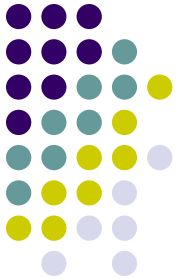
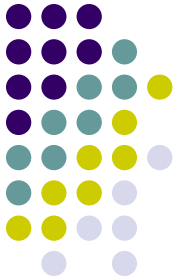
Grafická karta



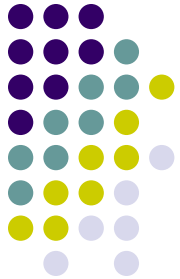
- **Grafická karta** neboli **grafický adaptér** – Vytváří digitální nebo analogový **obrazový signál** pro monitor dle zadaných obrazových dat
- Grafické možnosti počítačů PC byly po dlouhou dobu velmi chabé
- První grafické karty umožňovaly pouze bitovou mapu uloženou ve své paměti převést na obrazový signál
- IBM PC je navrženo tak, aby bylo modulární - každou součást počítače může vyrábět jiná firma
- Pro vytváření obrazového signálu pro monitor se používala **grafická karta**, která s mikroprocesorem komunikuje přes univerzální sběrnici – brzy nastává problém vysokého toku dat přes „úzkou“ sběrnici
- **Grafická karta** byla tedy samostatným zařízením, které se zapojovalo do příslušného slotu na základní desce
- Později se objevují **integrovaná řešení** – grafika byla přímo součásti chipsetu na základní desce (např. integrovaný grafický adapter v severním můstku)
- Moderní mikroprocesory mají **integrovaný grafický procesor (GPU)**, což umožňuje rychlý přenos obrazových dat přímo uvnitř čipu mezi jednotlivými sekcemi mikroprocesoru
- Grafický adapter integrovaný na základní desce nebo v mikroprocesoru lze deaktivovat a nahradit výkonnější grafickou kartou zasunutou ve slotu sběrnice PCI-e

Grafická karta



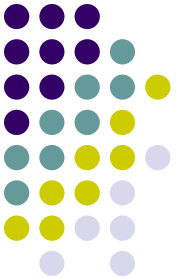
- Nejjednodušší grafické karty pouze mapují část operační paměti vyhrazené pro zobrazování (takzvané video paměti) a tuto informaci **převádějí na obrazový signál**
- Složitější grafické karty už obsahují **vlastní paměť**
- Moderní grafické karty umí i počítat se zadanými geometrickými objekty (úsečky, trojúhelníky…) a samy vytváří v paměti grafická data
- **Grafický akcelerátor** – grafický adaptér, který pouze nepřevádí data z videopaměti na obrazový signál, ale umí také provádět výpočty polohy zadaných těles
- Grafický akcelerátor obsahuje samostatný mikroprocesor, který umí provádět maticové výpočty, transformace souřadnic, výpočty používané při kódování a dekódování videa a další…
- Nejvyšším stupněm grafické akcelerace je podpora zobrazování **3D těles**

Základní parametry



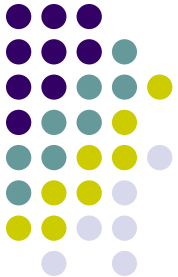
- Typ
  - Integrovaný grafický adaptér
    - Součást severního můstku
    - GPU integrovaná v mikroprocesoru
  - Samostatná karta
    - Připojení
      - ISA, EISA
      - VL-BUS
      - AGP
      - PCI-E
- Velikost videopaměti
- Rozlišení, barevná hloubka, režimy
- Obnovovací (snímková) frekvence
- Možnost akcelerace
  - 2D akcelerace
  - 3D akcelerace
- Výstupní signál
  - Analogový (Hercules, VGA, TV signál….)
  - Digitální (DVI, HDMI, DisplayPort)

Historie



- **Textový režim** umožňuje zobrazovat jen text a dříve byl hlavním používaným režimem zobrazení
- V textovém režimu je obrazovka rozdělena na rastr (mřížku) z buněk pro znaky uspořádaných do určitého počtu řádků a sloupců
- Obsah těchto „buněk“ je uložen ve videopaměti, což je paměťový prostor pro udržování aktuálního stavu obrazovky
- Datový tok obrazových dat je minimální
- Do paměti se zapisují pouze ASCII kódy znaků na jednotlivých řádcích
- Fyzicky jsou ve videopaměti znaky uloženy po řádcích za sebou, tj. nejdřív první řádek, pak druhý, atd. První bajt uložený ve videopaměti je tedy kód znaku, který bude zobrazen v levém horním rohu
- Zápis obsahu celé obrazovky do videopaměti (80x25 znaků) je vlastně pouze zápis 2000 bajtů
- Úkolem grafické karty bylo pouze tato data několikrát za sekundu přečíst (podle snímkové frekvence např. 50x za sekundu), každý znak převést dle definice jeho podoby na obrazové body a vygenerovat analogový signál pro monitor
- Toto zvládaly i jednoduché, levné, pomalé chipy

Historie

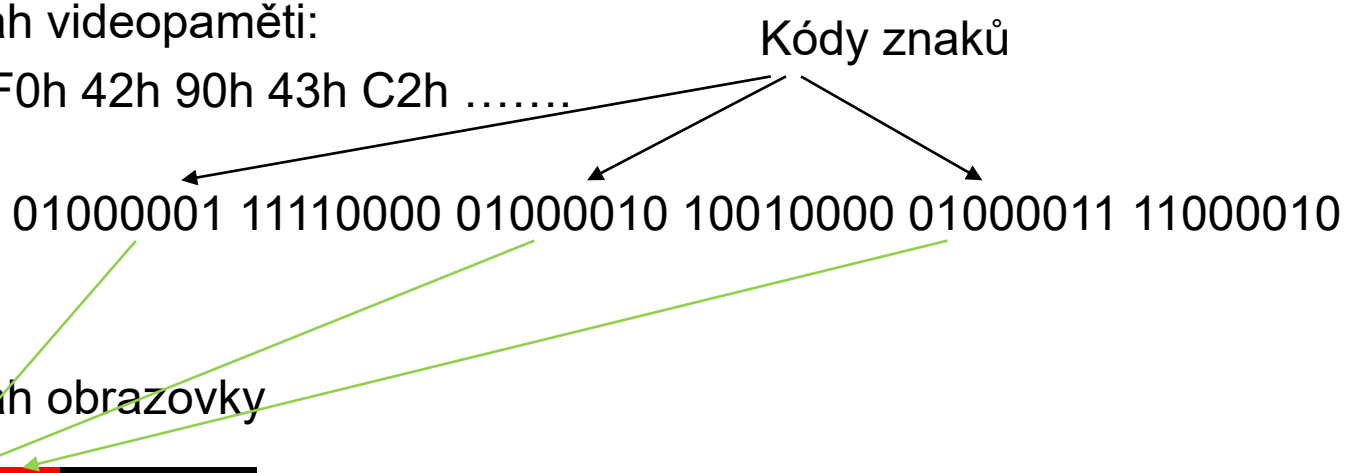


- Později se textový režim rozšířil o **barvy**
- Ke každém znaku je pak potřeba přidat také informaci o jeho barvě
- Informace o znaku bude zakódována pomocí dvou bajtů – ASCII kód znaku + barva
- Pokud je k zakódování barvy použit jeden bajt (8 bitů), pak lze používat 2^8 různých barev (256 barev)
- Nejstarší grafické karty většinou tak velký počet barev nezvládaly
- Bajt s informací o barvě tehdy kódoval pomocí 4 bitů barvu znaku (16 možných barev) a pomocí dalších 4 bitů barvu pozadí (16 možných barev)
- Nebo bývala běžná možnost kódovat pomocí 4 bitů barvu znaku (16 možných barev), pomocí 3 bitů barvu pozadí (8 možných barev) a posledním bitem zapnout/vypnout blikání – nastavit se tak daly bizarní kombinace, např. blikající žlutý znak na červeném pozadí

- Pokud se začnou používat barvy, k uložení obsahu obrazovky v běžném textovém režimu (rastr 80x25 znaků) již potřebujeme 4000 B (80x25x 2B)

Historie

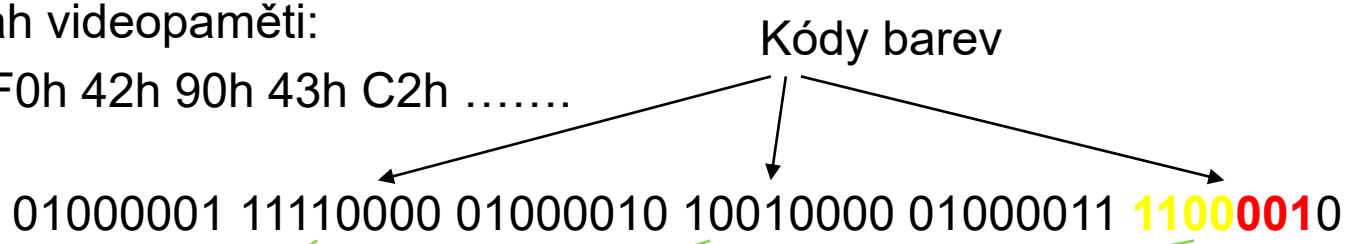
- Příklad
- Obsah videopaměti:
- 41h F0h 42h 90h 43h C2h …….



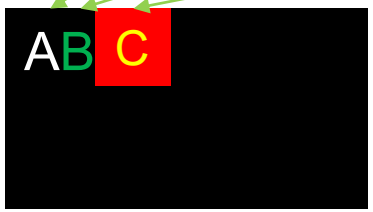
- Obsah obrazovky

Historie

- Příklad
- Obsah videopaměti:
- 41h F0h 42h 90h 43h C2h …….

