**1 - úvod**

* Jeden **bajt** má **osm bitů.**
* **Nejmenší adresovatelná jednotka paměti je kapacita místa v paměti, které má vlastní adresu**.
* Nejmenší adresovatelná jednotka paměti typicky je **8 bitů**
* 1 KB je 1024 B
* 210 bajtů je1 KB
* 216 bajtů je 64 KB
* 220 bajtů je 1 MB
* 232 bajtů je 4 GB
* Adresový registr obsahuje 4 bity. Kolik je schopen namapovat (zaadresovat) adres? Od.: 16
* Paměť o maximální kapacitě 1 M adresovatelných míst musí mít adresovací sběrnici širokou právě? Od.: 20 bitu
* Paměť o maximální kapacitě 1 G adresovatelných míst musí mít adresovací sběrnici širokou právě? Od.: 30 bitu
* Jaká je správná posloupnost seřazená podle velikosti uchovávané informace od nejmenší po největší? Od.: bit, bajt, slovo
* Paměť RAM je určena ke **čtení i k zápisu**
* Doslovný překlad **zkratky RAM je Random Access Memory**
* Vestavěný program řídící činnost automatického jednoúčelového zařízení patří typicky do kategorie **firmware**.
* Jednotka informace 1 slovo (1 word) odpovídá (80b, 2 B, 32b, 64b …. všechny odpovědi mohou být Správně
* Jedno slovo obyčejně nemá 1 slabiku
* Kontrolní bit například na děrné pásce se nazývá **paritní bit**
* 24bitová adresová sběrnice dokáže adresovat paměťový prostor o kapacitě maximálně (adresovatelná jednotka je bajt): 16 MB
* Mezi různými typy pamětí nejmenší kapacitu má obvykle **registr**
* Mezi různými typy pamětí je z hlediska přístupu nejrychlejší pamětí registr
* **Paměť se sekvenčním přístupem** při přístupu k místu s adresou n projde nejdříve adresy 0-(n-1)
* Která charakteristika neplatí pro paměť typu registr? Od.: *energeticky nezávislá*
* Která charakteristika platí pro paměť typu registr? Od.: *paměť s přímým přístupem*
* **Architektura počítače "von Neumann"** obsahuje pravidlo: Počítač obsahuje operační paměť, ALJ, řadič a V/V zařízení.
* Architektura počítače "von Neumann" obsahuje pravidlo: Údaje a instrukce jsou vyjádřeny binárně.
* V architektuře "von Neumann" má dekódování instrukcí na starost: *řadič*
* Které tvrzení NEplatí pro von Neumannovu architekturu? *Program je uložen v paměti oddělené od paměti pro data.*
* Stavová hlášení jsou v architektuře "von Neumann" zasílána: *řadiči*
* Které tvrzení o koncepci Johna von Neumanna NEplatí? Po skončení jsou výsledky poslány přes řadič na výstupní zařízení. [platí: Mezi výsledky jsou ukládány do operační paměti., Data se umístí do operační paměti přes ALJ pomocí vstupního zařízení., Data se umístí do operační paměti přes ALJ pomocí vstupního zařízení., Jednotlivé kroky výpočtu provádí aritmeticko-logická jednotka.)
* Ve von Neumannově modelu NEtečou data z řadiče do ALJ
* Ve von Neumannově modelu tečou data z paměti do ALJ
* Ve von Neumannově modelu tečou data z ALJ do paměti, řadiče
* Mezi typickou činnost **řadiče** patří: *transformuje instrukce na posloupnost signálů ovládající připojené zařízení .*
* DMA je určeno především pro přenos dat z disku do operační paměti

**2 – číselné soustavy**

• V polyadické soustavě je *číslo součet mocnin základu vynásobených číslicemi*.

• Čísla lze snadno (každou k-tici číslic nižší soustavy nahradíme číslicí soustavy vyšší) převádět mezi soustavami o základu 8 a 2

• Číslo 21 v desítkové soustavě po převedení do soustavy dvojkové je 10101

• Pascalovský typ *INTEGER* je celé číslo, které se na počítačích PC zobrazuje v doplňkovém kódu.

• **Znaménkový bit** v celém čísle je zpravidla bit nejvyššího řádu.

• Znaménkový bit bývá zpravidla roven nule, pokud se zobrazuje číslo kladné

• Rozsah zobrazení celého čísla uloženého ve dvojkovém doplňkovém kódu na 8 (celkem) bitech je <-128;127>

• Největší zobrazitelné celé číslo ve dvojkovém doplňkovém kódu má tvar 011...11

• Při sčítání dvou čísel v inverzním kódu jako korekci výsledku použijeme: **kruhový přenos**

• **Přeplnění (přetečení)** je stav, ve kterém výsledek spadá mimo rozsah zobrazení

• Vyberte NEpravdivé tvrzení týkající se zobrazení celého čísla: rozsah zobrazení doplňkového kódu je symetrický

• **Inverzní kód** pro zobrazení celého čísla nemá jednu nulu

• **Znaménkový bit** pro zobrazení celého čísla má hodnotu 0 pro kladné číslo

• **Přetečení** v celočíselné aritmetice ve dvojkovém doplňkovém kódu nastane pokud se přenos ze znaménkového bitu nerovná přenosu do znaménkového bitu

• Osmičkovou a šestnáctkovou soustavu používáme, protože: zápis čísla je kratší než ve dvojkové soustavě

• Binární hodnota 0,1001 odpovídá dekadické hodnotě desetinného čísla: 9/16

• Při sčítání ve **dvojkovém doplňkovém kódu** platí: vznikne-li přenos ze znaménkového bitu, tak se ignoruje

• Dvojkové číslo 1000 v přímém kódu v zobrazení se znaménkem na 4 bitech je: záporná nula

• Dvojkové číslo 1000 v inverzním kódu v zobrazení se znaménkem na 4 bitech je: nejmenší zobrazitelné

• Dvojkové číslo 1111 v doplňkovém kódu v zobrazení se znaménkem na 4 bitech je: [největší zobrazitelné ,nejmenší zobrazitelné , kladná nula , záporná nula , \*žádná odpověď není správná]

• **Kruhový přenos** je: přičtení přenosu z nejvyššího řádu k výsledku

• Kladná čísla v zobrazení se znaménkem mají na n bitech: ve všech kódech stejný rozsah

• Které z dvojkových čísel v reprezentaci se znaménkem na 4 bitech je kladné? 0100 v inverzním kódu

• Kladná čísla v reprezentaci bez znaménka mají na n bitech rozsah: <0;2n-1>

• Rozsah zobrazení směrem ke kladným číslům a směrem k záporným číslům je rozložen asymetricky v: doplňkovém kódu

• Dvě reprezentace nuly se vyskytují v: přímém a inverzním kódu

• Která z čísel jsou shodná (nejvyšší bit je znaménkový)? 1101 v inverzním a 1110 v doplňkovém kódu

• **Rozsah zobrazení dvojkového doplňkového kódu** na n bitech je: <-2n-1;2n-1-1>

• Dvojkové číslo 1001 v reprezentaci se znaménkem na 4 bitech se v inverzním kódu rovná: -6

• Jak při sčítání binárních čísel ve dvojkovém doplňkovém kódu poznám, že došlo k přetečení? přenos do znaménkového bitu se nerovná přenosu ze znaménkového bitu

• Číslo 14 v decimální soustavě odpovídá: E v hexadecimální soustavě

• Kruhový přenos v inverzním kódu se využívá pro korekci při přechodu přes nulu

• Jednoduše nelze převádět čísla mezi soustavami o základech 4 a 40

• Osmičková soustava se také nazývá oktalová

**3 - kódy**

• V ASCII kódu má ordinální hodnota znaku návrat vozíku (**CR**) menší hodnotu než ordinální hodnota znaku 'A'.

