**Informatika, informatizácia a informačná spoločnosť**

* Informatika je veda o získavaní, zbere, prenose, triedení, ukladaní, uchovávaní

(pamätaní), aktualizovaní, spracovaní, vyhodnocovaní a využívaní informácií na úrovni signálov, údajov, symbolov, správ, poznatkov a znalostí

**Výpočtová technika**

* skúma zákonitosti a princípy tvorby informačných procesov a spôsoby ich realizácie
* zaoberá sa vlastnosťami a zákonitosťami týchto procesov a ich algoritmickou realizáciou
* zahrňuje tvorbu, vývoj a využitie programových a technických prostriedkov

výpočtovej techniky, ako nástroja na automatizované a automatické spracovanie informácií

**Číslicový počítač (ČP)**

* je zložitý univerzálny číslicový systém určený na samočinné vykonávanie postupnosti operácií nad údajmi zobrazenými číslicovým kódom, na základe vopred pripraveného a v pamäti uloženého programu
* je súhrn prvkov komponovaných do jedného celku za účelom dosiahnutia definovaného cieľa
* Kompozícia systému - vlastnosť sa nazýva zložitosť systému
* Opísanie systému sa uskutočňuje na základe určenia jeho funkcie a štruktúry
* Funkcia systému predstavuje pravidlo na dosiahnutie požadovaného cieľa
* Štruktúra systému vyjadruje kompozíciu jeho prvkov a ich vzájomných väzieb
* Organizácia systému je spôsob vytvorenia systému na báze kompozície prvkov z ktorých pozostáva za účelom dosiahnutia zadaných funkčných vlastností
* Počítačový systém (PS) - systém vytvorený na báze jedného alebo viacerých ČP, je charakterizovaný svojou:

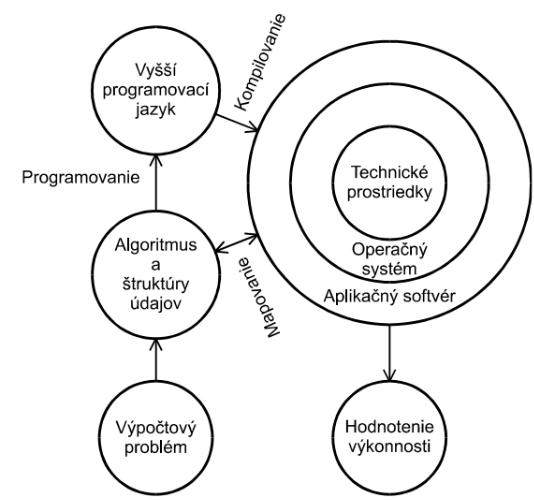
o funkčnou organizáciou o štruktúrnou organizáciou

* Funkčná organizácia PS je definovaná formou zobrazovania informácií, štruktúrou inštrukčných súborov, charakterom vnútorného a vonkajšieho riadenia operácií, spôsobom riadenia procesu spracovania informácií, prístupovým režimom používateľov k PS
* Štruktúrna organizácia PS je definovaná logickým a systémovým usporiadaním jeho komponentov, spôsobom ich vzájomnej komunikácie, konštrukčnou a technologickou realizáciou jeho komponentov
* Architektúra PS zahrňuje:
  1. požiadavky programových prostriedkov - inštrukčno orientovaná architektúra o vnútornú organizáciu jeho technických prostriedkov - implementačno

orientovaná architektúra

* Z hľadiska požiadaviek programovania inštrukčno orientovaná architektúra PS predstavuje abstrakciu prostredníctvom súboru inštrukcií - kódy operácií, adresovacie módy, špecifikáciu registrov, virtuálnu pamäť a pod
* Z hľadiska implementácie tech. prostriedkov implementačno orientovaná architektúra PS predstavuje abstrakciu na úrovni štruktúrnej organizácie - centrálnych procesorových jednotiek, komponentov pamäťového podsystému, zbernicových systémov, mikrokódu, prúdových funkčných jednotiek

**Model PS**



**Piliere číslicových systémov**

* Abstrakcia, Disciplína, Hierarchia, Modularita, Jednotnosť

**Analytický stroj**

* Charles Babbage pracoval na ňom od 1833 – do smrti (1871)
* Považuje sa za prvý číslicový počítač

**George Boole, 1815-1864**

* Zakladateľ matematickej logiky, vybudoval základy, na ktorých stojí dnešná

informatika

* Číslicové obvody používajú príslušnú hladinu napätia na reprezentáciu log. hodnôt 1 0
* Bit: Binary digit – základná jednotka informácie

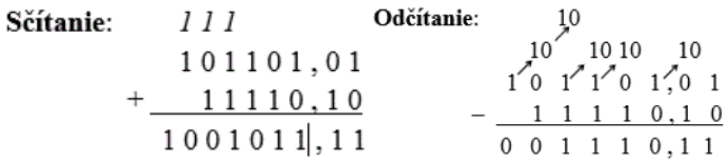
**Číselné sústavy (ČS)**

* Pozičné, Nepozičné, Symetrické, Nesymetrické

**Číselné sústavy**

* Desiatková, Binárna, Osmičková, Hexadecimálna

**Aritmetika v binárnej ČS**



**Pretečenie**

* ak hodnotu výsledku operácie nie je možné reprezentovať na N-bitoch

**Priamy kód (PK)**

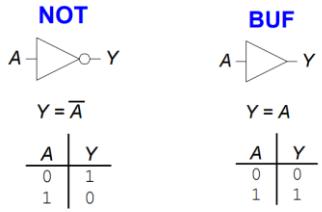
* 1 znamienkový bit, N-1 bit pre magnitúdu
* Najľavejší bit je znamienkový bit - Kladné čísla: z.bit = 0, Záporné čísla: z.bit = 1
* Dve „verzie“ nuly (± 0): 1000 (záporná), 0000 (kladná)
* Existuje len jedna reprezentácia nuly (1000)
* Najvýznamnejší bit reprezentuje z.bit (0 = kladné, 1 = záporné)

**Inverzný kód (IK)**

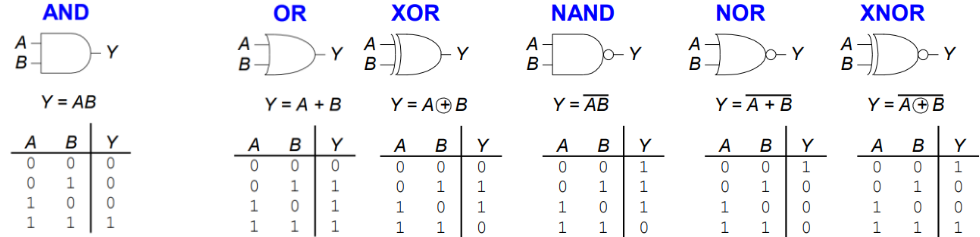
* Kopírovanie hodnoty z.bitu, Pridávanie núl sprava

**Hradlá**

* Realizujú logické funkcie o Jedno-vstupové:
  + NOT gate, buffer



1. Dvoj-vstupové:
   * AND, OR, XOR, NAND, NOR, XNOR

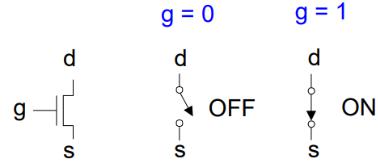


* 1. Viac-vstupové

**Logické úrovne**

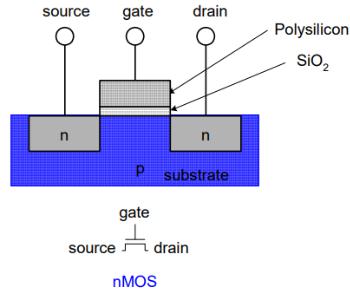
* Číslicové signály majú konečný počet diskrétnych hodnôt, väčšinou dve
* Rozdielne pásmo pre vstup a výstup
* Základným stavebným prvkom hradiel
* Má 3-elektródy:

1. bipolárne: C, B, E; unipolárne: D, G, S, d a s sú spojené (ON) ak g je 1

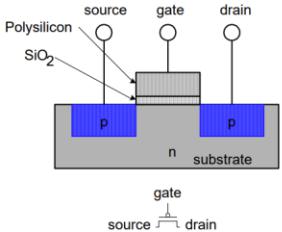


**MOS Tranzistor (Metal Oxide Silicon)**

* Najrozšírenejší druh tranzistorov riadených elektrickým poľom
* Vodivosť kanálu medzi s a d je riadená nepätím privedeným na g



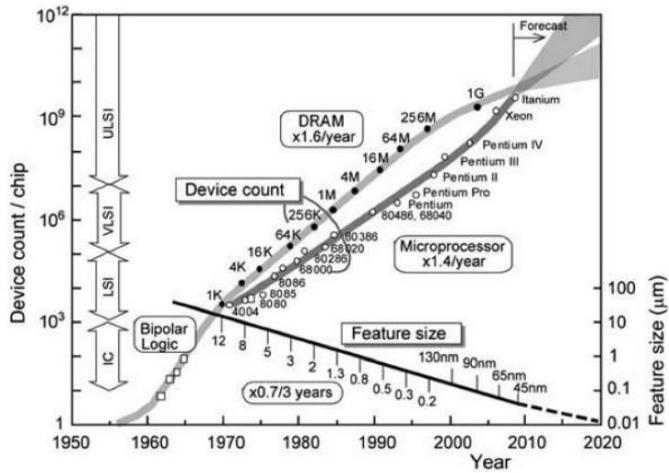
* pMOS tranzistor je komplementom k nMOS - je ON ak Gate = 0, OFF ak Gate = 1