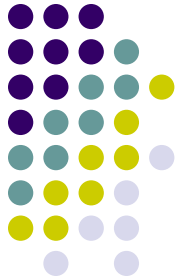


- Obsah obrazovky



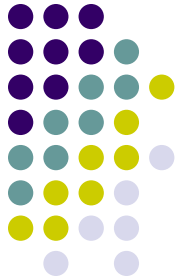
4 bity – barva znaku
3 bity – barva pozadí
1 bit blikání

Historie



- Pokud je velikost videopaměti větší, lze do ní uložit několik „stránek“
- Tak si lze připravit několik podob obsahu obrazovky a rychle mezi nimi přepínat
- Zápis do neaktivní stránky se okamžitě nijak neprojeví. Lze si tak připravit celou „budoucí obrazovku“ a pak jí naráz zobrazit
- Čím větší velikost videopaměti, tím více budoucích grafických dat lze mít připraveno

Historie

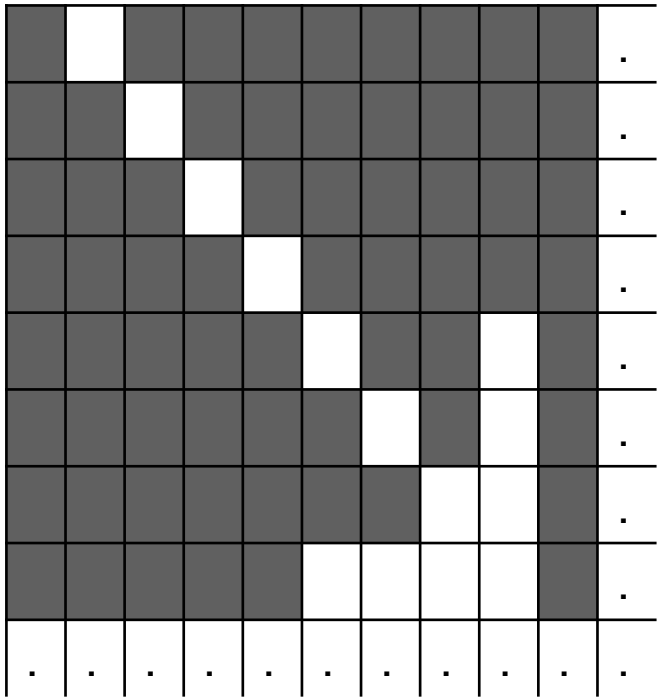


- **Grafický režim** – obsah obrazovky je vykreslen jako matice pixelů – obrazových bodů
- **Pixel** = picture element – obrazový bod
- Každý obrazový bod má určitou barvu
- Množství potřebné videopaměti se odvíjí od rozlišení obrazovky a barevné hloubky (počet bitů použitý pro zakódování barvy pixelu)
- Informace o pixelech jsou opět uložené v paměti postupně po jednotlivých řádcích
- V monochromatickém režimu pixel buď svítí nebo je zhasnutý
- V monochromatickém režimu stačí k uložení informace o každém pixelu jeden bit

- Například při **monochromatickém rozlišení 640x480 pixelů** je potřeba uložit informaci o 640x480=307200 pixelech
- Informace o každém pixelu je uložena jedním bitem (1/0 = svítí/nesvítí)
- Obsah obrazovky se uloží jako 307200 bitů, tj **38400 Bajtů**

Videopaměť

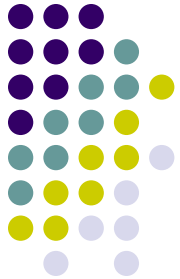
Na obrazovce



Videopaměť

0	1	0	0	0	0	0	0	0	0	.
0	0	1	0	0	0	0	0	0	0	.
0	0	0	1	0	0	0	0	0	0	.
0	0	0	0	1	0	0	0	0	0	.
0	0	0	0	0	1	0	0	1	0	.
0	0	0	0	0	0	1	0	1	0	.
0	0	0	0	0	0	0	1	1	0	.
0	0	0	0	0	1	1	1	1	0	.
.	.	.	.	.	.	.	.	.	.	.

Grafický režim



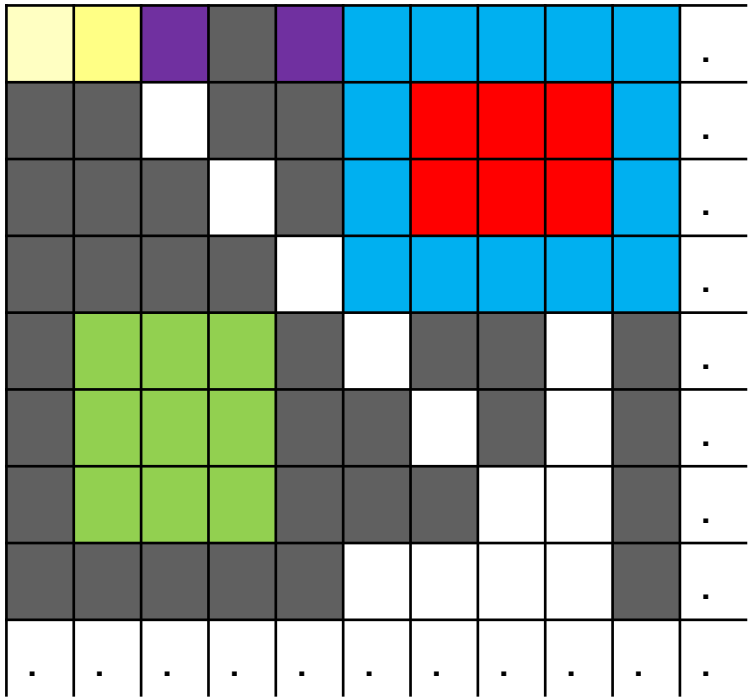
- Pokud se v grafickém režimu použijí barvy, je informace o každém pixelu uložena pomocí více bitů
- **Barevná hloubka** – počet bitů použitý k zakódování barvy pixelu
- 1 bit – monochromatické zobrazení
- 2 bity - 4 barvy
- 4 bity – 16 barev
- 8 bitů – 256 barev
- 16 bitů – 65536 barev
- 24 bitů - 16 777 216 barev (TrueColor)

- Čím větší je barevná hloubka, tím více barev se používá a obraz je věrnější, ale roste množství obrazových dat a k uložení obsahu obrazovky je potřeba více paměti

Barvy

Příklad
8-bitová barevná hloubka
256 barev
Každý pixel je uložen v paměti jako 1 Bajt

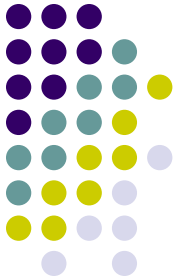
Na obrazovce:



Videopaměť:

21	73	09	00	09	33	33	33	33	33	.
00	00	FF	00	00	33	A0	A0	A0	33	.
00	00	00	FF	00	33	A0	A0	A0	33	.
00	00	00	00	FF	33	33	33	33	33	.
00	A1	A1	A1	00	FF	00	00	FF	00	.
00	A1	A1	A1	00	00	FF	00	FF	00	.
00	A1	A1	A1	00	00	00	FF	FF	00	.
00	00	00	00	00	FF	FF	FF	FF	00	.
.	.	.	.	.	.	.	.	.	.	.

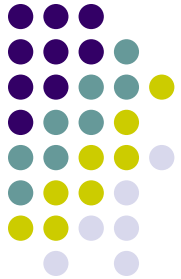
Příklad



- Kolik bajtů je třeba k uložení obsahu obrazovky při monochromatickém rozlišení 320x200 px ?

- Obraz je složen ze 320 x 200 = 64000 bodů
- Každý pixel je uložen v paměti jako 1 bit (svítí/nesvítí)
- Obsah obrazovky je tedy uložen jako 64000 bitů
- K uložení obsahu obrazovky je třeba 8000 B

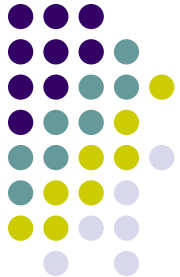
Příklad



- Kolik bajtů je třeba k uložení obsahu obrazovky ve čtyřbarevné hloubce v rozlišení 640x400 px?

- Obraz je složen ze 640 x 400 = 256000 bodů
- Pixel může mít jednu ze 4 barev, takže je třeba použít k zakódování barvy pixelu 2 bity
- Každý pixel je tedy uložen v paměti jako 2 bity
- Obsah obrazovky je uložen jako 256000 x 2 bity = 512000 b
- Obsah obrazovky je tedy uložen jako 64000 B

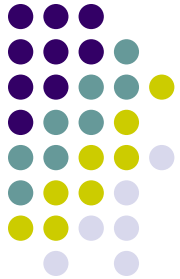
Příklad



- Kolik bajtů je třeba k uložení obsahu obrazovky v rozlišení 640x480 px, je-li barevná hloubka 256 barev?

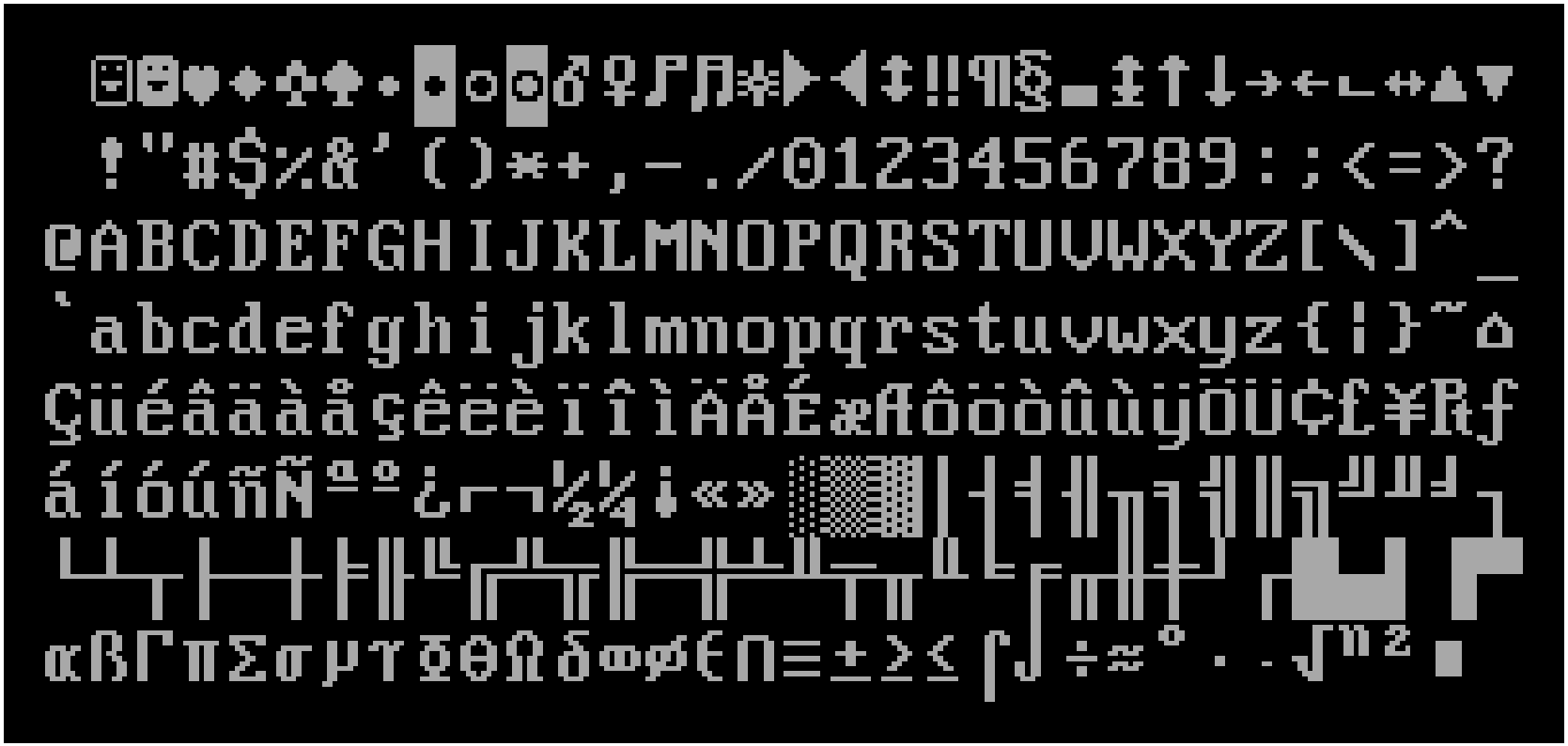
- Obraz je složen ze 640 x 480 = 307200 bodů
- Zobrazuje-li se 256 barev, je třeba k zakódování barvy pixelu použít 8 bitů ($2^8 = 256$)
- Každý pixel je tedy uložen v paměti jako 1 bajt
- Obsah obrazovky je tedy uložen jako 307200 B

