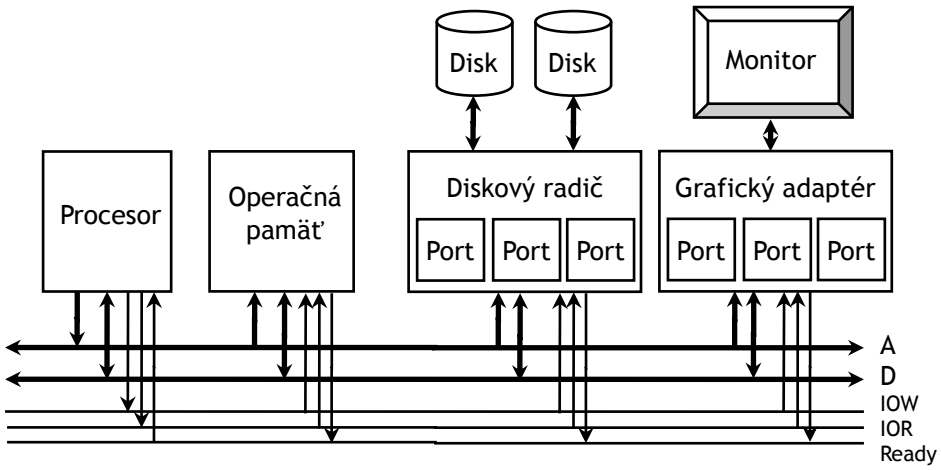
Porty bývajú riadiace, stavové a údajové. Do **riadiacich portov** posiela procesor hodnoty, ktoré radič transformuje do signálov pre zariadenie. Do **stavových portov** zapisuje radič stav komunikácie so zariadením, procesor môže tento stav čítať. Do **údajových portov** môže zapisovať i čítať procesor aj radič. Smer toku údajov určí procesor riadiacou zbernicou. Radič obsahuje obyčajne viacero portov.

Každý port je jednoznačne identifikovaný svojim číslom. **Číslo portu** musí byť priradené pri štarte systému tak, aby ho žiadne dva porty nemali rovnaké (aj v prípade, že pripojíme dva identické radiče). Kedysi sa kolízia čísel portov riešila manuálne nastavením v BIOS-e pri štarte systému. V súčasnosti je súčasťou systému automatickej konfigurácie PnP (Plug and Play). Aktuálne priradenia čísel portov radičom je možné zistiť pomocou programov pre správu počítača.

V systémoch MS Windows použite Správcu zariadení (Device Manager) a v pohľade na prostriedky podľa typu otvorte záložku Vstupno-výstupné (IO). Získate prehľad o priradení čísel portov jednotlivým radičom zariadení.

Jednoznačné číslovanie portov umožňuje pohľad na ne ako na súvislý **adresovateľný priestor portov** (*I/O address space*). Prístup k neobsadeným miestam je možné riešiť dohodnutým spôsobom. Niektoré architektúry procesorov používajú spoločný adresovací priestor pre pamäť aj V/V porty (*memory mapped I/O*). Výhodou je, že potom nepotrebujeme špeciálne inštrukcie pre čítanie a zápis z portov (IN, OUT), ale vystačíme len s inštrukciami typu MOV.

Mechanizmus zápisu a čítania z portov je z pohľadu procesora veľmi podobný mechanizmu prístupu do pamäte. Číslo portu signalizuje procesor cez adresovú zbernicu (podobne ako adresu pamäťovej bunky pri práci s operačnou pamäťou). Prenos údajov prebieha cez údajovú zbernicu. Riadiacimi signálmi **IOW** (I/O Write) a **IOR** (I/O Read) určí procesor smer toku údajov (zápis, čítanie – z pohľadu procesora) a oznamuje všetkým na zbernici, že bude prebiehať komunikácia s portami (na rozdiel od signálov MEMW a MEMR, ktoré za veľmi podobných podmienok signalizujú prístup do pamäte). V synchronizovaných systémoch sa na prenos čaká nastavený počet taktov časovača zbernice. Asynchrónne (hlavne v prípade pripojenia zariadení rôznych rýchlostí) je možné prenos ukončiť signálom **Ready** riadiacej zbernice, ktorý potvrdí ukončenie prenosu zo strany radiča.



Príklad zápisu obsahu registra AX do portu 80: (inštrukcia OUT 80,AX):

Procesor nastaví na adresovú zbernicu úrovne odpovedajúce hodnote 80.

Procesor nastaví na údajovú zbernicu úrovne podľa obsahu registra AX.

Procesor nastaví signál IOW riadiacej zbernice do aktívnej úrovne.

Na prechod signálu IOW do aktívnej úrovne reagujú všetky pripojené radiče. Každý radič porovná hodnotu na adresovej zbernici s rozsahom čísel portov, ktoré spravuje.

Radič, ktorý zistil prítomnosť portu s číslom 80 nastaveným na adresovej zbernici, pripojí tento port k údajovej zbernici a zapíše jej stav (obsah registra AX) do príslušného registra portu.

Po úspešnom zápise radič signalizuje signálom Ready riadiacej zbernice, že prenos bol ukončený a obsah všetkých zberníc je možné zmeniť.

Procesor zruší signál IOW.

Radič zruší signál Ready.

Príklad čítania z portu 80: (inštrukcia IN AX,80):

Procesor nastaví na adresovú zbernicu úrovne odpovedajúce hodnote 80.

Procesor nastaví signál IOR riadiacej zbernice do aktívnej úrovne.

Na prechod signálu IOR do aktívnej úrovne reagujú všetky pripojené radiče. Radič porovná hodnotu na adresovej zbernici s rozsahom čísel portov, ktoré spravuje.

Radič, ktorý zistil prítomnosť portu s číslom 80 nastaveným na adresovej zbernici, pripojí tento port k údajovej zbernici a nastaví ju podľa obsahu príslušného registra portu.

Radič signalizuje signálom Ready riadiacej zbernice, že požadovaný údaj je nastavený na údajovej zbernici.

Procesor zapíše stav údajovej zbernice do registra AX.

Procesor zruší signál IOR.

Radič zruší signál Ready.

Mechanizmus prerušenia a jeho využitie

Procesor vykonáva postupne inštrukcie zadané programátorom vopred do operačnej pamäte. Aj komunikáciu s ostatnými zariadeniami musí programátor predvídať a zapísať v programe (inštrukciami IN a OUT). Ako teda dosiahnuť to, že procesor zareaguje aj na nepredvídané udalosti ?

Jednou možnosťou by bolo pri vzniku udalosti prepísať program v operačnej pamäti. To ale spôsobuje mnoho konkurenčných problémov a je ťažko realizovateľné (robí sa len pre údajové prenosy, ktoré spomenieme v nasledujúcej kapitole). Dá sa tiež periodicky sledovať stav zariadení zo stavových portov a reagovať na ich zmenu. V zložitejších systémoch je však táto technika (*polling*) veľmi neefektívna.

Riešením je **mechanizmus prerušenia** (*interrupt*), ktorý je súčasťou väčšiny súčasných počítačových systémov. Jeho úlohou je reagovať na signály od vstupno-výstupných zariadení zmenou riadenia výpočtu. Je hlavným nástrojom na zabezpečenie interaktivity výpočtov, preemptívneho multitaskingu, viacpoužívateľských systémov, riadenia v reálnom čase, medzipočítačovej komunikácie a iných techník.

Cieľom mechanizmu je na základe signálu od radičov vstupno-výstupných zariadení prerušiť vykonávanie aktuálnej postupnosti príkazov, vykonať určenú procedúru a po jej skončení sa opäť vrátiť k vykonávaniu pôvodného programu.

Predpokladom úspešnej realizácie je samozrejme pripraviť procedúru vopred v prístupnej pamäti počítača. Aby bola možnosť reakcie na rôzne udalosti, adresy procedúr na ich ošetrenie sa zhromažďujú v **tabuľke popisovačov prerušení** (*interrupt descriptor table*). Tabuľka je v určenej časti operačnej pamäte (v architektúre x86 je jej adresa uchovávaná v IDTR registri) a obsahuje adresy procedúr v záznamoch pevnej dĺžky. Každá udalosť je charakterizovaná jednoznačným číslom tzv. **vektorom prerušenia**. Číslo (vektor) prerušenia udáva pozíciu záznamu v tabuľke, ktorý obsahuje adresu odpovedajúcej procedúry v operačnej pamäti. Čísla prerušení podliehajú štandardizácii, aby sa dosiahla kompatibilita vytvoreného softvéru.

Celý proces začína signálom žiadosti o prerušenie **INTR** (*interrupt request*). Je signálom riadiacej zbernice a posiela ho radič zariadenia, ktoré chce prerušenie vyvolať. Procesor nesleduje signál stále. Vyhodnocuje ho len na konci vykonávania každej inštrukcie (prerušenie výpočtu uprostred strojovej inštrukcie nie je možné).

Aby bolo možné zabrániť prerušeniam dôležitých častí programu (napríklad vykonávaniu procedúr na obsluhu iných prerušení), je možné pozastaviť akceptovanie signálu žiadosti o prerušenie. Zákaz sa obyčajne zadáva príznakom v stavovom, resp. príznakovom registri procesora (v architektúre x86 je to príznak IF – Interrupt-Enable Flag).

Zakazovanie (**maskovanie**) prerušení je dôležité pri činnostiach, ktoré je potrebné vykonať celé (tvoria tzv. transakcie). Napríklad, ak sa zmena obsahu vektora prerušení v tabuľke vykonáva vo viacerých krokoch, akceptovanie prerušenia pred ukončením celej zmeny môže spôsobiť skoky programu na nedefinované adresy. Tiež na prepnutie zásobníka potrebujeme dve inštrukcie (prepísanie selektora segmentu SS a prepísanie ukazovateľa na vrchol zásobníka SP), ktoré nemôžeme v polovici prerušiť.

Ak sú prerušenia povolené, potvrdí procesor žiadosť ďalším signálom riadiacej zbernice - signálom **INTA** (*interrupt acknowledge*) a očakáva, že zariadenie signalizujúce prerušenie (jeho radič) nastaví na údajovú zbernicu číslo prerušenia.

Po zverejnení čísla prerušenia procesor uloží do zásobníka adresu nasledujúcej inštrukcie (obsah registra ukazovateľa inštrukcie – je to vlastne návratová adresa, kde budeme pokračovať po ukončení procedúry na ošetrenie prerušenia) a príznakový register (aby sa uchovali príznaky medzivýpočtov, ktoré by sa inak v priebehu procedúry na ošetrenie prerušení porušili). Procesor nastaví zákaz ďalších prerušení a podľa príslušného vektora v tabuľke vektorov prerušení nájde adresu, na ktorej bude vo vykonávaní pokračovať.

Zmenou obsahu registra ukazovateľa inštrukcie sa prenesie riadenie do procedúry na ošetrenie prerušenia. Na jej konci by mal byť príkaz návratu z prerušenia (IRET), ktorý obnoví zo zásobníka pôvodný stav registra ukazovateľa inštrukcií a príznakového registra (najneskôr teraz je teda umožnené znova akceptovať prerušenia). Ďalšia inštrukcia sa potom vykonáva podľa obnoveného stavu inštrukčného registra, teda procesor pokračuje vo vykonávaní pôvodného programu s pôvodným stavom príznakov.

Príklad postupu pri signalizácii prerušenia 10:

Radič zariadenia, ktoré chce vyvolať prerušenie, nastaví signál INTR riadiacej zbernice.

Procesor ukončí vykonávanie inštrukcie a skontroluje úroveň signálu INTR.

Keď je signalizované prerušenie, procesor skontroluje obsah príznakového registra, či sú prerušenia povolené.

Pokiaľ by boli prerušenia zakázané (príznak IF=0) – žiadosť o prerušenie procesor ignoruje a pokračuje vykonávaním nasledujúcej inštrukcie (pokiaľ bude signál INTR aktívny, bude sa testovať príznak IF po každej ďalšej inštrukcii).

Ak sú prerušenia povolené (príznak IF=1), procesor potvrdí žiadosť signálom INTA riadiacej zbernice.

Keď dostane radič zariadenia signál INTA, nastaví na údajovej zbernici úrovne podľa kódu čísla (vektora) prerušenia, ktoré chce signalizovať (v našom prípade číslo 10) a zruší žiadosť INTR.

Procesor prečíta stav údajovej zbernice do vnútorných registrov a na základe čísla prerušenia a pozície začiatku tabuľky prerušení vypočíta pozíciu záznamu o adrese procedúry na ošetrenie prerušenia 10 (ak je záznam 8 bajtový, nájde príslušný popisovač adresy vo vzdialenosti 10.8 = 80 B od začiatku tabuľky popisovačov prerušení, uvedenej v registri IDTR).

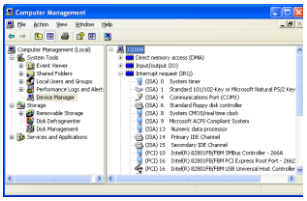
Procesor uloží do zásobníka aktuálny stav inštrukčného registra (ktorý obsahuje po ukončení spracovania inštrukcie adresu nasledujúcej vykonávanej inštrukcie), príslušného selektora segmentu a stav príznakového registra.

Nastaví zákaz prerušení (príznak IF na 0) a prenesie riadenie na miesto podľa vyhľadanej adresy v tabuľke prerušení (zapíše jej obsah do inštrukčného registra a selektora segmentu).

V procedúre na ošetrenie prerušení je potrebné uchovať pôvodný stav všetkých registrov (okrem príznakového). Ak je teda niektorý potrebný pri výpočte, treba jeho pôvodný obsah dočasne uložiť napr. v zásobníku. V architektúre x86 je možné urobiť aj prerušenie s prepnutím kontextu, ktoré uchová celý pôvodný kontext úlohy (ošetrenie prerušenia ale trvá omnoho dlhšie).

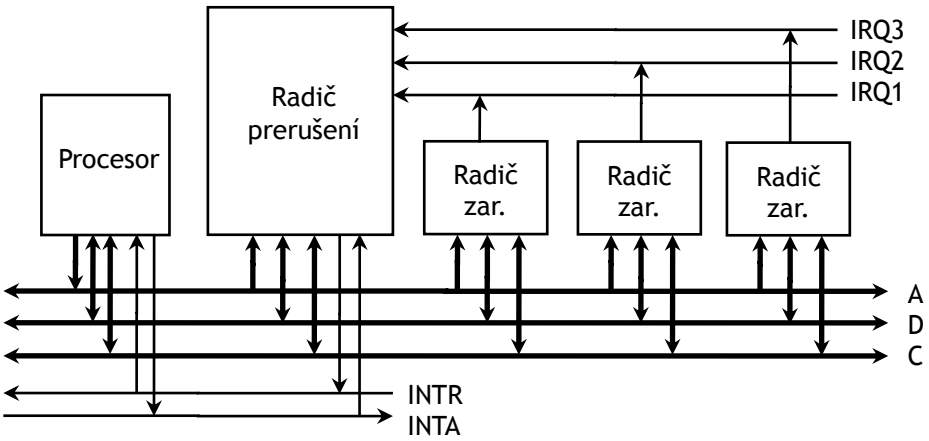
Ak nie je nutné vylúčiť ďalšie prerušenia, je dobré čo najskôr povoliť ich akceptovanie (inštrukciou STI – Set Interrupt-Enable Flag). Pokiaľ v procedúre povolíme prerušenia, môže ďalšie akceptované prerušenie vyvolať vnorené volanie procedúry na ošetrenie prerušenia. Pretože sa návratové adresy ukladajú do zásobníka, tak pokiaľ neprekročíme jeho kapacitu a pri ukončení každej procedúry vrátime registre v pôvodnom stave, bude vnáranie bez problémov.

Opísaný mechanizmus prerušenia predpokladá len jediný vstupný signál INTR pre procesor. Pokiaľ chceme ošetriť signály z viacerých zariadení, musíme použiť medzičlánok – **radič prerušení**. Úlohou radiča prerušení je organizovať komunikáciu žiadosti s procesorom (signálmi INTR a INTA) a komunikáciu (riešenie kolízii, priorít a selektívnych zákazov prerušenia) s ostatnými radičmi periférnych zariadení.



Aktuálne priradenia čísel prerušení (vektorov) je v systémoch MS Windows možné získať pomocou Správcu zariadení (Device Manager) a pohľadu na zdroje podľa pripojenia.

Radič prijíma signály žiadosti o prerušenia od radičov zariadení a preposiela ich procesoru. Na základe nastavenia svojich registrov (ktoré sú štandardne pripojené na systémovú zbernicu a programátor ich môže nastavovať inštrukciami IN a OUT) tento radič poskytuje sám číslo (vektor) prerušenia a v prípade kolízie žiadostí (keď ich vyšle niekoľko zariadení naraz) vyhodnocuje poradie spracovania podľa nastavenej priority (toto nastavovanie je tiež možné inštrukciami IN a OUT).



V špeciálnych prípadoch (napr. ladenie procedúr na ošetrenie prerušenia) je vhodné, aby sa niektorých žiadostí zákaz prerušenia netýkal. V architektúre x86 na to slúži ďalší signál riadiacej zbernice pre **nemaskovateľné prerušenia** NMI (Non-maskable Interrupt). Vtedy sa vykoná prerušenie INT 2.

Pôvodné radiče prerušení vedeli ošetriť 8 resp. 16 rôznych prerušení. Najnovšie používané radiče architektúry xAPIC (Advanced Programmable Interrupt Controller) vedia spracovávať 256 (v budúcej verzii x2APIC až 4 miliardy) hardvérových prerušení. Bývajú už aj zaintegrované v čipe procesora.

Mechanizmus prerušení dovoľuje, okrem reakcií na signály od vonkajších zariadení (**asynchrónne prerušenia**), reagovať aj na nepredvídané udalosti súvisiace s vykonávaním inštrukcií (**synchrónne prerušenia**), po ktorých nemá procesor štandardné pokračovanie (prístup mimo segment, delenie nulou, neznáma inštrukcia, výpadok stránky). Môžu vzniknúť len na konci spracovania inštrukcie a riešia sa podobne ako asynchrónne prerušenia s výnimkou kontroly signálov a príznakov povolenia. Tieto prerušenia sa nedajú maskovať. Nazývajú sa aj pasce (*traps*), resp. výnimky (*exceptions*).

Existuje aj možnosť vyvolania prerušenia priamo inštrukciou v programe. Hovoríme o **softvérovom prerušení** a v architektúre x86 sa realizuje inštrukciou typu INT n , kde n je vektor prerušenia. Procesor uloží aktuálnu pozíciu v práve vykonávanom programe spolu s obsahom príznakového registra do zásobníka a podľa vektora n nastaví z tabuľky vektorov prerušení vstupný bod obslužného podprogramu.

Medziprocesorové prerušenia IPI (Inter-processor Interrupt) sú hardvérové asynchrónne nemaskovateľné prerušenia, ktoré sa využívajú pre synchronizáciu medzi procesormi vo viacprocesorových počítačových systémoch. V najnovších typoch počítačov sa využívajú na synchronizáciu vyrovnávacích pamätí a jednotiek prístupov do pamäte jednotlivých procesorových jadier.

Aktivita 2	Monitorujte stav priradenia čísel portov a vektorov prerušení zariadeniam v systémovom programe Správca zariadení (Device Manager), analyzujte vlastnosti pripojených zdrojov.

Priamy prístup do pamäte

Prenosy údajov medzi V/V zariadeniami a operačnou pamäťou pomocou portov (inštrukciami IN a OUT) spomaľuje procesor tým, že údaje musí priebežne ukladať do svojich vnútorných registrov. Mechanizmus **priameho prístupu do pamäte DMA** (Direct Memory Access) umožňuje takéto prenosy vykonávať bez účasti procesora.

Cieľom tohto mechanizmu je, aby radič V/V zariadenia prebral na chvíľu riadenie systémovej zbernice (a tým aj prístup do operačnej pamäte). Žiadosť o pridelenie zbernice – **HOLD** posiela radič signálom riadiacej zbernice. Potvrdením (signálom **HLDA** – Hold Acknowledge) oznamuje procesor, že zbernicu uvoľňuje – možnosť riadiť zbernicu získa zariadenie (resp. jeho radič). To môže potom čítať a zapisovať do operačnej pamäte priamo (riadením signálov MEMR a MEMW).

Procesor v tomto prípade nemusí čakať na koniec inštrukčného cyklu (zariadenie nežiada o vykonanie iných inštrukcií, len o dočasné prevzatie komunikácie s pamäťou). Pokiaľ procesor nepotrebuje k ďalšiemu pamäť, môže pokračovať vo vykonávaní inštrukcií. Musí však vždy dokončiť strojový cyklus – teda pokiaľ začal prístup k pamäti, resp. portom, musí ho tiež dokončiť.

Radič zariadenia, ktoré chce priamo komunikovať s pamäťou (chce prevziať riadenie zbernice), nastaví signál HOLD riadiacej zbernice.

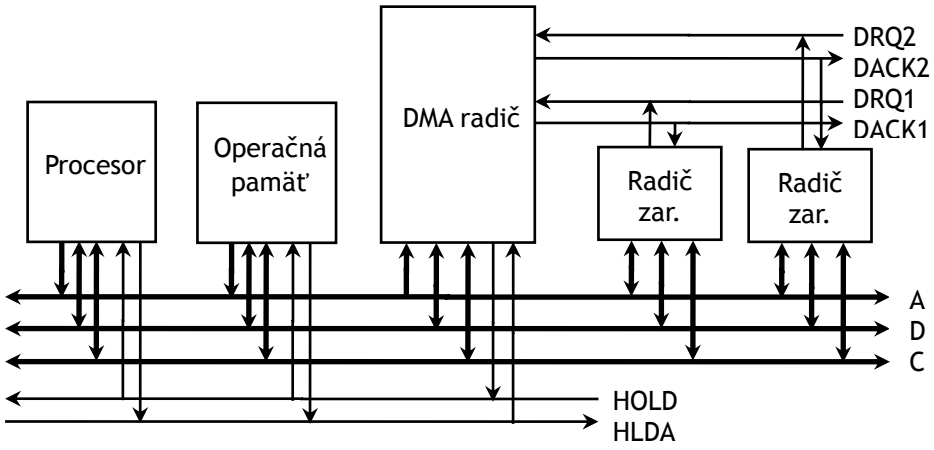
Po ukončení strojového cyklu procesor testuje stav žiadosti o DMA. Ak nie je aktívna resp. vykonávaná inštrukcia je zamknutá, pokračuje ďalej.