* nMOS: dobre prenáša „nuly“, pMOS: dobre prenáša „jednotky“

**Gordon Moore, 1929-**

* Spoluzakladateľ firmy Intel
* Moore-ov zákon: Počet tranzistorov sa zdvojnásobí každý „rok“ (empirické pravidlo, 1965)



**Logický systém**

* je charakterizovaný
  1. abstraktným opisom funkcie systému
  2. štruktúra systému → logický obvod
* Logický obvod (LO) je definovaný pomocou:
  1. svojich vstupov (X) a výstupov (Y), funkčnej špecifikácie a časových charakteristík
* Štruktúru logického obvodu tvoria logické členy a vzájomné väzby medzi nimi
* Uzly - spojenie dvoch alebo viacerých kanálov
* Typy logických obvodov

1. Kombinačný logický obvod (KLO)
   * Výstup závisí len od okamžitej kombinácii vstupných hodnôt o Sekvenčný logický obvod (SLO)
   * výstup závisí od kombinácie vstupov a od vnútorných stavov obvodu z

predchádzajúceho taktu

* Každý element obvodu je kombinačný obvod
* Každý vstup je spojený s výstupom len z jedného elementu obvodu
* Obvod neobsahuje slučky

**Booleovské rovnice**

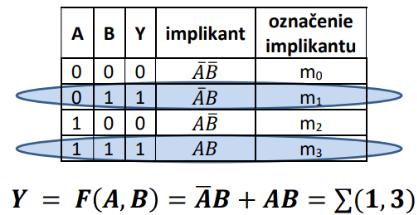
* Definujú vzťah medzi vstupnými a výstupnými b. premennými

**Definície**

* Komplement: negovaná b. premenná 
* Literál: b. premenná v priamej alebo v negovanej forme
* Implikant: súčin všetkých vstupných b. premenných v priamej alebo negovanej forme
* Implicent: súčet všetkých vstupných b. premenných v priamej alebo negovanej forme

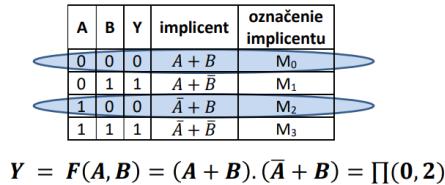
**Disjunktívna normálna forma**

* je súčet implikantov, Implikant je súčin literálov
* Každý implikant definuje pravdivostnú hodnotu TRUE



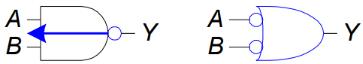
**Konjunktívna normálna forma**

* je súčin implicentov, Implicent je súčet literálov
* Každý implicent definuje pravdivostnú hodnotu FALSE



**Algebraické vyjadrenia a LO**

* Booleova algebra: Na báze: NOT, AND, OR
* Shefferova algebra: Na báze: NAND
* Peirceova algebra: Na báze: NOR
* Prevodov NAND na OR s neg. vstupmi:
* Prevod OR s neg. vstupmi na NAND:



**Pravidlá kreslenia schém LO**

* Vstupy sú naľavo, Výstupy sú napravo, Hradlá sú orientované zľava doprava, Každý kanál sa znázorňuje samostatnou čiarou
* Kríženie vodičov bez bodky: nie je vodivé spojenie medzi kanálmi
* Signál s hodnotou „X“:
  1. X reprezentuje hodnotu na ktorej nezáleží, X reprezentuje konflikt
* Vysokoimpedančná hodnota „Z“:
  1. Reprezentuje vysokoimpedančný stav zdroja signálu
  2. výstupný signál môže mať hodnotu 0 ako aj 1
* Trojstavové zbernice: Obvody sú v stave vysokej impedancie, odpojené od zbernice

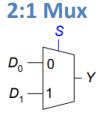
**Karnaughové mapy (K-mapy)**

* Princíp minimalizácie pomocou Karnaughových máp spočíva v aplikovaní pravidla spojovania na všetky dvojice susedných elementárnych súčinov
* Komplement: b. premenná v negovanej forme
* Literál: b. premenná v priamej forme alebo jej komplement
* Implikant: súčin literálov
* Prostý implikant: implikant definovaný s pravidelnou konfiguráciou s maximálnym možným stupňom k

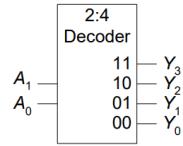
**Kombinačné logické obvody**

* Multiplexor, Dekodér, Demultiplexor, Komparátor, Prevodník kódu, Polovičná a úplná

sčítačka



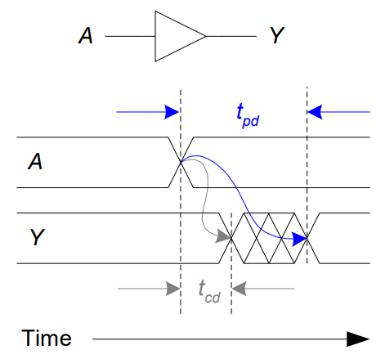
* Má M adresných (riadiacich), N dátových vstupov a jeden výstup, = log2N
* Implementácia: Pomocou hradiel, Trojstavové buffre



* Má N vstupov, 2 výstupov
* One-hot kódovanie: v danom čase je len jeden výstup HIGH

**Časové charakteristiky**

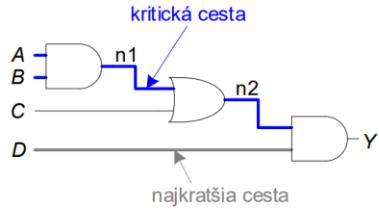
* Dynamické charakteristiky číslicových signálov
* Doba oneskorenia priechodu signálu
* Propagačné oneskorenie: = max oneskorenie signálu
* Kontaminačné oneskorenie: = min oneskorenie signálu



**Propagačné a kontaminačné oneskorenie**

* Oneskorenie je spôsobené: Vlastnosťami vodičov, Rýchlosť pohybu elektrónov

**Kritická a najkratšia cesta**



**Falošné impulzy, hazardy**

* Ak jedna zmena vstupného signálu vyvolá viacnásobnú zmenu výstupného signálu
* či vôbec vznikne rozhodujú časové charakteristiky vstupných signálov
* výstup závisí od kombinácie vstupov a od vnútorných stavov obvodu z predchádzajúceho taktu, má pamäť
* Definície:

1. Stav: konfigurácia nezávislých a závislých veličín
2. Preklápacie obvody: uchováva jednobitovú informáciu
   1. Sekvenčný obvod: pozostáva z KLO a z preklápacích obvodov

* výstupný vektor je závislý od sekvencie vektorov na vstupe v predchádzajúcich taktoch
* sekvenčný logický systém obsahuje podsystémy, ktoré sú schopné pamätať si informáciu z predchádzajúcich taktov

**Obvody určené na pamätanie**

* Na uchovávanie vnútorného stavu
* Bistabilné obvody
* Preklápacie obvody:

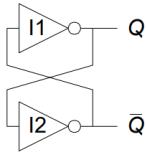
1. Hladinou riadené a hranou riadené (angl. flip-flop) o Synchrónne a asynchrónne

o D, T, SR, JK

**Bistabilný obvod**

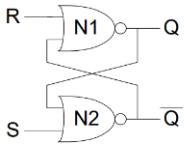
|  |  |  |
| --- | --- | --- |
|  | ̅ |  |
| Základ obvodov určených na pamätanie info., Má dva stabilné stavy: , |  |

* Uchováva 1 bit informácie pomocou stavovej premennej Q, Nemá žiadny vstup



**SR (Set/Reset) preklápací obvod**

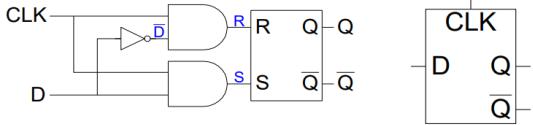
* Uchováva 1 bit informácie (Q) o stave systému
* Na kontrolu sa používajú vstupy S a R: Set: Nastaví výstup na 1, Reset: Nastaví výstup na 0, Je potrebné zabrániť kombinácií = =



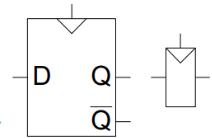
1. **PO**
   * Dva vstupy: CLK, D
   1. CLK - definuje moment vzorkovania vstupu
   2. D - na akú hodnotu má byť nastavený výstup

* Funkcia
  1. ak CLK = 1 , sa dostane na Q

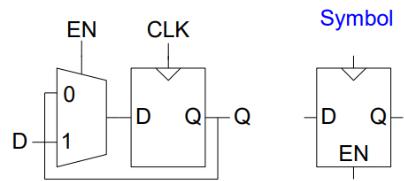
1. ak CLK = 0, si pamätá predchádzajúcu hodnotu



1. **Flip-Flop (D FF)**
   * Vstupy: CLK, D
   * Funkcia:

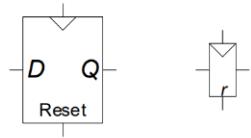


* 1. Vzorkuje pri prechode z jednej úrovne na druhú o sa mení len pri zmene CLK
* Hranou riadený PO
* Vstupy: CLK, D, EN
* Vstup able riadi moment, kedy sa dostane dátový vstup na výstup



**D FF s možnosťou a. resetu**

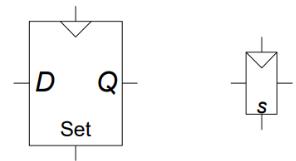
* Vstupy: CLK, D, Reset
* Dva typy:



1. Synchrónne: len pri zmene CLK
   1. Asynchrónne: hneď čo sa nastaví= 1

**D FF s možnosťou nastavovania**

* Vstupy: CLK, D, Set
* Funkcia
  1. =:=1



* 1. = : pracuje v „normálnom“ režime

**Sekvenčný LO**

* Bez vstupov 3 „vnútorné“ signály, Astabilný obvod → osciluje, Má to slučku

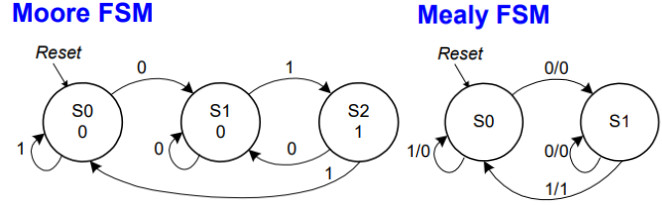
**Synchrónny sekvenčný LO**

* vkladáme registre, uchovávajú info. o vnútornom stave systému, Stav sa mení pri zmene CLK: systém je synchronizovaný s hodinovým signálom
* Obsahuje: Stavový register, Kombinačná logika
* Ďalší stav je definovaný aktuálny vnútorným stavom systému a momentálnym stavom na vstupe
* Dva typy:

1. Automat Moore: výstup je závislý len od vnútorného stavu
   1. Automat Mealy: výstup je závislý tak od vnútorného stavu ako aj od vstupu

**Graf prechodov**

* KSA Moore: výstup je značený v uzle, Stavy: Uzly, Prechody: Hrany



**Kódovanie stavov v KSA**

* Binárne kódovanie
* One-hot kódovanie

**Faktorizácia KSA**

* Rozdelenie „veľkého“ KSA na menšie interagujúce KSA

**Časové charakteristiky SLO**

* Hranou riadený PO

1. je nutné dodržať niektoré dynamické parametre vstupných signálov:
   * doba predstihu
   * doba presahu
   * minimálna strmosť hrany hodinového impulzu
   * minimálna doba jeho trvania
   1. Nedodržanie môže vyvolať metastabilný stav

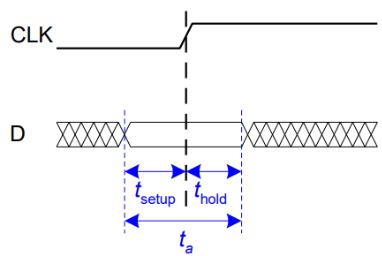
* Základné vlastnosti hranou riadeného PO:

1. Rozhodujúci je stav na vstupoch v tesnom okolí nábežnej hrany
   1. Metastabilný stav má za následok nespoľahlivú funkciu preklápacieho obvodu

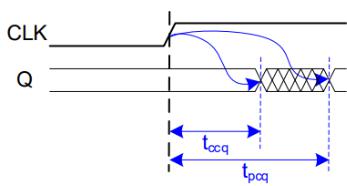
**Časovanie signálov v SLO**

* Doba nastavenia: = časový úsek pred zmenou CLK, počas ktorého vstup by mal byť stabilný
* Doba presahu: ℎ  = časový úsek po zmene CLK, počas ktorého vstup by mal byť stabilný
* Vzorkovací interval: = v tomto časovom intervale sa nesmie zmeniť hodnota

vstupu ( =+ ℎ )



* Propagačné oneskorenie: = časový úsek po uplynutí ktorého je garantované, že Q má stabilnú hodnotu
* Kontaminačné oneskorenie: = časový úsek po uplynutí ktorého sa začína prejavovať zmena na Q
* Vstupy do SLO musia byť stabilné počas vzorkovania:



* 1. minimálne počas doby pred zmenou CLK o minimálne počas doby ℎ  po zmene CLK
* Signál, kt. prechádza cez KLO medzi registrami je charakterizovaný kontaminačným a propagačným oneskorením

**Doba nastavenia**

* Závisí od propagačného oneskorenia signálu

**Doba presahu**

* Závisí od kontaminačného oneskorenia signálu

**Odchýlka v časovaní**

* Hodinový impulz prichádzajúci na vstupy rôznych registrov nie v rovnakom momente

**Porušenie dynamickej disciplíny**

* Asynchrónne vstupy môžu mať za následok nepredvídateľné správanie sa obvodu

**Metastabilný stav**

* Bistabilné zariadenia: dva stabilné stavy a jeden metastabilný stav
* Flip-flop: dva stabilné stavy (1 a 0) a jeden metastabilný stav
* Token: dátový vstup
* Doba odozvy: čas prechodu tokenu log. systémom
* Priepustnosť: počet tokenov spracovaných za jednotku času
* Paralelizmus zvyšuje priepustnosť

* Dva typy paralelizmu:

1. Priestorový paralelizmus - Zvýšenie počtu funkčných jednotiek
2. Temporálny paralelizmus - Úloha sa rozdelí na dielčie kroky, ktoré sa vykonávajú súbežne

**Návrh LS**

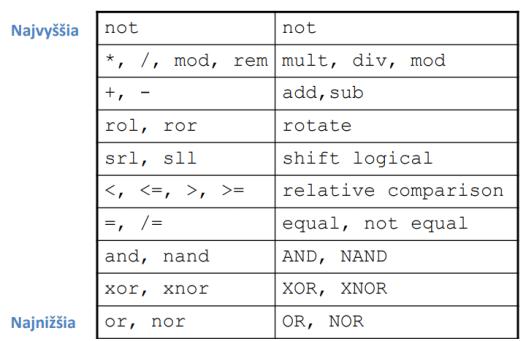
* Jazyk pre opis hardvéru (Hardware description language; HDL)
* HDL nie je programovací jazyk, je to jazyk, ktorý slúži na opis HW
* Dva hlavné HDL:

1. VHDL (VHDL 2008)
   1. Verilog (SystemVerilog)

* Syntéza - Prevod HDL kódu do netlist reprezentácie
* Simulácia - Verifikácia činnosti obvodu
* Spôsoby opisu LS:

1. Behaviorálny: na čo slúži / z pohľadu chovania
   1. Štrukturálny: modulárny opis / z akých komponentov pozostáva o Toku dát (dataflow): tok dát medzi komponentmi

* VHDL Je „case insensitive“



**Sekvenčná logika**

* VHDL používa návrhové vzory na modelovanie správania sa preklápacích obvodov a stavových automatov
* Príkaz process - Príkazy vo vnútri bloku process sú vykonané len vtedy, ak je zaznamenaná zmena aspoň jedného signálu
* Sekvenčný príkaz je súčasťou príkazu/bloku process: if / else, case, case?, loop
* Na modelovanie kombinačných logických obvodov (KLO) je vhodnejšie použiť konkurenčné (paralelné) príkazy
* Príkaz case vedie na syntézu KLO len vtedy ak sme pokryli všetky možné vstupné kombinácie
* Príkaz case? na rozdiel od „obyčajného“case dokáže pracovať aj s
* „don’t care“ hodnotami
* SLO: Použi príkaz process(clk)a neblokujúce priradenie (<=) pre modelovanie SLO
* KLO: Použi konkurenčné (paralelné) príkazy na tvorbu KLO

**Konečno-stavové automaty(FSMs)**

* Tri bloky: Vstupný logický obvod, Stavový register, Výstupný logický obvod

**Parametrizované moduly**

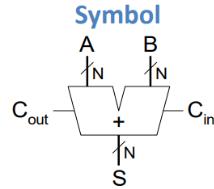
* Kľúčové slovo generic - generická konštanta vo vnútri bloku entity
* Kľúčové slovo generate - Genreujúci príkaz

**Simulácia / testbench súbory**

* Na simuláciu modulov použijeme testbench súbor
* Testbench nie je určený na syntézu
* Typy:
  1. Jednoduchá funkčná
  2. S autokontrolou
  3. S autokontrolou na báze testovacieho súboru s testovacími vektormi
* Hodinový signál:
  1. Inicializuj vstupy (počas nábehovej hrany)
  2. Porovnaj výstup s očakávanými výstupmi (počas dobehovej hrany)
* Hodinový signál je nevyhnutný v prípade synchrónnych sekvenčných obvodov

**Komponenty číslicových systémov:**

* hradlá, multiplexory, dekodéry, obvody na realizáciu aritmetických a logických operácií, počítadlá, registre, pamäte, ...
* Rozdelenie systémov do modulov
* Moduly majú dobre zadefinované funkcie a rozhrania
* Možnosť znovupoužitia



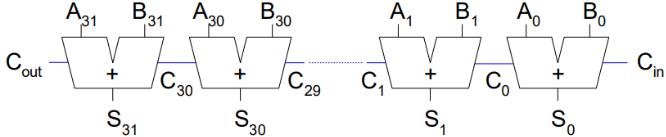
* Sčítanie dvoch N-bitových čísel

1. Sčítačka so sériovým prenosom (pomalá)
   1. Sčítačka so zrýchleným prenosom (rýchlejšia) o Paralelné prefixové sčítačky (najrýchlejšia)

* Sčítačky so ZP a prefixové sčítačky sú rýchle ale potrebujú viac hradiel

**Sériová sčítačka**

* Kaskáda 1b sčítačiek, Prenos sa šíri od najnižšieho až po najvyšší rád
* Výhoda: jednoduchý návrh, Nevýhoda: pomalá