Historie



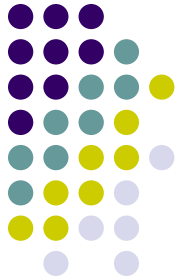
- **MDA** – monochrome display adapter – grafický adapter instalovaný v prvních počítačích PC. Umožňoval pouze černobílý obraz (monitor pak obvykle zobrazoval místo bílé zelenou)
- **Textový režim** – 25 řádků, 80 znaků na řádek
- Jeden znak má pevný rozměr 9x14 pixelů
- Adapter obsahoval paměť, ve které byla uložena **pevná podoba jednotlivých znaků** (jaké pixely mají svítit a jaké mají být zhasnuté, aby byl zobrazen znak s příslušným ASCII kódem)
- Rozlišení obrazu bylo **720 x 350 pixelů**, snímková frekvence **50 Hz** (viditelně to blikalo a unavovalo oči)
- Zobrazení různých typů písma nebylo možné
- Zobrazení znaků s českou diakritikou nebylo možně (grafický adapter je neznal)
- Po přepnutí zobrazení do grafického režimu přestal adapter tvořit obraz jednotlivých znaků a umožnil černobílou grafiku v rozlišení 320x200 pixelů
- Veškerá obrazová data byla uložena v paměti 4 kB
  - Z toho 2 kB obsahují informace o tom, jak vypadají jednotlivé znaky (definice zobrazitelných znaků bitmapou 9x14 pixelů)
  - Další 2 kB obsahují 80x25 znaků – tzn. obsah obrazovky
  - Tato obrazová paměť byla namapována do segmentu s počáteční adresou B0000h

Pevná znaková sada MDA
Každý znak je definován v rastru 9x14 px

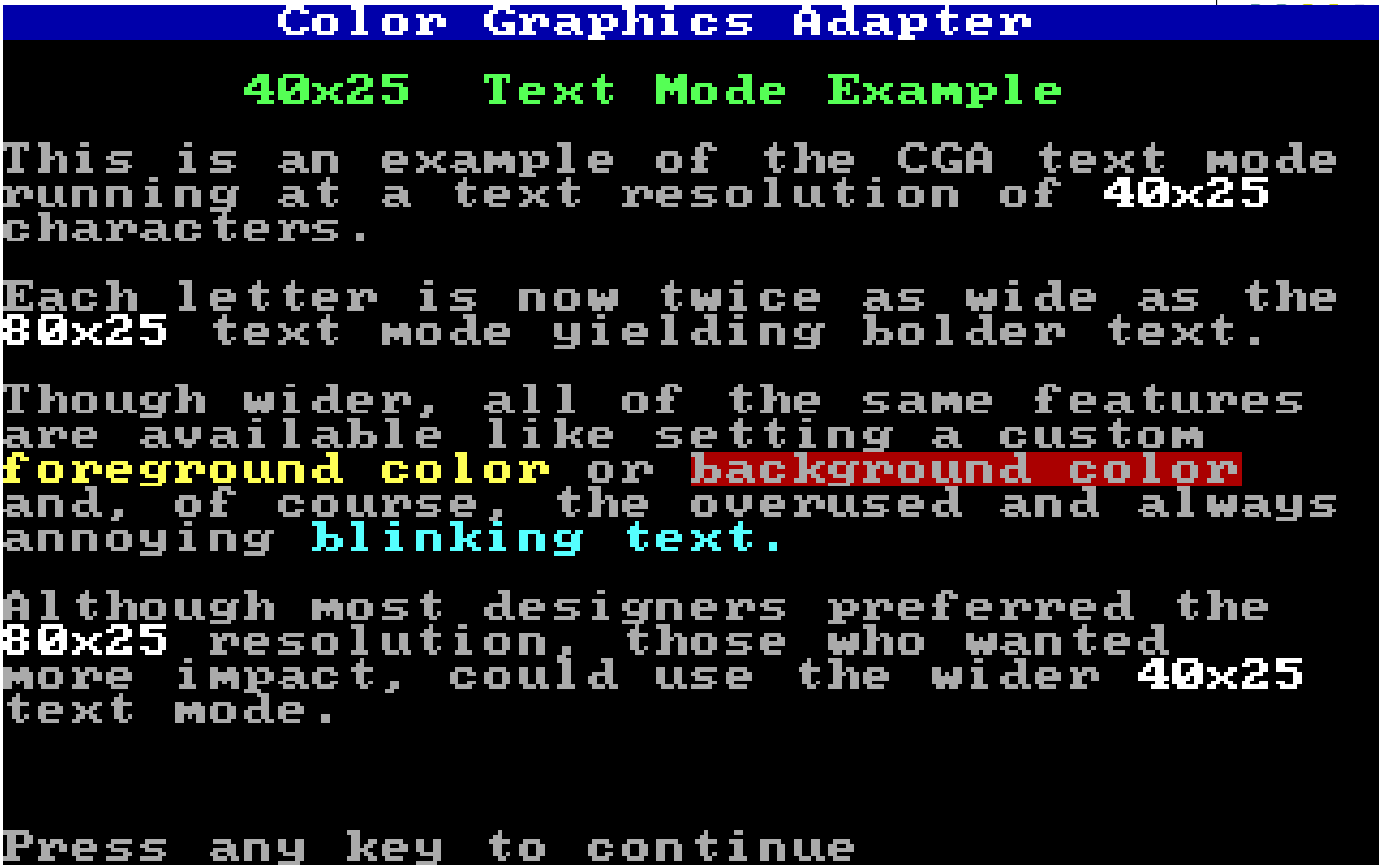


Monochromatické monitory pro MDA karty nebyly pouze černo-bílé. Používala se běžně i varianta černo-zelená a černo-oranžová.

Historie

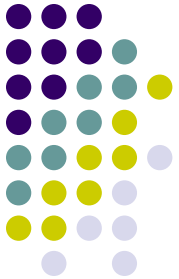


- **CGA** – color graphics adapter
- Používal se u prvních počítačů PC a umožňoval v grafickém režimu zobrazení ve **čtyřech barvách** při rozlišení 640x200 px (pixel byl dvakrát vyšší než jeho šířka)
- V textovém režimu měl znak rozměry 8x8 pixelů – nevypadalo to moc pěkně
- Rozlišení obrazovky v textovém režimu bylo pouze 320x200 px
- Znaková sada v paměti EPROM se dala přeprogramovat, aby bylo možné v textovém režimu zobrazovat i české znaky s diakritikou
- Karta je řízena čipem Motorola MC6845
- Ve videopaměti byl každý znak uložen dvoubajtově
  - 1. bajt – ASCII kód zobrazovaného znaku
  - 2. bajt barva popředí (spodní 4 bity) a barva pozadí (horní 3 bity) + blikání (1 bit)
- Bylo možné přesně nakonfigurovat vertikální a horizontální frekvenci, prodlevu pro návrat paprsku zpět do horního rohu
- Nastavením nesmyslných parametrů bylo možné zničit monitor (mnohokrát se to stalo)


Color Graphics Adapter
40x25 Text Mode Example
This is an example of the CGA text mode running at a text resolution of 40x25 characters.
Each letter is now twice as wide as the 80x25 text mode yielding bolder text.
Though wider, all of the same features are available like setting a custom foreground color or background color and, of course, the overused and always annoying blinking text.
Although most designers preferred the 80x25 resolution, those who wanted more impact, could use the wider 40x25 text mode.
Press any key to continue

CGA monitor

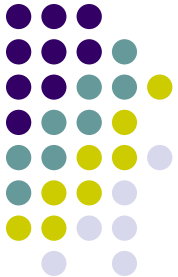

HIGH SCORE: 0
Tapper
TM
Designated Trade Mark of Bally Midway Mfg. Co.
©1983 Bally Midway
IBM
Personal Computer Color Display
NFS