Ak je signál HOLD nastavený a inštrukcia nie je zamknutá, procesor vráti signál HLDA a prestane používať zbernicu.

Radič zariadenia presne rovnako ako procesor číta a zapisuje do pamäte.

Po ukončení prenosu radič zruší signál HOLD.

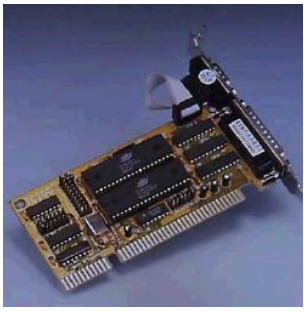
Aj v tomto prípade sa rieši kolízia viacerých zariadení, čakajúcich na priamy prístup, pomocou **radiča DMA**. Radič zachytáva žiadosti, vykomunikuje s procesorom prístup a potvrdenie prepošle zariadeniu s najvyššou prioritou.



Zabrániť odovzdaniu zbernice v priebehu spracovania inštrukcie je možné v architektúre x86 inštrukčným prefixom LOCK. Tento zámok treba použiť v prípade vykonávania sekvencie inštrukcií pre atomické operácie, prípadne pre inštrukcie, pracujúce s adresovým miestom, do ktorého sa bude pri DMA pristupovať.

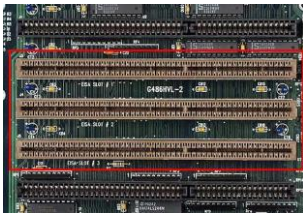
Rozšírenia systémovej zbernice

Zmeny v konfigurácii zariadení počítača sa v súčasných systémoch riešia zásuvnými modulmi s integrovanými radičmi zariadení. Tieto sa k systémovej zbernici pripájajú zasunutím do zásuvky, inštalovanej na základnej doske počítača. Zásuvky sú trvalo pripojené k zberniciam procesora a navyše bývajú rozšírené o signalizáciu žiadosti o konkrétne prerušenie a priamy prístup do pamäte. Štandardizácia rozšírených zberníc (a rozloženia kontaktov v príslušných zásuvkách) je dôležitá pre kompatibilitu pridávaných radičov a zariadení.

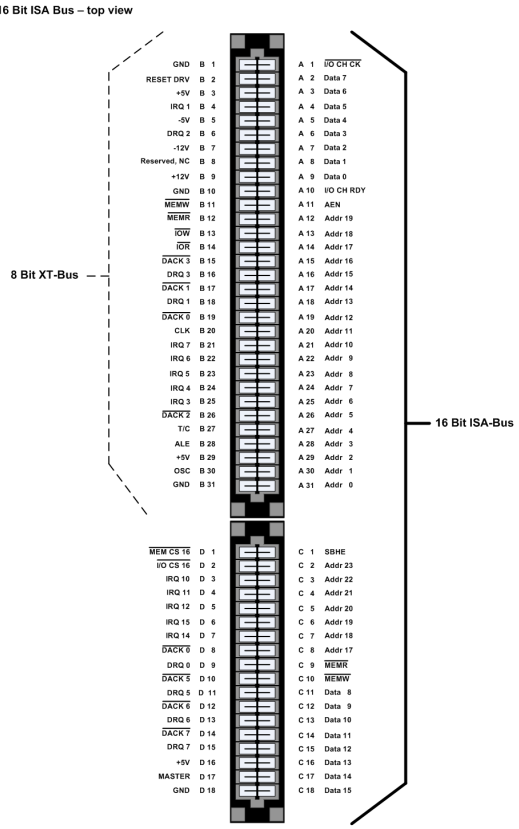


Zásuvné moduly sa v slangu niekedy nazývajú aj "karty"

Zásuvky 8 a 16 bitovej ISA zbernice



Zásuvky EISA zbernice



V prvých "PC kompatibilných" počítačoch triedy PC/XT sa začala (v osemdesiatych rokoch) používať rozšírená systémová zbernica ISA (Industry Standard Architecture). Obsahovala okrem signálov systémovej zbernice (8 údajových vodičov a 20 adresových vodičov s adresovacím priestorom 1 MiB, signálov pre čítanie a zápis do pamäte a registrov V/V zariadení) aj 7 signálov žiadosti o prerušenie (vedených k radiču prerušení) a signály pre 4 DMA kanály (žiadosti a potvrdenia k radiču DMA).

Jej 16 bitové rozšírenie (pôvodne pre modely PC/AT s procesorom Intel 80286) používalo 92-pinovú zásuvku, kompatibilnú s 8-bitovým formátom PC/XT, rozšírenú o ďalšie 4 adresové vodiče (celkom 24 – t.j. adresovateľný priestor maximálne 2^{24} = 16 MiB), 8 ďalších údajových vodičov (celkom dovoľuje paralelný prenos 16-bitových slov) a pridalo 5 prerušení a štyri DMA kanály. Na adresáciu V/V portov sa využívalo väčšinou len 10 adresových bitov (1024 adresovateľných portov). Taktovanie 4,77 MHz bolo nahradené 8 MHz taktom s prenosom max. 16 MB/s. Dnes sa používa hlavne pre zabudované (*embedded*) riešenia.

IBM v snahe udržať dominanciu na trhu použila pre svoj model PS/2 zbernicu MCA (Micro Channel Architecture), chránenú licenciami - ktorá sa však na trhu nepresadila a IBM tým svoju dominanciu stratila.

Pozostatkom MCA je konektor pre PS/2 myš a klávesnicu.

ISA zbernice (obyčajne čiernej farby) boli neskôr (združením nezávislých výrobcov PC na čele s firmou Compaq) rozšírené na hnedé EISA 32 bitové zbernice s 32 adresovými vodičmi (s adresovateľným priestorom 2^{32} = 4 GiB), ktoré vyhrali konkurenčný boj o kompatibilné PC s MCA zbernicami od IBM.

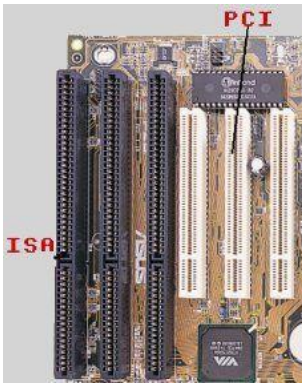
Všetky staršie rozšírenia zbernice boli v roku 1993 nahradené štandardom PCI (Peripheral Computing Interconnect), ktorý chráni a rozvíja nezávislé združenie PCI-SIG. PCI je (na rozdiel od ISA) synchrónnou zbernicou a v 32 bitovej verzii pri taktovaní 33,33 MHz prenesie 133 MB/s (v 64 bitovej verzii pri frekvencii 66 MHz až 533 MB/s). Pre prenos adresy aj údajov sa používajú rovnaké vodiče. Najskôr sa nastaví adresa a potom sekvencia údajov, ktoré sa od určenej adresy prenesú. Zbernica PCI podporuje 32-bitový adresovací priestor pre registre portov, identifikovaných číslom zbernice, zariadenia, funkciou a číslom registra. Každé zariadenie má tiež vlastný 256 bajtový priestor na ukladanie konfigurácií.

Združenie PCI-SIG zverejňuje špecifikácie na adrese www.pcisig.com.

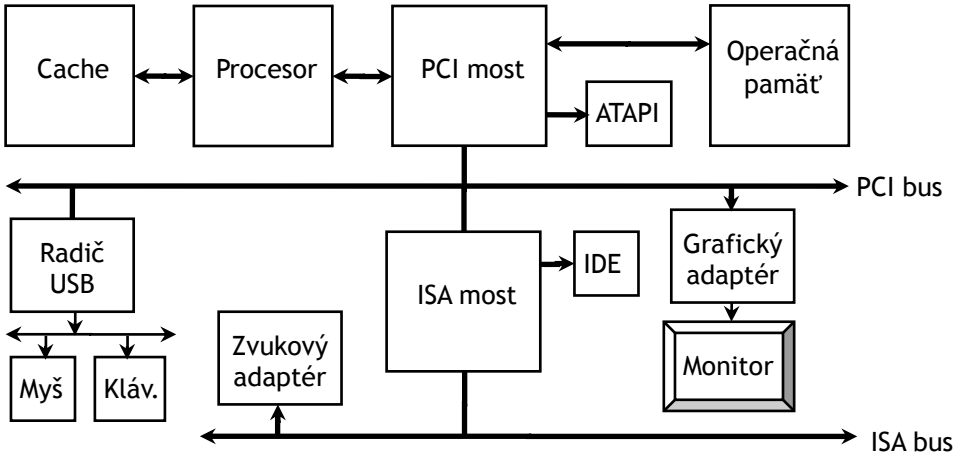
PCI obsahuje plnú podporu automatického softvérového nastavovania zdrojov (nie je potrebné riešiť kolízie čísel portov a prerušení v BIOSe resp. prepínačmi na základnej doske). Podporuje štyri prerušenia, viazané na konkrétne zásuvky. Radič môže žiadať o prerušenie resp. DMA prístup tiež pomocou správ MSI (Message

Signaled Interrupts), odosielaných do registra radiča APIC (Advanced Programmable Interrupt Controller), integrovaného v PCI moste. Ten vybavuje komunikáciu s CPU.

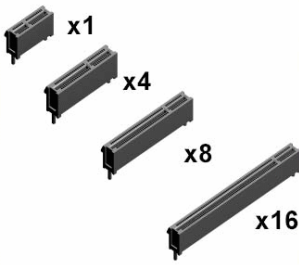
PCI zbernicu je možné rozvetvovať cez ďalšie mosty a vytvárať tak celú hierarchiu, rozdelenú podľa parametrov zariadení. Prístup k operačnej pamäti je odčlenený samostatným kanálom, aby nebol obmedzovaný pomalšími zariadeniami. V ďalšom sa odčlenil tiež prístup ku grafickému adaptéru cez samostatnú AGP (Accelerated Graphics Port) zbernicu (v najrýchlejšej verzii s prístupom 2,1 GB/s).

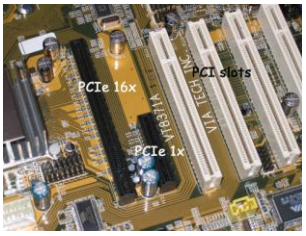


92-pólové zásuvky ISA a 120-pólové PCI zásuvky



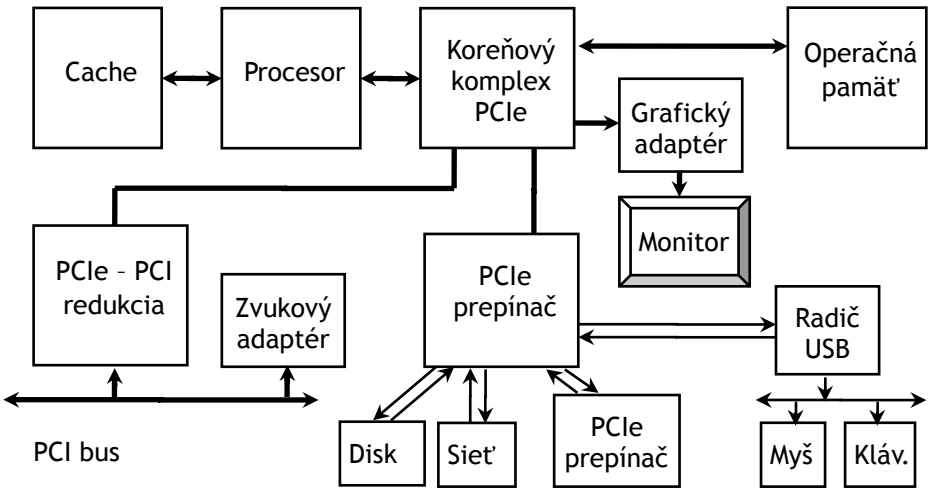
Vysoké nároky súčasných systémov rieši zbernica **PCI Express** (resp. **PCIe**). Na rozdiel od PCI zbernice je to sériová zbernica. Medzi dvoma komponentmi podporuje komunikáciu dvojicou protismerných komunikačných kanálov. Dva páry diferenčných signálov spolu s uzemnením (spolu 8 vodičov) tvorí jeden **pruh** (lane). Zariadenie môže tieto pruhy agregovať a vytvárať rýchlejšie prepojenia. Počet pruhov, ktoré podporuje radič resp. zásuvka na základnej doske, je označovaný v tvare x1, x2, x4, x8, x16, x32. Vo verzii 2.0 je deklarovaná priepustnosť pruhu 500 MB/s, čo znamená, že prepojením PCIe x16 možno preniesť až 8 GB/s.





Rôzne zásuvky pre PCIe zbernicu

Prenos údajov, príkazov a správ medzi pripojenými zariadeniami sa uskutočňuje (podobne ako v počítačových sieťach) paketmi s definovanou štruktúrou v sieti so stromovou topológiou prostredníctvom prepínačov (*PCI Express switch*). Pre každé prepojenie zariadení je možné vyjednať počet pruhov a prenosovú rýchlosť. PCI Express teda vlastne už ani nie je zbernica, ale sieť radičov zariadení.

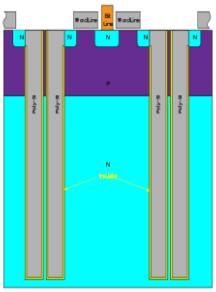


Zvukový adaptér s pripojením PCIe x1

Pamäť a procesor (jadro systému) oddeľuje od zvyšku zariadení koreňový komplex (*root komplex*). Na neho sa môžu napájať ďalšie prepínače a mosty. Prenos údajov prebieha v zhlukoch (*burst mode*) – sekvenciami bajtov.

2 Pamäťová architektúra počítačového systému

Konštrukcia operačnej pamäte, dynamická pamäť DRAM



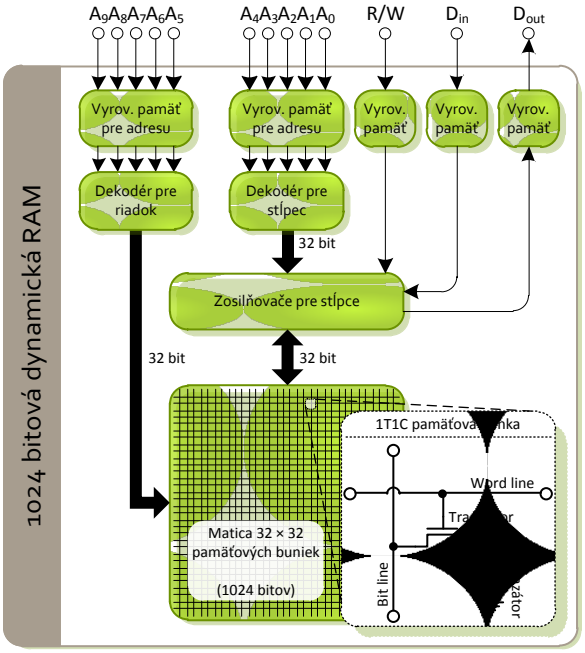
Pamäť DRAM v CMOS technológii - parazitné náboje na rozšírených elektródach CMOS obvodu spôsobujú kondenzátorový efekt

Pamäť, tvorená bunkami z logických preklápacích obvodov RS (spomínaná podrobnejšie v prvom module), nazývaná tiež **statická** - SRAM (Static Random Access Memory), je relatívne zložitá a náročná na výrobu - používa sa väčšinou tam, kde je dôraz na rýchlosť prístupu k údajom (menšie a rýchle vyrovnávacie *cache* pamäte). Súčasné operačné pamäte (s kapacitami niekoľko miliárd bajtov) sa obyčajne realizujú konštrukčne jednoduchšou (teda menej nákladnou) **dynamickou pamäťou** DRAM (Dynamic Random Access Memory).

Konštrukčným základom bunky dynamickej pamäte je kondenzátor. Kondenzátor je schopný určitý čas udržať elektrický náboj a môže sa teda nachádzať v dvoch základných stavoch – nabitom a nenabitom. Tieto stavy môžeme využiť na reprezentáciu jedného bitu informácie (napríklad nabitý kondenzátor reprezentuje logickú jednotku a vybitý logickú nulu).

Kondenzátor ale po nabití svoj náboj postupne stráca. Preto je nutné pravidelne jeho náboj obnovovať – z čoho pochádza aj názov "dynamická pamäť".

Podrobnosti o konštrukcii rôznych typov pamätí je možné nájsť v [9,10].



Vzhľadom na to, že bitové stĺpce sú veľmi dlhé (reálne pamäte majú tisíce riadkov), na prečítanie informácie z jednotlivých bitových stĺpcov sú potrebné zosilňovače (*sense amplifier*).

Najjednoduchším variantom bunky dynamickej pamäte je zapojenie typu 1T1C (bunka, zložená z jedného pamäťového kondenzátora a jedného prístupového tranzistora – na obrázku v bielom štvorci vpravo dole).

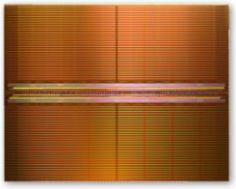
Pred otvorením tranzistorov sa jednotlivé bitové stĺpce prednabijú (*precharge*) na hodnotu presne medzi logickou 0 a 1. Potom po pripojení kondenzátora k bitovému stĺpcu príde k veľmi malému poklesu (ak bola uložená nula) alebo veľmi malému zvýšeniu napätia (ak bola uložená jednotka) na bitovom stĺpci.

Zosilňovače postupne zosilňujú túto odchýlku až po úroveň logickej 0 alebo 1. Ich úlohou je okrem toho obnoviť pôvodnú úroveň náboja na pamäťovom kondenzátore (pripojením kondenzátora k bitovému stĺpcu prišlo k jeho deštrukcii). Po ustálení zosilňovačov je riadok otvorený - pripravený na zápis alebo čítanie.

Privedením elektrického napätia na vstup riadku slova (*word line*) sa privedie napätie aj na bázu prístupového tranzistora. Ten sa otvorí a pripojí pamäťový kondenzátor na bitový stĺpec (*bit line*). Pomocou bitového stĺpca je potom možné čítať, alebo zapisovať jeden bit do pamäťového kondenzátora. Keď sa napätie z riadku slova odpojí, tranzistor sa uzavrie a izoluje kondenzátor od bitového stĺpca.

Jednobitové pamäťové bunky sa spájajú do väčších **pamäťových matíc**. Adresa pamäťovej bunky z adresovej zbernice sa rozdelí na dve časti: na adresu riadku a na adresu stĺpca. V našom príklade má adresa riadku (A_9 až A_5) päť bitov a adresa stĺpca (A_4 až A_0) tiež päť bitov. Preto pole pamäťových buniek bude mať 32 riadkov a 32 stĺpcov. Jednotlivé bunky v riadku majú prepojené svoje riadky slova (*word line*). Bunky v jednom stĺpci zas majú spojené bitové stĺpce (*bit line*). Spolu tak pamäť obsahuje 1024 (32 × 32) bitov adresovaných 10 bitovou adresou (A_9 až A_0).

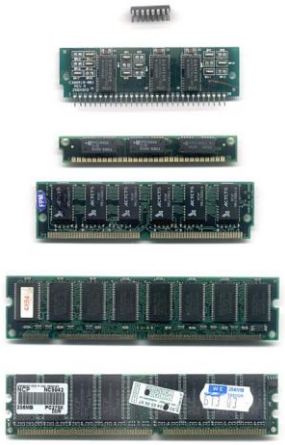
Po zadaní a rozdelení adresy sa pomocou dekodéra riadku otvoria všetky prístupové tranzistory v adresovanom riadku slova. To spôsobí pripojenie kondenzátorov z pamäťových buniek v tomto riadku na jednotlivé bitové stĺpce. V každom bitovom stĺpci je teraz pripojený práve jeden kondenzátor z adresovaného riadku. Následne pomocou dekodéra stĺpca sa z nich vyberie adresovaný bitový stĺpec. Ak je riadiaci vstup R/W nastavený na R (*read*), tak sa bit z tohto stĺpca prenesie do výstupu D_{out}. Ak je riadiaci vstup R/W nastavený na W (*write*), tak sa do bitového stĺpca zapíše bit informácie nachádzajúci sa na vstupe D_{in}.



Jednoduchá konštrukcia dynamických pamäťových buniek umožňuje ich vysokú integráciu

Pre zvýšenie efektívnosti sa v súčasných typoch pamätí posielajú adresy riadku i stĺpca po tých istých adresových vývodoch – potrebujeme ale pridať riadiace signály, ktoré signalizujú, ktorá časť adresy je práve pripojená (RAS – Row Address Strobe, CAS – Column Address Strobe).

Obnovovanie (*refresh*) obsahu pamäte DRAM (z dôvodov postupnej straty náboja) sa realizuje pravidelným otváraním riadkov slov. Ak pamäťová bunka vyžaduje obnovenie každých 64 ms (najčastejšia požiadavka výrobcov) a ak má pamäť napríklad 8 192 riadkov, tak sa musí každých 7,8125 µs (*refresh rate)* obnoviť jeden riadok. Pri obnovovaní sa používa počítadlo (*refresh counter*), ktoré obsahuje číslo riadku, ktorý sa má najbližšie obnoviť. Obnovovanie spravidla prebieha v nekonečnom cykle (po pretečení počítadla sa pokračuje odznova nultým riadkom).