• V ASCII kódu jsou znaky s ordinální hodnotou 0 až 31 označeny jako **řídící znaky**

• **Písmena s diakritikou** nejsou součástí vnějšího kódování ASCII

• Jaké kódování je korektní pro zobrazení všech českých znaků s diakritikou **ISO-8859-2**

• Znak **"Line feed"** je řídící znak s ordinální hodnotou nižší než 30

• Řídící znak "Line feed" znamená posun na další řádek se zachováním sloupce

• Řídící znak "**Carriage return**" znamená přesun na začátek téhož řádku

• Pro označení konce řádku v textovém souboru **MS-Windows** slouží kombinace znaků*: CR+LF*

• Konec řádku v textovém souboru operačního systému **UNIX** se označuje kombinací řídicích znaků: *LF*

• **Unicode** je vnější kódování znaků

• **UTF-8** zobrazuje jeden znak různým počtem bajtů

• **Unicode je vnější kódování a UTF-8 je způsob uložení**

• *Česká písmena s diakritikou jsou v UTF-8 uložena nejvíce na dvou bajtech*

• UTF-8 uloží znak z ASCII 7 na 1 bajtu

• *Počet bajtů, v kolika je uložen znak v UTF-8 (je-li uložen ve více než jednom bajtu), je vyjádřen počtem binárních jedniček v bitech nejvyšších řádů*

• Kolik bitů z UNICODE kódování lze zobrazit pomocí UTF-8? 31

• *Zobrazíme-li znak z kódování ASCII-7 v kódování UTF-8, co bude v bitu nejvyššího řádu? vždy 0*

• Vnější kódy ISO-8859-2 a Windows-1250 se liší v ordinální hodnotě znaku š

• Ve kterém vnějším kódování NEjsou česká písmena s diakritikou? ASCII

• **Detekční kód** je kód, který rozpozná chybu v uložené či přenášené informaci.

• **Opravný kód** je kód, který opraví chybu v uložené či přenášené informaci.

• Hammingova trojrozměrná krychle má 6 stěn.

• **BCD** (Binary Coded Decimal) znamená jedna desítková číslice uložená vždy na čtyřech bitech.

• BCD znamená Binary Coded Decimal

• BCD kód v každé čtveřici bitů ukládá jednu desítkovou číslici

• Kladné číslo v **rozvinutém BCD tvaru** je F7F1F3F4C6

• Číslo, které je v rozvinutém BCD tvaru uloženo na 5 bajtech, bude ve **zhuštěném BCD tvaru** uloženo ve 3 bajtech

• V čem je uznávaná *výhoda* zobrazení čísel v BCD kódu oproti zobrazení čísel v přímém binárním kódu? jednodušší převod čísla do desítkové soustavy

• Co znamená kód **2 z 5**? způsob zabezpečení, právě dva bity jsou rovny jedné

• Při Hammingově vzdálenosti (d) pět mohu kód opravit, pokud vzniknou maximálně dvě chyby

• Při Hammingově vzdálenosti (d) dva jsem schopen detekovat chybu a nejsem schopen ji opravit

*• Sudá parita znamená počet bitů vč. paritního obsahujících hodnotu 1 je sudý*

• Mějme detekční kód 2 z 5. Které z následujících čísel obsahuje chybu? 11010

• **Ztrojení** je příkladem opravného kódu

*• Kódová (Hammingova) vzdálenost je: počet bitů, v nichž se liší dvě sousední platné kódové kombinace*

• Pro Hammingovu vzálenost 1 platí žádnou chybu nelze detekovat, tedy ani opravit

• Kolik chyb jsme schopni detekovat, jestliže kódová vzdálenost d=3? dvě

• Kolik chyb jsme schopni opravit, jestliže kódová vzdálenost d=3? jednu

• *V opravném kódu v případě ztrojení každého bitu jsme schopni dvě chyby detekovat a jednu chybu korektně opravit*

• Jaké ordinální hodnoty mají číslice v EBCDIC (vnější kód BCD)? F0 až F9

• Co znamená **Big-Endian**: bajt nejvyššího řádu je na nejnižší adrese

• Co znamená použití pořadí **Little-Endian**? Bajt nejnižšího řádu je uložen na nejnižší adrese.

• Little-Endian a Big-Endian jsou způsoby ukládání bajtů ve slově

• Jak na čísle ve dvojkovém doplňkovém kódu poznáme, zda je uloženo v Big-Endian nebo Little-Endian: nelze to ze zápisu čísla jednoznačně poznat

**4 - Obvody**

• Mezi operace **Booleovy algebry** nepatří logický rozdíl

• *Sériové zapojení* vyjádřené v Booleově algebře znamená *logický součin*

• *Paralelní zapojení* vyjádřené v Booleově algebře znamená *logický součet*

• Který z uvedených způsobů se nepoužívá pro minimalizaci výrazu? jednotková kružnice

• Proč není *Booleova* algebra vhodná pro technickou realizaci? obsahuje příliš mnoho operací

• Jaké operace využívá **Shefferova algebra**? jedinou operaci a to negovaný logický součin (NAND)

• Shefferova algebra (NAND) se používá místo Booleovy algebry v technických zapojeních, protože má jen jednu operaci.

• **Zakázané pásmo** v obvodech vymezuje hodnoty signálu, ve kterých se signál nesmí nacházet během jeho vzorkování

• Zakázané pásmo v obvodech je rozsah hodnot, ve kterém se signál nesmí nacházet v okamžiku vzorkování.

• Napájecí napětí technologie **TTL** je 5 V

• **Invertor** je logický člen měnící logickou 0 na logickou 1 a opačně

• Výstupní hodnota logického členu NOR je rovna 1, když všechny vstupní hodnoty jsou 0.

• Výstupní hodnota logického členu NOR je rovna 0, když aspoň jedna vstupní hodnota je 1.

• Výstupní hodnota logického členu NAND je rovna 0, když všechny vstupní hodnoty jsou 1.

• Výstupní hodnota logického členu NAND je rovna 1, když aspoň jedna vstupní hodnota je 0.

• Mezi kombinační logické obvody patří NAND, NOT, multiplexor

• Mezi kombinační logické obvody patří **sčítačka pro jeden binární řád**

• Kombinační logický obvod "**nonekvivalence**" má stejnou funkci jako: sčítačka modulo 2

• **Klopný obvod RS** v obecném případě nesmí mít na vstupu kombinaci 00, pokud je řízen nulami

• **Parita** je způsob zabezpečení informace proti chybě.

• **Multiplexor** se čtyřmi datovými vstupy je obvod, který dle zadané adresy vybere jeden ze vstupních signálů a předá jej na výstup.

• Multiplexor se 16 datovými vstupy potřebuje 4 adresové vstupy.

• Multiplexor, který má celkem 6 vstupů, má 4 datové a 2 adresové vstupy

• **Dekodér**, který má 2 vstupy, má 4 výstupy.

• Dekodér, který má 8 vstupů, má 256 výstupů

• Dekodér, který má 4 vstupy, má 16 výstupů

• **Úplná sčítačka** pro jeden binární řád má dva bity sčítanců a přenos na vstupu a jeden bit součtu a přenos na výstupu.

• **Sekvenční logické obvody** mají vnitřní stav.

• Zakázaný stav u **klopného obvodu R-S** řízeného jedničkami je stav, kdy R=1 a S=1.

• **Klopný obvod** je název obvodu ze skupiny sekvenčních logických obvodů.

• **Sčítačka** pro jeden řád BCD kódu se realizuje pomocí dvou čtyřbitových sčítaček. Pokud je součet dvou BCD číslic klasickou sčítačkou větší než 9 provádí se korekce přičtením čísla 6

• Žádný bit se neztrácí při rotaci bitů.

