Teoretické otázky z testů 2017 – 2019

IEEE754

- Výsledkem operací, které přetečou z rozsahu čísel ukládaných dle IEEE754, například podíl 1/0, je hodnota NaN (Not-a-Number).
  NE
  **(1/0 = INF)*

- Čísla v plovoucí řádové čárce dle IEEE754 musí být vždy uložena v normalizovaném tvaru, tzn. před binární tečkou je vždy jednička. *(test 2017 – 6.6 – 16)*
  NE

Architektury a jejich vlastnosti

- V případě architektury MIPS je možné vykonávat jednoduché aritmetické operace nad operandy uloženými v paměti bez nutnosti nahrát je do registrů procesoru.*(test 2017 – 6.6 – 16)*
  NE

- Architektura AMD64 používá 64-bitovy formát virtuální adresy, ale pro překlad využívá pouze 48 bitů *(test 2017 – 6.6 – 17)*
  ANO

- Dle konceptu voň Neumanna se v počítači programy a výsledky (data) se ukládají do téže paměti - což je významným rozdílem v porovnání s tzv. Harvardskou architekturou *(test 2017 – 6.6 – 19)*
  ANO

- Obvyklou vlastností architektur RISC je, že obsahují instrukce pro snadnou práci s obsahem paměti jako například instrukce, která přímo modifikuje obsah paměti o libovolnou konstantu (omezení rozsahem datového typu) na zadané adrese *(test 2017 – 6.6 – 19)*
  NE

- Obvyklou vlastností architektur CISC je, že obsahují instrukce pro snadnou práci s obsahem paměti jako například instrukce, která přímo modifikuje obsah paměti o libovolnou konstantu (omezení rozsahem datového typu) na zadané adrese *(test 2019)*
  ANO

- RISC procesory pouzivaji tzv Load/Store model, v nemz nelze v jedne instrukci kombinovat pametove a aritmeticke (ALU) operace – coz neplati pro CISC
  ANO

Virtualni/ fizicke adresy

- V případě 64-bitove virtuální adresy je obvyklé používat méně bitů pro fyzickou adresu - například 48, nebo 40 (*test 2017 – 6.6 – 16)*
  ANO

- V případě 64-bitove virtuální adresy je nezbytné použít stejně dlouhou fyzickou adresu, aby nedocházelo ke konfliktům adres při překladu
  NE

- 64-bitove procesory pouzivaji 64-bitovy format virtualni adresy, ale jejich virtualni strankovani vyuziva mensi pocet bitu (zpravidla 48). Duvodem je zmenseni poctu urovni strankovacich tabulek *(test 2019)*
  ANO

PCI

- PCIe již dnes vytlačilo PCI sběrnice, se kterými není kompatibilní jak po hardwarové stránce tak z pohledu software (*test 2017 – 6.6 – 16)*
  NE

- V BAR registru č. 0 je uložena fyzická adresa PCI karty a je určena dle číslování (adresace) karet na sběrnici. Tato adresa slouží k přístupu ke kartě (*test 2017 – 6.6 – 16)*
  NE

- Konfigurační BAR registry PCI karty mají svůj obsah jednoznačně určen výrobcem (Vendor ID) a nastaven v závislosti na identifikačním číslu karty (Device ID). (*test 2017 – 6.6 – 18)*
  NE

- Adresový dekodér připojený na PCI sběrnici pomáhá dekódovat (převádí) virtuální adresu vyskytující se na této PCI sběrnici na adresu fyzickou, se kterou dané zařízení již dále pracuje (*test 2017 – 6.6 – 27)*
  NE

- PCI-e sbernice dovoluje full-duplex prenos dat *(test 2019)*
  ANO

- Klasicka paralelni PCI sbernice dovoluje full-duplex prenos dat *(test 2019)*
  NE

- Klasicka PCI sbernice je seriova *(test 2019)*
  NE

RAID

1. RAID úrovně 1 nepracuje s redundancí (*test 2017 – 6.6 – 17)*
   NE

2. RAID 6 rozkládá paritní informace napříč jednotlivými disky (*test 2017 – 6.6 – 17)*
   ANO

3. RAID 1 slouží k zvýšení spolehlivosti systému pevných disků, RAID 0 k zvýšení výkonu
   NE

Assembler

- Při psaní programů v assembleru je běžnou konvencí, že volaný (=podprogram/funkce/rutina) musí zálohovat všechny registry procesoru, které používá. To znamená, že volající se může spolehnout na to, že všechny své mezivýsledky uložený v registrech nebudou voláním podprogramu dotčeny. (Toto se provádí ukládáním jejich hodnot na stack a následným obnovením původních hodnot před návratem z podprogramu.) (*test 2017 – 6.6 – 17)*
  NE

- Je možné instrukcí aritmetického posunu vpravo nahradit instrukcí pro logický posun (uvažujte doplňkovou aritmetiku)? (*test 2017 – 6.6 – 18)*
  NE

- V odpřednášené verzi jedno-cykloveho procesoru je ADDI instrukce nejdelší instrukcí procesoru a od ní odvíjí délka periody hodinového signálu
  NE

Typ otazek na typicke vlastnosti pro jednotlive generace a typy architektur CPU
(test 2019 + 2017)