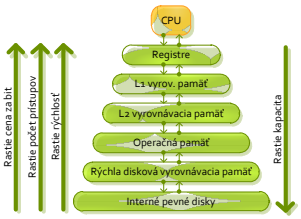
DRAM pamäťové zásuvné moduly rôznych typov

SDRAM (Synchronous Dynamic Random Access Memory) je dynamická RAM pamäť synchronizovaná signálmi systémovej zbernice. Synchronizácia umožňuje procesoru získavať údaje v presne stanovenom čase, čím sa zvýši efektivita prenosu. Súčasné operačné pamäte využívajú synchronizované prenosy na nábehovú aj na dobehovú hranu časových synchronizačných impulzov. Dosahuje sa tak hneď dvojnásobok prenosovej rýchlosti. Ich označenie je DDR (Double Data Rate) SDRAM a v závislosti na počte súčasne prenesených bajtov sa ešte rozlišujú na DDR2, DDR3, DDR4 SDRAM.

Pamäťová hierarchia

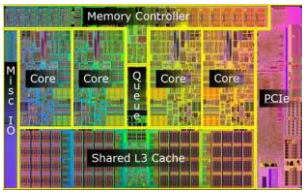
Ideálna pamäť by bola dostatočne veľká, lacná a primerane rýchla na to, aby CPU pri práci nezdržovala. Žiaľ, dnes neexistuje pamäť, ktorá by spĺňala všetky tieto požiadavky. Vo všeobecnosti platia nasledovné princípy:

- Čím má pamäť väčšiu kapacitu, tým má väčšiu prístupovú dobu.
- Čím má pamäť väčšiu kapacitu, tým je jej cena za bit nižšia.
- Čím má pamäť menšiu prístupovú dobu, tým je jej cena za bit vyššia.



Pamäťová hierarchia

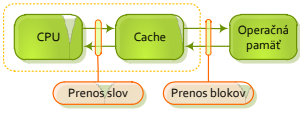
Počítačové systémy na dosiahnutie tohto cieľa využívajú celú pamäťovú hierarchiu od malých, rýchlych a drahých pamätí až po veľké, lacné a pomalé pamäte. Pamäťová hierarchia sa vyznačuje tým, že čím je pamäť bližšie k CPU, tým častejšie k nej CPU vďaka princípu lokálnosti referencií pristupuje. Princíp lokálnosti referencií (*locality of reference*) hovorí, že prístupy do pamäte majú tendenciu sa zhlukovať. Počas dlhšieho časového obdobia dochádza k zmene používaných zhlukov adries, ale počas krátkeho obdobia pracuje CPU väčšinou iba s niekoľkými zhlukmi.



Vyrovnávacia pamäť súčasných procesorov zaberá podstatnú časť plochy procesorového čipu

Rýchla vyrovnávacia pamäť (*cache*)

Diametrálny rozdiel medzi rýchlosťou prístupu CPU k registrom (rýchla SRAM s kapacitou niekoľko stovák bajtov) a k operačnej pamäti (relatívne pomalá a lacná DRAM s kapacitou niekoľko miliárd bajtov) je v súčasných počítačoch riešená rýchlou vyrovnávacou pamäťou (niekedy aj viacvrstvovou), riadenou hardvérovo. Je pomalšia ako registre a rýchlejšia ako operačná pamäť a jej kapacita je niekoľko miliónov bajtov.



Zapojenie rýchlej vyrovnávacej pamäte (cache)

Úlohou vyrovnávacej pamäte je uchovať čo najväčšiu časť obsahu aktuálne používaných pamäťových adries. Je zapojená medzi CPU a operačnou pamäťou a reaguje na všetky žiadosti CPU o prístup do pamäte. Ak má uchovanú hodnotu z požadovanej adresy, odpovedá na žiadosť sama, pričom sa úplne obíde pomalý prístup do operačnej pamäte. Nastáva takzvaný „*cache hit*“. Ak sa v rýchlej

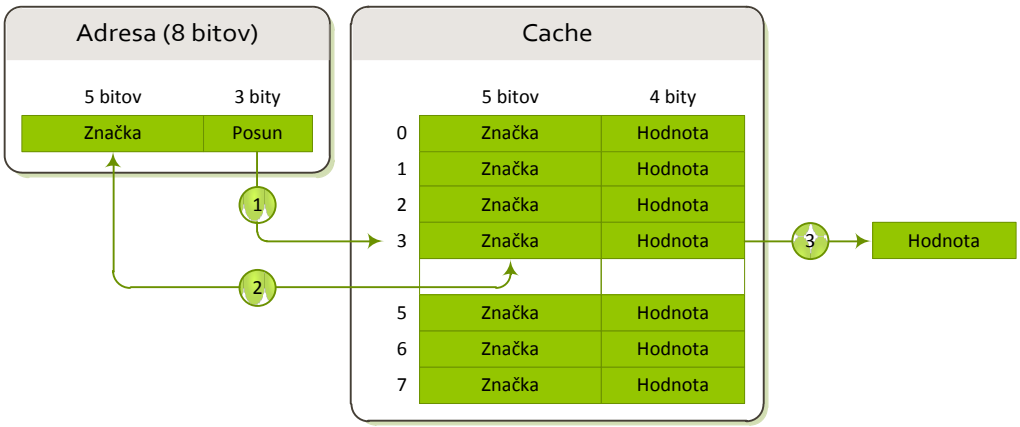
vyrovnávacej pamäti požadovaná hodnota nenachádza - „*cache miss*“ - prečíta radič vyrovnávacej pamäte z operačnej pamäte celý blok hodnôt okolo požadovanej adresy. Až potom pošle CPU požadovanú hodnotu. Niekoľko susedných prístupov bude takto obslúžených priamo z vyrovnávacej pamäte.

Keď je vyrovnávacia pamäť plná a nastane „*cache miss*“, je potrebné vybrať údaje, ktoré budú z vyrovnávacej pamäte vyhodené a nahradiť ich novými. Najjednoduchší nahradzovací algoritmus je **FIFO** (First In First Out), pri ktorom sa vyberie na nahradenie údaj, ktorý je vo vyrovnávacej pamäti najdlhšie. Jeho nevýhodou je, že často používané údaje budú po čase nahradené, aj keď málo používané ostanú. Algoritmus **LRU** (Last Recently Used) vyhodí najdlhšie nepoužívaný údaj. V tomto prípade robí najväčší problém uchovávanie informácií o poslednom prístupe a ich prehľadávanie. Postup **NFU** (Not Frequently Used) priradí každému miestu počítadlo, kde sa priebežne zaznamenáva počet prístupov. Vyhadzuje sa údaj z miesta s najmenšou hodnotou počítadla. Aby intenzívne využívané miesta neostali navždy neodstrániteľné, je možné zaviesť tzv. starnutie (všetky počítadlá raz za čas posunieme o jeden bit vpravo – vydelíme ich hodnoty dvoma).

Zápis do pamäte sa môže realizovať dvoma spôsobmi. Pri **priamom zápise** (*write-through*) sa zapisuje hodnota súčasne do vyrovnávacej a aj do operačnej pamäte. Hodnota v operačnej pamäti je stále aktuálna. no pri každom zápise je nutné počkať na pomalú operačnú pamäť. K urýchleniu práce dochádza len pri čítaní a nie pri zápise. Pri **odloženom zápise** (*write-back*) sa nová hodnota do operačnej pamäte neukladá. Výhodou je, že pokiaľ sa hodnota opakovane mení, nie je potrebné ju opakovane ukladať do pamäte. Do operačnej pamäte sa uloží až výsledná hodnota, keď sa toto slovo vyberie na nahradenie. Nevýhodou je, že hodnota v operačnej pamäti ostane dovtedy neaktualizovaná. Pokiaľ by s ňou pracovala aj iná súčasť počítača (napr. ďalší procesor alebo radič DMA) môže dôjsť k nekonzistencii.

Priamo mapovaná pamäť (*direct mapped cache*)

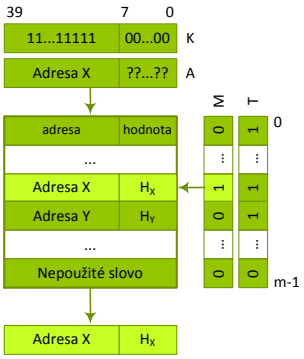
Je realizovaná pamäťou typu SRAM a pre zadanú adresu vráti hodnotu uloženú na tejto adrese (alebo oznámi „*cache miss*“). Adresa sa delí na dve časti: Značka (*tag*) a Posun (*offset*). Posun reprezentuje menej významnú časť adresy. Veľkosť posunu v bitoch (n) sa volí tak, aby korešpondovala s veľkosťou vyrovnávacej pamäte (2^n). Slová vo vyrovnávacej pamäti pozostávajú z dvoch častí: Značka (*tag*) a Hodnota (*value*).



Postup zistenia hodnoty pre zadanú adresu je takýto:

1. Z vyrovnávacej pamäte sa načíta slovo s indexom Posun.
2. Porovná sa Značka z adresy so Značkou zo získaného slova. Ak sa nerovnajú, rýchla vyrovnávacia pamäť neobsahuje požadovanú hodnotu - „*cache miss*“.
3. Pokiaľ nastala zhoda značiek, vráti sa hodnota uložená v slove získanom v predošlom kroku.

V takto realizovanej vyrovnávacia pamäti môže byť uložená ľubovoľná súvislá postupnosť adries nepresahujúca jej veľkosť – čo je výhodné vzhľadom na princíp lokálnosti referencií. Nevýhodou je, že vo vyrovnávacej pamäti nemôžu byť súčasne uložené žiadne dve slová, vzdialené o násobok jej veľkosti.



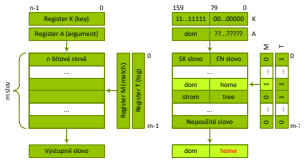
Plne asociatívna rýchla vyrovnávacia pamäť

Plne asociatívna pamäť (*fully associative cache*)

Riešenie postavené na drahšej asociatívnej pamäti nekladie obmedzenia na adresy uložených slov. Pamäť typu CAM (Content Addressable Memory) sa použije ako slovník dvojíc (Adresa, Hodnota). Pri hľadaní hodnoty na zadanej adrese sa Adresa vyhľadá v slovníku. Zo získanej dvojice sa vezme jeho druhá časť, ktorá obsahuje hľadanú hodnotu - *„cache hit“*. Pokiaľ zhoda nenastala, tak vyrovnávacia pamäť požadovanú hodnotu neobsahuje, nastáva *„cache miss“*.

Množinovo asociatívna pamäť (*set associative cache*)

Tento prístup v sebe kombinuje oba vyššie uvedené prístupy tým, že každé slovo z priamo mapovanej pamäte nahradí asociatívnou pamäťou veľkosti n. Potom hovoríme o n-cestnej (*n-way*) množinovo asociatívnej pamäti. Pamäť pracuje takto:



CAM vyhľadáva paralelne na základe nastavenia masky

1. Sprístupní sa asociatívna pamäť s indexom Posun.
2. Porovná sa Značka z adresy so značkami v sprístupnenej asociatívnej pamäti. Ak nenastane žiadna zhoda, rýchla vyrovnávacia pamäť neobsahuje požadovanú hodnotu, nastáva *„cache miss“*.
3. Pokiaľ nastala zhoda, vráti sa Hodnota uložená v slove získanom z asociatívnej pamäte v predošlom kroku.

Takéto riešenie umožňuje, aby vo vyrovnávacej pamäti mohlo byť uložených, na rozdiel od priamo mapovanej pamäti, až n rôznych slov s rovnakým posunom (*offset*).

Viacúrovňové rýchle vyrovnávacie pamäte (*multi-level caches*)

Viacúrovňové rýchle vyrovnávacie pamäte pracujú vo všeobecnosti tak, že pri prístupe do pamäte sa najprv skontroluje obsah vyrovnávacej pamäte prvej úrovne (*Level 1 cache*, alebo *L1 cache*). Ak sa požadovaná informácia nájde, CPU ju získa maximálnou možnou rýchlosťou. Ak sa nenájde, pokračuje sa väčšou ale pomalšou vyrovnávacou pamäťou druhej úrovne (*L2 cache*). Ak sa informácia nájde na druhej úrovni, získa ju CPU síce pomalšie ako v predošlom prípade, stále však rýchlejšie, ako by ju mal načítať z operačnej pamäte. V prípade, že sa informácia nenájde ani na druhej úrovni, môže sa hľadať na ďalšej úrovni (ak existuje), až nakoniec sa vykoná prístup do operačnej pamäte.

Virtuálna operačná pamäť

Pokiaľ nestačí veľkosť fyzicky realizovanej operačnej pamäte, umožňujú počítačové systémy pracovať s **virtuálnou pamäťou**. Princíp spočíva v tom, že proces nepoužíva fyzické adresy, ale virtuálne (pomyselné) adresy. Virtuálne adresy mapuje (prepočíta pomocou tabuľky) jednotka pre správu pamäte na fyzické adresy pri každom prístupe do pamäte. Tým získa fyzickú adresu, na ktorej je údaj skutočne uložený.

Virtuálna pamäť je obyčajne väčšia ako operačná pamäť. Údaje, ktoré sa nevmestia do operačnej pamäte sa ukladajú na disk. V prepočítavacej tabuľke je potom zaznamenané ich umiestnenie na disku, odkiaľ sa v prípade potreby načítajú späť do operačnej pamäte. Ak už v operačnej pamäti nie je voľné žiadne miesto a treba uložiť ďalšie údaje, uvoľní sa operačná pamäť odložením jej časti na disk. Pre výber kandidátov na vyhodenie sa používajú podobné algoritmy ako v prípade cache.

Virtualizácia adresového priestoru tiež umožňuje rozdelenie procesu na časti, ktoré nemusia byť v operačnej pamäti umiestnené za sebou, dokonca nemusia byť ani umiestnené v pôvodnom poradí. Operačný systém zavedie do operačnej pamäte len malú časť kódu, ktorý je vyžadovaný pri štarte programu. Keď sa počas vykonávania programu narazí na virtuálnu adresu, ktorá nemá mapovanie na fyzickú adresu,

vyvolá CPU prerušenie (pri stránkovaní sa toto prerušenie nazýva výpadok stránky (*page fault*). Operačný systém obslúži prerušenie zavedením časti programu, ktorá obsahuje požadovanú virtuálnu adresu do operačnej pamäte a vytvorí potrebný záznam v tabuľke mapovaní. Pokiaľ nie je v operačnej pamäti voľné miesto, najprv sa na disk odsunú nepotrebné údaje. Potom sa pokračuje v prerušenom programe.

Virtualizácia vyžaduje hardvérovo podporované prepočítavanie fyzických adries (so špeciálnym režimom na uchovanie tabuliek vo vyrovnávacích pamätiach). V prípade, že sú výpadky stránok časté, treba počítať so spomalením celkového výkonu o čas prístupu k diskovej pamäti (princíp lokálnosti referencií by aj tu mohol pomôcť).

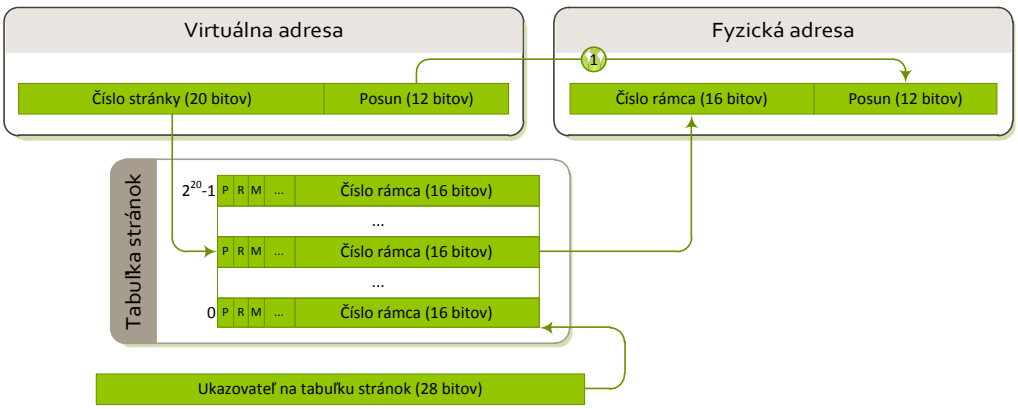
Stránkovanie

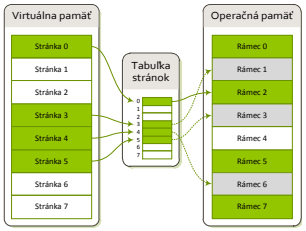
V praxi sa často virtuálna pamäť implementuje pomocou **stránkovania** (*paging*). Nedochádza tu k mapovaniu jednotlivých adries, ale celých blokov adries. Virtuálny adresový priestor je rozdelený na **stránky** s rovnakou veľkosťou. Na rovnako veľké časti sa rozdelí aj operačná pamäť. Tieto časti sa nazývajú **rámce** (*frame*). Na rovnako veľké časti je rozdelený aj odkladací súbor (*page file*, *swap file*), prípadne aj celý odkladací oddiel (*swap partition*) na disku. Do odkladacieho priestoru sú uložené stránky, ktoré sa nezmestili do operačnej pamäte.

Pre zjednodušenie prepočtov sa veľkosti stránok navrhujú ako mocniny dvojky.

Pri 32-bitovej virtuálnej adresácii do 28-bitového fyzického adresovacieho priestoru s veľkosťou stránky 4 kB = 2^{12} bajtov by mapovanie mohlo vyzerať tak, že prvých 20 bitov virtuálnej adresy určí číslo stránky = poradové číslo záznamu v tabuľke stránok. Záznam potom obsahuje prvých 16 bitov fyzickej adresy, ku ktorým sa pridá (šípka 1 na obrázku) 12 bitov z pôvodnej virtuálnej adresy (tzv. posunutie – *offset*).

Pre mapovanie virtuálnej adresy na fyzickú stačí zameniť vo virtuálnej adrese číslo stránky za číslo rámca. Na to sa použije tabuľka stránok.





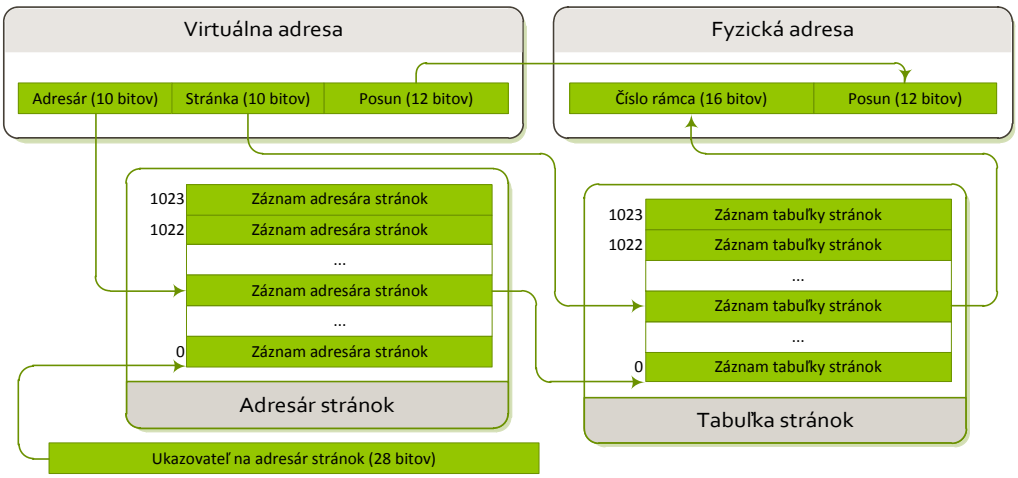
Systém môže obsahovať jedinú globálnu tabuľku stránok zdieľanú všetkými procesmi (všetci zdieľajú jeden virtuálny adresový priestor) alebo je možné vytvoriť samostatné tabuľky stránok pre každý proces. Pri mapovaní adries sa použije tá, na ktorú aktuálne ukazuje ukazovateľ na tabuľku stránok (realizovaný napr. špeciálnym registrom CPU, aktualizovaným pre každý proces). Každý proces potom má svoj vlastný, samostatný, virtuálny adresový priestor pokrývajúci celý rozsah virtuálnych adries. Pritom je možné elegantne realizovať aj zdieľanie časti operačnej pamäte viacerými procesmi pomocou rovnakého mapovania na zdieľaný rámec.

Záznam o stránke v tabuľke stránok obsahuje aj ďalšie informácie – v našom príklade príznak P (Present), ktorý signalizuje, či je príslušná stránka mapovaná do operačnej pamäte, príznak R (Referenced) sa nastaví na 1 pri každom prístupe (čítanie aj zápis) do danej stránky a použije sa ako počítadlo prístupov, príznak M (Modified) sa nastaví na 1 pri každom zápise do stránky (pokiaľ sa stránka od posledného načítania nezmenila, netreba ju pri odsunutí prepisovať). Môžu tu byť aj informácie o prístupových právach procesu k stránke, prípadne či je dovolené do nej zapisovať.

Pretože pri prideľovaní stránok sa nemusí zachovávať ich susednosť (a sú rovnako veľké), neostávajú vo fyzickej pamäti nevyužité malé kúsky (tzv. vonkajšia fragmentácia). Pokiaľ ale proces požaduje menej pamäte ako je veľkosť stránky, ostane zvyšok rámca nevyužitý. Tento jav sa nazýva **vnútorná fragmentácia**.

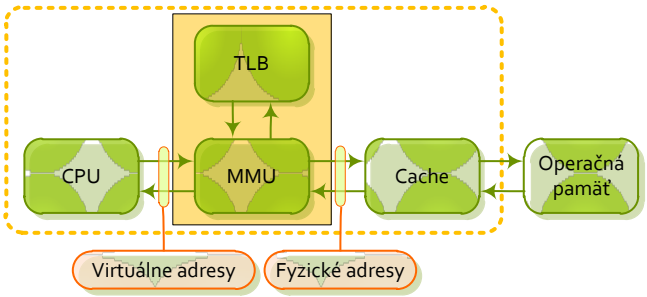
Viacúrovňové tabuľky stránok

So zväčšovaním virtuálneho adresového priestoru sa zväčšuje aj veľkosť tabuľky stránok. Ak chceme priradiť tabuľku stránok každému procesu a umožniť, aby virtuálna pamäť bola čo najväčšia, bude veľa záznamov v tabuľkách nevyužitých. Tomu je možné zabrániť viacúrovňovými tabuľkami stránok.