• **Aritmetickým posunem** doleva se násobí

• Násobení dvěma lze realizovat aritmetickým posunem o jeden bit doleva

• Operaci celočíselného dělení dvěma lze provést aritmetickým posuvem obsahu registru doprava

• Aritmetickým posunem doprava dojde u kladného čísla typicky ke zmenšení čísla

• **Boolova algebra** není vybudována na operaci negovaného logického součinu

• Boolova algebra užívá tři základní operace

• Boolova algebra je nauka o operacích na dvouprvkové množině

• Technologie TTL používá jako svůj základní prvek tranzistor NPN

• Pro technickou realizaci je nejméně vhodná Booleova algebra

• **Shefferova algebra** je vybudována pouze na jediné logické operaci, a to NAND

• **Piercova algebra** je vybudována pouze na jediné logické operaci, a to NOR

• Základním stavebním prvkem technologie TTL je bipolární tranzistor

• Logický obvod NAND pro vstupy 0 a 0 dá výstup 1

• Logický obvod NOR pro vstupy 1 a 0 dá výstup 0

• Logický obvod XOR (nonekvivalence) pro vstupy 0 a 0 dá na výstup 0

• Negaci bitu provádí: *invertor*

• Pro výběr jednoho z n vstupů slouží: *multiplexor*

• n adresových vstupů a 2n datových výstupů má: *dekodér*

• **Impuls** je dočasná změna hodnoty signálu

• Mezi **sekvenční logické obvody** patří klopný obvod JK, klopný obvod RS, klopný obvod D

• Zakázaný stav se nachází u žádná z uvedených možností

• Sekvenční logické obvody se vyznačují tím, že výstup závisí na předchozí posloupnosti změn

• Výstupy z eventuální sčítačky **Modulo 4** mohou nabývat hodnoty 0, 1, 2, 3

• Pro **kombinační logické obvody** platí, že výstupy nezávisí na předchozí posloupnosti změn

• Signálem **Reset** je návrat do předem definovaného stavu

• Mezi kombinační logické obvody nepatří žádná z uvedených možností

• Zakázaný stav **klopného obvodu JK** nastane když: žádná z uvedených možností [spravne: J=0, K=0 … J=1, K=1 … J=1, K=0]

• Korekce pro BCD sčítačku nepřičítá šestku, když přenosový bit součtu je roven nule

• **Logický posun** nenulového obsahu registru doprava nejnižší bit se ztrácí

• Aritmetický posun nenulového obsahu registru doleva způsobí obsah registru se celočíselně vynásobí dvěma, nezmění se znaménko, nedošlo-li k přetečení

• Pokud se obsah registru posune aritmeticky doprava a číslo se blíží k maximální hodnotě, kterou lze do registru uložit, pak obsah bude celočíselně vydělen dvěma

• **Jednotka Baud** udává počet změn stavů přenesených za sekundu

• Při stejné přenosové rychlosti je vždy počet bitů přenesených za sekundu větší nebo roven počtu baudů

• Jako sčítačka modulo 2, která neřeší přenosy, funguje logický člen XOR

• **Polosčítačka** se dvěma vstupy dává na výstup přenos do vyššího řádu

• **Klopný obvod RS** řízený nulami pro hodnoty 1 a 1 setrvává v předchozím stavu

• "R" v názvu klopného obvodu RS znamená reset

• **Registry** jsou typicky konstruovány z klopného obvodu D

• Při dvoustavové komunikaci je rychlost přenosu udávaná v baudech (Bd) stejná jako rychlost udávaná v bitech za sekundu

• Při čtyřstavové komunikaci je rychlost přenosu udávaná v baudech (Bd) menší než rychlost udávaná v bitech za sekundu

• Pod pojmem "**zakázané pásmo**" při přenosu signálu rozumíme rozsah napětí, v jehož rámci je hodnota signálu nedefinovaná

• Pro **multiplexor** neplatí: má adresový výstup (Platí: má adresové vstupy, má datové vstupy,

má datový výstup)

• Jaký zakázaný stav má klopný obvod RS řízený jedničkami? 1,1

• Pod **rotací bitů** vlevo rozumíme posuv z nižšího řádu do vyššího, žádná hodnota bitu se neztrácí

• Pod rotací bitů vpravo rozumíme posuv z vyššího řádu do nižšího, žádná hodnota bitu se neztrácí

• Pod pojmem **logický posun** vlevo rozumíme posuv z nižšího řádu do vyššího, ztrácí se hodnota některého bitu

• Pod pojmem logický posun vpravo rozumíme posuv z vyššího řádu do nižšího, ztrácí se hodnota některého bitu

• Při **aritmetickém posunu** se nemění hodnota znaménkového bitu, nedojde-li k přetečení

• V techologii TTL při použití tranzistoru NPN se kolektor a emitor otevírá když je na bázi přivedena vysoká úroveň -- logická jednička

• K čemu se využívá **Karnaughova mapa** k minimalizaci počtu operací B-algebry

• Pokud jsou 1 a 1 na vstupu sčítačky modulo 2, pak na výstupu je 0

• Mám 16 zařízení, zařízení číslo 10 chci poslat signál 1, ostatním 0. Co použiji? dekodér

• Pro úplnou sčítačku pro jeden binární řád platí má 3 vstupy a 2 výstupy

• Co platí pro klopný obvod D? je to paměť na jeden bit

• *NOXOR* je stejný jako: *ekvivalence*

Které zapojení nelze popsat pomocí Booleovy algebry? Můstkové

Sekvenční logické obvody mají vnitřní stav. Kombinační nemají.

**5 – Paměti**

* Která paměť musí být **energeticky nezávislá**? vnější paměť (závislé: vnitřní paměť, registry)
* *Obsah adresového registru* paměti se na výběr jednoho z výběrových (adresových) vodičů převádí *dekodérem 1 z N*.
* K *destruktivnímu nevratnému zápisu do permanentní paměti* pomocí přepalování tavných spojek proudovými impulsy je určena **paměť PROM**
* Parametr pamětí "*vybavovací doba - čas přístupu*" bude nejvyšší u diskové paměti (ostatní: operační paměti, vyrovnávací (cache) paměti , registru)
* Paměť, která svůj obsah *adresuje klíčem*, který je uložen odděleně od obsahu paměti a vyhledává se v klíči paralelně, se nazývá **asociativní paměť.**
* Paměť typu **cache** nebývá umístěna *mezi procesorem a registry*
* Do paměti typu **PROM** lze zapsat data pouze jednou
* Které tvrzení NEplatí pro popis fyzické **struktury vnitřní paměti**? Datový registr má na vstup přivedeny adresové vodiče. (platí: Dekodér na jeden z adresových vodičů nastaví hodnotu logická 1. Informace je na koncích datových vodičů zesílena zesilovačem. Adresa je přivedena na vstup dekodéru. Podle zapojení buněk na řádku projde/neprojde logická 1 na datové vodiče.)
* Máme-li vnitřní paměť o kapacitě 16 bitů zapojenou jako matici paměťových buněk 4x4 bity, pak *nejmenší adresovatelná jednotka* je 4 bity
* Působením UV záření je možné vymazat obsah paměti **EPROM** .
* Statickou, energeticky nezávislou pamětí není paměť typu - žádná z odpovědí není správně . (Správně: ROM , PROM , EPROM , EEPROM)
* Vybavovací doba paměti znamená čas přístupu k jednomu záznamu v paměti .
* Pro **paměť s přímým přístupem** platí doba přístupu k libovolnému místu v paměti je konstantní .
* **Energeticky závislá paměť** obecně obsahuje po obnově napájení - obsah paměti je nedefinovaný .
* Energeticky závislá paměť typicky je paměť **RAM** .
* Správný *postup čtení dat z paměti* je: procesor vloží adresu do adresového registru, příkaz čti, procesor převezme informaci z datového registru
* Paměť určená pro čtení i pro zápis má zkratku **RWM** .
* Zpětnému proudu v **ROM** pamětech zabraňuje použití *polovodičů*
* Kolikrát je možno zapisovat do paměti **PROM**? lze jednou naprogramovat
* Ultrafialovým světlem lze přemazat paměť **EPROM**
* Elektrickým proudem lze přemazat paměť **EEPROM**
* Paměť, ze které se většinou čte, maže se elektrickým proudem a dá se do ní i zapisovat má zkratku **RMM**
* Pro **asociativní paměť** NEplatí : používá se jako operační paměť . (Platí: v paměti se plní klíč a obsah, paměť klíčů se prohledává paralelně, zkratka je **CAM**)
* **CAM paměti** předám adresu. Nejdříve ji hledá v paměti klíčů
* Jaké **sběrnice** jsou mezi procesorem a pamětí? **datová a adresová**
* Jakou funkci u paměti má **refresh cyklus**? obnovuje data uložená v dynamické paměti
* Mezi paměti s výhradně s **přímým přístupem patří operační paměť .**
* Která z uvedených pamětí není programovatelná? **ROM .** (ostatní: PROM,EPROM ,EEPROM)
* Pro **statickou paměť** NEplatí: informace se udržuje, i když není napájení . (Platí: informace se neudržuje, když není napájení ; informace se udržuje, pokud je napájení )
* **ROM** je paměť pouze pro čtení
* **ROM** je zkratka pro read only memory .
* Páska je paměť se **sekvenčním přístupem**
* Na libovolnou adresu v paměti s **přímým přístupem** se dostanu typicky za konstantní čas