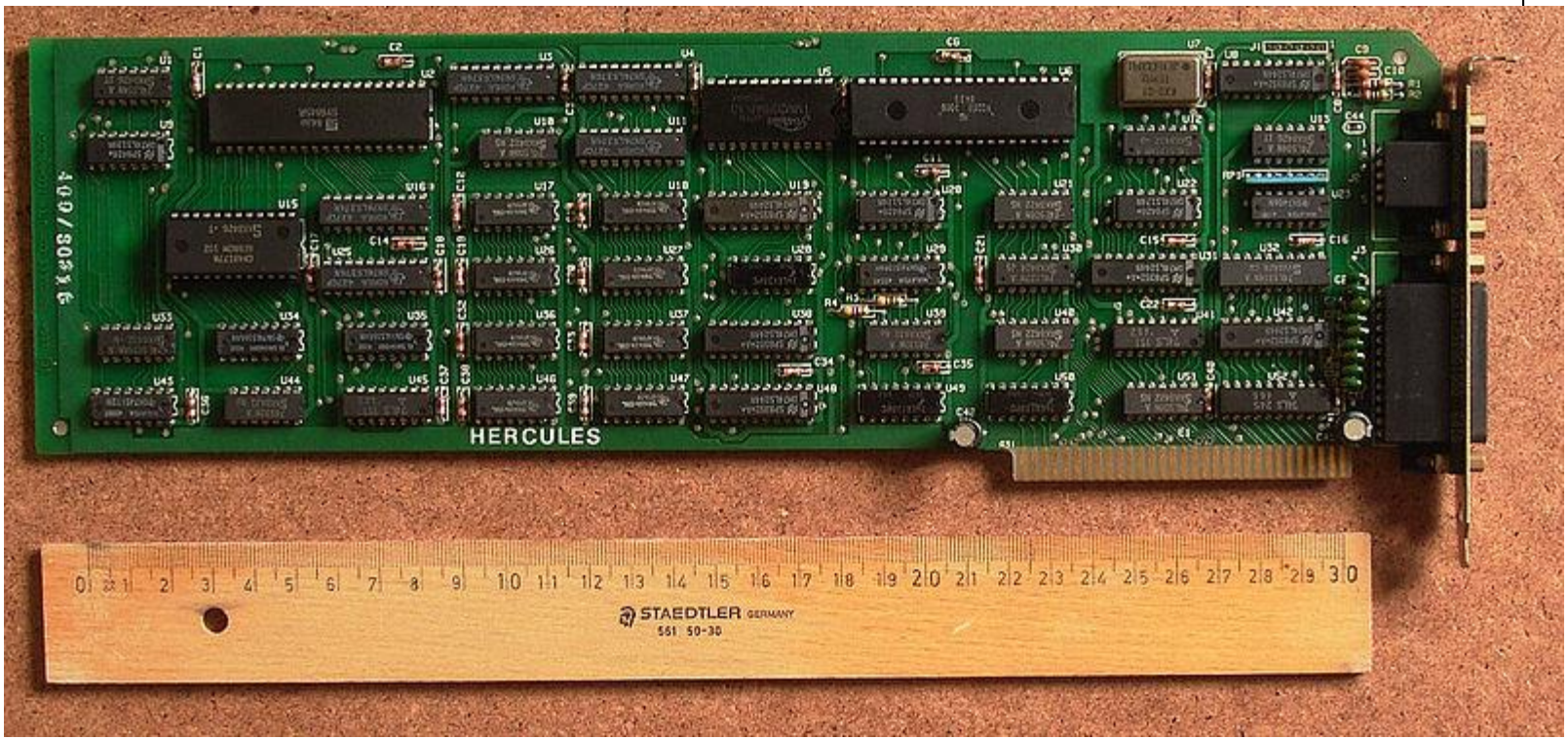
- **Jak vypadaly CGA hry používající 4 barvy**

https://www.youtube.com/watch?v=Z1isu8oMxTw

Historie



- **HGC** – **Hercules** graphics card
- Vzniká ve stejné době jako CGA a MDA – jde o jednu ze tří alternativ, mezi kterými mohl uživatel tehdy volit
- V textovém režimu **80 znaků na 25 řádcích** v celkovém rozlišení **720×350** pixelů (kompatibilní s MDA)
- V grafickém režimu pouze dvoubarevné zobrazení, ale ve vyšším rozlišení 720 x 348 px
- Obrazová paměť s kapacitou 32 kB mapovaná do segmentu s počáteční adresou B0000h
- Grafická karta je řízena také čipem Motorola MC6845 (stejně jako CGA)
- Každý z uvedených adapterů MDA, CGA, Hercules používá jiný způsob volání grafických služeb a jiný typ výstupního signálu
- **Pro každý adapter je potřeba jiný monitor**, monitory byly navrhovány speciálně pro určité typy grafických karet
- Pro každý grafický adapter bylo potřeba napsat jinou verzi programu (např. hra naprogramovaná pro CGA nešla hrát na počítači s kartou Hercules)



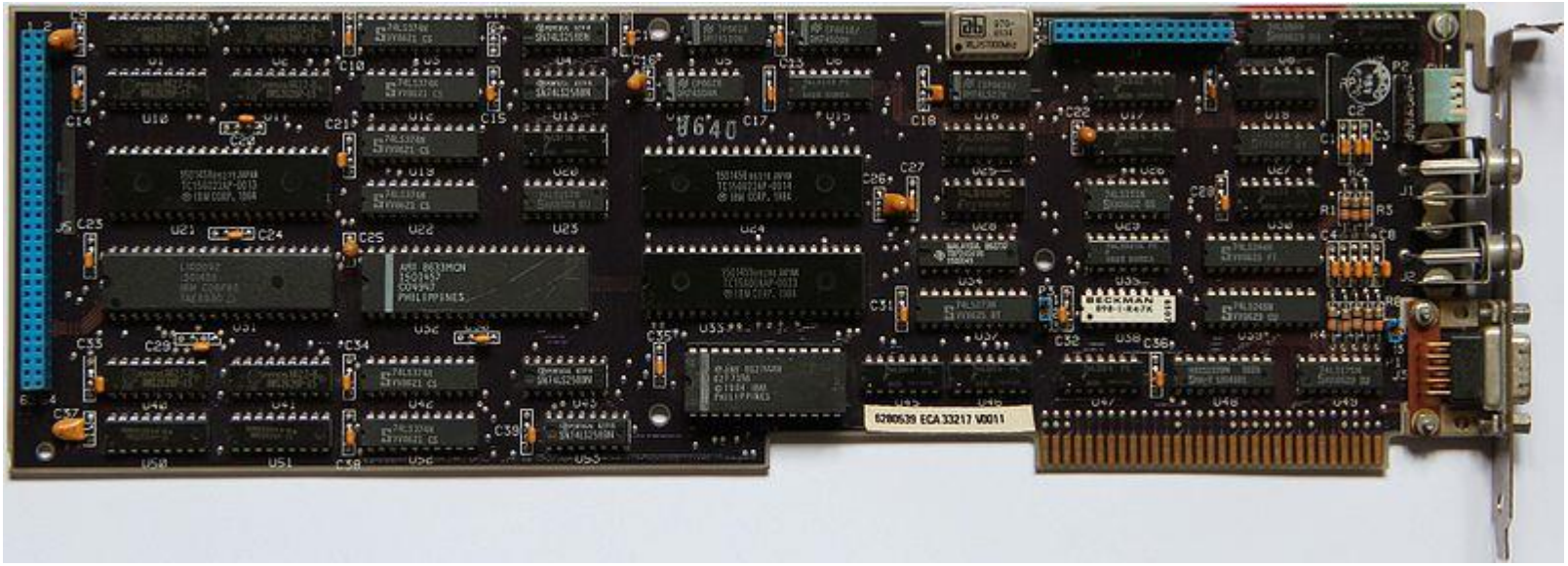
Grafická karta Hercules

Jako bonus byl v této kartě řadič paralelního portu, takže instalací této grafiky se počítač rozšířil i o paralelní port, který v té době nebyl běžný. Většina počítačů měla pouze sériové porty (COM1, COM2)

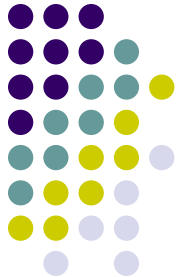
Takto vypadala hra v monochromatickém rozlišení 720x350 px

Historie

- **EGA** – Enhanced graphic adapter
- Používané v počítačích IBM PC/AT
- Cena cca 500 $
- Rozlišení 640x350 px, **16 barev**
- Existovalo více různých verzí, které se lišily velikostí grafické paměti (64kB – 256 kB)
- Zapojovala se do slotu sběrnic ISA nebo EISA

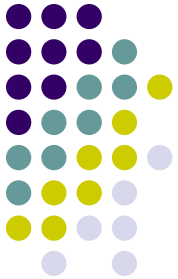


Historie



- **PGC** – první grafický akcelerátor pro počítače PC
- Vznik v roce 1984
- Používána byla hlavně v počítačích pro **CAD** systémy
- Cena této karty byla přibližně stejná jako cena celého počítače PC
- Jedná se o celý výpočetní subsystém tvořený **3 kartami**, které jsou přes 3 (!!!) ISA sloty zapojeny do počítače PC
- Všechny tři desky se mezi sebou propojovaly kabelem, který tvořil interní sběrnici grafického akcelerátoru
- PGC funguje jako vykonavatel grafických příkazů
- Akcelerátor dovedl vykonávat okolo 80 grafických povelů jako např
  - CIRCLE – vykreslení kružnice
  - AREA – vyplnění plochy barvou
  - DRAW – vykreslení úsečky
  - CLEARS – smazání celé obrazovky (je třeba si uvědomit, že bez grafického akcelerátoru by bylo potřeba provést tisíce zápisů nulových bajtů do obrazové paměti)
  - PROJCT – nastavení úhlu pohledu na rovinu
  - MDROTX – výpočet rotace objektu

Grafická akcelerace

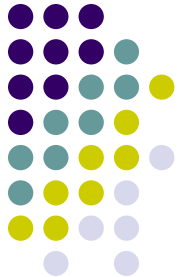


Vykreslení kružnice s poloměrem 50 px a středem na pozici x=100 y=100

- **Bez grafické akcelerace**
- Mikroprocesor musí vypočítat souřadnice všech bodů kružnice x,y = (100,50);(101,50);(102;49);(103,49);(104,48)…… Celkem to bude cca 314 bodů
- Mikroprocesor musí pro každý vypočítaný bod nalézt adresu, na které je ve videopaměti uložený pixel odpovídající daným souřadnicím
- Mikroprocesor musí do videopaměti na všechny tyto adresy zapsat bajt, kterým nastaví odpovídající barvu pixelů tvořících kružnici
- Mikroprocesor tedy musí provést stovky operací a dochází také zatížení sběrnice přenosem dat při zápisech do paměti