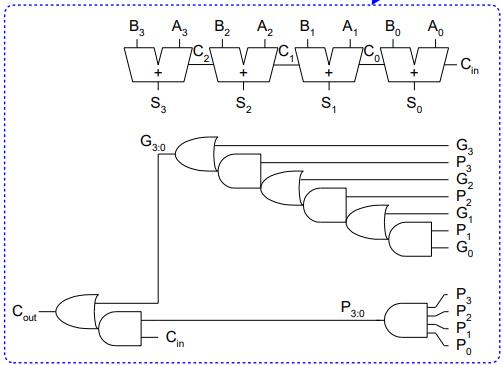


**Sčítačka so zrýchleným prenosom**

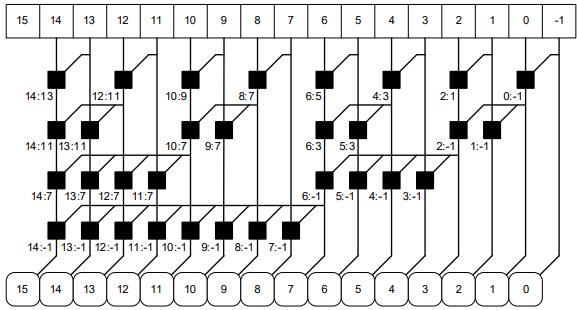
* Výpočet prenosu (Cout) z k-bit veľkého bloku pomocou vlastného a tranzitívneho prenosu
* Prenos z i-teho rádu nastane vtedy ak je to dané vlastným prenosom (Gi) alebo v

súčinnosti s indikovaným príznakom tranzitívneho prenosu (Pi)

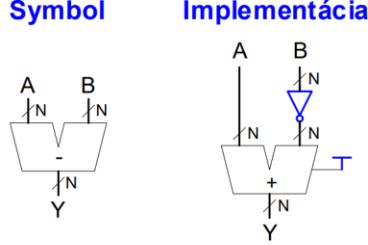
**32b:4b CLA**



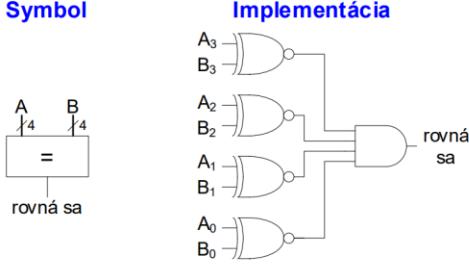
**Schéma prefixovej sčítačky**



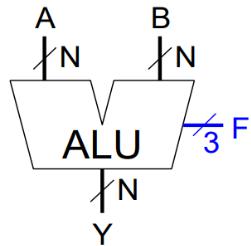
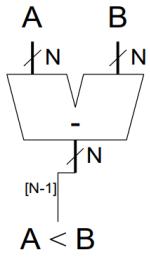
**Odčítačka**



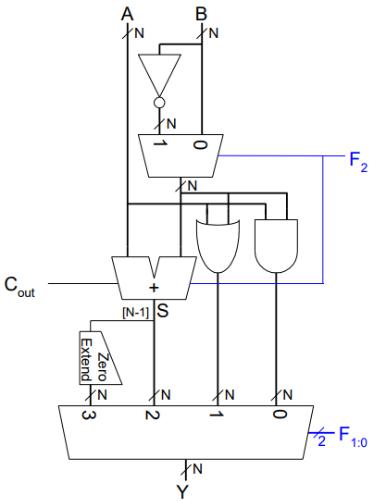
**Komparátor : rovnosť**



**Komparátor : menší** **Aritmeticko-logická jednotka (ALJ)**

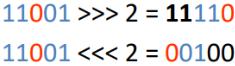
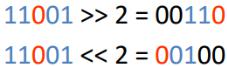


**Jednoduchá ALJ**

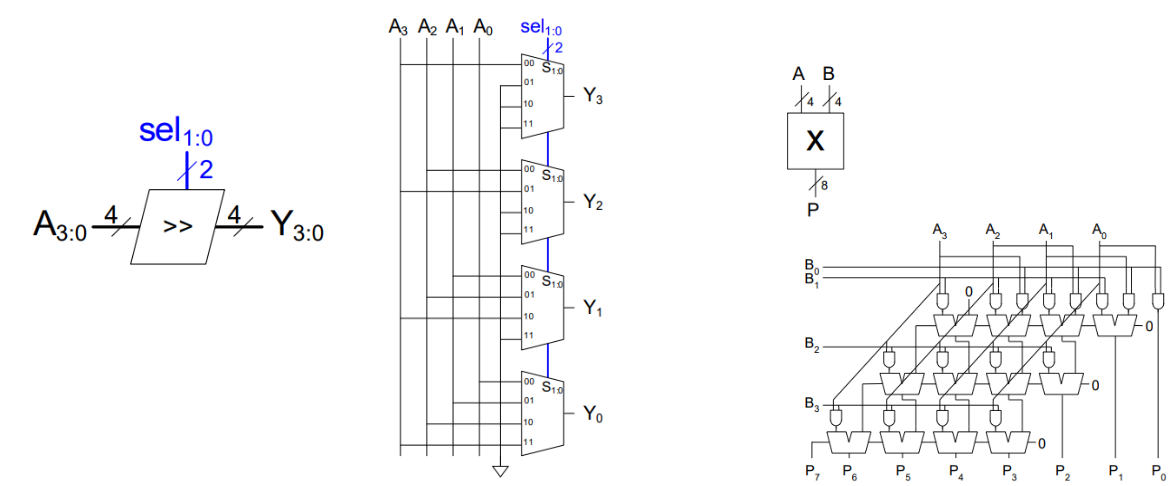


**Posuvné (funkčné) jednotky (PFJ)**

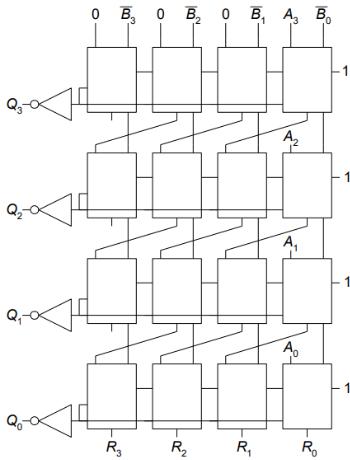
* Logický posun: posun doprava alebo doľava s pridaním 0
* Aritmetický posun: posun doľava je ako logický posun doľava; pri posune doprava sa kopíruje hodnota msb.
* Rotácia: bit ktorý sa pri rotácií dostane mimo rámec zobrazenia sa objaví v pozícií msb (doprava) alebo lsb (doľava)



**Návrh posuvnej jednotky (KLO)** **Násobička 4 x 4**



**Delička 4 x 4**



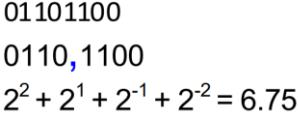
**Reprezentácia čísel**

* ČP používa binárnu č. sústavu o Prirodzené čísla
  + Bezznamienkové binárne čísla o Celé čísla
  + Priamy kód
  + Doplnkový kód
  + Kód s posunutím

* Reálne čísla

1. Pevná rádová čiarka (PRČ) – Q notácia o Pohyblivá rádová čiarka (PHRČ)

* Vyjadri 6,75 v Q4.4 (= 4 bity pre celú časť vrátane znamienka a 4 pre desatinnú časť):
* Pozícia desatinnej čiarky je striktne určená
* Je potrebné sa dohodnúť na počte bitov vyhradených pre celú a pre desatinnú časť



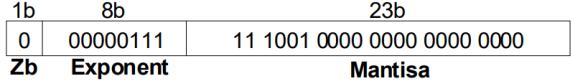
čísla

**Pohyblivá rádová čiarka**

* Čísla v pohyblivej rádovej čiarke sa vyjadrujú v tvare 
* M = mantisa, b = základ, E = exponent
* napr. : 273 = 2,73 × 10^2
* Príklad: Vyjadrite reálne číslo 228,010 v 32-bitovom formáte



1. Binárna reprezentácia: 22810 = 111001002
2. Vedecký binárny zápis: 111001002 = 1,110012 × 27 o 32-bitovú reprezentácia čísla v pohyblivej rádovej čiarke
   * Znamienkový bit = 0 (kladné)
   * 8-bitový exponent = 7
   * Mantisa na 23 bitoch (doplnené 0)



**IEEE-754 formáty**

* Formát jednoduchej presnosti: o 32 bitov

o 1b znamienko, 8b charakteristika, 23b mantisa o bias = 127

* Formát dvojitej presnosti:

1. 64 bitov
2. 1b znamienko, 11b charakteristika, 52b mantisa o bias = 1023

* Pretečenie: číslo je príliš veľké pre zobrazenie v danom formáte
* Podtečenie: číslo je príliš malé pre zobrazenie v danom formáte
* Techniky zaokrúhľovania:

1. Nadol (down) o Nahor (up)

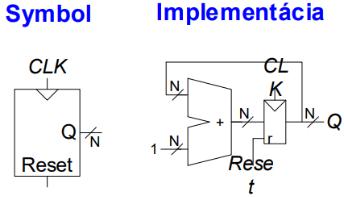
o Smerom k nule (toward zero)

o Na najbližšiu hodnotu (to nearest)

**Počítadlo**

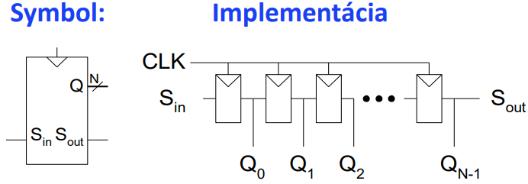
* Hodnota sa inkrementuje/dekrementuje pri každom takte
* Prípady použitia:

1. Riadenia synchronizačných impulzov v digitálnych zobrazovačoch o Programové počítadlo



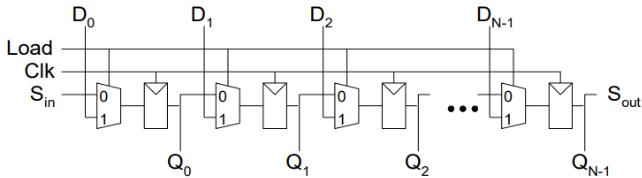
**Posuvný register**

* Pri každom takte vstupuje do a vystupuje z registra 1b
* Prevod sériového kódu na kód paralelný