**6 – Procesor**

• **Registr PC** -- čítač instrukcí v procesoru obsahuje adresu právě prováděné instrukce.

• Jednou z **fází zpracování instrukce procesorem** NEní: kopírování instrukce do paměti [Ostatní, které jsou: výběr operačního kódu z paměti , výběr adresy operandu z paměti , provedení instrukce , zápis výsledků zpracované instrukce ]

• Pro adresaci operační paměti mající kapacitu 64 K adresovatelných jednotek (bajtů) je třeba adresová sběrnice šířky: 16 bitů.

• Pro adresaci operační paměti mající kapacitu 1 M adresovatelných jednotek (bajtů) je třeba adresová sběrnice šířky: 20 bitů.

• Pro adresaci uvnitř 4KB stránky potřebuji adresu širokou: 12 bitů.

• Pro adresaci 4 MB paměti potřebuji adresu širokou: 22 bitů

• *Jak široká musí být adresa, pokud chceme adresovat 1 M stránek a každá stránka má velikost 4 K adresovatelných jednotek. 32 bitů.*

• PC -> AR, 0 -> WR, DR -> IR

PC+1 -> AR, 0 -> WR, DR -> TAL

PC+2 -> AR, 0 -> WR, DR -> TAH

TA -> AR, 0 -> WR, DR -> A

PC+2 -> PC: …. tyto mikroinstrukce jsou nekorektní

• Mezi **aritmetické instrukce** fiktivního procesoru definovaného na přednáškách patří pouze tyto **ADD, CMA, INR**

• Příznaky pro větvení programu vždy nastavují tyto instrukce fiktivního procesoru definovaného na přednáškách **ADD, INR, CMA**

• Příznaky pro větvení programu nikdy nemění tyto instrukce fiktivního procesoru definovaného na přednáškách **MOV, STA, JMP**

• Pro nastavení příznaků v procesoru definovaném na přednáškách použiji instrukci **CMP**

• Instrukce mající zkratku **LDA** typicky znamená naplň obsah registru A hodnotou z paměti.

• Instrukce mající zkratku **JMP** typicky provádí nepodmíněný skok.

• **Příznakový registr** procesoru se používá na realizaci podmíněných skoků.

• Instrukce **CMP** pro porovnání typicky pouze nastaví příznaky.

• Posloupnost instrukcí:

LDA x

MOV B,A

LDA y

CMP B

JP ne

ano: ...

JMP ven

ne: ...

ven: ...

o vyjadřuje příkaz: IF x>y THEN ano ELSE ne

• Posloupnost instrukcí:

LDA y

MOV B,A

LDA x

CMP B

JM ne

ano: ...

JMP ven

ne: ...

ven: ...

o vyjadřuje příkaz: IF x>=y THEN ano ELSE ne;

• Posloupnost instrukcí:

LDA y

MOV B,A

LDA x

CMP B

JP ne

ano: ...

JMP ven

ne: ...

ven: ...

o vyjadřuje příkaz: IF x<y THEN ano ELSE ne;

• Posloupnost instrukcí:

LDA x

MOV B,A

LDA y

CMP B

JM ne

ano: ...

JMP ven

ne: ...

ven: ...

o vyjadřuje příkaz: IF x<=y THEN ano ELSE ne

• PC -> AR, 0 -> WR, DR -> IR

PC+1 -> AR, 0 -> WR, DR -> TAL

PC+2 -> AR, 0 -> WR, DR -> TAH

TA -> AR, 0 -> WR, DR -> TAXL

TA+1 -> AR, 0 -> WR, DR -> TAXH

TAX -> AR, A -> DR, 1 -> WR

PC+3 -> PC ….. jsou mikroinstrukce STAX (nepřímé naplnění)

• *Instrukce podmíněného skoku provede následující instrukci, pokud podmínka splněna není.*

• Operace **PUSH** nad zásobníkem vloží položku do zásobníku.

• Instrukce **PUSH** *osmibitového procesoru definovaného na přednáškách vkládá do zásobníku 1 dvoubajtové slovo*

• Jaký je správný postup operací? *PUSH sníží SP a uloží položku na adresu podle SP; POP vybere z adresy podle SP a zvýší SP.*

• *Instrukce volání podprogramu musí uchovat návratovou adresu*

• Pojem **'time-out'** při provádění **V/V** operací znamená, že např. zahájená výstupní operace neodpověděla 'hotovo' do definované doby.

• Posloupnost instrukcí:

START

opak: FLAG opak

IN

STA x

je podle toho, jak jsme si na přednáškách definovali vlastní procesor (pomíjíme otázku time-outu, neefektního využití procesoru): korektní operace čtení ze vstupního zařízení

• Posloupnost instrukcí:

LDA x

START

OUT

opak: FLAG opak

je podle toho, jak jsme si na přednáškách definovali vlastní procesor (pomíjíme otázku time-outu, neefektního využití procesoru),žádná z ostatních odpovědí není správná

• Ve kterém z následujících okamžiků by mělo dojít ke *vzniku přerušení*? konec tisku znaku

• Které z konstatování vztahujících se k *okamžiku přerušení procesu* je NEsprávné? Přerušení

nastane ihned po žádosti signálem INTERRUPT. [spravne: Přerušit nelze bezprostředně po zahájení obsluhy přerušení. Přerušit lze pouze tehdy, je-li to povoleno (nejde-li o nemaskovatelné přerušení). Přerušit nelze během provádění instrukce.]

• Jaké je správné modelové chování obsluhy vzniku přerušení? Mikroinstrukce musí uložit PC a vynulovat IF. Programem se ukládají všeobecné registry.