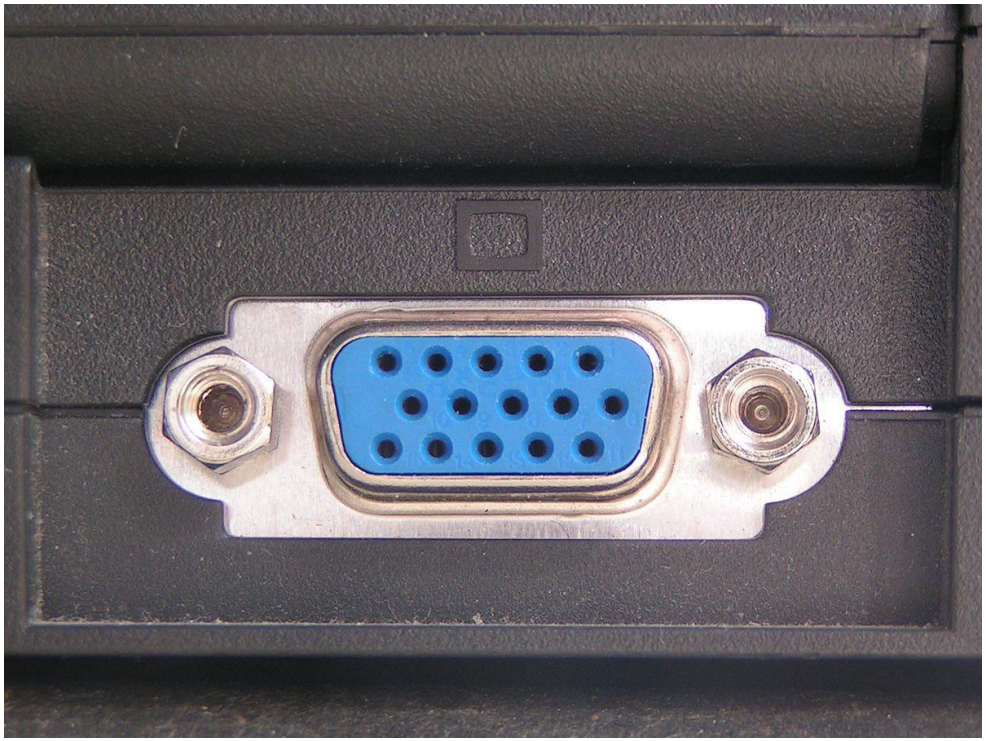
- **S grafickou akcelerací**
- Mikroprocesor zadá grafické kartě povel k vykreslení kružnice a předá jí odpovídající parametry
- Tím veškerá práce pro mikroprocesor končí
- Grafická karta si sama nastaví ve videopaměti odpovídající obsah tak, aby byla zobrazena zadaná kružnice. Veškeré výpočty a nastavení pixelů probíhá uvnitř grafické karty

VGA



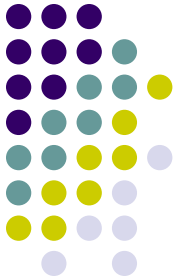
- **VGA** – Video graphic array
- Vyvinuto firmou **IBM** pro počítače řady **IBM PS/2**
- Karty tohoto typu vyrábí v různých variantách velké množství výrobců (ATI, S3, Matrox, Trident…)
- Všechny VGA karty mají jednotný analogový výstup přes nový 15-pinový VGA konektor
- Běžné rozlišení až **640x 480 px**
- **16** nebo **256 barev**
- Podporuje výpočty s grafickými daty – prolínání a překrývání objektů se počítá pomocí logických operací XOR, AND
- Do grafické paměti lze uložit **více stránek** (snímků) a mezi nimi lze přepínat
- Pro textový režim lze definovat **vlastní fonty** - znak o rozměru 8x16 pixelů
- Znaková sada v textovém režimu mohla obsahovat dokonce 512 různých znaků (znak tedy mohl mít více podob – např. tučný, kurziva)
- V textovém režimu rozlišení obrazovky **80x25 znaků**
- Obnovovací frekvence **60 Hz** nebo **70 Hz**
- Programování a ovládání tohoto grafického adapteru bylo velmi komplikované

VGA konektor



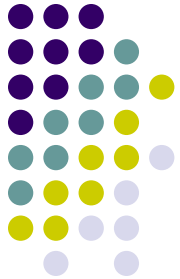
VGA paleta – 256 barev

VESA



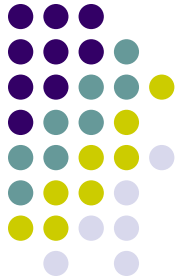
- Ve stejném období (rok 1989) vzniká **VESA** – Video Electronics Standard Association – asociace výrobců grafických karet, monitorů a dalších komponent týkajících se grafiky, zobrazování a videa
- Konsorcium snažící se sjednotit grafické formáty, výstupy grafických karet a jejich grafické a textové režimy
- Grafické karty mají jednotný výstup a stejný způsob uložení grafických dat v paměti
- Není již potřeba pro každou grafickou kartu programovat aplikaci jinak
- Není již potřeba pro každou grafickou kartu kupovat jiný monitor
- Výrobci se dohodli na způsobu kódování barev do budoucna a na dalších standardních rozlišeních pro budoucí grafické karty (aby to bylo jednotné a nedělal každý výrobce své vlastní rozlišení)
- U předchozích grafických adapterů neexistoval jednotný způsob nastavení grafických režimů, lišily se režimy přístupu do video paměti, způsob přepínání snímků…
- Výsledkem činnosti VESA je mimo jiné VL-BUS – sběrnice pro připojení grafických karet používaná v počítačích s mikroprocesorech 80486
- Dalším cílem VESA bylo, aby operační systémy nepotřebovali mít ovladač pro každý monitor zvlášť¨, ale při připojení monitoru si automaticky zjistí jeho rozlišení a zobrazení grafiky se mu přizpůsobí

Standardizace



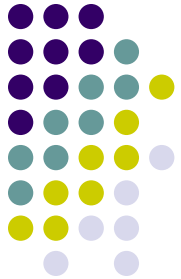
- **Nastává období sjednocení a standardizace grafických režimů, rozlišení a výstupů**
- **VESA 2.0 –** normy, kde byla uvedena čísla všech různých používaných grafických režimů tak, aby je bylo možné nastavit jednotným způsobem pomocí služeb takzvaného VESA BIOSu.
- Dále byly specifikovány služby pro zjištění parametrů grafických karet, tj. výrobce grafického čipu, typu grafické karty, podporovaných rozlišení a podporovaných obnovovacích frekvencí obrazu
- Dále je zde popsáno uspořádání grafických dat v obrazové paměti a komunikace s grafickým adapterem přes sběrnice

SVGA



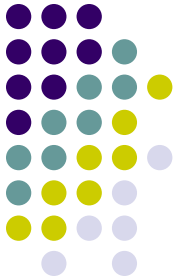
- **Super VGA**
- Výsledek práce asociace **VESA**
- Rozlišení 800x600 až **1024x768 px**
- Poprvé podporována barevná hloubka **TrueColor** – 24 bitů (16,7 mil. barev)
- První programy pracující v této barevné hloubce se objevily v polovině 90. let (začíná se používat komprese JPEG a na počítačích se poprvé dají ukládat a upravovat fotografie)
- Kapacita obrazové paměti byla různá a běžně překračovala 1 MB
- Pro uložení obrazu 1024x768 v TrueColor barevné hloubce je potřeba 1024x768x3 B, což je 2,25 MB

TrueColor



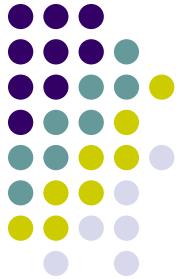
- 24-bitová barevná hloubka
- V počítačové grafice běžná přibližně od roku 1990
- Umožňuje fotorealistické barvy
- Pixel může mít jednu z 2^{24} = 16 777 216 barev
- Použitelných barev je více, než kolik dokáže rozlišit lidské oko (dvě sousední barvy palety jsou pro oko nerozlišitelné)
- Barva pixelu je uložena pomocí 3 Bajtů (3 x 8 bitů)
- Barva pixelu je složena ze tří základních barev – červené, zelené a modré – jejichž intenzitu lze nastavit v různém poměru
- 24 bitů jsou vlastně tři bajty – tedy tři hodnoty <0;255>
- První číslo říká, jak hodně je pixel červený
- Druhé číslo říká, jak hodně je pixel modrý
- Třetí číslo říká, jak hodně je pixel zelený

Grafická akcelerace



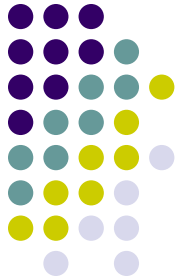
- Důvodem pro rozvoj **grafické akcelerace** v 90. letech byl rozvoj **interaktivní grafiky** – hra, animace, video nebo ovládaný program musí okamžitě reagovat na ovládání uživatelem a působit plynulým dojmem
- Dosud byly sledovanými parametry pouze rozlišení a barevná hloubka.
- Nově se stává zásadním parametrem počet zobrazitelných nových snímků za sekundu
- **FPS – frames per second**
- Vznikají aplikace, kde je důležitý počet zobrazitelných snímků za sekundu, který by měl pro dobrou plynulost dosahovat hodnoty alespoň **15 FPS**
- Množství obrazových dat vznikajících každou sekundu je ohromné a není možné, aby je počítal mikroprocesor a ještě je přesouval pomalou sběrnicí do grafického adapterů
- Grafický adaptér se sám stává i **výpočetní jednotkou** obrazových dat, mikroprocesor se výpočtu obrazu neúčastní (pouze zadá grafickému adaptéru úkoly, co má zobrazovat a jak to má počítat)

2D akcelerace



- 2D akcelerace byla nutná pro možnost plnohodnotného využití operačních systémů s GUI
- Windows bez akcelerace byla prakticky nepoužitelná
- Vykreslování oken bez akcelerace bylo velmi pomalé
- Plynulé scrollování obsahu okna bylo nemožné
- Procesor musel pracovat s grafikou pixel po pixelu
- Při tehdejší rychlosti procesorů to znamenalo zátěž i při pouhém přesunu okna po obrazovce
- Posouvaná okna nebo jiné objekty se vykreslily vždy až na cílové pozici – průběh posuvu (mezisnímky na trase) se nestíhaly

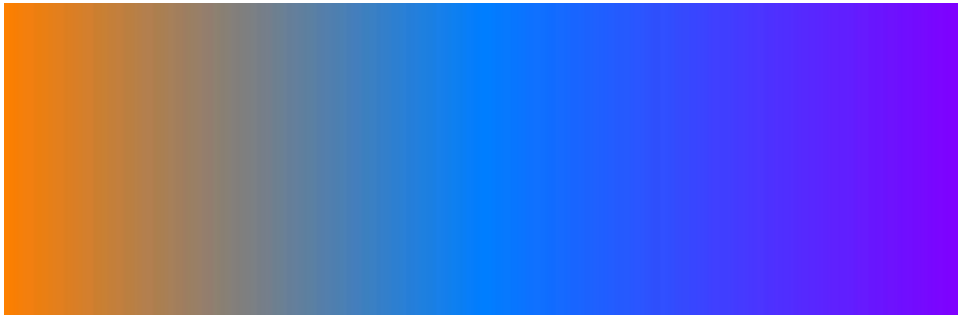
Grafická akcelerace



- V první polovině 90. let se nejprve objevuje 2D akcelerace
- **2D akcelerace** = výpočty s grafickými daty úseček, křivek, textu, obdélníků, kružnic, souvislých ploch, barevných přechodů, bitmapových obrázků, fotografií, snímků videa apod.