V zobrazenom príklade dvojúrovňového stránkovania (podobne ako v architektúre x86) sa virtuálna adresa na rozdiel od predošlého prípadu rozdelí nie na dve, ale na tri časti. Pravých (najnižších) 12 bitov bude opäť tvoriť posun (*offset*). Ľavých 20 bitov, ktoré predtým tvorili číslo stránky sa však rozdelí na 10 bitové číslo adresára (ľavá časť) a 10 bitové číslo stránky (pravá časť).

Hľadanie čísla rámca začína adresárom stránok. Každý proces má ukazovateľ na začiatok tohto adresára. Z adresára sa načíta záznam adresára stránok s indexom rovnajúcim sa 10 bitovému číslu adresára získaného z virtuálnej adresy. Tento záznam obsahuje položku, ktorá ukazuje na začiatok tabuľky stránok. Z tejto tabuľky sa získa záznam tabuľky stránok s indexom rovnajúcim sa číslu stránky. V tomto zázname je položka udávajúca číslo rámca. Nevýhodou tohto riešenia je, že na získanie čísla rámca sú potrebné dva prístupy do pamäte. Jeden do adresára stránok a druhý do príslušnej tabuľky stránok.



Preklad virtuálnych adries na fyzické vykonáva špeciálna časť procesora, nazývaná **jednotka pre správu pamäte MMU** (Memory Management Unit). Pracuje samostatne bez programátorských zásahov. V prípade, že požadovaný preklad adresy zatiaľ neexistuje, vyvolá prerušenie (výpadok stránky), ktoré obslúži operačný systém zavedením požadovanej stránky. Preklad adries si MMU urýchľuje využitím TLB (Translation Lookaside Buffer), čo je špeciálna rýchla vyrovnávacia pamäť využívaná iba na ukladanie tabuliek stránok. Pri neoprávnenom prístupe k pamäti vyvolá prerušenie, ktoré operačný systém rieši väčšinou ukončením procesu.

Ak by záznam v adresári stránok mal veľkosť 4 bajty, tak veľkosť adresára stránok v našom modeli je 4 kB. Rovnako veľkosť jednej tabuľky stránok je 4 kB. Každý proces má práve jeden adresár stránok (na ktorý ukazuje ukazovateľ na adresár stránok). Jeden proces však môže mať niekoľko tabuliek stránok, na ktoré ukazujú jednotlivé ukazovatele z adresára stránok. Týchto ukazovateľov je v našom modeli 1024. Každý proces tak môže mať od nula po 1024 stránok veľkosti 4 kB.

Celá štruktúra v pamäti bude zaberať od 4 kB (iba prázdny adresár) až po 4 MB + 4 kB, ak by boli alokované všetky tabuľky stránok. Koľko pamäte bude reálne obsadenej je priamo úmerné počtu použitých tabuliek stránok.

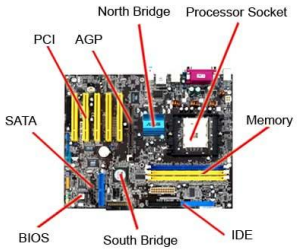
3 Funkcia, pripojenie a spôsob využívania periférnych zariadení

Vstupno-výstupné zariadenia *(I/O devices)*, alebo tiež **periférne zariadenia** *(peripheral devices*), sú časti počítačového systému, ktoré sprostredkúvajú prirodzenú komunikáciu medzi výpočtovými jednotkami (procesormi) a okolím (používateľmi resp. inými počítačovými systémami). Pre človeka tak interpretujú vstupné a výstupné údaje spôsobom jemu blízkym. Sú to napríklad klávesnica, myš, tablet a iné polohovacie zariadenia, mikrofón, kamera pre **vstup údajov** a monitor, reproduktor, tlačiareň, kresliace zariadenie pre **výstup údajov**.

Príklady periférnych zariadení môžete nájsť aj v materiáloch k modulu Základy hardvérového a softvérového vybavenia počítača 2DG3.

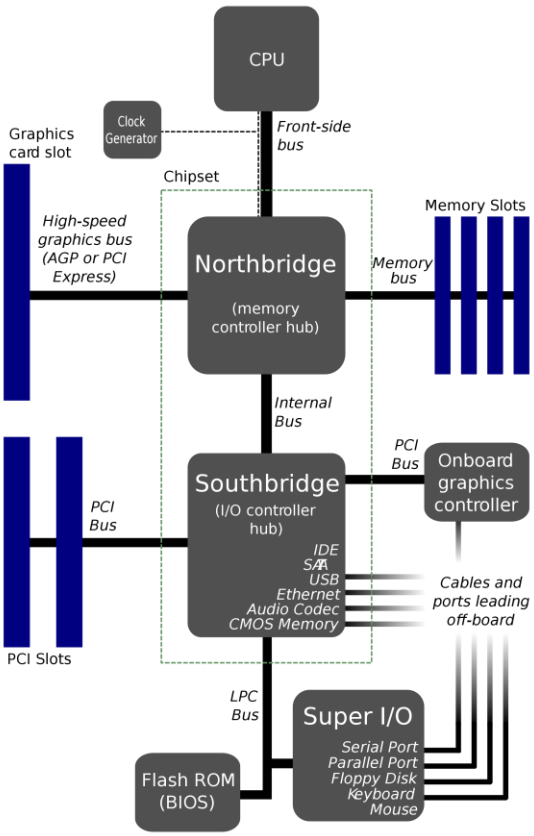
Počítač môže byť aj prostriedkom na prenos údajov a **komunikáciu s inými zariadeniami** – napr. elektronickými meracími zariadeniami, automatizovanými výrobnými linkami, ale aj inými počítačmi pripojenými pomocou počítačovej siete. Môže tiež využívať špecializované externé **zariadenia na ukladanie údajov**, ktoré presahujú rozsah operačnej pamäte. Ukladacie zariadenia môžu používať trvalo pripojené záznamové médiá, na ktoré je možné zapisovať a čítať zapísané údaje, prípadne údaje na záznamovom médiu prepisovať (magnetické disky, diskové polia). Pre archiváciu údajov môžeme používať tiež výmenné médiá (optické a páskové záznamové zariadenia).

Paleta periférnych zariadení je teda veľmi rôznorodá. Navyše sa môže časom meniť a rozširovať. Prepojovací systém (pôvodne realizovaný jedinou systémovou zbernicou), je preto v súčasných systémoch hierarchizovaný a štandardizovaný. V architektúrach Intel je oddelená časť, kde komunikuje CPU s pamäťou a zariadeniami s vysokou rýchlosťou prenosu údajov (grafickým adaptérom) od časti, komunikujúcej s pomalšími zariadeniami. Signály pre rýchle zariadenia sa spracúvajú v pamäťovom rozbočovači MCH (Memory Controller Hub) resp. severnom moste (*northbridge*), signály pre pomalšie zariadenia vo V/V rozbočovači ICH (I/O Controller Hub) resp. južnom moste (*southbridge*).



Rozmiestnenie rozhraní zberníc a prepojovacieho systému na základnej doske počítača

V najnovších procesoroch sa riadenie komunikácie s pamäťou (severný most) presúva priamo na procesorový čip. Táto časť prepojovacieho systému (procesor a vlastná vyrovnávacia pamäť) sa nazýva tiež **jadro**. Pri viac-jadrových systémoch sa tu navyše musí riešiť synchronizácia prístupu k spoločnej pamäti.



Oganizácia zberníc v súčasných typoch počítačov architektúr Intel
http://en.wikipedia.org/wiki/Motherboard

Prístup k ostatným perifériám sprostredkúvajú väčšinou zbernice PCI a PCIe. Už v prvej časti sme spomenuli, že PCIe funguje v podstate ako počítačová sieť so sériovým prenosom a že je ju možné rozširovať pomocou prepínačov a mostov aj ďalej. Mosty súčasne vykonávajú funkcie radičov prerušení aj radičov priamych prístupov do pamäte. Do pripojených zberníc púšťajú len signály, potrebné pre prácu s radičmi zariadení konkrétneho typu.

Univerzálne rozhrania

Radiče (adaptéry) periférnych zariadení pripájame buď priamo k rozšíreniam systémovej zbernice (PCI, PCIe) prostredníctvom zásuvných modulov, alebo sú pre určité typy zariadení zaintegrované na základnej doske (napr. v južnom moste). Niektoré typy radičov sú naopak zaintegrované priamo v zariadení (napr. disk).

Nesprávne sa niekedy rozhranie označuje ako "port" (čo vytvára problémy napr. pri programovaní).

Hranicu medzi zariadením a radičom (prípadne radičom a rozšírenou zbernicou) nazývame **rozhranie** (*interface*). Špecifikácia rozhrania spočíva v definovaní signálov a komunikačného protokolu, pomocou ktorých sa prenášajú informácie medzi zariadením a jeho radičom (väčšinou tiež aj typom **konektora**, ktorým sa zariadenie k radiču pripája). Rozhrania sú špecializované, určené na prenos špeciálnych signálov, napr. k monitoru, pre pripojenie klasickej myši a klávesnice a pod., alebo univerzálne (napr. USB, FireWire, SCSI), umožňujúce pripojiť celý rad zariadení. Radiče univerzálnych rozhraní bývajú súčasťou základnej dosky.

Pripomenieme, že zariadenia sa nepripájajú k systémovej zbernici procesora priamo, ale pomocou svojich radičov (resp. adaptérov). Úlohou radiča je komunikácia s procesorom a so zariadením, synchronizácia, ukladanie dočasných údajov (*buffering*), predspracovanie údajov, detekcia chýb.

Typickým **univerzálnym rozhraním** v starších typoch počítačov bolo sériové rozhranie (COM) štandardu RS-232C (resp. EIA/TIA-574) s 9 pólovým konektorom DB-9P, dovoľujúce prenosy do 115200 b/s súčasne smerom od počítača aj k počítaču. Radič sériového rozhrania obsahoval 12 registrov, zaberajúcich 8 čísel portov (niektoré čísla určili iný register pri čítaní a iný pri zápise). 8-bitové bloky údajov sa zapísali do posuvného vysielacieho registra, z ktorého radič postupným posúvaním odoslal údaje cez príslušný vodič rozhrania (TXD). Podobne posuvný register, určený na príjem, postupne posúval vstupné sériové bity z rozhrania (RXD) až do jeho zaplnenia, kde si vstup mohol prečítať procesor. Sériové rozhranie sa obyčajne používalo na asynchrónnu komunikáciu so zariadením, pripojeným k diaľkovému komunikačnému kanálu – modemom (ktorého úlohou bolo ďalej modulovať a spätne demodulovať signály do určeného prenosového pásma - typicky telefónneho rozsahu).

Konektory sériového a paralelného rozhrania starších typov počítačov

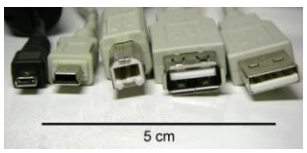
Paralelné rozhranie s 25 pólovým konektorom DB-25S protokolu Centronics (označované aj LPT – Line Printer Terminal), neskôr štandardu IEEE 1284, dovoľovalo paralelné prenosy 8 údajovými kanálmi, ktoré bolo možné nastaviť na prenos od počítača alebo k počítaču. Toto rozhranie sa používalo na pripojenie tlačiarní, skenerov, ploterov, ale aj na prepojenie dvoch počítačov v sieti LapLink. Dosahovalo prenosovú rýchlosť až 2 MB/s ale len do vzdialenosti 2 m.

V prepojovacom systéme súčasných počítačov sa pre kompatibilitu so staršími typmi rozhraní používa zbernica LPC (Low Pin Count).

Tieto rozhrania boli postupne nahradzované novým univerzálnym sériovým rozhraním **USB** (Universal Serial Bus) s prenosovými rýchlosťami 1,5 Mb/s a 12 Mb/s, neskôr vo verzii 2.0 s prenosom rýchlosťou až do 480 Mb/s. Okrem rýchlejších prenosov tento štandard umožňuje súčasné pripojenie viacerých zariadení (až 127) s rôznou rýchlosťou prenosu, pripájanie a odpájanie zariadení počas chodu systému, kontrolu prenesených údajov a dynamickú konfiguráciu zdrojov pre pripojené zariadenie (*Plug-and-Play*). Garantované synchrónne (izochrónne) prenosy údajov do 24 MB/s dovoľujú pripojenie videokamier, digitálnych zvukových zariadení (reproduktorov, mikrofónov), záznamových a prehrávacích zariadení zvuku a videa (CD, DVD). Verzia 3.0 ohlásená v roku 2008 pripúšťa prenosy rýchlosťou až 3,2 Gb/s (400 MB/s).

Špecifikáciu USB koordinuje USB Implementers Forum (www.usb.org)

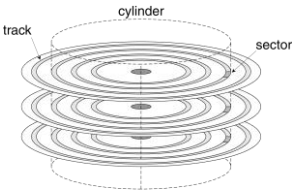
USB zbernicu je možné ďalej stromovito rozširovať pomocou USB rozbočovačov (*USB hub*), ktoré opakujú signály z jedného rozhrania na ostatné a vytvárajú tak pre koreňový USB radič dojem, že každé zariadenie je pripojené priamo k nemu. Komunikácia po USB zbernici prebieha prostredníctvom rámcov a mikrorámcov. Je možné posielať tiež žiadosti o prerušenia (od myši, klávesnice) a o priamy prístup do pamäte. USB rozhranie je realizované 4 pólovou zásuvkou (dva vodiče pre komunikáciu, ďalšie dva pre napájanie +5V a uzemnenie) viacerých typov.



USB mikro, mini, typ B, typ A (zásuvka a zástrčka)

Alternatívnym rýchlym univerzálnym sériovým rozhraním je tiež rozhranie IEEE 1394 FireWire (s maximálnou prenosovou rýchlosťou 400 Mb/s, vo vylepšených verziách 1394a a 1394b tiež až 3,2 Gb/s, podobne ako USB). V budúcnosti sa počíta tiež s pripojením optickým vláknom rozhraním IEEE P1394d.

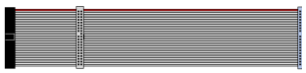
Externé pamäťové jednotky a ich rozhrania



Pevný disk a jeho štruktúra

Platne pevného disku sa otáčajú konštantnou rýchlosťou. Magnetické hlavy (schopné čítať i zapisovať magnetický záznam) sa pohybujú (plávajú) tesne po povrchu platní lineárne od kraja po stred disku. V každej z pozícií môžu pracovať údajmi na jednom cylindri.

V súčasnosti sa prechádza na veľkosť bloku 4096 B. Použitím lepšieho ECC kódovania (s veľkosťou 100 B) sa tak úspešnejšie detegujú, prípadne opravujú chyby (aj väčšie ako 50 B). Súčasné radiče aj tak čítajú disk po osmiciach blokov, ktoré ukladajú do svojej vyrovnávacej pamäte.



Prvé ATA jednotky používali na pripojenie plochý 40 žilový kábel a dosahovali prenosové rýchlosti do 60 kB/s (dnes je toto pripojenie označované ako Parallel ATA)

Pre uloženie údajov a programu procesor používa operačnú pamäť. Operačná pamäť má však len ohraničenú veľkosť a je energeticky závislá (volatilná) – neuchová svoj obsah pri prerušení dodávky elektrického prúdu. Tieto problémy rieši **externá pamäť** (alebo tiež sekundárna pamäť). Tá je nezávislá na zdroji napájania (*non-volatile*) a umožňuje (aj prostredníctvom výmenných médií) dosahovať takmer neobmedzenú kapacitu. Tvoria ju pevné disky, disketové jednotky, optické disky, magnetické pásky a iné pamäťové jednotky.

Externá pamäť sa pripája k počítaču cez štandardizované rozhrania. Rozhranie ATA (AT Attachment for Disk Drives) bolo vyvinuté pre počítače IBM PC AT so zbernicou ISA. Z dôvodov efektívnejšieho prístupu bol už tu radič disku presunutý priamo do zariadenia (IDE – Integrated Drive Electronics) a ATA je len rozšírením systémovej zbernice. ATA umožňuje pripojiť až dve jednotky pevných diskov, z ktorých jeden má vedúce postavenie (*master*) a druhý je mu podriadený (*slave*). Registre radičov obsadzujú ten istý adresový priestor, výber konkrétneho zariadenia sa robí nastavením bitu v registri jednotky. Signály teda zachytávajú obidve zariadenia, ale reaguje len jedno z nich.

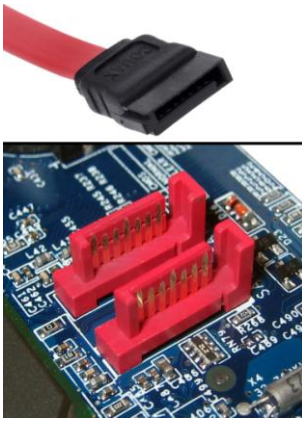
Pre prístup k diskom sa používa adresovanie CHS – trojicou čísel cylindra, hlavy a sektora. Od začiatku toto trojčíslie presne odrážalo reálne priestorové usporiadanie. **Cylinder** (s číslovaním od 0) odpovedal pozícii vystavenia hláv - určoval stopy, ktoré boli v danej pozícii hláv prístupné. Číslo **hlavy** (tiež od 0) určovalo, ktorá hlava bude aktívna (bude čítať resp. zapisovať) – určovalo teda povrch, s ktorým sa bude pracovať a spolu s číslom cylindra jedno medzikružie na tomto povrchu – **stopu** (*track*). Číslo **sektora** odpovedalo pozícii zápisu na tejto stope.

Údaje na diskových zariadeniach sú uložené po blokoch (veľkosti 512 B) – hovoríme tiež o zariadeniach s **blokovým prístupom** (čítať resp. zapisovať sa musí vždy celý blok). Ak chceme zmeniť len jeden bajt v bloku, je potrebné celý blok prečítať, urobiť zmenu a potom blok zase zapísať. Stopa disku je rozdelená na niekoľko sektorov, do ktorých sa zapisujú bloky údajov. Sektor je na stope vymedzený záhlavím (obsahujúcim základné informácie o pozícii a veľkosti sektora), potom nasleduje miesto pre 512 B údajov a zakončenie ECC (error correction code) – chybovým korekčným kódom. ECC umožňuje detegovať a opravovať chyby (obmedzené množstvo), ktoré by mohli vzniknúť pri čítaní bloku. Stopa stále začína špeciálnym sektorom s informáciami o stope, preto sektory sa pre používateľov číslujú až od 1. Sektory sa na disku vymedzujú pri **formátovaní**.

Prístup k bloku údajov signalizuje procesor zápisom CHS údajov do príslušných portov radiča a zvolením smeru prenosu (tiež zápisom jeho kódu do príslušného portu). Radič (pokiaľ nenájde zvolený blok v svojej vyrovnávacej pamäti) vystaví hlavy na požadovanú pozíciu C (relatívne vzhľadom k ich predchádzajúcej pozícii), aktivuje hlavu podľa súradnice H a čaká, kým sa disk nedostane do polohy, v ktorej pod hlavou začína štartovací sektor stopy. Táto poloha je signalizovaná elektronicky, resp. u starších zariadení excentrickou značkou alebo indexovým otvorom v pružných diskoch. Podľa obsahu záhlavia štartovacieho sektora skontroluje radič, či sú hlavy vystavené v správnej polohe. Potom aktivovaná hlava číta postupnosť sektorov na stope až príde určený sektor s požadovaným číslom, do ktorého hlava zapíše resp. prečíta požadovaný blok údajov.

CHS dáva možnosť adresovať celkom 65536 cylindrov, 16 hláv a 255 sektorov. Zväčšovaním kapacity diskov sa rýchle narazilo na hranice niektorých rozmerov, niektoré stopy (na okraji disku) sa začali zapisovať hustejšie, niektoré sektory mohli byť vyhradené ako chybné. Prešlo sa teda k virtuálnej adresácii, kde konkrétnu pozíciu na disku vyhľadáva radič v zariadení. Logicky je takto možné adresovať celkom 65536 x 16 x 255 sektorov čo je asi 137 GB. Dnes sa používa logická adresácia LBA (Logical Block Addressing), identifikujúca sektor 28 bitovým číslom, prípadne 32 bitová adresa rozšírenia ATAPI (AT Attachment Packet Interface) do kapacity až 2 TB (pre vyššie kapacity už bude potrebný 64 bitový systém).

Postupom času vzniklo niekoľko ATA štandardov umožňujúcich paralelné prenosy s podporou DMA až do prenosovej rýchlosti 133 MB/s. Kvôli ďalšiemu zvýšeniu rýchlosti (a odstráneniu množstva vodičov redukovaním z 80 na 7) sa prešlo k sériovým prenosom. Pôvodné rozhranie dostalo názov PATA (Parallel ATA). Nové rozhranie SATA (Serial ATA) dovoľuje pripojiť ku konektoru len jedno zariadenie, to však je možné počas prevádzky vymieňať (*hotplug*), prípadne ovplyvňovať jeho príkon (uspať). Rozšírenie AHCI (Advanced Host Controller Interface) umožňuje tiež optimalizovať prístup k sektorom a rôzne režimy zabezpečenia. Pokiaľ ich zariadenie nepodporuje, je možné zvoliť aj režim emulácie klasického ATA/IDE rozhrania.