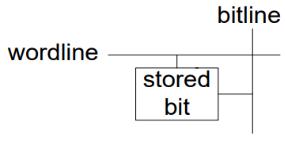
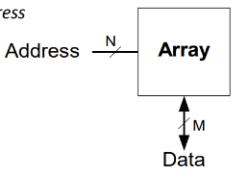
**Posuvný register s paralelným nastavením**

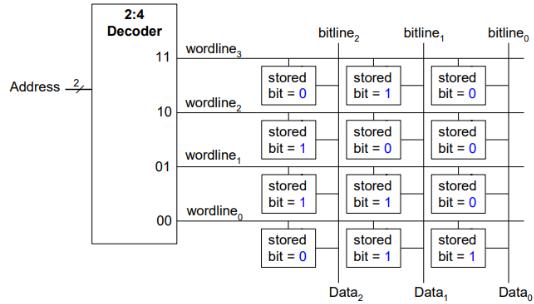
* Ak Load = 1, pacuje ako (bežný) N-bitový register
* Ak Load = 0, pracuje ako posuvný register



**Pamäte**

* Slúži na pamätanie údajov
* 3 základné typy:
  1. Dynamic random access memory (DRAM)
  2. Static random access memory (SRAM)
  3. Read only memory (ROM)
* M-bit široký dátový výstup/vstup Data pre čítanie/zápis do pamäte na adresu definovanú N-bit širokým vstupom Address
* 2D pole pamäťových buniek, Každý element dokáž uchovať 1b údaj
* realizácia pamäti
* Adresná linka (wordline)
  1. unikátna adresa slova, jeden riadok v pamäti pre čítanie/zápis





**Typy pamätí**

* Random access memory (RAM) - volatilná pamäť, energeticky závislá
* Read only memory (ROM) - nevolatilná pamäť, energeticky nezávislá

**RAM: s náhodným prístupom**

* Stráca obsah po odpojení elektrickej energie
* Čítanie a zápis sú relatívne rýchle operácie
* Hlavná pamäť počítača je typu RAM (DRAM)
* NVRAM = Nestráca obsah po odpojení elektrickej energie
* Rýchle čítanie, zápis je pomalý, BIOS, PROM, EPROM, EAROM, EEPROM, Flash pamäť

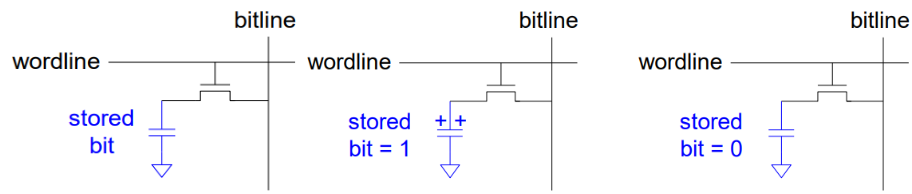
**Pamäte RAM**

* DRAM (Dynamic random access memory)
* SRAM (Static random access memory)
* Rozdiel je v spôsobe ukladania dát:

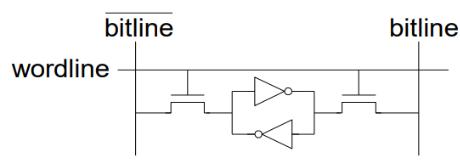
1. DRAM používa kondenzátory
2. SRAM používa dvojicu invertorov v slučke

**DRAM**

* Konštrukčným základom bunky je kondenzátor

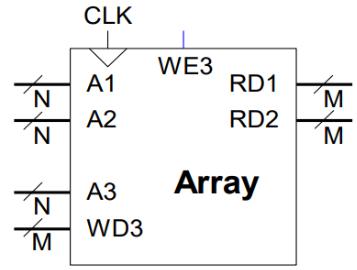


**SRAM**



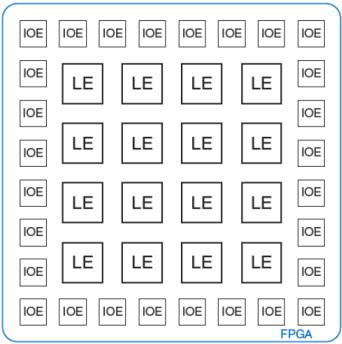
**Viacportové pamäte**

* Port: zabezpečuje vstup/výstup
* Súbor registrov: malá viacportová pamäť



**Logické polia**

* PLAs (Programmable logic arrays)
  1. Pole AND členov napojené na pole OR členov, Kombinačná logika
* FPGAs (Field programmable gate arrays)
  1. Pole logických elementov, Kombinačná/sekvenčná logika, Reprogramovateľné
  2. Pozostáva:
     + LE (Logic elements): realizácia logiky
     + IOE (Input/output elements): rozhranie pre komunikáciu s vonkajším svetom
     + Prepojovacia sieť: spája LE a IOE



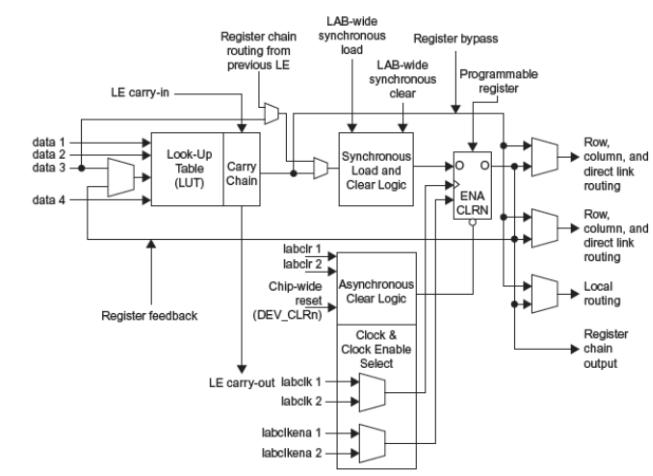
* LE: Logický element pozostáva

o LUT (lookup tables)

o Preklápacie obvody (flip-flops)

1. Multiplexory

**Altera Cyclone IV LE**



**ISA: inštrukčnoorientovaná architektúra**

* abstrakcia prostredníctvom súboru inštrukcií

**IA: implementačnoorientovaná architektúra**

* abstrakcia na úrovni štruktúrnej organizácie

**Asembler**

* Asembler: jazyk symbolických inštrukcií
* Strojový jazyk/kód: inštrukcie sú v podobe binárneho reťazca
* MIPS architektúra

1. Vývoj 80-tych rokoch na Stanfordskej univerzite pod vedením Johna Hennessyho
2. Použitá v zariadeniach od Silicon Graphics, Nintendo, a Cisco

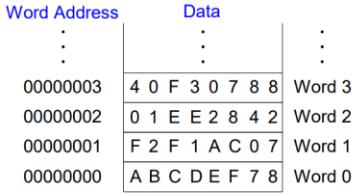
**Princípy návrhu počítačových architektúr**

* Hennessy a Patterson navrhujú postupovať pri návrhu architektúry tak, aby boli dodržané nasledujúce odporúčania:

o Jednoduchosť podporuje jednotnosť

1. Jadro musí byť rýchle
2. Zníženie počtu komponentov zvyšuje rýchlosť o Dobrý návrh vychádza z kompromisov

* Každé 32b slovo má svoju unikátnu adresu

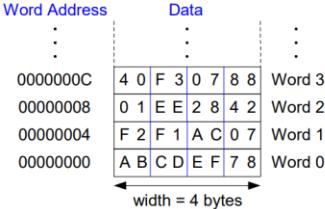


**Čítanie z pamäte – load**

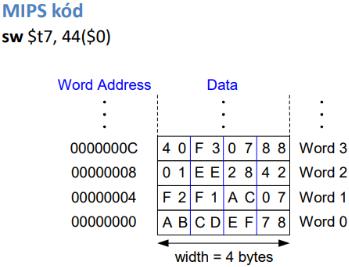
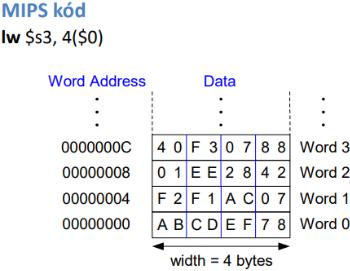
**Zápis do pamäte – store**

**Organizácia zobrazovania objektov v pamäti**

* Zarovnávanie viacbajtových objektov
  1. Usporiadanie bajtov objektov v informačných slovách
* Ukladací endian - Malý endian, Veľký endian
* Každý bajt v slove má svoju unikátnu adresu
* MIPS používa bajtovú organizáciu pamäte pri ukladaní 32b slov

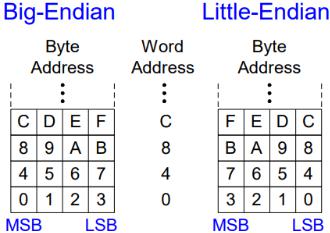


**Čítanie slova z pamäti** **Zápis slova do pamäti**



**Ukladací endian**

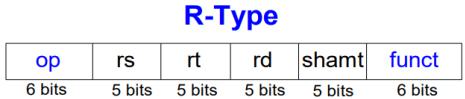
* Malý endian: v smere od najnevýznamnejších (LSB) po najvýznamnejšie bajty (MSB)
* Veľký endian: v smere od jeho najvýznamnejších bajtov po najnevýznamnejšie bajty



**Strojový jazyk / kód**

* Binárna reprezentácia inštrukcie
* 3 inštrukčné formáty (IF):