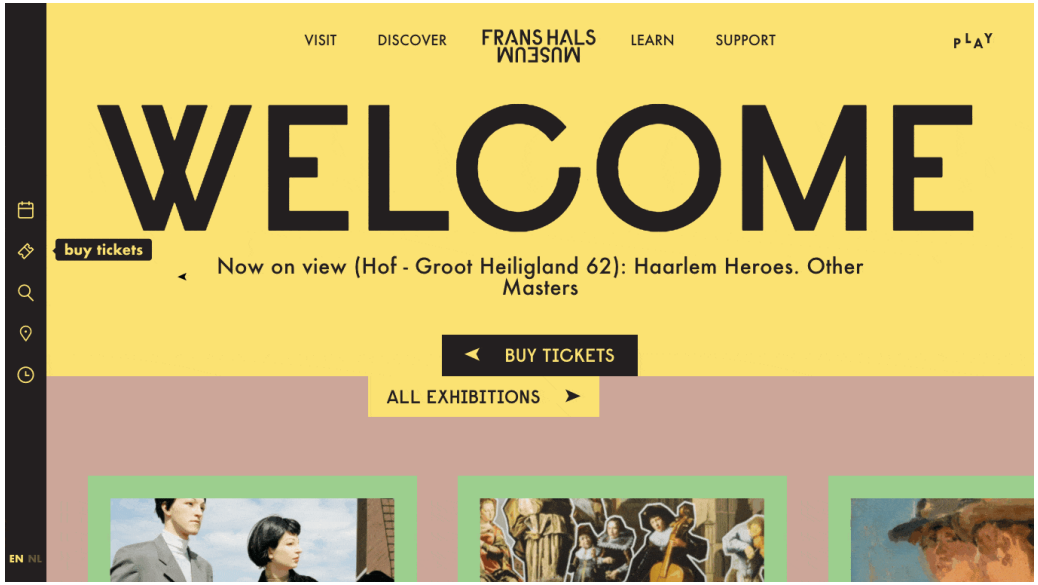
2D akcelerace

- **Color blending** – vyplnění plochy plynulým barevným přechodem od jedné barvy k druhé



2D akcelerace

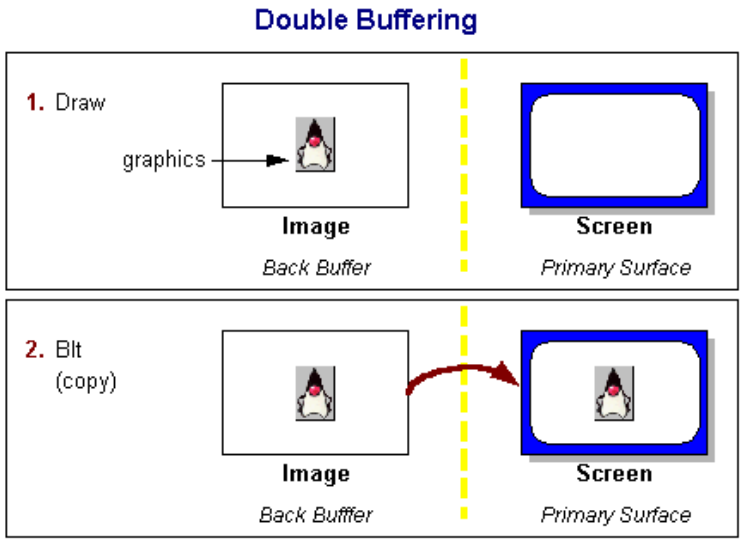
- **Scrolling** – vertikální, horizontální, všesměrový posuv celého obrazu
- Bez akcelerace by musel mikroprocesor počítače přesouvat statisíce bajtů ve videopaměti z místa na místo – to by byla ohromná zátěž pro procesor a sběrnici
- Díky akceleraci pošle procesor pouze informace o oblasti a směru posuvu grafický akcelerátor si vše spočítá a přesune sám



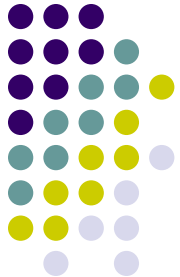
2D akcelerace

- **Double-buffering** – viditelná obrazovka je uložena v předním bufferu.
- Zapisovat lze také to tzv. zadního bufferu, který je neviditelný.
- Obraz snímku lze postupně vypočítat a vykreslit v zadním bufferu a následně prohodit roli obou bufferů velmi jednoduchou operací nevyžadující žádné přenosy dat.
- Uživatel tak nevidí postupné vykreslování ale rovnou se na displeji objeví celý výsledný snímek.

- **Multiple-buffering** – ve videopaměti lze mít dopředu vypočítáno a připraveno několik snímků v několika bufferech. Přepínáním bufferů se pak snímky zobrazují a vidíme rychlou animaci



2D akcelerace

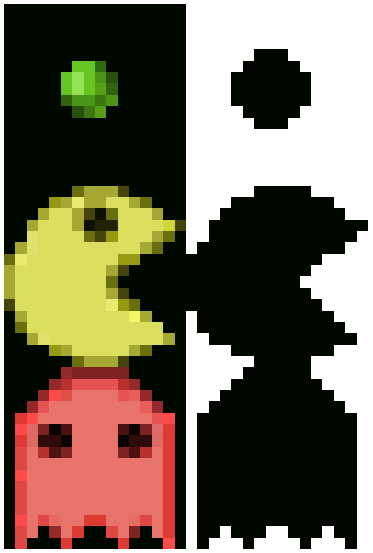


- **Blitter** (anglicky BLock Image Transfer) je v informačních technologiích označení pro specializovaný obvod, který nezávisle na procesoru velmi rychle přesunuje data v paměti počítače. Typicky slouží jako akcelerace pro přesun části obrazu (bitmapy) zobrazené na monitoru počítače – provede přesun nebo zkopírování bloku obrazových dat v paměti na jiné místo

- **Blitting**
- Do grafické paměti se uloží nejčastěji používané rastrové obrazce (např. ikony, obrazy tlačítek apod.)
- Následně je možné pomocí operací blokového přesunu dat (Bit-Block Transfer – BitBlt – Blit) provést vykreslení těchto obrazců na různých místech obrazovky bez dalšího zbytečného zatížení mikroprocesoru a sběrnice

2D akcelerace

- **Spriting** – přesuny a klonování jednoduchých rastrových 2D objektů s nepravidelným okrajem
- V podstatě jde o blitting s objekty libovolného tvaru, kterým říkáme **sprite**
- Vytváření kopií existující části obrazu (např. kosmická loď v počítačové hře je popsána jednou a pak jsou zobrazeny její kopie)

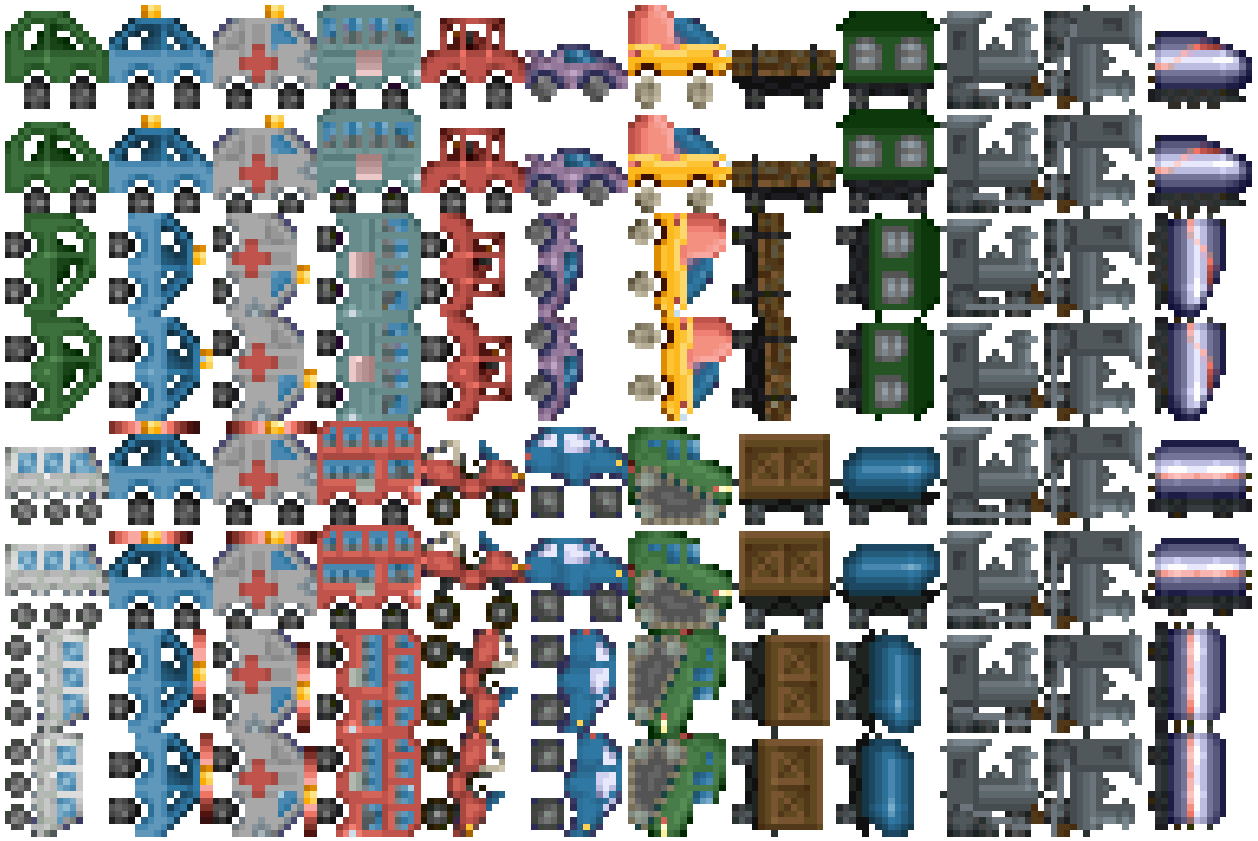


maska

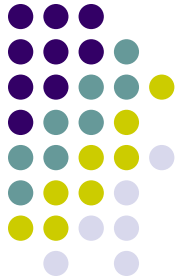
Blitter u spritů používá masku, určující jaké pixely přesune. Maska pracuje jako šablona ukazující tvar spritu

2D akcelerace

- **Spriting** – ukázka spritů



2D akcelerace



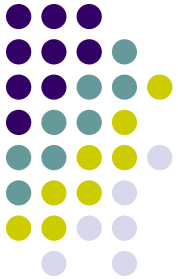
- **Cutting** – ořezávání obrazu – zobrazí se pouze část. Např. ve zmenšeném okně se zobrazí jeho oříznutý obsah

- **Mirorring** – zrcadlení
  - Velmi problematické, pokud by ho měl provádět mikroprocesor místo grafické karty. Obrazová data by bylo třeba nejprve všechna přečíst, někam přesunout a pak v opačném pořadí přesunout zpět.