• **Operační kód** (operační znak) je numerické vyjádření konkrétní instrukce, má typicky proměnlivou délku

• **Operační kód** je: operační znak , numerické vyjádření konkrétní instrukce, které má proměnlivou délku , součást instrukce

• Pro **čítač instrukcí procesoru** platí: může mít zkratku PC , může mít zkratku IP , obsahuje adresu prováděné instrukce

• Která instrukce naplní registr A obsahem slabiky z paměti? **LDA**

• Instrukce **STA** uloží registr A do paměti

• Instrukce **JMP** je nepodmíněný skok

• Osmibitový procesor se 64KB pamětí má 8bitovou datovou sběrnici a 16bitovou adresovou sběrnic

• Registr PC procesoru naplníme instrukcí JMP

• Pomocný **16bitový registr TA** procesoru definovaného na přednáškách se skládá z 8bitového TA High a 8bitového TA Low

• *První fází každé instrukce je výběr operačního znaku*

• Pro mikroinstrukci výběr operačního znaku NEplatí: cílem je vložit do datového registru data [Plati: je součástí např. instrukce LDA , je vždy 1. fází instrukce, cílem je vložit do instrukčního registru instrukci ]

• Mikroinstrukce výběr operačního znaku znamená: procesor zjistí, kterou instrukci provádí

• Mezi mikroinstrukce instrukce LDA NEpatří: naplnění registru PC hodnotou operandu instrukce . [Patri: výběr operačního znaku , výběr operandu , aktualizace registru PC zvýšením o délku instrukce ]

*• Instrukce* **INR** procesoru definovaného na přednáškách způsobí: zvýší obsah registru o jedna

• *Instrukce* **CMA** procesoru definovaného na přednáškách způsobí: inverzi bitů v registru A

• Která instrukce sníží obsah registru o jedna: žádná z uvedených možností [ADD, CMA , INR ]

*• Instrukce* **ADD** procesoru definovaného na přednáškách přičte obsah registru k registru A

• Příznak procesoru definovaného na přednáškách není: žádná z uvedených možností [CY (Carry) , Z (zero) , jednobitový indikátor ]

• **S (Sign)** je *příznak* procesoru definovaného na přednáškách, kterým je kopie znaménkového bitu výsledku operace

• Pro **příznaky** procesoru definovaného na přednáškách platí: nastavuje je procesor .

• *Příznaky* procesoru definovaného na přednáškách mění instrukce **INR, ADD, CMA**

• *Instrukce* procesoru definovaného na přednáškách **CMP B** porovná obsah registru A s obsahem registru B a změní podle toho příznaky

• Mezi *příznaky* procesoru definovaného na přednáškách NEpatří: TA [Patri: CY, A, Z]

• Změnu znaménka u čísla v registru A procesoru definovaného na přednáškách provedeme **posloupností instrukcí CMA, INR A**

• Pro **zásobník** procesoru definovaného na přednáškách NEplatí, že: je datová struktura fungující systémem FIFO [Plati: je datová struktura fungující systémem LIFO , vkládá se do ní operací PUSH , vybírá se z ní operací POP ]

• **PUSH** procesoru definovaného na přednáškách je *instrukce*, vkládá obsah registru do zásobníku

• **PSW** procesoru definovaného na přednáškách je *stavové slovo procesoru*, tvořeno z registru A a příznaků

• Pro **zásobník** procesoru definovaného na přednáškách platí: nemá kontrolu podtečení

• **LXISP** procesoru definovaného na přednáškách definuje dno zásobníku

• *Instrukce* **PUSH** procesoru definovaného na přednáškách numericky snižuje ukazatel vrcholu zásobníku

• *Instrukce* **POP** procesoru definovaného na přednáškách : žádná z uvedených možností [Moznosti: definuje dno zásobníku , snižuje ukazatel vrcholu zásobníku , dekrementuje SP]

• Pro *instrukci* **RET** procesoru definovaného na přednáškách NEplatí: vrátí se na absolutní začátek programu [Plati: vrátí se z podprogramu do těla programu , obsah vrcholu zásobníku je vložen do registru PC , používá se na konci podprogramu ]

• Která *posloupnost* instrukcí může korektně obsloužit time-out při programování V/V operace procesoru definovaného na přednáškách:

100 START

101 INR

102 JZ 108

105 FLAG 101

• *Instrukce* **OUT** procesoru definovaného na přednáškách zapíše obsah reg. A na datovou sběrnici pro v/v zařízení

• Která *instrukce* procesoru definovaného na přednáškách skočí na adresu, není-li operace hotova? FLAG

• Posloupnost instrukcí procesoru definovaného na přednáškách - LDA x, OUT, START, FLAG – je: korektní operace zápisu do výstupního zařízení

• Posloupnost instrukcí procesoru definovaného na přednáškách - START, IN, STA x, FLAG – je: žádná z ostatních odpovědí není správná [moznosti: korektní operace čtení ze vstupního zařízení , korektní operace zápisu do výstupního zařízení ]

• Co je **time-out**? doba, kterou jsme ochotni čekat na dokončení V/V operace

• *Signál* **INTERRUPT (INTR)** žádá o přerušení v procesoru

• Která činnost se vykonává jako poslední při návratu z přerušení procesoru definovaného na přednáškách? obnovení PC, A, ...

• Pro přerušení platí: *o přerušení se musí požádat signálem* ***INTERRUPT***

• Instrukce, která zakáže přerušení procesoru definovaného na přednáškách, se nazývá: **CLI**

*• Co je v registru PC* procesoru definovaného na přednáškách při uplatnění žádosti o přerušení: **adresa instrukce, která nebyla provedena v důsledku přerušení**

• *Během uplatnění přerušení NEní provedeno:* povolení přerušení [je provedeno: uložení registru PC do zásobníku , vynulování IF, uklizení registru A a dalších do zásobníku ]

• Která z instrukcí NEpatří mezi instrukce procesoru definovaného na přednáškách, které se použijí při návratu z přerušení: CLI [Patri: POP, STI, RET]

• Co NEplatí pro instrukci STI procesoru definovaného na přednáškách: povolí přerušení po svém dokončení [Plati: nastaví IF na hodnotu 1 , povolí přerušení až po provedení následující instrukce ]

• *Signál* **RESET** procesoru definovaného na přednáškách NEzpůsobí: vynulování příznaků procesoru [Zpusobi; nastavení procesoru do počátečních podmínek , vynulování IF, zakázání přerušení , předání řízení na adresu ukazující zpravidla do permanentní paměti ]

• Pro *signál* **RESET** procesoru definovaného na přednáškách NEplatí: provede se pouze při přerušení [Plati: provede se kdykoliv , nastaví IF na nulu , předá řízení na adresu ukazující zpravidla do v permanentní paměti ]

• *Výběr instrukcí procesoru definovaného na přednáškách je řízen* **registrem: PC**

• Který z registrů procesoru definovaného na přednáškách NEní 16bitový: IR [Jsou: PC,TA, AR]

• Která instrukce procesoru definovaného na přednáškách NEnastavuje příznaky: LDA [Nastavuje: INR, ADD, CMA]

• Která instrukce procesoru definovaného na přednáškách *nastavuje příznaky*: **ADD**

• Která instrukce procesoru definovaného na přednáškách *porovná zadaný registr s registrem A*: **CMP**

• Zásobník má strukturu: **LIFO** (last in, first out)

• Fronta má strukturu: **FIFO**

• Pro *instrukci* **CALL** procesoru definovaného na přednáškách NEplatí: přečte obsah zadaného registru [Plati: uloží návratovou adresu do zásobníku¸provede nepodmíněný skok na zadanou adresu , provede totéž co posloupnost instrukcí PUSH a JMP]

*• Procesor rozlišuje komunikaci s pamětí a se V/V zařízeními*: **signálem M/IO**

• Jak široká musí být adresa, pokud chceme adresovat 1 K stránek a každá stránka má velikost 4 K adresovatelných jednotek. 22 bitů.

• Pokud používáme **virtualizaci paměti**, pak šířka virtuální adresy by měla být větší nebo rovna šířce reálné adresy.