1. Typ R: operandy sú v registroch



1. Typ I: používa priamy operand

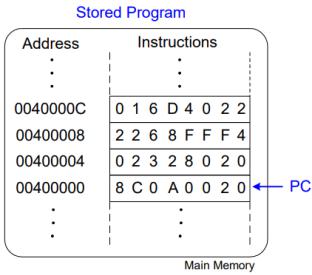


1. Typ J: vetvenia



**Program v pamäti**

* Inštrukcia a aj dáta sú v pamäti
* Rozdielna sekvencia inštrukcií = rozdielny program
* Spracovanie programu:
  1. Procesor číta inštrukcie z pamäte počítača o Procesor vykonáva inštrukcie
* Programové počítadlo (PC): definuje poradie spracovania inštrukcií



**Logické inštrukcie**

* and, or, xor, nor
  1. and: používa sa pri maskovaní bitov o or: používa sa pri rekombinácií bitov o nor: používa sa pri invertovaní bitov
* andi, ori, xori
* sll/sllv: logický posuv doľava
* srl/srlv: logický posuv doprava
* sra/srav: aritmetický posuv doprava

**Vetvenie**

* Ovplyvňuje poradie vykonania inštrukcií
* Typy vetvení:

1. Podmienené – beq, bne o Nepodmienené – j, jr, jal

**Polia**

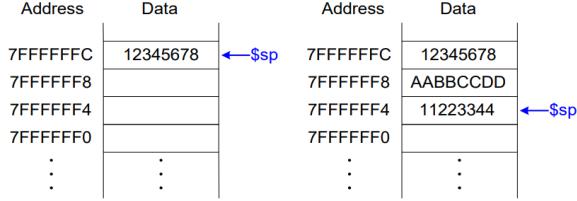
* Kompozitná dátová štruktúra
* Index: určuje pozíciu prvku v poli
* Veľkosť: počet prvkov v poli

**ASCII kód**

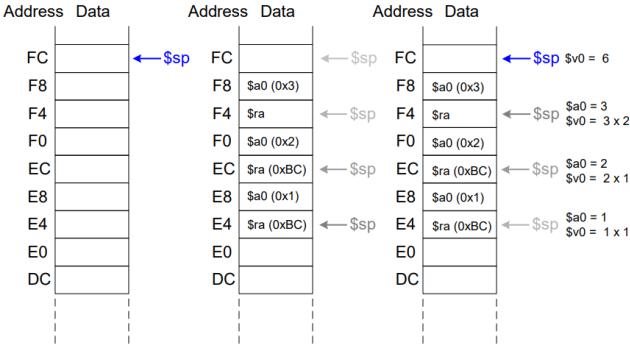
* American Standard Code for Information Interchange
* Každý znak má svoj unikátny kód
* Volajúci:
  1. odovzdáva argumenty volanej funkcie
* Volaný:
  1. Vykoná funkciu, Vráti výsledok, Zabezpečí návrat, Nesmie prepísať obsah registrov a pamäťových miest

**Zásobník**

* Pamäť pre dočasné premenné, Je typu LIFO (last-in-first-out)
* Zväčšuje sa dynamicky: Ak potrebujem ukladať viac dát
* Zmenšuje sa dynamicky: Ak už nepotrebujem tie dáta
* Rastie smerom nadol
* Smerník zásobníka: $sp ukazuje na vrchol zásobníka

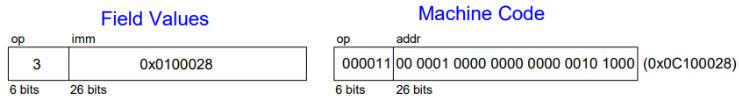
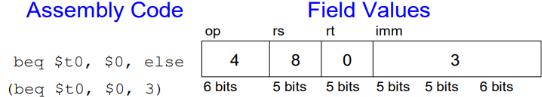


**Zásobník počas rekurzie**



**Adresovacie módy**

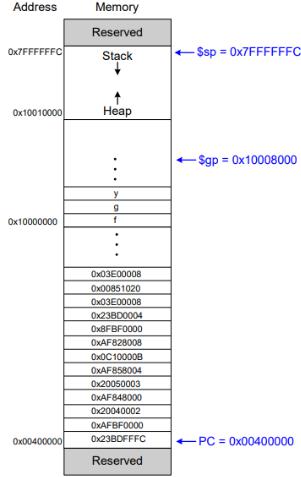
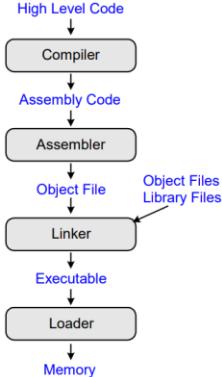
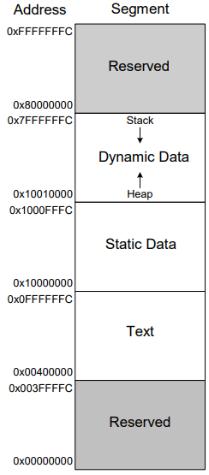
* Registrový mód - Operand je uložený v registri
* Bezprostredný mód - 16b konštanta je súčasťou inštrukcie
* Posúvací mód - Adresa operanda sa počíta ako: Bázová adresa + posuv
* PC-Relatívny mód
* Nepriamy indexovací mód



**Čo je uložené v pamäti?**

* inštrukcie, dáta

**MIPS pamäť** **Preklad & spúšťanie programu** **Spustiteľný kód v pamäti**



**Mikroarchitektúra**

* abstrakcia na úrovni štruktúrnej organizácie

**Procesor**

* Tok dát/operandov: spájanie funkčných jednotiek
* Tok riadenia: distribúcia riadiacich signálov

**Mikroarchitektúra**

* Rôzne IA pre tú istú ISA:

1. Jednocyklová: Každá inštrukcia sa vykoná za jeden strojový cyklus o Viaccyklová: Inštrukcie sa vykonajú za jeden alebo viac cyklov

o Prúdová: Vykonanie inštrukcie sa rozloží na dielčie kroky

* Čas vykonania programu
* Definícia: CPI: počet SC procesora v priebehu ktorých sa vykoná jedna inštrukcia
* Výzvou je nájsť správnu rovnováhu medzi parametrami: cena – spotreba – výkon

**MIPS Procesor**

* Nech je daná podmnožina MIPS inštrukčnej sady:

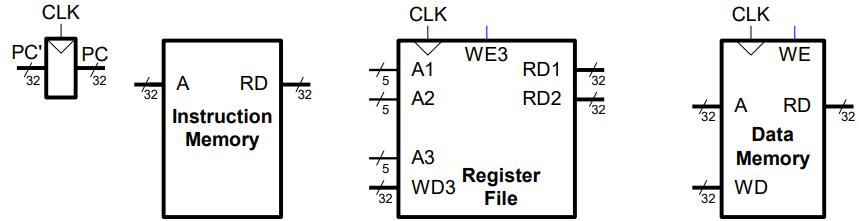
1. Inštrukcie typu R: and, or, add, sub, slt o Inštrukcie pre prácu s pamäťou: lw, sw o Inštrukcia vetvenia: beq

**Vnútorný stav architektúry**

* V ktorom stave sa architektúra nachádza závisí od:

1. Hodnoty programového počítadla (PC)
2. Hodnôt uchovaných v registroch architektúry o Obsahu pamäte

**Stavové elementy MIPS**

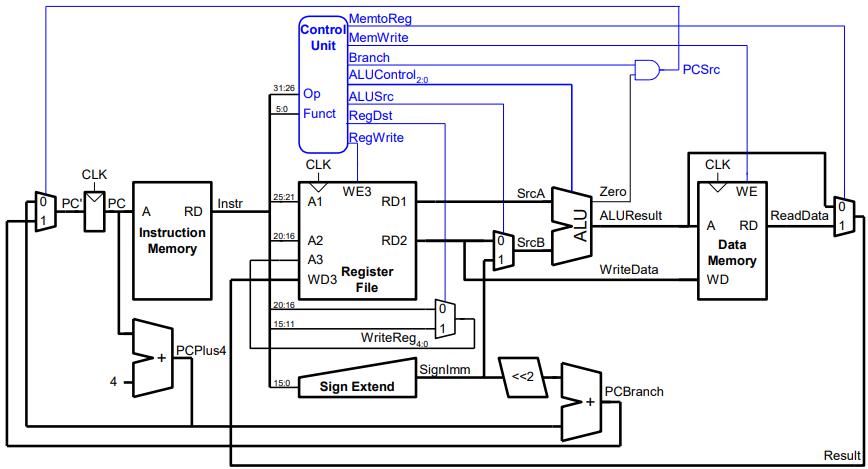


**Jednocyklový MIPS procesor**

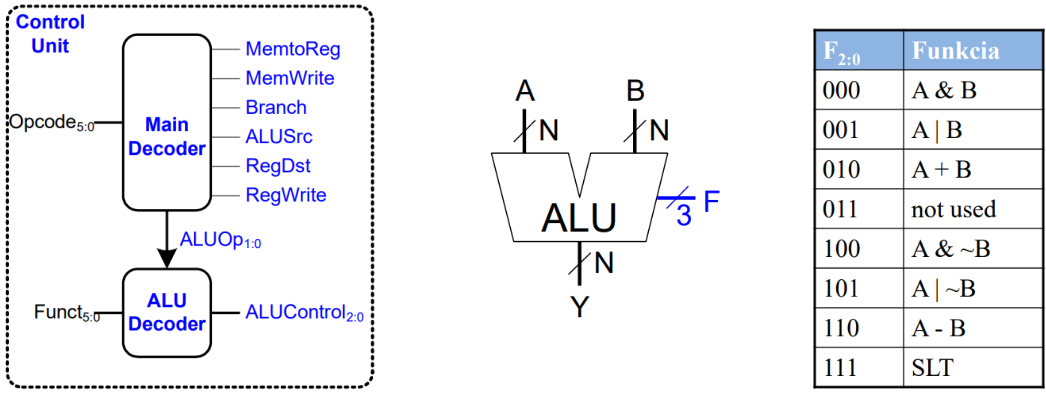
* Návrh procesora vychádza z návrhu jeho komponentov a prepojovacej sústavy, pričom sa zohľadňuje

o tok dát/operandov medzi funkčnými jednotkami procesora o tok riadenia

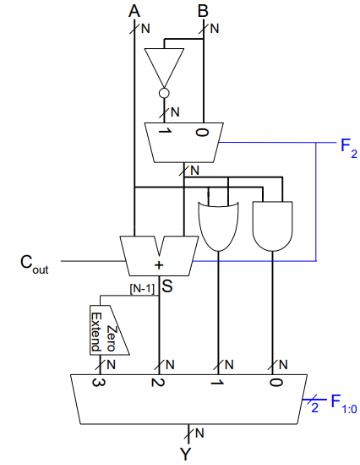
* čítanie inštrukcie:



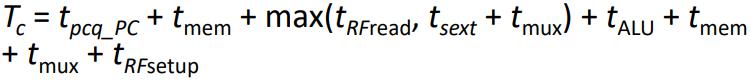
 riadenie Aritmeticko-logická jednotka



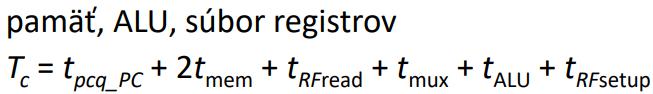
* Aritmeticko-logická jednotka



* Výkon jednocyklového procesora o Kritická cesta:

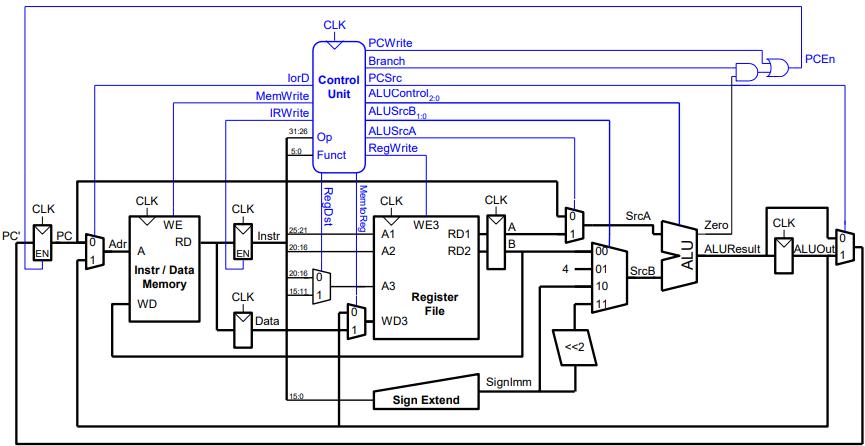


1. vo väčšine prípadov

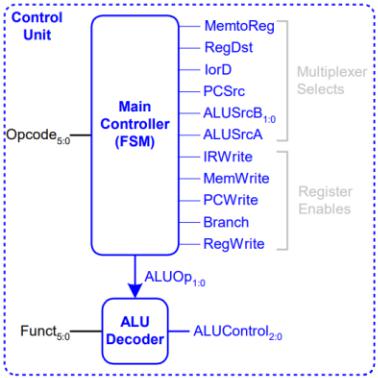


**Viaccyklový MIPS procesor**

* + vyššia taktovacia frekvencia
* + jednoduchšie inštrukcie vyžadujú kratší čas
* + znovupoužiteľnosť niektorých funkčných jednotiek počas realizácie inštrukcie
* - zložitejšie riadenie
* Podobný postup: návrh smerovania toku dát/operandov & smerovania riadiacich signálov



* Riadiaca jednotka viaccyklového MIPS procesora



* Výkon viaccyklového procesora o Kritická cesta



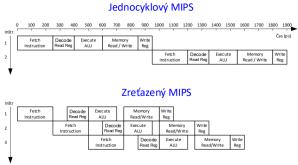
**Prúdový MIPS procesor**

* Temporálny paralelizmus
* Rozdeľ vykonanie inštrukcie v jednocyklovom procesore na 5 fáz: o Čítanie inštrukcie / Fetch

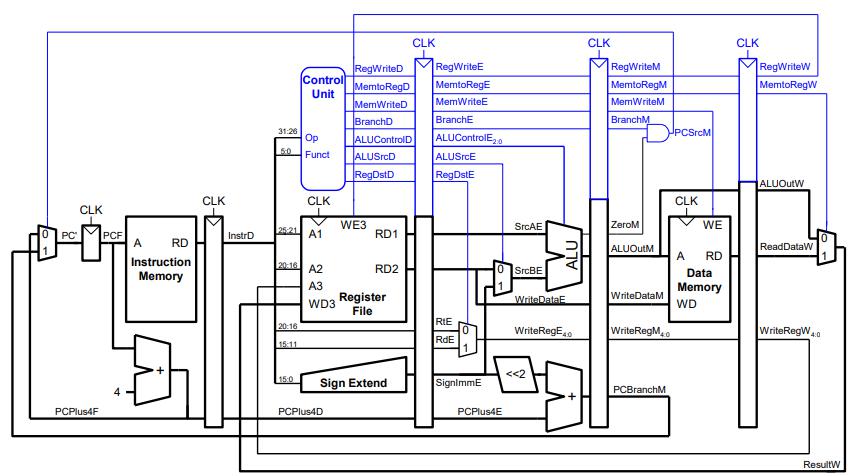
o Dekódovanie inštrukcie / Decode o Vykonanie inštrukcie / Execute o Pamätanie výsledku / Memory o Zápis do pamäte / Writeback

* Pridaj záchytné registre medzi jednotlivé stupne zreťazenia

**Jednocyklový vs. prúdový MIPS**



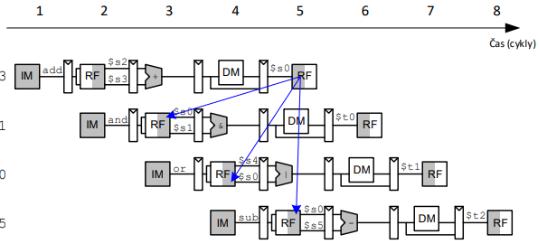
**Riadenie prúdového MIPS**



**Hazardy prúdového spracovania**

* Ak vykonanie inštrukcie je závislé od inštrukcie, ktorá ešte nebola dokončená
* Typy:

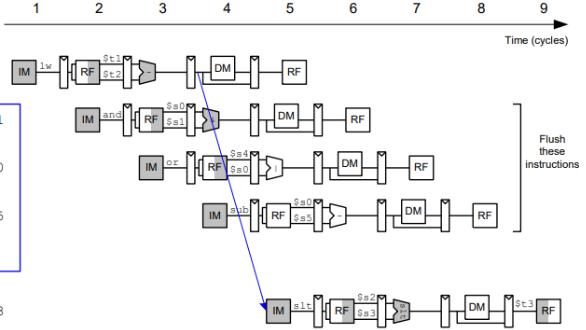
1. **Údajový hazard**: v dôsledku nedostupnosti operandov pri vykonaní inštrukcií



1. eliminácia: sw prostriedky – vloženie NOP po inštrukcií, preusporiadanie inštrukcii, hw prostriedky – blokovanie prúdového spracovania, dopredné

generovanie výsledku predchádzajúcej inštruckie

1. **Hazard riadenia**: v dôsledku vetvení
2. Pokutovanie nesprávneho odhadu skoku
   * Vyradenie inštrukcií z procesu spracovania ak sa skok uskutoční
   * Predikovanie skoku



* 1. **Zdrojový hazard**: nie sú dostupné zdroje na vykonanie inštrukcie

**Moderné mikroarchitektúry**

* „Hlboké“ prúdové funkčné jednotky: 10-20 stupňov
* Špekulatívne vetvenie: Ideálny prúdový procesor: CPI = 1, Zlá predikcia zvyšuje CPI
  1. Statická predikcia vetvenia: Kontrola smeru skoku
  2. Dynamická predikcia vetvenia: Na základe histórie skokov
* Superskalárne procesory : Viac inštrukcií je spracovaných naraz, Hazardy môžu byť problém
* Vykonávanie mimo poradia: Analýza niekoľkých inštrukcií, Priprav čo najviac inštrukcií na spracovanie, Spracuj inštrukcie aj mimo poradia
  1. závislosti:
     + RAW (read after write): údajová závislosť
     + WAR (write after read): údajová antizávislosť
     + WAW (write after write): výstupná závislosť
  2. Plánovanie spracovania inštrukcií
     + Instruction level parallelism (ILP)
     + Prehľadávací obvod (Scoreboard)
* Premenovávanie registrov
* SIMD : Single Instruction Multiple Data
  1. Jeden prúd inštrukcií spracováva viacnásobný tok dát
* Multivláknové procesory: mikro, nano, ...
* Multiprocesory: jeden čip ale viac procesorov
* Proces: program vykonávaný na počítači
  1. Je možné spúšťať niekoľko procesov súčasne
* Vlákno: časť programu
  1. Proces môže pozostávať z niekoľkých vlákien
* Vlákna v konvenčných procesoroch
  1. V danom čase „beží“ len jedno vlákno
  2. Ak sa pozastaví činnosť vlákna
     + Uloží sa stav vlákna
     + Načíta a spustí sa druhé (čakajúce) vlákno
     + Volá sa to zmena kontextu
* Multithreading
  1. Aktivuje sa niekoľko vlákien súčasne