2D akcelerace

- **Scaling** – změna měřítka obrazu (zvětšení/zmenšení)
- Např. přepočet videa 4:3 → 16:9
- Bez akcelerace grafickou kartou by procesor nestíhal v reálném čase při přehrávání videa přepočítat snímky na jiný poměr stran

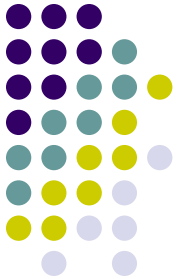
2D akcelerace



- **Práce s více vrstvami** – pozadí, popředí
- **Maskovací bitmapy** – bitová mapa, kterou lze nastavit, které pixely nějaké části obrazu se zobrazí a které ne
- **Průhlednost -** bitmapa s informací, zda se má korespondující pixel vykreslit, či zda se ponechá stávající pozadí

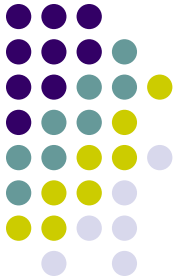
- **Konverze mezi různými barevnými prostory** – barva pixelu může být reprezentována i jiným způsobem než pomocí R G B hodnot
- Používají se například také barevné prostory C M Y K (vhodný pro tisk) nebo YUV (vhodný pro fotografii a video a jejich kompresi)

2D akcelerace



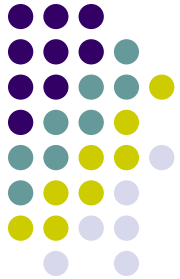
- **Komprese** a **dekomprese** s pomocí diskrétní kosinové transformace (DCT)
- Objem obrazových dat při použití barevné hloubky TrueColor a vyšším rozlišení je ohromný
- Kompresí lze snížit množství dat potřebných k uložení obrazu
- **JPEG** Komprese se snaží popsat obraz matematicky – průběh jasu a barev v jednotlivých řádcích a sloupcích lze vyjádřit jako matematickou funkci a tu lze rozložit na kosinusoidy s různou amplitudou a frekvencí.
- *Například všechny vaše známky, které jste postupně získali z předmětu HW by šly popsat jako nějaká matematický funkce (v ideálním případě y=1, reálně to ale bude funkce, která bude různě oscilovat) a tato funkce by se dala rozložit na součet sinusoid nebo kosinusoid (DCT). Průběh vašeho prospěchu by se pak dal zakódovat amplitudou a frekvencí těchto sinusoid/kosinusoid.*
- Diskrétní kosinová transformace umí libovolnou funkci převést na řadu kosinusoid, jejichž složením by taková funkce vznikla
- Obraz pak už není zakódován jako jednotlivé pixely, ale jako amplitudy a frekvence kosinusoid, jejichž sečtením vzniká funkce popisující průběh jasu jednotlivých barevných složek v horizontálním a vertikálním směru
- Kvalitní grafické karty podporují výpočty používané při tomto způsobu komprese/dekomprese a výrazně jí tím urychlí

2D akcelerace



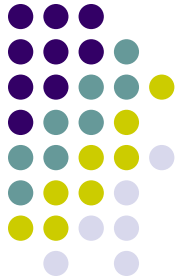
- JPEG Komprese
- https://www.youtube.com/watch?v=Q2aEzeMDHMA
- https://www.youtube.com/watch?v=Ba89cI9eIg8

2D akcelerace - video



- Pro kompresi videa byla zavedena komprese **MPEG** (podobný princip jako JPEG, navíc se zde pracuje s podobností po sobě jdoucích snímků)
- Jednou z metod 2D akcelerace se pak stává i podpora výpočtů při dekompresi MPEG videa
- Kolem roku 1995 se objevují první grafické karty, které díky 2D akceleraci umí přehrát **video v TV kvalitě** (25 snímků/s v rozlišení 720x576 px)

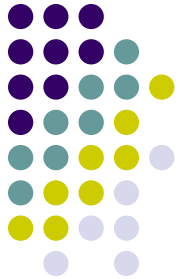
GUI a grafický akcelerátor



- S příchodem dalších moderních operačních systému s grafickým uživatelským rozhraním (Windows95) stoupla potřeba grafické akcelerace
- Bylo třeba řešit
  - **Vykreslování oken** a jejich vzájemné překrývání
    - Překrytá (neviditelná část) obsahu okna musí být někde uložena, ale nesmí být vidět
  - Možnost **scrollování** obsahu okna
    - Celý obsah okna není vidět, ale je zapsán ve videopaměti a při pohybu posuvníku se postupně zobrazuje
  - Urychlení vykreslování **kurzoru myši**
    - bez grafického akcelerátoru, tedy při programovém vykreslování kurzoru myši by bylo zapotřebí při každé grafické operaci testovat, zda vykreslovaný obrazec nebo dialog nezasahuje do oblasti se zobrazeným kurzorem. Pokud by kolize nastala, musela by se dotčená část obrazu nejprve neviditelně vykreslit do operační paměti počítače a poté provést kombinaci s kurzorem myši.

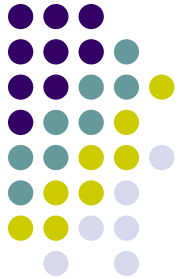
- Vznikají tedy grafické karty s hardwarovou akcelerací výše popsaných aktivit – Cirrus Logic, Trio 32, Trio 64, Matrox, S3 Vision

3D akcelerátory



- 3D akcelerace se objevuje na grafických kartách v druhé polovině devadesátých let minulého století
- Umožňuje rychlé vykreslení prostorových těles popsaných pomocí **hraniční reprezentace** (nezajímá nás, co je uvnitř, ale jak to vypadá zvenku)
- základní operací, kterou musí tyto grafické akcelerátory provádět, je zobrazení **trojúhelníků** vyplněných zadaným **povrchem** (barva, barevný přechod, textura) a řešení jejich vzájemného **překrytí** podle vzdálenosti od pozorovatele a úhlu pohledu
- Nejjednodušším zobrazitelným 3D objektem je plocha popsaná jako trojúhelník (bod ani přímka by nebyly vidět kvůli „nulové tloušťce“)
- Plochu je třeba definovat pomocí tří bodů
- Každý bod v prostoru – **vertex** - je pak třeba definovat pomocí tří souřadnic x, y, z

3D akcelerace



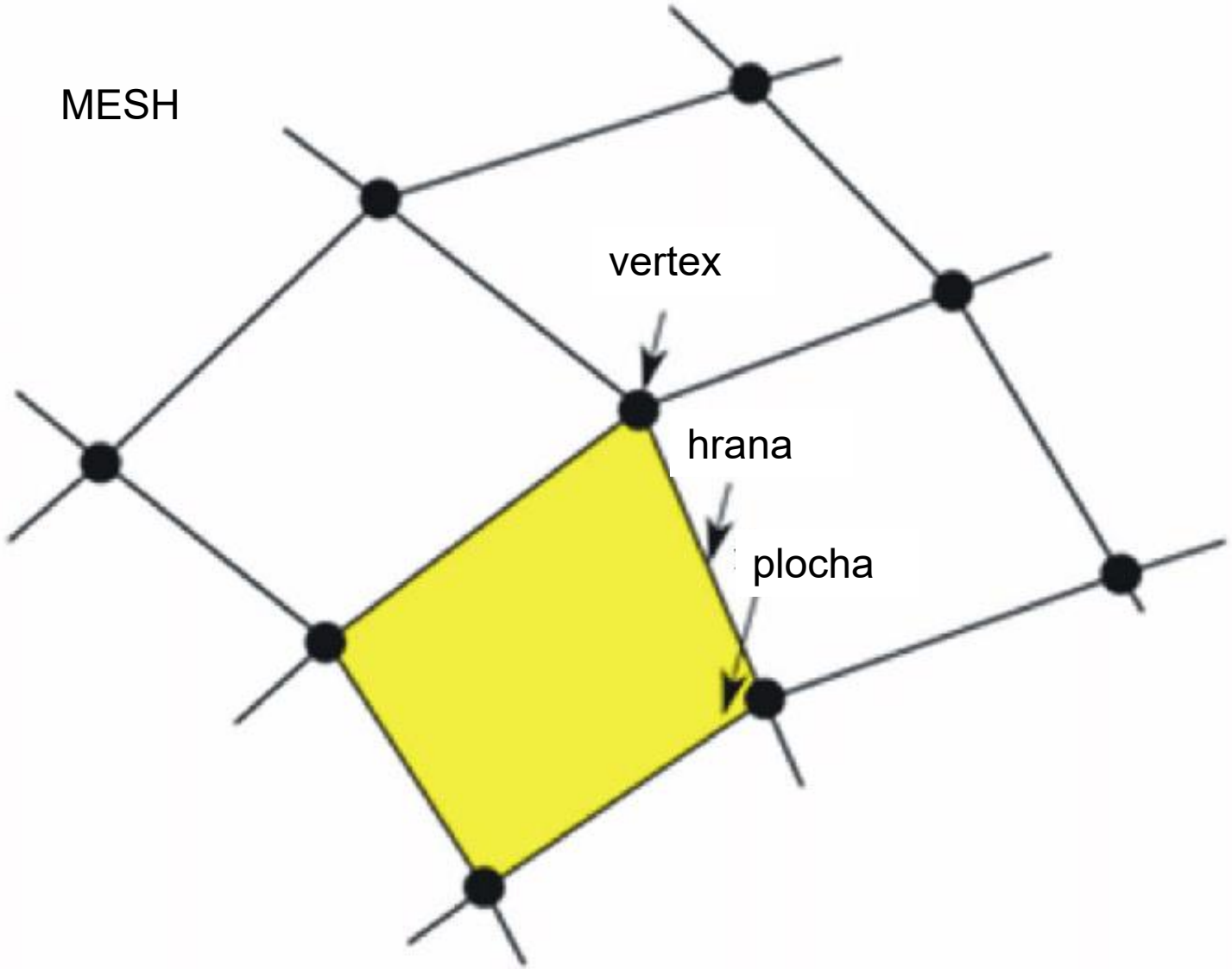
Základní pojmy

- **Vertex** – bod v prostoru, jehož poloha je dána souřadnicemi (x, y, z) Vzhledem k jeho singularitě (absenci rozměrů) nemá smysl u něj kromě souřadnic zaznamenávat další informace (např. barvu, průhlednost, odraz světla…) Jedná se vlastně o „Uzlový bod“ (neboli vrchol dvou dotýkajících se hran nějakého objektu)