Zástrčka a zásuvka sériového rozhrania SATA s prenosovou rýchlosťou až 6 Gb/s

Radič (hostiteľský adaptér) SATA rozhrania (je väčšinou súčasťou južného mosta) riadi viacero SATA pripojení. Kvôli dosahovaniu vyšších rýchlostí prístupu obsahuje tieňové registre všetkých pripojených radičov zariadení. Komunikuje s nimi prostredníctvom rámcov pomocou dvoch párov vodičov (podobne ako PCIe). V prvej verzii bolo možné prenášať údaje prenosovou rýchlosťou 150 MB/s, neskôr 300 MB/s resp. 600 MB/s (s rýchlosťou 6 Gb/s a kódovaním 10b/8b).

V súčasnosti najpoužívanejšie zariadenia externej pamäte sú pevné disky (HDD – Hard Disk Drive). Využívajú magnetický zápis na otáčajúce sa platne s magnetizovateľnými povrchmi. Platne sú uzavreté v pevnom puzdre a umožňuje mechanicky vysúvateľným hlavám zápis aj niekoľko stovák GB na platňu (v súčasnosti sú dostupné disky do kapacity 3 TB). Nevýhodou pevných diskov je mechanické obmedzenie, dané časom vystavenia hláv (pôvodne 600 ms, dnes asi 3 ms), ktoré sa čiastočne kompenzuje vyrovnávacou pamäťou priamo na diskovej mechanike. Platne sa otáčajú stálou rýchlosťou 5400 resp. 7200 otáčok/min.

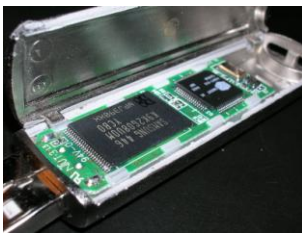
Vnútro SSD Solid-State Drive pamäte

Najrýchlejšie disky sa otáčajú rýchlosťou 10000 resp. 15000 otáčok/min a dosahujú sekvenčný prenos viac ako 1,6 Gb/s (s podporou vyrovnávacej pamäte na disku prenos do počítača až 300 MB/s). Používajú zbernicové rozhranie SCSI – Small Computer System Interface, prípadne jeho sériovú verziu SAS – Serial Attached SCSI.

Pre vysokorýchlostné prenosy sa prechádza k externým pamätiam typu SSD (Solid-State Drive) bez pohyblivých častí. Mechanické disky sú nahradené statickou pamäťou typu flash (schopnou uchovať elektrický náboj aj bez napájania elektrickou energiou), čím sa obchádza problém vystavenia hláv. Zatiaľ ich všeobecnému rozšíreniu bráni vyššia cena.

Externé pamäťové médiá možno pripájať aj prostredníctvom univerzálnych zberníc – v súčasnosti najčastejšie USB zbernicou (flash pamäťové jednotky, pamäťové karty MS, MMC, SD a iné).

Rozhrania SATA resp. SAS sa používajú aj pre zariadenia s výmennými pamäťovými médiami. Umožňujú čítanie z týchto médií, prípadne aj zápis a prepisovanie. Pôvodne sa používali magnetické výmenné médiá (magnetické pásky, magnetické pružné (*floppy*) disky), dnes väčšinou optické disky (CD, DVD, Blu-ray). Optické výmenné pamäťové médiá používajú laserový záznam na dlhú špirálu od stredu disku k jeho okrajom. Súvislý záznam tu nedovoľuje selektívne prepisovanie. Jednotlivé sektory v prepisovateľných variantoch môžu byť len zneplatňované, alebo sa musí kvôli prepisu najskôr premazať celé médium.



Polovodičové pamäte nezávislé na zdroji napätia (flash pamäte) s USB pripojením

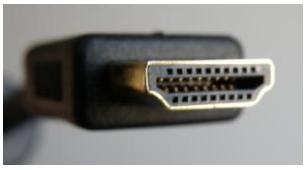
Grafický adaptér

Úlohou grafických adaptérov je vytvárať signály pre zobrazovacie zariadenie (monitor) na základe digitálnych údajov o farbe zobrazovaných bodov na obrazovkovom rastri. Informácie o bodoch sú prístupné v pamäti radiča, ku ktorej má prístup aj procesor pomocou špeciálneho premapovania niektorých adries operačnej pamäte. Radič sa pripája k procesoru pomocou rýchlejších typov zberníc (ISA, AGP, PCI), po ktorých je možné údaje prenášať s dostatočnou rýchlosťou (v najnovších typoch býva radič integrovaný priamo v procesorovom čipe).

Niekedy sa používa aj jednoduchšia paleta **Hi-color** s dvojbajtovou informáciou o farbe - 5 bitov pre červenú, 6 bitov pre zelenú – tú vie ľudský zrak najlepšie rozlíšiť - a 5 bitov pre modrú farbu.

Adaptér štandardizovaného analógového rozhrania VGA (Video Graphics Array) generuje videosignál v jednotlivých farebných zložkách (červenej, zelenej a modrej) pomocou 8 bitových digitálne-analógových prevodníkov. Zmiešaním odtieňov troch základných farieb možno na monitore znázorniť až 256x256x256 = 16 777 216 farebných odtieňov (paleta **True-color)**. Aj keď je rozlišovacia schopnosť farebných senzorov v ľudskom oku oveľa menšia, používajú sa dnes aj 30 a 48 bitové palety.

VGA rozhranie



Kábel k HDMI rozhraniu

Niektoré štandardizované VESA rozlíšenia
pre pomer strán 4:3
VGA 640x480
SVGA 800x600
XGA 1024x768
UXGA 1600x1200
pre pomer strán 16:9
WXGA 1280x720
HDTV 1920x1080

Podrobnosti o hardvérových rozhraniach možno dobre naštudovať z knihy [5].

Aktuality o PC zberniciach a rozhraniach nájdete aj na www.interfacebus.com.

Základné ovládače počítača (pre klávesnicu, systémový čas, myš, monitor v textovom režime, kontrolný reproduktor, disk) sú súčasťou programu BIOS (Basic Input/Output System), ktorý sa nachádza v malej ROM pamäti mapovanej na konci operačnej pamäte.

Procesor začína vykonávať po štarte inštrukcie práve tam. BIOS okrem testu základných funkcií počítača zapíše vstupné body týchto ovládačov do tabuľky vektorov prerušení a umožní tak interaktívne nastavenie konfiguračných parametrov počítača pred zavedením operačného systému.

Adaptér vysiela súvislý signál pre lúč CRT monitora postupne v úrovniach hodnôt farebných zložiek jednotlivých bodov v riadku, zatemní na čas prechodu lúča na nový riadok a nakoniec zatemní pre prechod lúča späť na začiatok obrazovky. Pomocou ďalších signálov radič synchronizuje vychyľovacie cievky monitora pre prechod lúča elektrónov na nasledujúci riadok (HSync) resp. nasledujúcu snímku (VSync). Pri 60 - 100 zobrazených snímok za sekundu ľudské oko zotrvačnosťou prestane vnímať postupnosť zobrazovania a jeho prekrývanie.

Digitálne LCD (Liquid Crystal Display) monitory nepoužívajú elektrónový lúč na aktiváciu fosforeskujúcich bodiek povrchu obrazovkovej elektrónky. Obraz vytvára matica farebných segmentov z tekutých kryštálov. Každý segment je možné adresovať a nastaviť pomocou elektrického prúdu jeho svietivosť (priehľadnosť). Zobrazovací mechanizmus monitora postupne vyhodnocuje analógový signál na vstupe a upravuje svietivosť jednotlivých segmentov obrazovky.

Pre digitálne monitory je lepšie prenášať priamo digitálny signál. Výhodnejšie je tiež neposielať celú informáciu o obrazovke, ale len zmeny v jednotlivých jej pixeloch. Takéto prenosy dokážu sprostredkovať digitálne rozhrania DVI (Digital Visual Interface), HDMI (High-Definition Multimedia Interface) alebo DP (DisplayPort) – posledné dva súčasne aj s informáciou o zvuku. Rozlíšenia zobrazovanej plochy sú štandardizované združením VESA (Video Electronic Standards Association) a udávajú počet bodov v riadku a počet zobrazovaných riadkov na obrazovke.

Výkonné grafické adaptéry obsahujú tiež vlastné grafické procesory – GPU (Graphics Processing Unit), ktoré sú schopné z objektového opisu zobrazovanej scény samostatne vypočítavať farby bodov povrchu objektov pri ich osvetlení, zisťovať prekrývanie, počítať polohy objektov po otočení a posunutí a mnoho iných.

Úloha ovládačov, začlenenie do operačných systémov

Jedinou možnosťou ako ovládať periférne zariadenia procesorom je pomocou inštrukcií IN a OUT cez porty príslušného radiča. Programátor ale často dopredu nepozná ani konkrétne čísla portov ani do detailu kódy jednotlivých funkcií zariadenia, ktoré treba do príslušných portov zapisovať. Navyše v inom počítačovom systéme, prípadne s iným typom zariadenia, sa môže ovládanie líšiť. Program už nebude prenositeľný na iný počítač.

Problémy prenositeľnosti programov medzi počítačmi tej istej architektúry s rôznymi konfiguráciami pripojených zariadení sa riešia pomocou ovládačov. Firemne napísaný **ovládač** (*driver*) pre konkrétne zariadenie (tvorí väčšinou súčasť dodávky zariadenia) je program (súbor procedúr), pomocou ktorých je možné toto zariadenie ovládať. Aby bol ovládač použiteľný, musí sa vopred **inštalovať**. Inštalácia spočíva v jeho uložení do operačnej pamäte a začlenení do operačného systému. Niekedy zmena v konfigurácii operačného systému vyžaduje aj jeho reštart.

Inštalovaný ovládač môžeme použiť prostredníctvom služieb operačného systému, alebo priamo volaním jeho procedúr (inštrukciami CALL) alebo transparentnejšie pomocou volania softvérových prerušení (inštrukcia INT) – vtedy pri inštalácii potrebujeme zapísať do tabuľky vektorov prerušení aj vstupný bod tohto ovládača. Parametre volania procedúry odovzdáme v registroch, prípadne uložíme do zásobníka. Pokiaľ dodržíme konvencie o priradení čísel softvérových prerušení konkrétnym zariadeniam, dosiahneme **prenositeľnosť** nášho programu aj na systémy s inými typmi zariadení a inými ovládačmi.

Aktivita 3

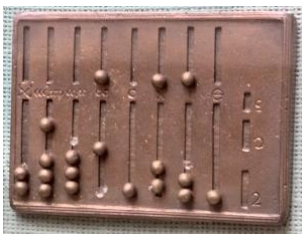
Ukážka práce s ovládačom grafického adaptéra, nastavovanie režimov pomocou prerušení, odovzdávanie parametrov prostredníctvom registrov resp. naplnením zásobníka.

Demonštrácia nastavenia parametrov v systéme BIOS.

4 História vývoja počítačových systémov

Výroba prvých nástrojov, schopných pomáhať pri spracovávaní informácie súvisí s rozvojom fyzickej reprezentácie číselných údajov. Od začiatku to bolo len priame priradenie objektov prstom, ktoré pomáhali pri počítaní. Počítanie uľahčovali neskôr záznamy na stenách jaskyne, zárezy na paliciach.

S vynájdením pozičných sústav sú spojené **abakusy**. O ich používaní v Mezopotámii siahajú zmienky až do obdobia 2400 rokov pred Kristom. Používali sa tiež v starovekom Egypte, Grécku a Ríme v tvare drevených alebo hlinených doštičiek, do ktorých sa vkladali kamienky (*calculli*), slúžiace na reprezentáciu čísel. Vhodnou manipuláciou s kamienkami bolo možné realizovať aj zložitejšie výpočty.



Rekonštrukcia rímskeho abakusu

Indický pozičný dekadický zápis a príslušné algoritmy pre násobenie a delenie pomocou abakusu sa dostali do Európy prostredníctvom traktátu Kniha o indickom počítaní (*Kitáb al-džám'a wa-l-tafríq bil-hisáb al-hindi*) z roku 825 od perzského matematika Al-Chvárizmího (Abū ʿAbdallāh Muḥammad ibn Mūsā al-Khwārizmī). Traktát v roku 1145 preložil Robert z Chesteru do latinčiny pod názvom *Algorithmi de numero indorum* a spolu s traktátom o riešení lineárnych a kvadratických rovníc (*Kitab al-Jabr wa-l-Muqabala)* podstatne ovplyvnil rozvoj európskej matematiky (pozri aj [11]). S menom Al-Chvárizmí je spojený aj vznik pojmu **algoritmus** (v latinských textoch sa formula *Dixit Algorizmi* – tak riekol Algorizmi – používala na konci dôkazu ako potvrdenie jeho jasnosti a spoľahlivosti).

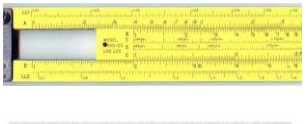
Fragment mechanizmu z Antikythery

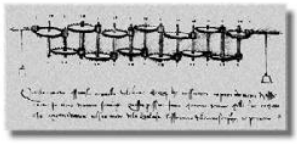
Za dômyselné analógové mechanické zariadenia na spracovanie astronomických výpočtov možno považovať rôzne **astrolaby**. Kuriózne sú nálezy fragmentov starogréckeho astrolabu z vraku lode Antikythera z druhého storočia pred Kristom (zložitosťou porovnateľný s hodinami 18. storočia). Astrolaby uľahčovali astronomické a geografické výpočty arabským učencom od začiatku 11. storočia.

Programovateľné astronomické hodiny (orloj) Al-Jazari (1206)

Funkcie dobových mechanizmov zaznamenal arabský konštruktér, matematik a astronóm Al-Jazari (Abū al-ʿIz Ibn Ismāʿīl ibn al-Razāz al-Jazarī). V traktáte Kniha poznatkov o dômyselných mechanických zariadeniach (*Kitáb fí ma'rifat al-hiyal al-handasiyya)* opísal konštrukciu viac ako 50 zariadení vrátane mechanických vodných hodín, kódových zámkov, robotov. Jeho astronomické hodiny na vodný pohon, zabudované do orloja (1206), sa považujú za prvé programovateľné analógové zariadenie. Boli schopné zobraziť zverokruh, dráhy slnka, mesiaca, jeho fázy a každú hodinu piati robotickí hudobníci zahrali nastavenú muziku. Hodiny bolo možné nastavovať podľa aktuálnej dĺžky dňa.

Špeciálnym spôsobom reprezentácie čísel pomocou ich **logaritmov** sa zaoberal škótsky matematik John Napier (1620). Zápis čísel v logaritmickej forme umožnil previesť operácie násobenia a delenia na jednoduché pripočítanie a odpočítanie, ktoré je potom možné mechanicky interpretovať posúvaním bežca logaritmického pravítka. **Logaritmické pravítko** (*slide rule*) skonštruoval o niekoľko rokov neskôr William Oughtred (1632). Až do čias vreckových kalkulačiek sa stalo najpoužívanejším výpočtovým nástrojom pre vedcov a inžinierov.





Leonardo da Vinci - kresba

Mechanické výpočtové zariadenia

Mechanické číslicové stroje boli založené na princípe reprezentácie čísla v dekadickom zápise postupnosťou desaťzubových koliesok, pričom jednotlivé cifry boli znázornené ich pozíciou. Otočenie kolieska pre jednu cifru zapríčinilo posun kolieska vyššieho rádu o jeden zub. Tento princíp bol opísaný už Herónom z Alexandrie (1. storočie). Nákresy zariadení, používajúcich tento princíp, možno nájsť už u Leonarda da Vinciho (okolo roku 1500). K ich realizácii sa však nedostal.

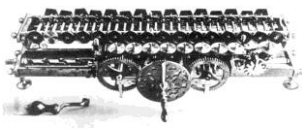
Shickardove počítacie hodiny (1623)

Plnú funkčnú **sčítačku** a **odčítačku** (počítacie hodiny – používali kolieska hodinového stroja) zostrojil Wilhelm Shickard v roku 1623, ktorá však pri požiari zhorela. Jej konštrukčné nákresy sa našli až v roku 1957 v Schickardovom liste Keplerovi, v ktorom mu vysvetľuje ako použiť stroj na výpočet astronomických tabuliek.

Pascaline - sčítačka a odčítačka (1642)

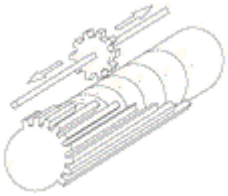
Shickardove hodiny umožňovali aj poloautomatické násobenie a delenie (využívali postup, navrhnutý J. Napierom, známy ako Napier's Bones).

Blaise Pascal skonštruoval mechanickú sčítačku a odčítačku v roku 1642 (mal 19 rokov), keď sa nemohol dívať, ako jeho otec, daňový kontrolór, spočítaval ručne stĺpce čísel dlho do noci. Pascal navrhol a skonštruoval niekoľko desiatok modelov sčítacích strojov. Svoj definitívny model patentoval v roku 1645 pod názvom *Pascaline*. Vstup sa uskutočňoval vytáčaním čísel, násobenie a delenie sa dalo urobiť viacnásobnými sčítaniami a odčítaniami.



Leibnitz (1672)

Ťažkopádne delenie a násobenie odstránil Gottfried Wilhelm Leibnitz, ktorý v roku 1672 navrhol a neskôr aj zostrojil svoj kalkulátor, kde použil na **násobenie** a **delenie** ozubené kolieska so stupňovitou šírkou zubov, neskôr tiež cylindrické valce. Pri násobení a delení sa otáčacia kľučka posúvala v závislosti od rádu cifry spracovávaného čísla. Kalkulátor sa podľa tohto mechanizmu nazýval tiež "*Stepped reckoner*". Leibnitz sa ako prvý zaoberal aj možnosťami binárnej aritmetiky – a jej využití v kalkulátoroch.

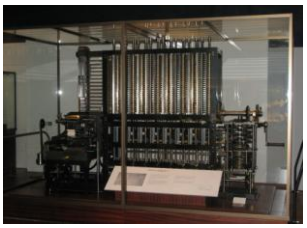


Leibnitzov valec

Pre potreby rýchlych výpočtov v astronómii (tabuľky pozícií Mesiaca, výpočet dráhy Neptúna), fyzike a rozvíjajúcej sa technike (logaritmické tabuľky) sa komerčne presadil až *Arithmometer* Charlesa Thomasa (1820), ktorý dokázal sčítať, odčítať, násobiť a deliť. Výpočty sa realizovali otáčaním ozubených koliesok s podobným princípom ako Leibnitz, používal tiež Leibnitzov valec a princíp postupného upravovania výsledku posúvaním vozíka a otáčaním kľučky prístroja.

Thomasov Arithmometer (s pohyblivým vozíkom)

Princíp ozubených koliesok pri konštrukcii mechanických výpočtových zariadení priviedol k dokonalosti anglický matematik Charles Babbage. Jeho prvým cieľom bolo skvalitniť prácu výpočtárom (angl. *computers*) spresnením existujúcich logaritmických tabuliek. V tom čase už bol známy princíp Weierstrassovej aproximácie spojitých funkcií pomocou polynómov. Svoj **diferenčný stroj** (*Difference Engine*) preto navrhol tak, aby vedel Newtonovou diferenčnou metódou postupne počítať (tabelovať) hodnoty polynómov so zadanými koeficientami (každý krok tabelácie vyžaduje toľko sčítaní, koľko je stupeň polynómu).



Diferenčný stroj rekonštruovaný podľa Babbageovych plánov v londýnskom múzeu

Prototypom pre kvadratické polynómy presvedčil v roku 1822 britskú vládu, ktorá v tom čase našla v používaných navigačných tabuľkách viac ako tisíc numerických chýb, aby financovala stavbu stroja pre tabeláciu polynómov šiesteho stupňa s tlačou výsledkov na papier. Po desiatich rokoch neúspechov však vláda financovanie zastavila a Babbage od stavby ustúpil (podľa plánov mal stroj pozostávať z 25000 dielov a vážiť viac ako 13 ton). Modifikáciu diferenčného stroja pre polynómy štvrtého stupňa neskôr zostrojili vo Švédsku. Diferenčný stroj podľa pôvodných Babbageovych plánov bol zostrojený až v roku 1991 a možno ho nájsť v londýnskom vedeckom múzeu. Svojou presnosťou na 31 cifier prekoná mnohé vreckové kalkulačky.

Tkáčsky stroj, riadený dierkovanými kartičkami (Jacquard 1801)

Babbagea neodradili neúspechy, naopak, začal rozmýšľať o novom zariadení, ktoré nazval **analytický stroj** (*Analytic Engine*). V roku 1837 sa tak fakticky dopracoval k myšlienke univerzálneho číslicového počítača, ktorý mal obsahovať vstup a výstup (na dierkovaných kartičkách), pamäť ("*store*" - v ktorej malo byť 50 riadkov po 100 ozubených koliesok), aritmetickú jednotku (nazývanú "*mill*" a vytvorenú z ozubených koliesok a prevodov). Riadenie výpočtov sa malo realizovať tiež prostredníctvom dierkovaných kartičiek, známych v tom čase ako riadiaci prvok pre Jacquardove tkáčske stroje. Analytický stroj mal spracovávať programy zostavené z kartičiek pre premenné, kartičiek pre operácie a špeciálnych "indexových" a "kombinatorických" kartičiek pre riadenie cyklov. Babbage strávil veľa času vylepšovaním svojho "mlynu" a zefektívnením realizácie aritmetických operácií (+,-,*,/). Uvažoval aj o dvojnásobnej aritmetike. Analytical Engine mal byť na ručný pohon, aj keď Babbage uvažoval aj o parnom pohone.

Dôležitý je článok, ktorý na základe Babbageovho výkladu a nákresov napísal taliansky inžinier Luigi Menabrea (1842). Tento článok v roku 1843 preložila do angličtiny a doplnila významnými poznámkami Babbageova priateľka Augusta Ada, dcéra Lorda Byrona (http://www.fourmilab.ch/babbage/sketch.html). Adine poznámky, dvakrát tak dlhé ako Menabreaov článok, vznikli v spolupráci s Babbageom. Týkali sa hlavne principiálnych možností a ohraničení navrhovaného stroja (Ada pripojila aj program na výpočet Bernoulliho čísel). Babbage si uvedomoval ďalekosiahle možnosti stroja, aj keď ho stále považoval za stroj na zostavovanie tabuliek. Vedel, že môže napríklad "riešiť ľubovoľnú algebrickú rovnicu" a dokonca "pripravovať (a dierovať) vlastné programy". Uvedomoval si aj potrebu podmienených príkazov pre realizáciu cyklov, no univerzálnosť stroja (v zmysle Turinga) ešte presnejšie neformuloval.