- Instrukce volani podprogramu uklada navratovou adresu do pameti
  Historicka 8-bit CPU **16-bit CISC** ~~MIPS~~ ~~SPARC~~

- ALop je mozne provozovat pouze mezi operandy v registrech
  ~~Historicka 8-bit CPU~~ ~~16-bit CISC~~ **MIPS** **SPARC**

- ALop jsou typycke 3operandove
  ~~Historicka 8-bit CPU~~ ~~16-bit CISC~~ **MIPS** **SPARC**

- Prenosy dat mezi pameti a registry obstarava omezena skupina instrukci (load/store)
  ~~Historicka 8-bit CPU~~ ~~16-bit CISC~~ **MIPS** **SPARC**

- Pro ulozeni navratove adresy se pouziva specializovany registr
  ~~Historicka 8-bit CPU~~ ~~16-bit CISC~~ **MIPS** **SPARC**

- Veskere instrukce v zakladnim rezimu maji jednu pevnou delku
  ~~Historicka 8-bit CPU~~ ~~16-bit CISC~~ **MIPS** **SPARC**

DMA

- Přenos DMA (Direct Memory Access) je méně výhodný pro velké objemy dat než použití mechanizmu přerušení v důsledku zátěže (overhead), která je potřeba pro inicializaci DMA
  NE

- Při použití DMA data neprocházejí skrze procesor. Program/OS naplánuje parametry přenosu, procesor nastaví adresy do DMA řadiče - ten po ukončení operace vyvolá přerušení
  ANO

- Program /OS se behem DMA cteni nebo zapisuvenuje jinym operacim. Pouze naplanuje parametry prenosu a nastavi adresy do DMA radice - ten po ukonceni operace vyvola preruseni
  ANO

- Při pouziti DMA periferie primo zapisuje/čte data z/do cache *(test 2019)*
  NE

I/O

- V případě použití programového kanálů s přerušením (interrupt-driven) procesor již pouze periodicky sleduje stavový bit/registr daného I/O zařízení (a v případě potřeby zareaguje - vyvolá přerušení), což je v protipříkladů s metodou tzv. "poiling-u", kdy procesor musí neustále ve smyčce sledovat dané zařízení a reagovat na jeho stav
  NE

- V případě paměťově mapovaných I/O jsou použity speciální instrukce (in, out) pro přístup k I/O registrům
  NE

- Každé I/O zařízení na PCI sběrnici obsahuje své vlastní speciální registry, ve kterých je uložena bázová adresa (adresy), na které má toto zařízení reagovat
  ANO

Řadiče

- Mikroprogramový řadič NENÍ vhodný pro řízení činnosti zřetězeného procesoru
  ANO

- Mikroprogramový řadič realizuje každou programátorovi viditelnou instrukci pomocí vlastního mikroprogramu uloženého v řídicí paměti řadiče
  ANO

Cache

- Data, která jsou uložena v L1 datové cache pamětí dnešních moderních procesorů, jsou vždy stejná jako data uložená na odpovídající adrese ve fyzické paměti.
  NE

- Nezávisle na organizaci cache, pokud nastane **valid page fault** (adresa je součástí virtuálního adresního prostoru procesoru), pak typicky dochází k načtení (z disku do hlavní paměti) jednoho: *(test 2019)*
  RAMCE

- Nezávisle na organizaci cache, pokud nastane **invalid page fault** (adresa není součástí virtuálního adresního prostoru procesoru), pak typicky dochází k načtení (z disku do hlavní paměti) jednoho: *(test 2019)*
  PROGRAM JE OBVYKLE UKONČEN

- Algoritmus **LRU** v pripade konfliktu platnych radek v cache nahrazuje *(test 2019)*
  NEJDELE NEPOUZITY RADEK

- Algoritmus **LFU** v pripade konfliktu platnych radek v cache nahrazuje *(test 2019)*
  NEJMENE POUZIVANY RADEK

RAM

- Paměťová buňka DRAM je rychlejší než SRAM
  NE
  **(DRAM ma vetsi kapacitu nez SRAM)*

- Paměťová buňka dinamicke pameti pouziva pro svou cinnost kondenzator
  ANO
  **(DRAM pouziva kondenzator, SRAM ne)*

Ostatni

- 32-bitove integer cisla se znamenkem ve dvojkovem doplnku ukladaji mensi pocet ciselnych hodnot nez cisla bez znamenka *(test 2019)*
  NE

- Dynamický prediktor skoků po inicializační fázi (počátečně učení se prediktoru) je vždy lepší než statický predictor
  NE

- Datovy typ **float**, ktery se pouziva jazyce C se v pameti obvykle uklada *(test 2019)*
  PODLE IEEE754

- Datovy typ **int**, ktery se pouziva jazyce C se v pameti obvykle uklada v *(test 2019)*
  DVOJKOVÉM DOPLŇKU

Segmenty adresniho prostoru

- Co se uklada do Text segmentu**? - Program**
- Co se uklada do Data segmentu**? - Inicializovane globalni nebo staticke promennne**
- Co se uklada do BSS segmentu**? - Neinicializovana data**
- malloc()? **- Heap**
- new? **- Heap**