1. Nezvyšuje mieru paralelizmu na úrovni inštrukcií (ILP) v rámci vlákna, ale zvyšuje priepustnosť

**Multiprocesory**

* Multiprocesorové systémy s definovanou komunikačnou topológiou

1. Tesne viazané multiprocesory o Voľne viazané multiprocesory

**Výkonnostné parametre**

* Výkon PS závisí od

1. Výkonu procesora
2. Pamäťového podsystému



**Návrh pamäťových modulov**

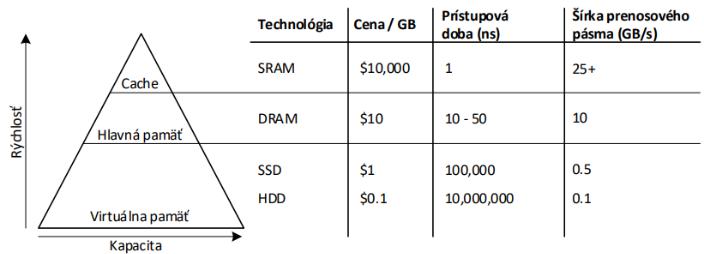
* Cieľom je navrhnúť pamäťový podsystém tak rýchly ako je sám procesor
* Aplikuje sa hierarchická stavba PaP
* Vlastnosti ideálnej pamäte:

1. rýchla

o lacná dve sú splniteľné

1. veľká kapacita

**Hierarchia PaP**



**Lokalita**

* Temporálna lokalita:
  1. Časová charakteristika prístupu
* Priestorová lokalita:
  1. Priestorová charakteristika prístupu

**Výkonnostné parametre**

* Úspešné sprístupnenie (hit): údaj sa našiel na danej úrovni PaP
* Neúspešné sprístupnenie (miss): údaj sa nenašiel
* Koeficient úspešnosti (Hit Rate; HR)
  1. HR = # úspešné sprístupnenie / # prístupov do pamäte = 1 – MR
* Koeficient neúspešnosti (Miss Rate; MR)
  1. MR = # neúspešné sprístupnenie / # prístupov do pamäte = 1 – HR
* Priemerná doba prístupu do pamäte (AMAT):



o

**Gene Amdahl, 1922-**

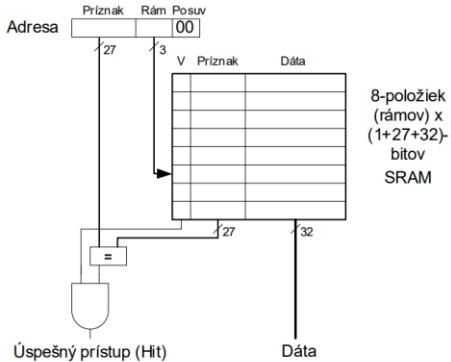
* Amdahlov „zákon“: úsilie vynaložené na zvýšenie výkonu subsystému minie účinkom, pokiaľ subsystém neovplyvní veľké percento celkového výkonu

**Vyrovnávacia pamäť typu „cache“**

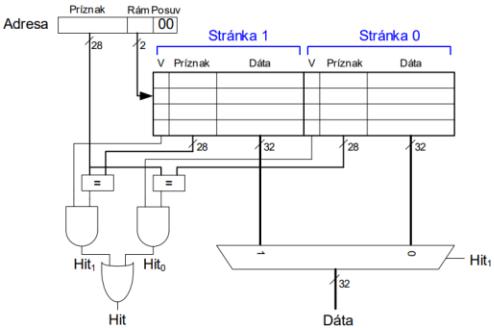
* Pamäť určená na vyrovnávanie rýchlosti prenosu údajov medzi procesorom a hlavnou pamäťou
* Je to rýchla asociatívna pamäť
* Parametre cache

1. Kapacita (C)
   1. Veľkosť bloku / skupiny slov v pamäti (b)
   2. Počet blokov v pamäti (B = C/b)
   3. Stupeň asociativity (N)
   4. Rám bloku cache (S = B/N)

* Ideálne, často používané dáta sú uložené v cache
* Cache sa delí podľa stupňa asociativity:
  1. Cache s priamym mapovaním



1. N-cestná asociatívna pamäť



* 1. Plne asociatívna cache – zníži počet konfliktov, je to drahé riešenie
* Kapacita pamäte cache
  1. Pamäť cache má obmedzenú kapacitu
* Metódy vyraďovania
  1. Náhodný výber

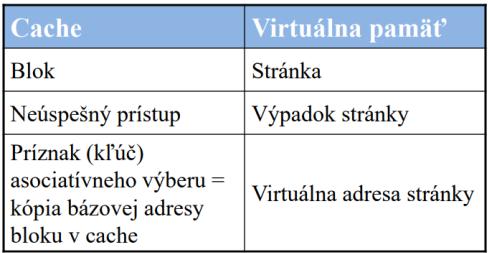


1. Výber podľa príznaku aktivity
   1. Stratégia LRU (Least recently used)

* Viacúrovňová organizácia cache
  1. Veľká cache má nižší koeficient neúspešnosti a prístupová doba je väčšia v porovnaní s menšou cache

**Virtuálna pamäť**

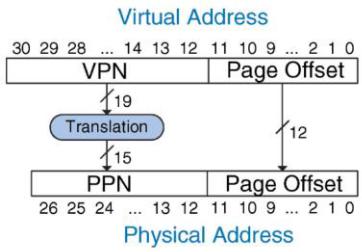
* Virtuálnou pamäťou sa získa väčší pamäťový priestor než čo nám ponúka hlavná pamäť
* Virtuálna adresa
  1. V programe sa používa virtuálna adresa na určenie miesta kde sa údaj nachádza
* Ochrana pamäte
  1. Každý program má vlastnú schému prekladu virtuálnych adries na fyzické
* Cache vs virtuálna pamäť



* transformácia virtuálnych adries na fyzické adresy sa nazýva mapovanie adries
* Virtuálna adresa je adresa, ktorou sa identifikuje slovo (blok) programu v logickom adresovom priestore
* Fyzická adresa predstavuje adresu kópie tohto slova (bloku) lokalizovaného v HP
* Existujú dve skupiny VP

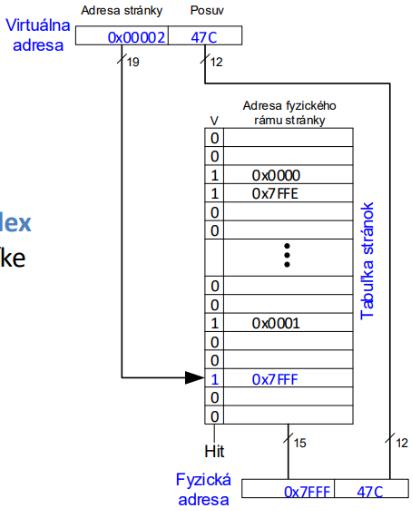
1. s pevnou dĺžkou blokov, ktoré sa nazývajú stránky
2. s premenlivou dĺžkou blokov, ktoré sa nazývajú segmenty

**Mapovanie pamäte**



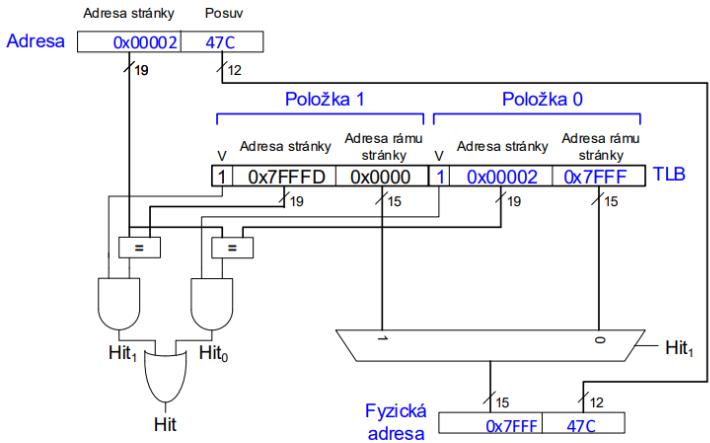
**Tabuľka stránok**

* Priebežný stav pamäte ČP v procese mapovania je registrovaný pomocou tabuľky stránok
* Adresa stránky funguje ako index položky v tabuľke stránok



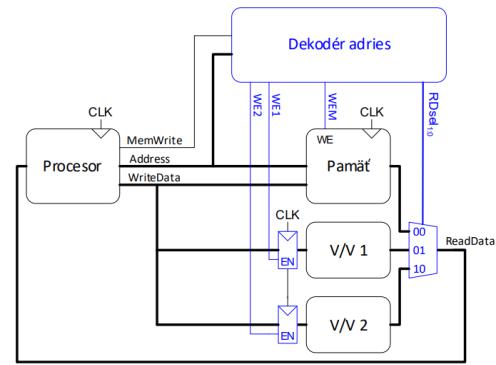
**Pamäť preložených adries (TLB) - Translation Lookaside Buffer**

* mapovanie prostredníctvom pamäte preložených adries
* sú uložené adresy najčastejšie používaných stránok
* Ukážka TLB



**Vstupno-výstupný podsystém**

* Každé V/V zariadenie má svoju adresu
* Čítanie/zápis z/do V/V zariadenia vyžaduje adresu V/V zariadenia
* Rozhranie V/V podsystému



* Komponenty V/V podsystému

1. Dekodér adries - Zisťuje adresu zariadenia
2. V/V registre - Obsahujú dáta určené pre V/V zariadenia
3. Multiplexor ReadData - Zabezpečuje smerovanie medzi V/V zariadením