Model časti Babbageovho analytického stroja

V návrhu analytického stroja pokračoval aj Babbageov syn Henry. Percy Ludgate navrhol v roku 1915 vlastný analytický stroj, riadený postupnosťou štvoradresových inštrukcií na diernej páske. Leonardo Torres rozpracoval schému pre podmienený príkaz a uvedomoval si možnosť zostrojiť jeho elektromechanickú verziu.

Diferenciálny analyzátor (Bush 1876, 1930)

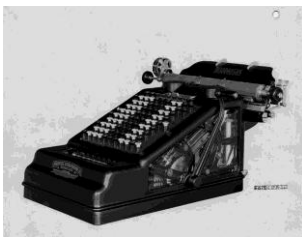
Je zaujímavé, že poprední fyzici 19. storočia sa neinšpirovali Babbageovými ideami, ale obrátili svoju pozornosť na konštrukciu podstatne rýchlejších, aj keď jednoúčelových **analógových strojov,** využívajúcich nový zdroj energie – elektrinu. Analógové stroje pracovali na princípe analógie meraných a modelovaných veličín a veľkosti prúdu a napätia. Patrí sem Maxwellov integrátor (1855), Kelvinov harmonický analyzátor, ktorý bol prvým strojom na výpočet Fourierovych koeficientov a tiež Kelvinove (Thompsonove) a Bushove diferenciálne analyzátory (1876 a 1930). Bushov diferenciálny analyzátor (*Differential analyser*) dokázal riešiť diferenciálne rovnice s osemnástimi nezávislými premennými. Používal sa väčšinou na výpočet delostreleckých tabuliek, neskôr dráh balistických rakiet.

Tabelačný stroj (Herman Hollerith 1889)

Význam automatizácie v hromadnom spracovaní údajov sa prejavil pri pravidelných sčítaniach ľudu s desaťročnou periódou, ktoré boli deklarované v ústave USA. Ukázalo sa, že výsledky desiateho sčítania z roku 1880 (50 miliónov obyvateľov) neboli spracované ani v roku 1887. Bolo zrejmé, že plánované sčítanie v roku 1890 by existujúcou technológiou manuálneho sčítania nezvládli. Vo vypísanej súťaži vyhral diernoštítkový sčítavací stroj Hermana Holleritha. Využíval princíp diernych štítkov s elektrickou indikáciou otvorov a registráciou pomocou elektromagnetických krokových relé. Pomocou neho bolo sčítanie ľudu v USA v roku 1890 spracované za 6 týždňov s predbežným výsledkom 62 622 250 (spresneným po dvoch rokoch na 62 947 714).

Príprava diernych štítkov zo sčitacích hárkov

Ťažkosti so sčítaním v roku 1890 predstavujú asi prvý prípad, kedy si ľudia uvedomili, že svet sa stáva príliš zložitý a že presahuje schopnosti ľudského mozgu analyzovať ho bez pomoci strojov. Hollerith založil veľmi úspešnú spoločnosť Tabulating Machine Company. Tá sa neskôr zlúčila s ďalšími dvomi spoločnosťami a vytvorili Computing Tabulating Recording Corporation. Pod vedením Thomasa J. Watsona sa v roku 1924 spoločnosť premenovala na International Business Machines (IBM), ktorá sa sústredila na výrobu rýchlych tabelačných strojov.

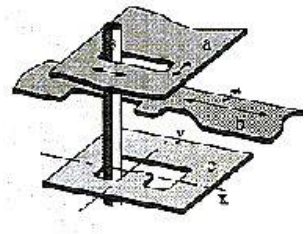


Začiatkom 20. storočia sa začali v mechanických kalkulátoroch, účtovacích strojoch a pokladniach postupne presadzovať elektrické motory namiesto ručného otáčania kľukou. Vo veľkom sa začali vyrábať stolné elektromechanické kalkulátory, schopné sčítať, odčítať, násobiť a deliť. Slovo "computer" označovalo pracovnú pozíciu, v ktorej pracovník (väčšinou žena) vykonával rôzne matematické (účtovné) výpočty pomocou týchto mechanických kalkulátorov.

Pracovňa "počítačov" (1933)

Vlastnosti **elektromagnetického relé**, ktoré je možné pomocou elektrickej energie nastaviť a udržať v dvoch rôznych stavoch inšpirovali rozvoj binárnych a digitálnych technológií. Claude Shannon (1937) ukázal, že pomocou elektromagnetických relé je možné realizovať akúkoľvek logickú funkciu.
(http://dspace.mit.edu/bitstream/handle/1721.1/11173/34541425.pdf?sequence=1)

Z1 v obývačke rodičov (Konrad Zuse 1938)



Kovové pliešky s otvormi ako základný konštrukčný prvok Zuseho Z1

Zuseho spôsob zápisu čísel s pohyblivou rádovou čiarkou je základom štandardu IEEE 754.

Rekonštruovaný Z3 v múzeu v Mníchove

Mark I (1944)

Jedným z prvých, ktorí sa začali zaoberať využitím relé na konštrukciu kalkulátorov, bol Konrad Zuse. Ako inžinier sa stretával s mnohými nudnými výpočtami, ktoré si chcel zjednodušiť použitím elektromechanického zariadenia. V roku 1938 zostrojil v obývačke u rodičov prvý binárny, elektricky poháňaný, mechanický kalkulátor Z1 s obmedzenými možnosťami programovania (9 inštrukcií) pomocou dierovaného filmového pásu (35 mm). Ako základné konštrukčné elementy použil drobné kovové pliešky (patentoval ich v roku 1936). Bol to prvý kalkulátor, ktorý používal Booleovu logiku a binárnu reprezentáciu reálnych čísel s plávajúcou rádovou čiarkou.

Za Z1 (ktorý bol ešte dosť nespoľahlivý v dôsledku nedostatočne presných komponentov) nasledovali Z2 (revidovaná Z1 s telefónnymi elektromagnetickými relé v roku 1939) a Z3, ako jeho vylepšenie, v roku 1941. Zuse bol v tom čase (v čase 2. svetovej vojny) už v totálnej intelektuálnej izolácii a pracoval úplne nezávisle od ostatných svetových konštruktérov a matematikov. Napriek tomu bol Z3 **prvým reálne fungujúcim univerzálnym počítačom** (v zmysle Turingovych definícií z roku 1936) na svete. Pozostával z 2000 relé, reprezentujúcich 22 bitové čísla. Počítač Z3 bol riadený programom (presnejšie lineárnym aritmetickým programom – bez podmienených príkazov), ktorý sa skladal z 8 bitových jednoadresových inštrukcií na diernej páske. Aritmetická jednotka (600 elmg relé) pracovala v binárnej sústave, v pohyblivej rádovej čiarke s dĺžkou slova 22 bitov (znamienko, 7 bitov exponent a 14 bitov mantisa), so zabudovanými operáciami +,-,*,/, √ a násobenie konštantami 2, 0.5, 10, 0.1, -1. Pri časovacej frekvencii 5,3 Hz trvalo sčítanie 0,8 s, násobenie 3 sekundy. Pamäť mala 64 22-bitových slov, vstup bol klávesnicou, výstupom bol lampový display so 4 desiatkovými ciframi a desatinnou čiarkou.

V roku 1943 dostal Zuse objednávku ministerstva letectva, ktoré podporilo dokončenie Z3 a výstavbu ďalších počítačov S1 a S2, určených na testovanie parametrov krídel vo výrobných leteckých závodoch. Ďalej pracoval na univerzálnom modeli Z4 s 32 bitovým slovom a mechanickou pamäťou (ako v Z1), ktorý sa mu podarilo vyviezť v roku 1945 do Švajčiarska (ostatné modely boli zničené pri bombardovaní) a používali ho na ETH v Zürichu až do roku 1955 (dlhú dobu bol jediným funkčným počítačom v Európe). V roku 1945 rozpracoval Zuse prvý programovací jazyk **Plankalkül** s viacrozmernými poľami, priraďovacími, podmienenými a iteratívnymi príkazmi a procedúrami, ktorý ovplyvnil konštrukciu jazyka ALGOL 58.

Babbageovymi prácami sa inšpiroval Howard Aiken, fyzik z Harvardu, keď v svojej dizertácii narazil na diferenciálne rovnice, ktoré sa dali riešiť len zložitými výpočtami. Pre svoj návrh počítača získal v roku 1939 T. J. Watsona, vtedajšieho šéfa IBM, ktorý návrh podporil finančne (milión dolárov) aj spolupracovníkmi. Aikenov projekt považovala firma IBM za demonštráciu svojich technických možností a bol to jej prvý vstup do sveta výpočtovej techniky (dovtedy vyrábala len diernoštítkové stroje).

Aiken zostrojil monštrum, poháňané elektromotorom s výkonom 3,7 kW, napojeným na dlhý hriadeľ (16 m), sprostredkujúci pohon jednotlivých častí počítača, obsahujúceho 765 000 elektromechanických prvkov. Počítač pracoval v desiatkovej sústave s pevnou čiarkou, v statickej pamäti bolo možné manuálne nastaviť 60 konštánt, dynamickú časť tvorilo 72 registrov na prácu s 23 miestnymi číslami. Registre, v ktorých prebiehal výpočet, boli realizované elektromechanicky ovládanými kolieskami. Násobenie trvalo asi 6 sekúnd, delenie 15 sekúnd a výpočet funkcie sínus viac ako minútu.

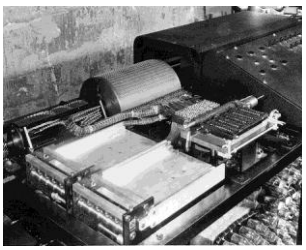
Počítač bol riadený inštrukciami na 24-stopej papierovej diernej páske. Podľa Babbageovho vzoru Aiken oddelil pamäť pre údaje a riadiaci program, čo dalo za vznik tzv. Harvardskej architektúry. Pôvodne neobsahoval ani možnosť podmienených skokov, cykly sa realizovali zlepením papierovej pásky do slučky. Funkčný počítač bol prezentovaný v roku 1944 a prevezený z IBM na Harvard. Aiken zmenil pôvodný názov ASCC (Automatic Sequence Controlled Calculator) na Harvard Mark I. Jeho nasledovník Mark II už nepoužíval mechanické prvky (obsahoval len 13 000 relé) a vedel násobiť za 0,25 s.

Elektronické počítače prvej generácie

Elektromechanické počítače sa síce vyrábali až do 50-tych rokov, ale ich budúcnosť vzhľadom k veľkým oneskoreniam mechanických prvkov relé, opotrebovaniu súčiastok a vysokou spotrebou energie prestala byť perspektívna vynálezom **elektrónky** (*vacuum tube*), ktorá umožnila porovnateľné logické funkcie realizovať oveľa vyššou rýchlosťou.

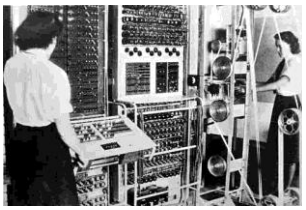
Replika ABC na Iowa State University

Prvé jednoduché elektronické výpočtové zariadenie predstavil John Athanasoff v roku 1939 na Iowa State University. Vďaka grantu ho spolu so svojim doktorandom Cliffordom Berrym dobudovali na zariadenie na riešenie systémov lineárnych rovníc a predstavili ho v roku 1941 ako Atanasoff-Berry Computer (ABC). Používal binárnu aritmetiku, realizovanú logickými obvodmi z 280 elektrónok. Na uchovávanie výsledkov použil obnoviteľnú kondenzátorovú pamäť (podobnú dnešnej DRAM) na rotujúcich bubnoch s kapacitou 60 50-bitových čísel v binárnom zápise. Zariadenie skúmal aj John Mauchly a niektoré myšlienky prevzal aj pre ENIAC.



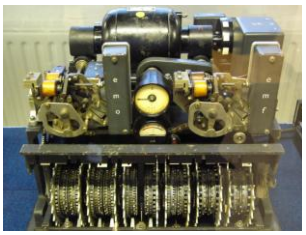
Detail ABC s bubnovou DRAM pamäťou (1941)

Po elektronickom zariadení siahli aj britskí lúštitelia šifrovaných nemeckých správ v Bletchley Park počas 2. svetovej vojny. Po úspechoch s elektromechanickými "bombami" na lúštenie správ zo zariadení Enigma zostrojil tím vo veľkej tajnosti elektronický počítač, určený na kryptoanalýzu ďalekopisných správ zo zariadení Lorenz SZ42 (čo boli rotorové šifrátory, zabezpečujúce 5 bitové ďalekopisné znaky Vernamovou šifrou). Prvý prototyp Colossus Mark 1 začal pracovať začiatkom roku 1944, po ňom sa postupne pridávalo deväť vylepšených Colossus Mark 2 až do konca vojny. Colossus Mark 2 obsahoval 2400 elektrónok, jeho činnosť spočívala v simulácii šifrátora pre veľké množstvo potenciálnych kľúčov. Tie boli zadávané na diernej páske a rýchlosť výpočtu ohraničovala práve rýchlosť čítania pásky. Pokiaľ by sa použili len elektromechanické prvky, rýchlosť snímania nemohla presiahnuť 2000 znakov/s. Pri použití elektroniky bola dosiahnutá rýchlosť 5000 znakov/s, v testoch až 9700 znakov/s. Uložený program sa síce na prepájacích paneloch dal meniť, ale konštrukcia neumožňovala realizovať všetky funkcie univerzálneho počítača.



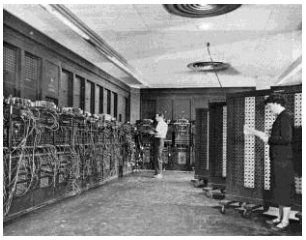
Colossus (1944)

Prvý univerzálny elektronický počítač ENIAC (Electronic Numerical Integrator And Computer) bol postavený na Pensylvánskej univerzite pod vedením Johna Mauchlyho a J. Prespera Eckerta v rokoch 1943 – 1945 na objednávku Balistického výskumného laboratória (*Ballistic Research Laboratory*). Počítač obsahoval 17 468 elektrónok, 70 000 odporov, 10 000 kondenzátorov, bol 24 m dlhý, vážil 30 ton a spotreboval 150 kW elektrickej energie.



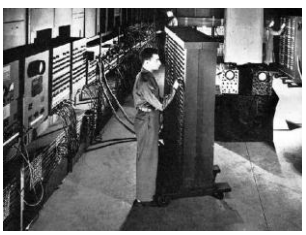
Šifrátor Lorenz SZ42

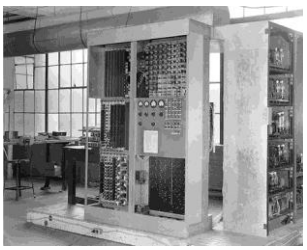
ENIAC pracoval s desaťcifernými číslami v desiatkovej sústave, každá cifra využívala 36 elektrónok. Desať z nich tvorilo kruhový posuvný register, riadený impulzmi. Po posune cez celý kruh sa generoval impulz pre cifru vyššieho rádu, čím sa simulovalo chovanie ozubených koliesok v mechanickom počítadle. Mal 20 akumulátorov (pre sčítanie a odčítanie), v ktorých bol schopný urobiť 5000 operácií za sekundu. V špeciálnej jednotke so štvoricou akumulátorov bolo možné vykonať 385 násobení za sekundu (teda 2000 krát viac ako Mark I), resp. v ďalšej 40 delení alebo 3 odmocnenia za sekundu. Programovanie spočívalo v prepájaní týchto jednotiek medzi sebou, prípadne s ručne nastavovateľnou pamäťou pre 312 hodnôt. Vstup a výstup bol na diernych štítkoch (zostavený v spolupráci s IBM). Pôvodné obavy z veľkej pravdepodobnosti zlyhania elektrónok sa nenaplnili - poruchy nastávali po odstránení počiatočných problémov priemerne raz za dva dni, dokonca sa dosiahlo až 116 hodín bez poruchy. S viacerými prestavbami sa v balistickom laboratóriu využíval ENIAC až do roku 1955.



ENIAC (1946)

Už v období ukončovania prác na ENIACu bolo jasné, že je možné zostrojiť ďaleko lepší počítač. Vo veľmi pracovnej atmosfére vznikol celý rad významných nápadov. Veľkým nedostatkom ENIACu bolo veľmi zložité programovanie, ktoré v podstate spočívalo v "predrôtovaní". ENIAC bol pritom paralelný počítač, každý register mal vlastné riadenie, ktorá bolo treba zosynchronizovať. Herman Goldstine, ktorý bol jedným z vedúcich ENIACovho tímu v BRL, sa náhodou stretol s Johnom von Neumannom, ktorý vtedy spolupracoval na vývoji atómovej bomby a silne potreboval počítač na riešenie parciálnych diferenciálnych rovníc. Bol tak nadšený rýchlosťou





EDVAC (Pennsylvania 1951)

SSEM (Manchester 1948) s výstupom na obrazovku

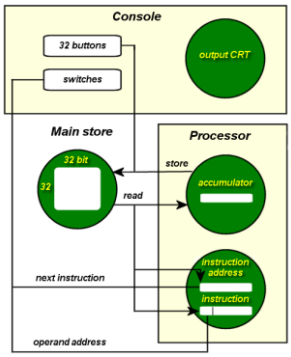
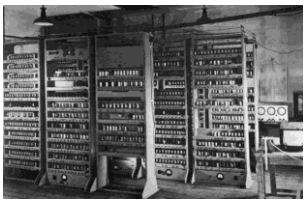
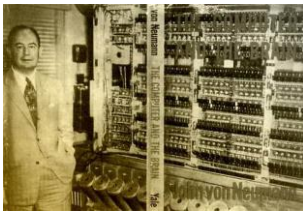


Schéma SSEM so štyrmi obrazovkovými elektrónkami



EDSAC (Cambridge 1949)



John von Neumann IAS (Princeton 1952)

ENIACu, že hneď začal spolupracovať ako konzultant s ENIACovým tímom a stal sa vedúcou postavou v diskusiách, z ktorých vznikla predstava nového počítača, nazývaného EDVAC (Electronic Discrete Variable Automatic Computer), so zapamätaným programom a s pamäťou na oneskorovacích linkách (to bola Eckertova idea, ktorá umožnila 100 násobne zredukovať počet elektrónok pre pamäť). Von Neumann obrátil pozornosť ENIACovho teamu na problémy logického návrhu počítača. Výsledkom diskusie bola správa *First draft of a report on the EDVAC* [12] napísaná von Neumannom (30. 6. 1945), v ktorej sa jasne poukazuje na:

- potrebu používať binárnu aritmetiku,
- potrebu sekvenčného spracovania inštrukcií,
- potrebu veľkej pamäte ako pre čísla, tak aj pre inštrukcie.

Táto správa sa čoskoro stala známa po celom svete a začala modernú éru počítačov ako systémov na uchovávanie a spracovanie informácie.

EDVAC bol tiež postavený za peniaze Ballistic Research Laboratory (podobne ako ENIAC za cca $500000). Vstup a výstup bol realizovaný magnetickými páskami. Riadiaca jednotka riadila prenos inštrukcií a údajov medzi zdvojenou pamäťou (64 akustických oneskorovacích jednotiek po 8 44-bitových slov – spolu viac ako 5 kB) a zdvojenou aritmetickou jednotkou. EDVAC mal už len 6000 elektrónok, spotreboval 56 kW a vážil skoro 8 ton. Po problémoch s patentmi začal pracovať až v roku 1951 a bol intenzívne využívaný až do roku 1961.

Prvým funkčným počítačom s **vloženým programom** v pamäti sa tak stal Small-Scale Experimental Machine (SSEM), dokončený v roku 1948 na univerzite v Manchestri dvojicou Frederic Williams, Tom Kilburn. Nebol určený na praktické účely, ale na testovanie pamätí, využívajúcich sekundárne emisie v obrazovkových elektrónkach - tzv. *Williams tube*. Snahou bolo vyriešiť hlavný problém von Nemannovej koncepcie – nedostatok dostatočne rýchlej a spoľahlivej pamäte pre ukladanie programu. V SSEM testovaná pamäť o kapacite 32 32-bitových slov obsahovala údaje aj program, ktorý sa postupne vykonával v jednoduchej aritmetickej jednotke, schopnej realizovať 8 rôznych inštrukcií. Prvý program na hľadanie najvyššieho deliteľa čísla 2^{18}, zapísaný do 17 inštrukcií a 8 slov pre ukladanie medzivýsledkov, bežal 52 minút. Počítač vážil tonu a na aritmetiku spotreboval 550 elektrónok. Jedna obrazovková elektrónka slúžila aj ako výstupné zariadenie.

V roku 1949 zostrojil Maurice Wilkes na univerzite v Cambridge podľa projektu EDVAC reálne používaný univerzálny počítač EDSAC (Electronic Delay Storage Automatic Calculator). Obsahoval 1024 18-bitových pamäťových miest, vstup z diernej pásky a výstup na obrazovkovú elektrónku. Používal 5-bitový inštrukčný kód a aj jednoduchý assembler. V roku 1951 bol použitý na nájdenie do toho času najväčšieho 79-ciferného prvočísla. V roku 1952 bola na ňom naprogramovaná prvá hra OXO (tic-tac-toe) s výstupom na obrazovku.

Aktivita 4

Pozrite ukážku programu pre EDSAC na diernej páske na http://www.cl.cam.ac.uk/~mr10/Edsac/edsacposter.pdf.