• K obecnému mechanismu virtuální paměti: Co je obvyklé? **Počet stránek je větší než počet rámců.**

• K obecnému mechanismu virtuální paměti: *Která z adres může být širší (má se na mysli, že je více bitová)*: **virtuální**

• K obecnému mechanismu virtuální paměti: Co platí? **Stránky jsou uloženy na disku, rámce jsou v reálné paměti.**

• K obecnému mechanismu algoritmu LRU: **Algoritmus LRU vybírá nejdéle nepoužitou položku**

• **Algoritmus LRU** *pro výběr oběti např. při virtualizaci paměti vybírá nejdéle nepoužitý obsah rámce.*

• Při **virtualizaci** paměti se používají pojmy rámec a stránka.

**• Špinavá stránka** je stránka změněná v reálné paměti

• K obecnému mechanismu algoritmu LRU: K úplnému ošetření osmi položek algoritmem LRU (pomocí neúplné matice) bychom potřebovali kolik bitů v neúplné matici? 28

• K obecnému mechanismu algoritmu LRU: K úplnému ošetření šesti položek algoritmem LRU (pomoc í neúplné matice) bychom potřebovali kolik bitů v neúplné matici? 15

• Pro virtualizaci paměti NEplatí: rámec není stejně velký prostor jako stránka [Plati: paměť dělíme do rámců a disk na stránky, reálná adresa ukazuje do reálné paměti , počet stránek je větší nebo roven počtu rámců ]

• Při virtualizaci paměti NEplatí: rámec je špinavý, pokud má příznak parity nastaven na jedničku [Plati: obsah špinavého rámce musím před jeho smazáním zapsat na disk…, označení čistý rámec odpovídá označení rámec, do kterého nebylo zapsáno … do špinavého rámce bylo něco zapsáno ]

• Pro **virtualizaci** paměti se používá stránka

• Co platí pro **segmenty a stránky:** segmenty jsou různé velikosti, stránky jsou stejné velikosti

• Co znamená **LRU: least recently used**

• Jaká je NEsprávná konfigurace virtuální paměti u obecného procesoru?: 32bitová reálná adresa a 24bitová virtuální adresa [Spravna: 32bitová reálná adresa a 48bitová virtuální adresa , 24bitová reálná adresa a 24bitová virtuální adresa ]

**7 – 86**

* Jaká je maximální hodnota adresy reálné paměti v procesoru Intel 8086: 104857510
* Adresa 02AB:00A416 reálného režimu procesorů Intel se vyčíslí na hodnotu: 2B5416
* Na jaké hodnoty se nastaví bity příznakového registru provedením instrukce ADD v procesorech Intel řady 86 s operandy -5 a 8: CF=1, ZF=SF=OF=0
* Jakou instrukcí v procesorech Intel x86 naplním registr příznaků F? POPF
* Jaký je korektní postup činností při přerušení v procesoru Intel 8086? do zásobníku se uloží obsah reg. příznaků vynulují se příznaky IF a TF do zásobníku se uloží CS registr CS se naplní obsahem z n x 4+2 do zásobníku se uloží IP registr IP se naplní obsahem z n x 4
* Jaká je poslední (20bitová) adresa tabulky přerušovacích vektorů v procesoru Intel 8086 a reálných režimech procesorů vyšších: 102310
* Adresový prostor adres V/V zařízení v procesorech Intel (typicky 8086) je 16bitový .
* Kolik různých přerušení může vzniknout v procesoru Intel 8086 a reálných režimech procesorů vyšších 256 .
* Která z uvedených variant instrukce MOV v procesorech Intel je nekorektní? MOV prom1,prom2
* Která z uvedených variant instrukce MOV v procesorech Intel je nekorektní? MOV AL,BX
* Jaké dvě různé operace se v procesorech Intel realizují jedinou instrukcí? SAL a SHL provádí SHL (arit. a logický posun bitů vlevo se vždy provádí jako logický posun vlevo)
* Které varianty instrukce JMP v procesorech Intel přiřazují (nepřičítají) operand do registru IP? vzdálený (far) skok a nepřímý skok
* Které varianty instrukce JMP v procesorech Intel přičítají (nepřiřazují) operand k obsahu registru IP? krátký (short) skok a blízký (near) skok
* Programujeme cyklus typu REPEAT, ve kterém na konci bloku testujeme, zda je hodnota i>5. Pokud ano, pak provádění bloku opakujeme. Neznáme však velikost bloku, který musíme opakovat. Blok začíná návěštím "Blok" a programujeme jej na procesoru Intel 8086. Jaká bude správná a nejbezpečnější realizace podmínky?
  + CMP i,5
  + JLE Dále
  + JMP Blok
  + Dále:
* Čím se procesor 8088 liší od procesoru 8086: 8088 je určen pro vnější osmibitové prostředí
* NMI - nemaskovatelné přerušení se používá například při hlášení chyb parity paměti
* Při adresaci paměti procesoru 8086 neplatí: používá se 32bitová adresa složená z dvou 16bitových komponent [plati: adresu zapisujeme ve tvaru segment: offset , používá se 20bitová adresa složená z dvou 16bitových komponent ]
* Mezi segmentové registry nepatří: PC [patri: CS , SS , DS ]
* Pro registr CS platí je určen pro výpočet adresy instrukce
* Pro registr IP neplatí: obsahuje pomocný datový segment [ plati: obsahuje část adresy právě prováděné instrukce , je ekvivalentem registru PC ]
* Adresu paměti u procesoru 8086 zapisujeme ve tvaru segment: offset.
* Jakou velikost má jeden segment v procesoru 8086 64 KB
* Segment procesoru 8086 začíná na adrese dělitelné 16
* Jaká je korektní posloupnost operací při uplatnění přerušení v procesoru 8086? PUSH F; IF:=0; PUSH CS; PUSH IP
* Instrukce IRET procesoru 8086 obnovuje ze zásobníku obsahy registrů IP, CS, F
* Jaký rozsah adres v procesoru 8086 bude přepsán, pokud se v nekonečné smyčce zacyklí použití instrukce PUSH AX? SS:0000-SS:FFFF
* V trasovacím režimu (TF=1) procesoru 8086 se provedení jedné instrukce spustí instrukcí IRET
* Trasovací režim procesoru 8086 se spouští nastavením TF=1 v příznaku TF
* Trasovací režim procesoru 8086 se spouští nastavením TF=1 a ukončuje se neukončuje se
* Důvodem, proč po použití instrukce MOV SS,... v procesoru 8086 se zakazuje přerušení na dobu provádění jedné následující instrukce, je atomické naplnění adresy vrcholu zásobníku
* Nepovolená instrukce v procesoru 8086 je MOV CS,..
* Programátor procesoru 8086 nastavuje příznaky DF, IF, TF
* Příznak ZF procesoru 8086 je nastaven při nulovém výsledku operace
* Příznak TF procesoru 8086 uvede procesor do krokovacího režimu
* Všechny odkazy na zásobník procesoru 8086 jsou segmentovány přes registr SS (Stack segment)
* Pro vnější přerušení procesoru 8086 neplatí: vyvolá se např. při dělení nulou [plati: vyvolá se pomocí signálu INTERRUPT, vyvolá se pomocí signálu NMI, dělí se na maskovatelná a nemaskovatelná ]
* Pro vnitřní přerušení procesoru 8086 neplatí: je generováno řadičem přerušení . [plati: vyvolá se chybou při běhu programu , je generováno programově , vyvolá se instrukcí INT n ]
* Akce, která se neprovádí při přerušení procesoru 8086: provede se instrukce OUT [plati: vynulují se příznaky IF a TF , do zásobníku se uloží registr CS , do zásobníku se uloží registr IP]
* Pro tabulku adres rutin obsluhujících přerušení procesoru 8086 neplatí: začíná na adrese SS:0000 [plati: má 256 řádků , začíná na začátku adresového prostoru , začíná na adrese 0:0000 ]
* Při přerušení v procesoru 8086 se jako první operace provádí do zásobníku se uloží registr příznaků (F)
* Při návratu z přerušení v procesoru 8086 se provádí instrukce IRET, pro níž neplatí: ze zásobníku se obnoví registr SS [plati: ze zásobníku se obnoví příznakový registr , ze zásobníku se obnoví registr CS , ze zásobníku se obnoví registr IP]
* Návrat do přerušeného procesu v procesoru 8086 typicky zajistí instrukce IRET .
* Mezi rezervovaná přerušení procesoru 8086 nepatří: časovač [patri: ladící bod , krokovací režim , pokus o dělení nulou ]
* Pro trasovací režim procesoru 8086 neplatí: probíhá, když je TF nastaven na nulu [plati: po provedení instrukce je generováno přerušení INT 1 , procesor je uveden do krokovacího režimu příznakem TF (Trace Flag) , krokovací režim využívá instrukci IRET ]
* Příznak TF procesoru 8086 se nastaví na jedničku při obnově příznakového registru (F) ze zásobníku instrukcí IRET
* Signál RESET procesoru 8086 neprovede: nastaví TF = 1 [provede: vynuluje IP a SS, vynuluje příznakový registr ]
* Chci naplnit registr AH procesoru 8086 hodnotou 50, které řešení není správné: MOV AH,[50] [spravne: MOV AH,50; PADESAT DB 50, MOV AH,PADESAT]
* Chci naplnit registr AH procesoru 8086 obsahem adresy 50, které řešení je správné: MOV AH,[50]
* Instrukce procesoru 8086 MOV AH,[BX] provede: hodnota, která je na adrese v registru BX, se uloží do AH
* Instrukce procesoru 8086 MOV AH,[BX][DI] provede: hodnota, která je na adrese, jež vznikne součtem adres v registrech BX a DI se uloží do AH
* Který ze zápisů instrukcí procesoru 8086 je korektní operací? Všechny: MOV AX,BX ; MOV AX,[BX] ; MOV AX,PROM[BX][DI]
* Pro instrukci MOV procesoru 8086 neplatí: mění příznaky [plati: nelze s ní měnit registr CS ; má povolen tvar MOV BX,CX; nemá povolen tvar MOV adresa,adresa
* Který ze zápisů instrukcí procesoru 8086 je špatně: MOV CS,DS [spravne: MOV DS,adresa; MOV adresa,DS ; MOV CX,DX ]
* Pro instrukci procesoru 8086 MOV SS,... platí: po dobu trvání následující instrukce je zakázáno přerušení
* Pro aritmetické instrukce procesoru 8086 neplati: [všechny neplati] nesmí nastavovat příznaky; nepatří sem instrukce ADD ; nepatří sem instrukce INC
* Pro znaménkové rozšíření procesoru 8086 neplatí: všechny bity původního objektu se zkopírují do jeho nové horní poloviny [plati: znaménko je zachováno ; do všech bitů nové horní poloviny se zkopíruje znaménkový bit původního objektu]
* Instrukce procesoru 8086 ADC se používá při sčítání širších objektů
* Při násobení reálných čísel procesoru 8086 použijeme instrukci: [zadna z intrukci: IMUL, MUL, IDIV]
* Který z následujících skoků procesoru 8086 mění registr CS? vzdálenný (far)
* Který skok procesoru 8086 pracuje se 16bitovým operandem? blízký (near)
* Který skok procesoru 8086 pracuje s 8bitovým operandem? krátký (short)
* Pro podmíněný skok procesoru 8086 neplatí: vždy mění registr CS [plati: je vždy krátký, reaguje na obsah příznaků , cílová adresa se vytvoří 8bitovým přírůstkem ]
* Označení typu GE, LT, ... jsou vyjádřením porovnání velikosti čísel
* Do zásobníku procesoru 8086 se vkládají 16bitové objekty
* Pro instrukci POP SS,... platí po dobu trvání následující instrukce je zakázáno přerušení
* Instrukce HALT procesoru 8086 uvede procesor do stavu čekání
* Jaký je rozdíl mezi instrukcí RET a RETF procesoru 8086: RETF naplní i registr CS
* Proč instrukce STI procesoru 8086 nepovoluje přerušení ihned? aby byla nepřerušitelně ze zásobníku vybrána adresa přerušené instrukce
* Kde začíná segment reálného režimu (procesoru 8086)? na adrese dělitelné 16
* Adresa reálného režimu procesorů Intel x86 ve tvaru segment : offset 01B2:0015 představuje dvacetibitovou adresu (hexadecimálně) 01B35
* Registr IP procesoru Intel 8086 obsahuje offsetovou část adresy právě prováděné instrukce
* Všechny odkazy na zásobník v procesoru Intel 8086 jsou segmentovány přes registr SS
* Instrukce IRET reálného režimu procesoru Intel x86 zajišťuje návrat do přerušeného procesu a jeho pokračování
* Je-li v procesoru 8086 nastaven příznak OF=1 a následně je provedena instrukce INTO, nastane přerušení INT 4
* Procesor 8086 poskytuje pro adresování V/V bran 16bitovou adresu
* Která z následujících operací procesoru Intel x86 je nekorektní: MOV CS,AX [korektni : MOV AX,CS , MOV DI,10000, MOV BX,CX ]
* Instrukce AND Intel x86 provádí logický součin
* Instrukce IN AX,DX procesoru x86 zajišťuje přenos slova z V/V brány podle DX do registru AX
* Instrukce INC CL provede v procesoru x86 CL := CL + 1
* Kolika bitovou adresu při přístupu do paměti vytváří procesor Intel 8086? 20
* Adresa SS:SP ukazuje vždy na na vrchol zásobníku