Na stránke http://www.dcs.warwick.ac.uk/~edsac/ vyskúšajte simulátor činnosti EDSAC aj s výstupom na obrazovku.

John von Neumann pracoval na stavbe počítača v IAS (Institute for Advanced Study) v Princetone. Svoj návrh podrobne opísal začiatkom 1947 spolu s M. Goldstinom v správe *Planning and coding problems for an electronic computing instrument* [14]. Je historicky asi najvýznamnejším článkom o konštrukcii počítačov. Obsahuje opis "von Neumannovského počítača", analýzu problémov spojených s programovaním, základy programovania použitím blokových schém atď. (Von Neumann počítal s tým, že pri budovaní veľkých súborov sa budú používať magnetické pásky, a že funkcie sa budú zobrazovať graficky na obrazovke). "IAS machine" bol dokončený v roku 1952.

Mal 2000 elektrónok, pamäť 1024 40-bitových slov, násobenie zvládol za 600 μs a výber z pamäte za 25 μs. Na základe podrobnej a presnej von Neumannovej správy vzniklo mnoho ďalších počítačov v rôznych výskumných laboratóriách a univerzitách. Často ich označujeme termínom počítače von Neumannovho typu.

Konštruktéri ENIACu Eckert a Mauchly v spoločnosti Remington Rand realizovali v roku 1951 objednávku úradu pre sčítanie ľudu na dodávku počítača UNIVAC I (UNIVersal Automatic Computer). Obsahoval 5200 elektrónok, vážil 1,3 tony, spotreboval 125 kW elektrickej energie a vykonával 1905 operácií za sekundu s pamäťou veľkosti 1000 12-ciferných dekadických čísel. Pre vstup a výstup používal už aj magnetické pásky, elektrický písací stroj (Remington) a výstupný osciloskop. UNIVAC I stál od 15 do 50 tisíc dolárov a celkove bolo predaných 46 kusov. Stal sa tak prvým komerčne úspešným modelom počítača.

UNIVAC I (1951)

Myšlienka využitia magnetických pamätí bola zrealizovaná v roku 1950 v projekte Whirlwind (MIT). Bol pôvodne určený na leteckú simuláciu, neskôr sa stal základom systému SAGE (Semi Automatic Ground Environment) protivzdušnej obrany USA a Kanady. Umožňoval výpočty v **reálnom čase** a kontrolu priebehu výpočtu na obrazovke. Spočiatku používané magnetické bubny boli neskôr nahradené magnetizovateľnými feritovými jadierkami.

Whirlwind (MIT 1951)

Na svoje elektronické diernoštítkové tabelátory, vyrábané koncom 40-tych rokov nadviazala firma IBM univerzálnym počítačom IBM 701 v roku 1953 s pamäťou 2048 36-bitových slov, neskôr komerčne úspešnejším modelom IBM 704 s 4096 slovami a výpočtami s pohyblivou desatinnou čiarkou.

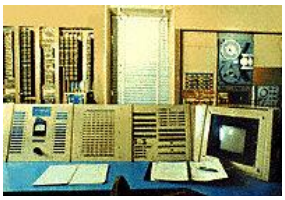
IBM 701 (1953)

Pre tieto typy bol navrhnutý aj prvý programovací jazyk s matematickým zápisom príkazov FORTRAN (FORmula TRANslation). John Backus pri jeho návrhu vychádzal z prvých prekladačov na úrovni asembleru (A0 od Grace Hooperovej z roku 1952). Vznikla tiež potreba vytvárania a uchovania dopredu pripravených segmentov kódu pre opakované použitie, ktoré sa stali zárodkom budúcich operačných systémov.

Snahy o vývoj prekladačov komentoval John von Neumann tým, že je to zbytočná práca, keďže stále bude dosť šikovných študentov, ktorí budú schopní programy prepísať priamo do strojového kódu.

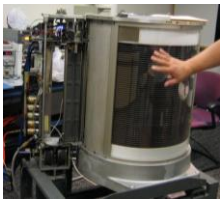
Tranzistorové počítače druhej generácie

Zásadný obrat v konštrukcii počítačov spôsobilo zavedenie **polovodičových** technológií. Stalo sa tak veľmi rýchlo po objave tranzistorového efektu (1947 Bardeen, Brattain, Shockley - Bell Telephone Laboratories). Prvýkrát na MIT v roku 1956 v počítači TX-0 Transistored eXperimental computer (prerobenom modeli Whirlwind). "Tixo" mal feritovú pamäť veľkosti 4096 18-bitových slov a bol schopný vykonať za sekundu viac ako 80 tisíc inštrukcií.



TX-0 (MIT 1956)

Koncom 50-tych rokov väčšina výrobcov prešla na tranzistorové verzie procesorov pre svoje počítače. Operačnú pamäť tvorila väčšinou mriežka z feritových jadierok. Ako externé pamäte boli používané magneticko-páskové jednotky, neskôr už aj magnetické diskové pamäte (vyvinuté v IBM už v roku 1956).



HDD IBM 305 RAMAC (1956)

Hlavne pre komerčné administratívne účely sa uplatnili počítače Univac, CDC 1604 a tiež IBM 7090 a 7094 (s feritovou pamäťou 32k 36-bitových slov, frekvenciou procesora 0,5 MHz). Na komunikáciu medzi riadiacou, výpočtovou jednotkou, pamäťou a periférnymi zariadeniami využívali kanálové prenosy, niektoré periférne zariadenia mali vlastné externé procesory. Počítače boli vybavené jednoduchými dávkovými operačnými systémami a kompilátormi programovacích jazykov Fortran a Cobol (Common Business Oriented Language).

V súvislosti s externým predspracovaním údajov sa používali už aj priame prístupy do pamäte (1958) a experimentálne aj prerušenia a systémy so zdieľaním času (1961). Rýchly a efektívny mechanizmus ošetrenia prerušení bol základným pilierom konštrukcie menších počítačov, zamýšľaných pre použitie na riadenie v reálnom čase. V tomto smere vykročila firma DEC (Digital Equipment Corporation) modelmi PDP, konštruovanými podľa vzoru počítačov TX-0 (využívali grafické monitory, terminálové konzoly a zbernicové prepojenie komponentov a radičov periférií).

DEC PDP-1

CDC 6600 (1964) prvý paralelný superpočítač z dielne Seymoura Craya

Vývoj integrovaných obvodov v 60-tych rokoch podporil aj program Apollo a najmä využitie v navádzacích systémoch medzikontinentálnych balistických rakiet.

IBM S/360 Model 50 (1964)

Stojan s diskovou magnetickou pamäťou

Pružné magnetické diskety (8", 5,25" a 3,5")

Prvý "stolný" minipočítač PDP-8 (1965)

Vývoj počítačov, určených pre náročné numerické výpočty (pre potreby vedy, armády, atómového priemyslu), nedával až taký dôraz na periférne zariadenia. Smeroval skôr k paralelizácii výpočtov a zvyšovaním ich presnosti používaním čísel s pohyblivou desatinnou čiarkou. Počítačové systémy IBM 7030 (1961), Ferranti Atlas (1962) a CDC 6600 (1964) s desiatimi výpočtovými jednotkami a výkonom 1 Mflops sa stali základom kategórie **superpočítačov**.

Integrované obvody, počítače tretej generácie

V polovici 60-tych rokov došlo k širšiemu využívaniu **plošných integrovaných obvodov**. Technológia vytvárania tranzistorov na povrchu kremíkovej podložky (čipu) umožnila miniaturizáciu predtým zložitých a nákladných počítačových komponentov. Hustota integrácie od počiatku stále rastie (podľa predpovede Gordona Moora z roku 1964 sa až doteraz každé dva roky zdvojnásobuje). Už v prvých rokoch umožnila realizáciu základných súčastí procesora (registre, počítadlá, dekódery, multiplexory) a rýchlej operačnej pamäte, ktorá začiatkom 70-tych rokov nahradila feritové jadrá.

Charakteristickým znakom počítačov tretej generácie bola snaha o štandardizáciu rozhraní a vznik sérií navzájom kompatibilných počítačov. Uplatňuje sa interaktívny prístup k výpočtu, využívanie prerušení, zdieľanie času procesora i ďalších spoločných zdrojov, terminálový prístup, paralelizácia výpočtu. V architektúre sa presadzuje mikroprogramovanie, virtualizácia pamäte, zreťazenie spracovania inštrukcií. Vznikajú viacpoužívateľské viacúlohové operačné systémy so zdieľaním času, využívajú sa procedurálne jazyky vyšších úrovní, systémy správy databáz.

V roku 1964 začala IBM svoju sériu System/360 - kompatibilných modelov počítačov rôznej výkonnosti na báze integrovaných obvodov. Počiatočné modely procesorov tejto série pracovali na frekvenciách 1-4 MHz so šestnástimi 32-bitovými registrami a 24-bitovou adresáciou operačnej pamäte (od začiatku osadzovanej 512 kB feritovou RAM, neskôr aj integrovanou pamäťou do kapacity 4 MB). System/360 zaviedol a presadil členenie pamäte (z dovtedajšieho šesťbitového) na 8-bitové bloky – **bajty** (byte). Pre urýchlenie komunikácie procesora s pamäťou bola v roku 1971 použitá prvýkrát aj rýchla vyrovnávacia pamäť (*cache*).

Pevné magnetické diskové pamäte vystriedali stojany s výmennými magnetickými diskami najskôr s kapacitou 2 MB, neskôr až 200 MB. Namiesto krabíc so štítkami sa začiatkom 70-tych rokov začali používať pružné (*floppy*) magnetické diskety, určené pôvodne na prenos mikrokódu procesora a štart systému. Prvé diskety s priemerom 8" a kapacitou 256 kB vystriedali menšie s priemerom 5,25" a 3,5" (1,44 MB).

Štandardizácia rozhraní, škálovateľnosť, kompatibilita hardvéru aj softvéru zaistila pre sériu IBM S/360 dominantné postavenie na trhu počítačov pre administratívu, nazývaných tiež **strediskové** (*mainframe*) počítače. Boli používané na spracovanie údajov, ale aj na vedecko-technické výpočty. Ich súčasťou bol aj operačný systém OS/360 (s orientáciou na dávkové spracovanie) a prekladač vlastného jazyka PL/1. Ostatné počítačové firmy (Hitachi, Fujitsu, NEC, Amdahl, Siemens, ICL) sa preorientovali na výrobu kompatibilných periférnych zariadení, prípadne klonov.

Na nekvalitu a množstvo problémov pri vývoji softvéru reagovala akademická komunita návrhom nových programovacích jazykov, podporujúcich metódy štruktúrovaného (Pascal, Ada), logického (Prolog) a objektového (Simula, Smalltalk) programovania. Urobili sa prvé kroky v oblasti formálnych špecifikácií a dokazovania správnosti programov. Položili sa základy **softvérového inžinierstva** ako súboru metód a postupov na efektívnu a kvalitnú realizáciu rozsiahlych softvérových projektov. Vývoj prenositeľného softvéru sa definitívne oddelil od vývoja hardvéru a vznikol samostatný softvérový trh.

Od stredného prúdu drahých strediskových počítačov (okolo milióna dolárov) s nutnosťou inštalácie v klimatizovaných miestnostiach s viacčlennou obsluhou sa koncom 60-tych rokoch výraznejšie odčlenili jednoduchšie stojanové zariadenia, ktoré vynikali spoľahlivosťou a flexibilitou, možnosťou experimentovania s rôznymi periférnymi zariadeniami. Dostali označenie **minipočítače** (*minicomputers*).

Na trhu minipočítačov sa presadila hlavne firma DEC pokračovaním svojej série PDP (Programmed Data Procesor). 12-bitový model PDP-8 (1965) a 16-bitový PDP-11 (1968) s adresovateľným priestorom 24 kB RAM (neskôr s virtualizovaným prístupom až 4 MB) opustil kanálové prenosy a používal univerzálnu rýchlu zbernice (Unibus). Mal rozvinutý hierarchický mechanizmus prerušení a výnimiek (*exception*), čo uľahčovalo interaktívne ovládanie a nasadenie v riadení výrobných procesov. Minipočítače (vďaka ich prijateľnej cene) kúpila aj väčšina vysokých a mnoho stredných škôl. Študenti sa takto ľahko dostali k práci s novou technikou, čo neskôr viacerí využili pri vývoji mikroprocesorov a osobných počítačov.

V prostredí minipočítačov tiež vznikol operačný systém Unix (inšpirovaný projektom Multics – Multiplexed Information and Computing Services, financovanou agentúrou ARPA, GE a AT&T Bell Labs, zameraným na zefektívnenie využitia počítačov a zdieľanie času procesorov). Jeho úspornosť a efektivita sa prejavila už v základnej 24 kB pamäti. Aby bolo možné systém prenášať aj na iné počítače, prepísali ho autori (Ritchie a Thompson, 1973) z PDP asembleru do nového vlastného jazyka C. Firma AT&T v snahe podporiť ďalší vývoj dovolila rozširovať systém bezplatne, vrátane zdrojového kódu. Vzniklo tak viacero jeho klonov (komerčné HP-Unix, AIX, Ultrix, ale aj voľne šíriteľné FreeBSD, OpenBSD, Minix a z neho neskôr aj Linux).

V kategórii **superpočítačov** (orientovaných na extrémne výkony) prevládalo zameranie na vektorové procesory – vykonávajúce súčasne rovnaký program na viacerých vstupných údajoch. ILLIAC IV (1968) dosahoval na 64 procesoroch výkon až 200 Mflops. Cray-1 (1976) dosiahol 250 Mflops a Cray X-MP (1983) až 1 Gflops.

Tvorcovia OS Unix Dennis Ritchie a Ken Thompson pri PDP-11 (1970)

Študenti, ktorí chceli laboratórne počítače čo najlepšie poznať a efektívne využiť všetky ich možnosti, pôvodne na MIT, neskôr aj na iných školách, si začali hovoriť **hackeri**.

Začiatkom 70-tych rokov nastal výrazný rozvoj polovodičových pamätí - niekedy sa označuje aj ako 3,5-ta generácia. IBM v nej prešla na 32-bitový S/370 s výkonom 10 MIPS, DEC na 32-bitový VAX.

Mikroprocesory a počítače štvrtej generácie

Sen o výpočte priamo na stole používateľa sa naplnil integráciou celého procesora na jedom čipe. Od prvých **mikroprocesorov** Intel 4004 (1971) a 8-bitového Intel 8008 (1972), pôvodne vyvíjaného ako procesor pre inteligentný terminál, bol len krok ku komerčne prístupným univerzálnym počítačom, určeným k osobnému používaniu (bez špeciálnej obsluhy, či špeciálnych prídavných zariadení a za prijateľnú cenu).

Koncom 70-tych rokov sa predali desiatky miliónov 8-bitových domácich počítačov, určených predovšetkým pre hry a jednoduché výpočty. Práca s textom, spracovanie zvuku a obrazu, vytváranie prezentácií, komunikácia, spôsobili zásadnú zmenu v nazeraní na využiteľnosť počítačov. Rozvojom komunikačných sietí koncom 90-tych rokov sa **zabudované** (*embedded*) počítačové systémy stali neoddeliteľnou súčasťou každodennej činnosti ľudí (inteligentné mobilné telefóny, hracie konzoly, netbooky, tablety, čipové platobné a identifikačné karty, aktívne prvky sietí, set-top-boxy).

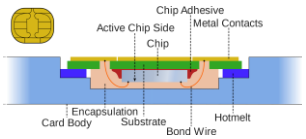
Profesionálne **osobné mikropočítače**, pre ich kompatibilitu s populárnou sériou IBM PC označované **PC** (Personal Computer), využívali mikroprocesory architektúry x86 (od 16-bitových Intel 8086 (1978) cez 32-bitové I80386 (1985) až k 64-bitovým Core2 (2006)). Kategóriu minipočítačov nahradila kategória **pracovných staníc** (*workstation*), stolných (*desktop*) a prenosných (*laptop*) počítačov. Škálovateľnosť mikroprocesorov umožnila nahradiť aj triedu strediskových počítačov mikropočítačovými viacprocesorovými servermi, spájanými podľa potreby do väčších celkov.

Vysoká integrácia (2 miliardy prvkov/čip) umožňuje dnes konštrukciu viacjadrových procesorov s viacúrovňovými vyrovnávacími pamäťami na čipe. Paralelizácia výpočtu dovoľuje virtualizovať systém už na ISA úrovni. Nutnosť efektívnej spolupráce počítačov viedla k vzniku rýchlych lokálnych počítačových sietí a veľkého globálneho sieťového systému Internet. Okrem paralelizácie je tak umožnená aj distribúcia výpočtov, vznikajú virtuálne sw platformy (JVM). V technológiách sa presadzujú bezdrôtové a optické komunikačné kanály, experimentálne aj optické logické prvky.

V dnešných superpočítačoch nájdeme tie isté mikroprocesory, ako v pracovných staniciach. Ich výkonnosť zabezpečujú rýchle zbernice a efektívna distribúcia zaťaženia. Hranicu výkonu 1 Tflops prekročil už v roku 1997 systém Intel ASCI Red a v roku 2008 IBM Roadrunner aj výkon 1 Pflops. Pre rok 2011 plánuje IBM oživiť systém Sequoia s výkonom 20 Pflops (1,6 milióna jadier, 1,6 PB RAM, príkon 6 MW).

Altair 8800 (1975)

O počítačoch 5. generácie sa hovorilo v 80-tych rokoch v súvislosti s japonským projektom umelej inteligencie, realizovanej logickým programom na paralelných inferenčných strojoch.



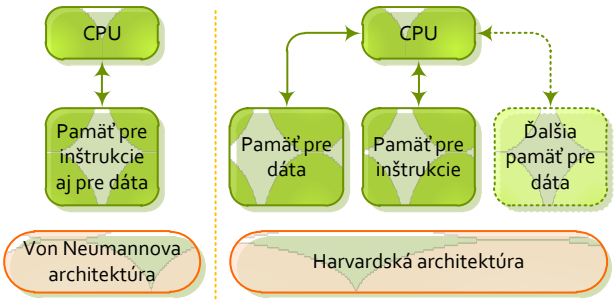
Zabudovaný procesor v čipovej karte

Pripravovaný superpočítač IBM Sequoia

5 Architektúry počítačových systémov

Klasické koncepcie

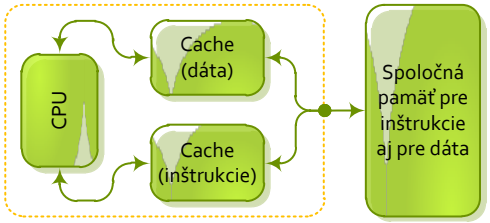
Drvivá väčšina dnešných počítačov sú počítače riadené vloženým programom (*stored-program concept*). To znamená, že ich činnosť je riadená inštrukciami vloženými v pamäti, namiesto nastavovania prepínačov a prepojovania káblov medzi ich funkčnými jednotkami. Umiestnenie programu je možné dvojakým spôsobom.



Označenie harvardská súvisí s jej používaním v prvých modeloch počítačov postavených na Harvarde (Aikenov Mark I).

Von Neumannova architektúra používa spoločnú operačnú pamäť pre údaje aj inštrukcie. Pamäť je pripojená k CPU jedinou zbernicou, po ktorej sa prenášajú údaje aj inštrukcie. V **harvardskej architektúre** je pamäť pre inštrukcie oddelená od pamäte pre údaje. Každý typ pamäte je pripojený k CPU samostatnou zbernicou. Zbernice môžu mať rôznu šírku, ktorú môžeme pre inštrukcie prispôsobiť ich veľkosti, navyše súčasne s načítaním inštrukcie môžeme prenášať údaje z údajovej pamäte (aj viacerých naraz, podľa potreby aj s rôznymi adresovými rozsahmi).

Striktne poňatá harvardská architektúra nevie do inštrukčnej pamäte zapisovať alebo z nej čítať (okrem aktuálneho kódu inštrukcie). Nie je preto možné jednoducho vytvoriť program, ktorý skompiluje, načíta a odštartuje iný program. Nie je možné tiež vytvárať samomodifikujúce sa programy.



Výhody obidvoch využíva v súčasnosti tzv. modifikovaná harvardská architektúra. Navonok sa javí ako von Neumannova, t.j. CPU je prepojené jedinou zbernicou s jedinou operačnou pamäťou, ktorá je spoločná pre údaje aj pre inštrukcie. Interne však CPU používa harvardskú architektúru s oddelenou internou vyrovnávacou pamäťou pre údaje a pre inštrukcie.

Paralelné architektúry, zdieľanie pamäte

Efektivitu paralelného výpočtu ovplyvňuje ďalej čas na synchronizáciu, stupeň granularity algoritmu, vybavenie globálnych systémových požiadaviek.

Limity efektivity paralelného spracovania vyjadruje aj Amdahlov zákon.

Pri zachovaní taktovacej frekvencie procesora (ktorá je pre súčasné sekvenčné spracovanie na technologickom maxime) je možné urýchliť spracovanie inštrukcií tým, že sa viaceré z nich budú vykonávať súčasne. Ak napríklad program s tisícom inštrukcií je možné rozdeliť na dve samostatné polovice a vykonanie každej inštrukcie trvá sekundu, bude jeho vykonanie trvať na paralelnom počítači s dvoma CPU len 500 sekúnd – teda dvakrát rýchlejšie ako na sekvenčnom.

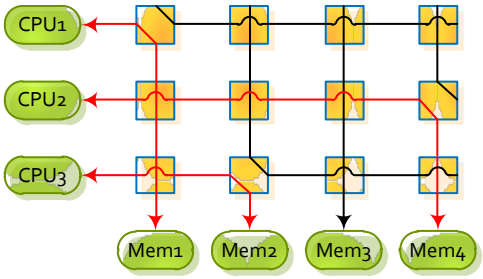
Väčšinou však nie je možné klasický sekvenčný program rozdeliť na potrebný počet rovnako veľkých nezávisle vykonateľných častí. Ak by v predošlom príklade boli dve nezávislé časti s 200 a 800 inštrukciami, výsledné urýchlenie nebude dvojnásobné, ale iba 1,25 násobné. Tiež sa môže stať, že pôvodný program nemožno rozdeliť na požadovaný počet častí, prípadne niektoré časti je nutné spracovať sekvenčne.

Paralelný výpočtový systém je charakteristický prístupom do spoločnej zdieľanej pamäte – hovoríme tiež o **silne spriahnutých** (*tightly coupled*) systémoch. Paralelizmus môžeme realizovať už na úrovni inštrukcií (VLIW architektúra), vláken (*multithreading*), procesorových jadier (*multicore*), procesorov (*multiprocessors*).

Jednotný prístup do pamäte

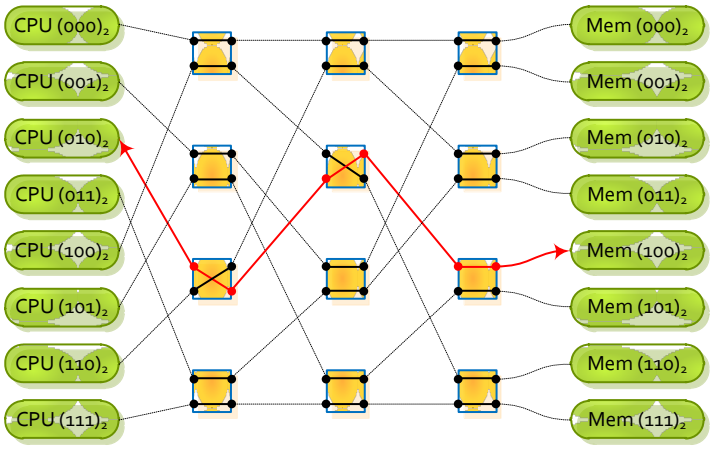
V paralelných systémoch s jednotným prístupom do pamäte UMA (Uniform Memory Access) pristupujú procesory k ľubovoľnému pamäťovému modulu v rovnakom čase.

V prípade jednozbernicovej organizácie komunikujú procesory s pamäťou pomocou jednej zdieľanej systémovej zbernice. Táto sa však môže ľahko stať úzkym hrdlom riešenia, pokiaľ potrebujú viaceré procesory pracovať s pamäťou súčasne.



Možným riešením tohto problému je rozdelenie pamäte a prepojenie s procesormi pomocou prepínacej matice (*m×n switching crossbar matrix interconnection*). Jeho výhodou je, že pokiaľ potrebujú procesory pristupovať do častí pamäte, ktoré sa nachádzajú v rôznych moduloch, je možné vykonať všetky tieto prístupy súčasne.

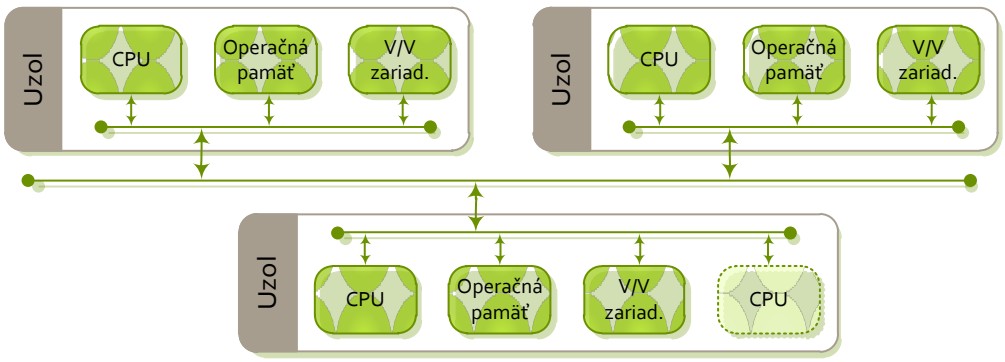
Veľký počet prepínačov je možné znížiť použitím viacstupňových prepojovacích sietí (*multistage interconnection networks*). Sú charakteristické tým, že medzi procesory a pamäťové moduly je umiestnených niekoľko stupňov prepínačov. Procesory sú prepojené s prepínačmi na prvom stupni, tie s prepínačmi na druhom stupni. Prepínače na poslednom stupni sú prepojené s pamäťovými modulmi.



Na obrázku je tzv. **omega sieť** pre osem procesorov a osem pamäťových modulov, v ktorej možno realizovať prepojenie každého procesora s ľubovoľnou pamäťou. Dokonca je možné prepojiť všetky procesory naraz s rôznymi pamäťovými modulmi, no nie všetky permutácie sú realizovateľné (pre sieť 8x8 je realizovateľných približne 10%). Výhodou však je, že táto sieť si vystačí iba s n/2 × log n prepínačmi (namiesto n^2). Pre n = 8 to znamená úsporu približne 81% plochy.

Nejednotný prístup do pamäte

Vo viacprocesorových systémoch môže byť operačná pamäť „decentralizovaná“ a rozdelená medzi jednotlivé procesory. V takomto systéme je prístup k pamäti nejednotný – NUMA (Non-Uniform Memory Access), lebo k svojej pamäti vie procesor zvyčajne pristupovať efektívnejšie ako k pamäti pridelenej iným procesorom.

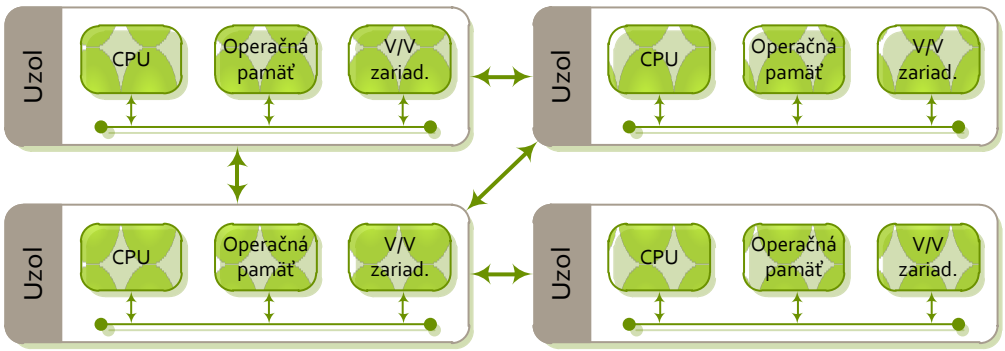


Jednotný a nejednotný prístup do pamäte je možné kombinovať. Ak do jedného uzla pridáme ďalší procesor (na obrázku je označený čiarkovane), v uzle môže zdieľať pamäť s pôvodným procesorom jednou zbernicou. Prístup procesorov k pamäti mimo ich uzla je pre ne stále menej efektívny (celkove je to NUMA architektúra).

Distribuované architektúry, komunikácia pomocou správ

Programovanie komunikácie pomocou správ uľahčuje štandardizovaný protokol MPI (Message Passing Interface).

Distribuovaný systém pozostáva z viacerých autonómnych počítačov (uzlov) s vlastnou lokálnou pamäťou, ktoré spolu komunikujú **posielaním správ** (*message passing*) prostredníctvom počítačovej siete a navzájom spolupracujú na dosiahnutí spoločného cieľa. Je charakterizovaný tiež ako **slabo spriahnutý** (*loosely coupled*) systém. Podporuje pridávanie nových a uberanie existujúcich uzlov počas výpočtu a je odolný voči výpadkom individuálnych uzlov, či spojení medzi nimi.



Distribuovaný systém, v ktorom sú jednotlivé uzly prepojené výkonnou lokálnou počítačovou sieťou zvykneme nazývať **klaster** (cluster). Špeciálny softvér **middleware** dovoľuje všetkým uzlom využívať všetky zdroje ostatných uzlov klastra. Umožňuje tiež ich správu a ochranu (používateľ sa prihlasuje do klastra ako do jedného veľkého výpočtového systému). Pokiaľ je prepojenie dostatočne rýchle, môže middleware vytvárať aj tzv. distribuovanú zdieľanú pamäť, ktorá dovoľuje efektívnu realizáciu paralelných algoritmov.

Grid je distribuovaný systém, v ktorom sú uzly prepojené menej výkonnou, no rozľahlejšou počítačovou sieťou (obyčajne Internet). Na rozdiel od stovák uzlov v klastri môže grid obsahovať až státisice uzlov. Vďaka geografickej rozľahlosti je možné efektívne využívať zdroje, ktoré by boli v nočnom čase na opačnej stane zemegule nevyťažené. Gridové siete majú vyššiu latenciu (oneskorenie), preto ich middleware je orientovaný viac na distribúciu celých úloh a väčších balíkov údajov.

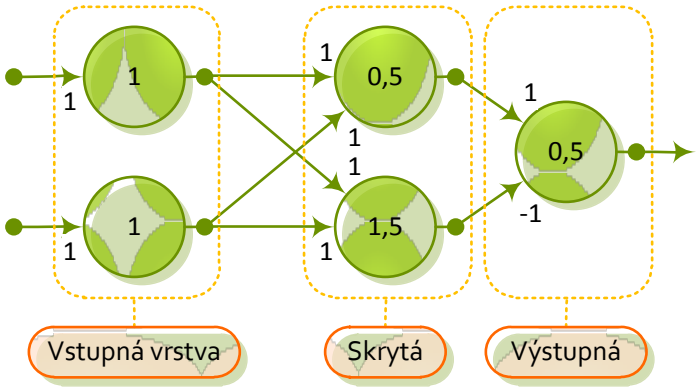
Existujú aj programy, ktoré po nainštalovaní na klasický operačný systém sprístupnia zdroje tohto počítača distribuovanému systému. Takýmto spôsobom môže aj technicky neskúsený majiteľ počítača pomôcť vybudovať veľký distribuovaný systém. Ako príklad možno uviesť projekt SETI@home, ktorý je postavený na myšlienke využitia voľného výkonu osobných počítačov. Jeho cieľom je prehľadávanie signálov z vesmíru, zaznamenaných v observatóriu v Arecibe, na prítomnosť rádiového vysielania pochádzajúceho od mimozemskej inteligencie.

Počítačové systémy riadené tokom údajov

Problémom pomalého a konfliktného prístupu k pamäti v klasických počítačových systémoch riadených programom sa snažia vyhnúť systémy, riadené tokom údajov.

Umelé neurónové siete

Umelá neurónová sieť je výpočtový model, ktorý sa snaží simulovať štruktúru a funkciu biologických neurónových sietí. Pozostáva z umelých neurónov, ktoré sú navzájom poprepájané a organizované do vrstiev.



Perceptrónová sieť na obrázku realizuje booleovskú funkciu XOR. Jej základným stavebným prvkom je perceptrón (znázornený je krúžkom). Každý perceptrón má prah excitácie (číslo vnútri krúžku), niekoľko vstupov a práve jeden výstup. Každý vstup má svoju váhu (číslo pri vstupe). Perceptrón sčíta hodnoty všetkých vstupov prenásobené ich váhou. Ak je súčet menší ako prah excitácie, je výstupom 0, inak 1 (perceptrón sa tzv. aktivoval).

Sieť perceptrónov je rozdelená do troch druhov vrstiev. Prvá je vstupná, posledná je výstupná. Ostatné sú tzv. skryté vrstvy (v tomto príklade je len jedna). Po privedení vstupných hodnôt na vstupné perceptróny sa začne šíriť výpočet po jednotlivých vrstvách, až nakoniec na výstupe perceptrónov v poslednej vrstve sa objaví výsledok.

Simulujme výpočet na vstupe 1, 1. Oba vstupné perceptróny budú aktivované. V ďalšom kroku budú aktivované oba perceptróny skrytej vrstvy (pretože vážený súčet vstupov oboch perceptrónov je 2, čo prekračuje ich prah excitácie). V poslednom kroku jediný výstupný perceptrón má vážený súčet vstupov 0, čo nestačí na jeho aktiváciu a výstupom je 0.

Voľba správnej topológie siete a zistenie správnych váh prebieha obyčajne procesom učenia. Počas neho sa upravujú jednotlivé váhy na základe trénovacej množiny (obsahujúcej rôzne vstupy a k nim zodpovedajúce výstupy). Umelé neurónové siete sa takto dokážu naučiť modelovať aj zložité vzťahy medzi vstupom a výstupom, ktoré potom dokážu aplikovať aj na vstupy, na ktoré neboli natrénované.

Existuje veľa rôznych architektúr umelých neurónových sietí. Používajú sa napríklad pri aproximácii funkcií, predpovedaní pokračovania časových radov, rozpoznávaní vzorov, rozdeľovaní do skupín a podobne.

Kvantové počítače

Kvantový počítač je zariadenie, ktorého výpočet je priamo postavený a ovplyvňovaný zákonmi kvantovej mechaniky, ako sú superpozícia, previazanosť či princíp neurčitosti. Stav výpočtu je uložený v kvantovom stave počítača. Namiesto bitov sa používajú kvantové bity (qubit). Klasický bit môže nadobúdať hodnotu 0 alebo 1. Kvantový bit môže byť v ktorejkoľvek kvantovej superpozícii týchto stavov. Akoby bol súčasne s pravdepodobnosťou x v stave 0 a s pravdepodobnosťou y v stave 1. Meraním však nevieme tieto pravdepodobnosti určiť. Nameriame len hodnotu 0 alebo 1 (s pravdepodobnosťou x, resp. y). Meraním však zničíme kvantový stav qubitu a preto opakované meranie na určenie pravdepodobnosti x a y nemá zmysel.

To, že sa jeden kvantový bit môže súčasne nachádzať akoby vo viacerých stavoch, umožňuje kvantovým počítačom vykonať jednu operáciu s veľkým počtom hodnôt v jednom okamihu. Preto sa očakáva, že kvantové počítače v sebe ukrývajú potenciál pre zrýchlenie niektorých algoritmov, ktoré na klasických počítačoch už nie je možné zrýchliť.

Ako príklad možno uviesť Groverov algoritmus, ktorý dokáže určiť pozíciu hľadaného prvku v neutriedenom poli v čase $O(n^{1/2})$. Pritom na klasických počítačoch to nie je možné spraviť rýchlejšie ako v lineárnom čase. Druhým veľmi zaujímavým problémom je faktorizácia čísla (pre dané celé číslo nájsť jeho rozklad na prvočinitele), ktorý je možné kvantovým počítačom riešiť v polynomiálnom čase (Shorov algoritmus). Jeho realizácia by umožnila napadnúť bezpečnosť niektorých šifrovacích algoritmov (RSA). Ďalšie informácie o kvantových počítačoch je možné nájsť napríklad v [17].

Biomolekulárne počítače

Biomolekulárne počítače využívajú DNA, proteíny, biochémiu a molekulárnu biológiu namiesto tradičných kremíkových technológií. Napríklad DNA počítače spoliehajú na schopnosť nukleotidov vytvárať páry s kompatibilným nukleotidom. Zakódovaním vstupu do vhodných podreťazcov nukleotidov je možné hľadať riešenie daného problému pomocou detekcie vzniku reťazca DNA s určitými vlastnosťami.

Takýmto spôsobom Leonard Adleman [18] v roku 1994 vyriešil na prvom DNA počítači, ktorý sám zostrojil, problém existencie hamiltonovskej cesty medzi dvoma vrcholmi v orientovanom grafe (prechádzajúcej práve raz všetkými vrcholmi grafu). Jeho postup bol takýto:

> Pre každý vrchol grafu náhodne vygenerujeme podreťazec DNA dĺžky 20, ktorý bude vrchol reprezentovať.
>
> Pre každý vrchol grafu vytvoríme komplementárny podreťazec (dĺžky 20).
>
> Pre každú hranu v grafe vytvoríme podreťazec dĺžky 20, ktorý vznikne spojením pravej polovice DNA podreťazca reprezentujúceho počiatočný vrchol s ľavou polovicou DNA podreťazca koncového vrcholu.
>
> Zmiešaním dostatočného počtu kópií takto vytvorených podreťazcov samovoľne vzniknú rôzne reťazce DNA, ktoré reprezentujú všetky možné cesty v orientovanom grafe.
>
> Odfiltrujeme reťazce DNA, ktoré nezačínali alebo nekončili vo zvolenom vrchole, prípadne nemali dĺžku, rovnajúcu sa počtu vrcholov v orientovanom grafe.
>
> Odstránime všetky DNA reťazce, v ktorých sa niektoré vrcholy opakovali viackrát.
>
> Ak zostal nejaký reťazec DNA, tak hamiltonovská cesta existuje, inak nie.

Výhodou DNA počítačov je, že môžu vykonávať preskúmanie veľkého priestoru možností časovo, priestorovo a energeticky efektívne. Ich nevýhodou je, že pre každý problém treba individuálne navrhnúť jeho zakódovanie do DNA podreťazcov, následne filtrovať a interpretovať vzniknuté DNA reťazce. DNA počítač vlastne len zabezpečí rýchle vytvorenie všetkých prípustných DNA reťazcov z dostupných DNA podreťazcov. Ich počet môže predstavovať riešenie problému.

Čo sme sa naučili v tomto module

Zhrnutie

Na jednoduchom komunikačnom modeli sme si ozrejmili základné komunikačné mechanizmy, používané v počítači. Poznáme štruktúru a funkcie pamäťovej hierarchie počítača, štruktúru zberníc a spôsoby pripojenia periférnych zariadení k procesoru.

Získali sme prehľad o historických medzníkoch vývoja počítačových systémov a máme predstavu o možných alternatívnych architektúrach.

Preverenie výstupných vedomostí

Zistenie hardvérovej konfigurácie konkrétneho počítača, popis inštalovaných súčastí a možností rozšírenia.

Príprava jednoduchej prezentácie konkrétneho historického modelu počítača.

Literatúra a použité zdroje

[1] Patterson, D.A., Hennesy J.L. (2009) *Computer Organization and Design*, 4. ed., Morgan Kaufmann, 2009, ISBN 978-0-12-374493-7.

[2] Stallings, W. (2005) *Computer Organization and Architecture: Designing for Performance*, 7. ed., Prentice Hall, 2005, ISBN 978-0-13-185644-8.

[3] Tanenbaum, A.S. (2005) *Structured Computer Organization*, 5. ed., Prentice Hall, 2005, ISBN 978-0-13-148521-1.

[4] Dandamudi, P.S. (2003) Fundamentals of Computer Organization and Design, Springer, 2003, ISBN 978-0387952116.

[5] Gook, M. (2004) *Hardwarová rozhraní : Průvodce programátora* (český preklad), Computer Press, 2006, ISBN 80-251-1019-2.

[6] Huck, T. (1983) *Comparative Analysis of Computer Architectures*, Stanford University Technical Report Number 83-243, 1983.

[7] Patterson, D., Sequin, C. (1982) *A VLSI RISC*, *Computer*, vol.15 (9), 1982.

[8] Stallings, W. (1998) *Operating Systems: Internals and Design Principles*, 3. ed., Pretice-Hall, 1998, ISBN 0-13-887407-7.

[9] Keeth, B., Baker, R. J., Johnson, B., Lin, F. (2007) *DRAM Circuit Design: Fundamental and High-Speed Topics, 2nd edition,* Wiley-IEEE Press, 2007, ISBN 978-0-470-18475-2.

[10] Jacob, B., Ng, S., Wang, D. (2007) *Memory Systems: Cache, DRAM, Disk,* Morgan Kaufmann, 2007, ISBN 978-0123797513.

[11] Al-Chvárizmí, *Aritmetický a algebraický traktát (s komentářem Petra Vopěnky)*, OPS - Prameny evropské vzdělanosti, Nymburk, 2009, ISBN 978-80-87269-01-5.

[12] von Neumann, J. (1945) *First draft of a report on the EDVAC*, http://www.virtualtravelog.net/entries/2003-08-TheFirstDraft.pdf.

[13] Turing, A. (1946) *Proposed Electronic Calculator*, The Turing Archive, http://www.alanturing.net/turing_archive/archive/p/p01/P01-001.html.

[14] Goldstine, H., von Neumann, J. (1947) *Planning and coding of problems for an electronic computing instrument*, IAS Princeton, New Jersey, 1947, http://library.ias.edu/ecp/planningcodingof0103inst.pdf.

[15] Gruska, J., Havel, I.M., Wiedermann, J., Zelený, J. (1982) *Počítačová revolúcia*. In: SOFSEM'82, 1982, pp. 7-64.

[16] Zelený, J., Mannová, B. (2006) *Historie výpočetní techniky*, Scientia, 2006, ISBN 80-86960-04-8.

[17] Gruska, J. (1999) *Quantum Computing,* McGraw-Hill Publishing Company, 1999, ISBN 007-709503-0.

[18] Adleman, L. (1994) *Molecular computation of solutions to combinatorial problems,* Science, 266:5187, pp. 1021-1024, 1994. Dostupné na http://www.usc.edu/dept/molecular-science/papers/adleman-science.pdf.

Tento študijný materiál vznikol ako súčasť národného projektu Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika v rámci Aktivity „Vzdelávanie nekvalifikovaných učiteľov informatiky na 2. stupni ZŠ a na SŠ“.

Autori © RNDr. Jozef Jirásek, PhD.
RNDr. Richard Ostertág, PhD.

Názov Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Podnázov Počítačové systémy 3

Študijný materiál prešiel recenzným pokračovaním.

Recenzenti doc. Ing. Ľudovít Trajteľ, PhD.
doc. Ing. Peter Fabián, PhD.

Počet strán 40

Náklad 300 ks

Prvé vydanie, Bratislava 2010

Vydal Štátny pedagogický ústav, Pluhová 8, 830 00 Bratislava, v súčinnosti s Univerzitou Pavla Jozefa Šafárika v Košiciach, Univerzitou Komenského v Bratislave, Univerzitou Konštantína Filozofa v Nitre, Univerzitou Mateja Bela v Banskej Bystrici a Žilinskou univerzitou v Žiline

Vytlačil BRATIA SABOVCI, s r.o., Zvolen

ISBN 978-80-8118-062-0