**7 – 286**

* Jaká je maximální dosažitelná adresa v reálném režimu procesoru Intel 80286 a vyšších procesorů:10FFEFh
* Kolik řádků má tabulka popisovačů segmentů GDT nebo LDT procesoru Intel 80286 a vyšších procesorů: 8192
* Virtuální adresa procesoru Intel 80286 má celkem 30 bitů na adresaci virtuální paměti. Jak velká tato virtuální paměť může být? 1 GB
* S obsahem instrukčního segmentu procesoru Intel 80286 je povoleno následující: pouze provádět a možná i číst; mám-li potřebná práva
* Popisovač segmentu s LDT (tabulka popisovačů lokálního adresového prostoru) se v procesoru Intel 80286 smí nacházet v těchto tabulkách pouze v GDT
* Popisovač segmentu s GDT (tabulka popisovačů globálního adresového prostoru) se v procesoru Intel 80286 smí nacházet v těchto tabulkách: v žádné z nich [pouze v IDT , v GDT i v LDT. pouze v GDT]
* V reálném režimu procesoru Intel 80286 nelze provést instrukci: LLDT (plnění registru LDTR) [lze: LGDT (plnění registru GDTR), LIDT (plnění registru IDTR), LMSW (plnění registru MSW), HLT (zastavení procesoru)]
* Jaký je rozdíl mezi přerušením typu trap a fault v procesoru Intel 80286? Fault pracuje s adresou ukazující na instrukci, která přerušení způsobila. Trap poskytuje adresu ukazující na instrukci následující.
* Co znamená výjimka (přerušení) "Výpadek segmentu" v procesoru Intel 80286? procesor při vyčíslování virtuální adresy narazil na nulovou hodnotu bitu Present
* Procesor 80286 má 16bit. datovou a 24bit. adresovou sběrnici
* Procesor 80286 má chráněný a reálný režim
* Pro chráněný režim procesoru 80286 neplatí: tabulka vektorů přerušení má velikost 1 KB [plati: není možnost jej softwarově vypnout, poskytuje prostředky 4úrovňové ochrany, adresuje 16 MB reálné paměti]
* Pro registr MSW procesoru 80286 neplatí: slouží k zapnutí reálného režimu [plati: plní se instrukcí LMSW, slouží k zapnutí chráněného režimu, čte se instrukcí SMSW ]
* Signál RESET u procesoru 80286 zapíná reálný režim procesoru
* Bit P popisovače datového segmentu procesoru 80286 nastavený na 1 sděluje: obsah segmentu je uložen v reálné paměti
* Bit ED datového segmentu procesoru 80286 určuje zda datový segment obsahuje zásobník
* Bit C (Conforming) popisovače instrukčního segmentu procesoru 80286 může způsobit změnu úrovně oprávnění pro podprogramy volané v tomto segmentu
* Pro registr GDTR procesoru 80286 neplatí: označuje segment stavu procesoru [plati: má délku 5 bajtů, při spuštění chráněného režimu se do něj vkládá adresa tabulky GDT, naplňuje se instrukcí LGDT]
* Pro TSS (segment stavu procesoru 80286) neplatí: je to ukazatel, jestli je procesor 80286 v chráněném režimu [plati: na segment TSS ukazuje popisovač systémového segmentu, který může být umístěn pouze v GDT; slouží k uložení kontextu procesu, kterému bylo odebráno řízení;každý proces má vlastní TSS ]
* Interrupt Descriptor Table (IDT) procesoru 80286 nemá popisovač: brána pro V/V operace[ma: brána zpřístupňující TSS, brána pro maskující přerušení, brána pro nemaskující přerušení]
* Pro Interrupt Descriptor Table (IDT) procesoru 80286 neplatí: slouží k uložení kontextu procesu, kterému bylo odebráno řízení [plati: obsahuje až 256 popisovačů rutin obsluhujících přerušení, její adresu obsahuje IDTR,obsahuje nejvýše tolik popisovačů, kolik dovoluje limit segmentu ]
* Kolik řádků má Interrupt Descriptor Table (IDT) procesoru 80286? 256
* Který z následujících názvů nespecifikuje kategorii přerušení generovanou procesorem 80286? Flag [specifikuje: Fault, Trap, Abort]
* Která z možností nepatří mezi rezervovaná přerušení 80286? výpadek systému [patri: dělení nulou, přeplnění, chybný operační kód ]
* Zapnutí chráněného režimu procesoru 80286 neznamená: restart procesoru [znamena: změnu způsobu adresace, nastavení bitu PE=1 registru MSW, vypnutí reálného režimu ]
* Počet lokálních adresových prostorů procesoru 80286 typicky se rovná počtu procesů .

**7 – 386**

* Procesor Intel 80386 je 32bitový procesor **s 32bitovou adresovou a datovou sběrnicí**
* Selektor v chráněném režimu procesoru Intel 80386 je **16bitový**
* Procesor Intel 80386 pracuje s těmito možnými adresami 48bitovou logickou adresou, 32bitovou lineární adresou a 32bitovou fyzickou adresou
* Kolika bity plní programátor segmentové registry v procesoru Intel 80386 a vyšších typech? 16
* Registr EAX je použit od procesorů 386
* Stránkováním se v procesoru Intel 80386 transformuje: lineární adresa na fyzickou
* Největší možná velikost segmentu v procesoru Intel 80386 a vyšších typech je 4 GB
* Velikost stránky v procesoru Intel 80386 a vyšších typech je právě 4 KB
* Co znamená "Mapa přístupných V/V bran" v procesoru Intel 80386? Seznam V/V adres dostupných jednomu konkrétnímu (typicky V86) procesu.
* Jaká část adresy vstupující do stránkovací jednotky není stránkováním postihnuta (v procesoru Intel 80386)? dolních 12 bitů
* Kolik bitů je nezbytných pro uložení adresy stránkovací tabulky (zpravidla ve stránkovacím adresáři) a stránkovacího adresáře (zpravidla v CR3)? 20
* Pro procesor 80386 neplatí: adresová sběrnice má 24 bitů [plati: datová sběrnice má 32 bitů, lze použít stránkování , data se do/z paměti přenášejí po 4 bajtech]
* Pro adresaci v chráněném režimu procesoru 80386 neplatí: offset je 16bitový [plati: selektor je stejný jako v 80286, báze segmentu je 32bitová , limit segmentu může být až 4 GB ]
* Instrukční prefix je realizován jedním bajtem uloženým před prvním bajtem operačního kódu
* Co znamená, že stránkovací jednotka procesoru 80386 není zapnuta: fyzická adresa je totožná s lineární adresou
* Pro stránkování procesoru 80386 platí je-li zapnuto, tak se lineární adresa transformuje na fyzickou
* Pro stránkový adresář procesoru 80386 neplatí: žádná z uvedených možností [plati: má velikost právě jedné stránky, ukazuje na max. 1024 stránkových tabulek, je k dispozici pouze se zapnutým stránkováním]
* Pro bit D (Dirty) při stránkování procesoru 80386 neplatí: pokud je nastaven na jedničku, tak je rámec vybrán za oběť [plati: procesor ho nastaví na jedničku při zápisu do rámce, procesor jej nenuluje, to má na starost software;rozlišuje, jestli je rámec špinavý nebo čistý ]
* Pro TLB neplatí: je zapnuto pouze v chráněném režimu procesoru 80286 [plati: funguje na principu asociativní paměti,je to vyrovnávací paměť ,při vyprazdňování se vynulují bity V (validita) ]
* Procesor 80386 má 32 bitovou adresovou sběrnici A2 až A31, což znamená, že do paměti se jde alespoň pro 4 bajty

**7 – 486**

* Jaký má význam interní vyrovnávací paměť v procesoru **Intel 80486**? Pamatuje si několik posledních obsahů adres čtených z fyzické paměti vč. okolí.
* Kolik bitů by potřeboval **algoritmus LRU** v interní vyrovnávací paměti procesoru Intel 80486 k tomu, aby úplně fungoval pro výběr ze čtyř položek na řádku (předpokládejme, že by byl realizován neúplnou maticí)? 6
* **Procesor 80486** má: datovou sběrnici 32bitů , adresovou sběrnici 32bitů , integrovaný matematický koprocesor
* **Procesor 80486** se od procesoru 80386 NEliší v: velikosti vnějších sběrnic [lisi se v: interní vyrovnávací paměti, nové technologii, která se blíží k RISCovým procesorům , jednotce operací v pohyblivé řádové čárce ]
* V procesoru Pentium se dynamicky předvídá *výsledek vyhodnocení podmínky podmíněné skokové instrukce*
* Autorem architektury x86-64 je **firma AMD**
* **Reálný režim** se používá u procesorů: 286, , 386 , 486, x86-64
* V 64bitovém režimu architektury x86-64 se potlačuje segmentace
* Zástupcem architektury x86-64 je **procesor Opteron**
* Jakou velikost má **selektor** v 64bitovém režimu architektury x86-64? 16 bitů

**RISC**

Který rys je vlastní technologii procesorů RISC? poskytnutí velkého počtu registrů v procesoru

Který procesor má více různých instrukcí pracujících s operandy v paměti? CISC

**Koprocesor**

Základní šířka dat interně zpracovávaných koprocesorem pro výpočty v pohyblivé řádové čárce je 80 bitů.

Nejmenší záporné číslo (největší v absolutní hodnotě; číslo na levé hranici rozsahu zobrazení) v IEEE 754 má znaménko mantisy 1, největší kladné zobrazitelné číslo v exponentu.

**Připojování**

Signály STROBE a BUSY používá rozhraní Centronics

Paralelní rozhraní je Centronics.

Rozhraní Centronics: Signál !STROBE je v aktivní úrovni pevně stanovenou dobu

Rozhraní Centronics: Signál !STROBE je v aktivní úrovni, když přenáší hodnotu logická "0"

Rozhraní RS-232C: Přenos dat tímto rozhraním je: synchronní i asynchronní

Rozhraní RS-232C: Jaké zapojení nulmodemu je NEsmyslné? SG--SG, TxD--TxD, RxD--RxD

USB při komunikaci používá protokol Master-Slave