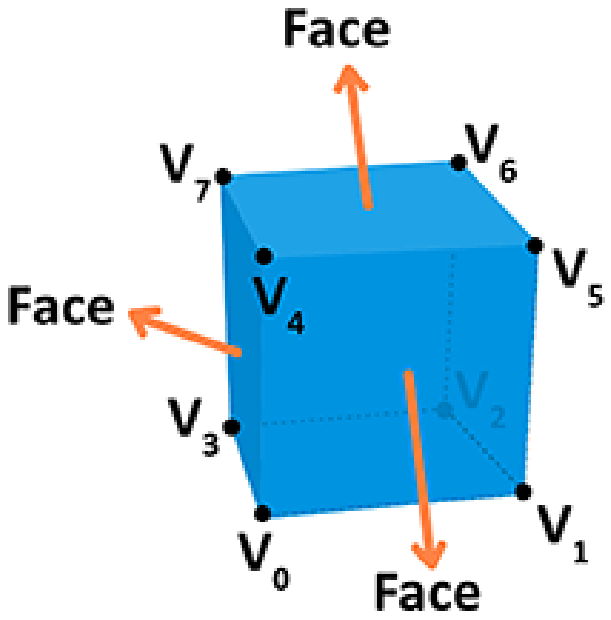
- **Polygon** - je nejjednodušší prostorové těleso. Podmínkou toho, aby prostorové těleso mohlo být považováno za polygon, je třeba, aby bylo tvořeno nejméně třemi vertexy (vrcholy) a 3 úsečkami (hranami)

- **Mesh** – Síť, která tvoří povrch 3D modelů. Je tvořena vertexy a plochami polygonů

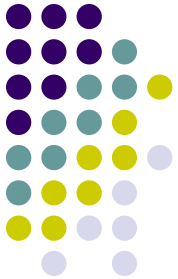
3D akcelerace



3D akcelerace

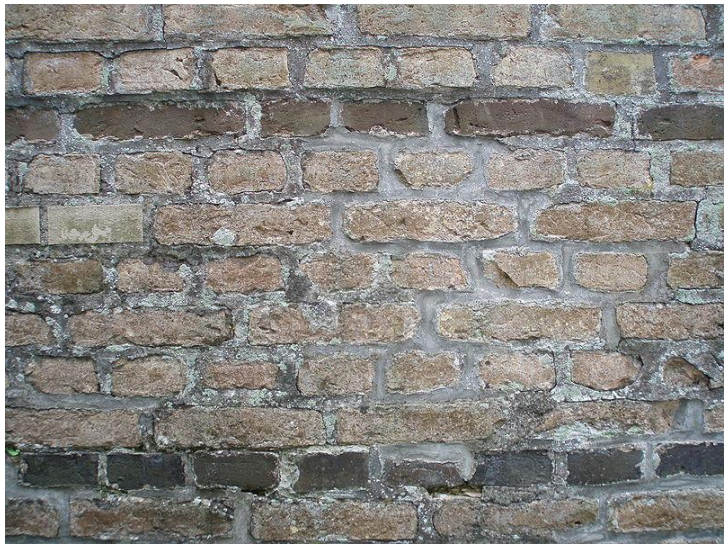


3D akcelerace

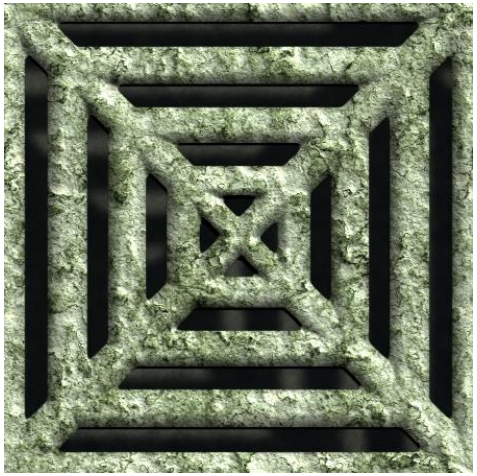


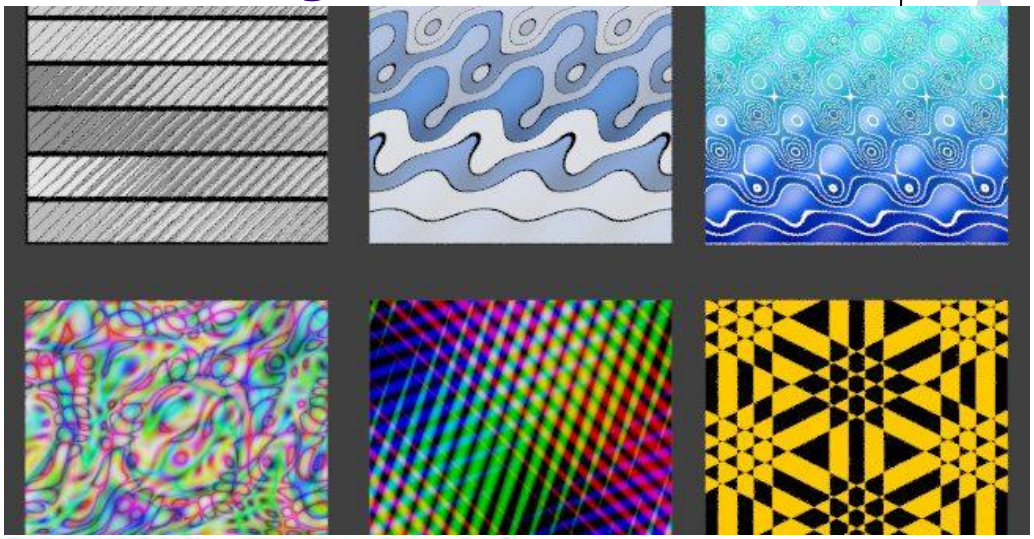
- **Textura** – **Povrch** 3D modelu. Díky textuře dostávají vytvářené objekty i prostředí, do kterého jsou umístěny, autentickou podobu.
- Textura je **2D rastrový obrázek** složený z pixelů, který se nanese na plochu v 3D prostoru
- Některé textury lze definovat i matematicky – **procedurální textura**. Taková textura pak není uložena jako 2D obrázek, ale jako matematický vzorec (např. fraktál)
- Proces nanášení textur provádí jednotka zvaná **pixel shader**
- Při aplikaci každé textury na 3D mesh je třeba definovat, jak velká má být daná textura na vybraných polygonech (měřítko) a jakou má orientaci a rotaci

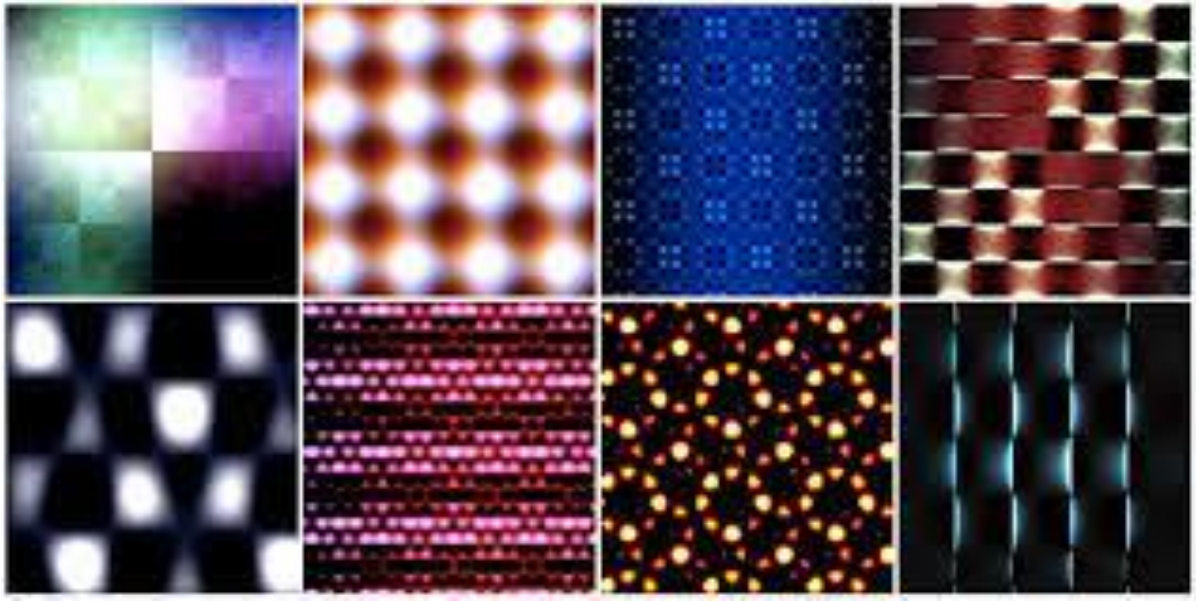
Rastrové textury



Procedurální textury



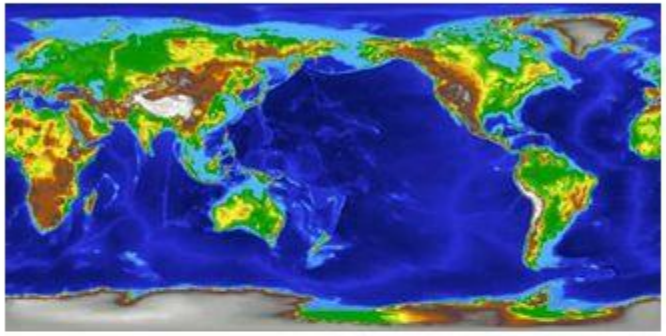




Texture Mapping

Object

+



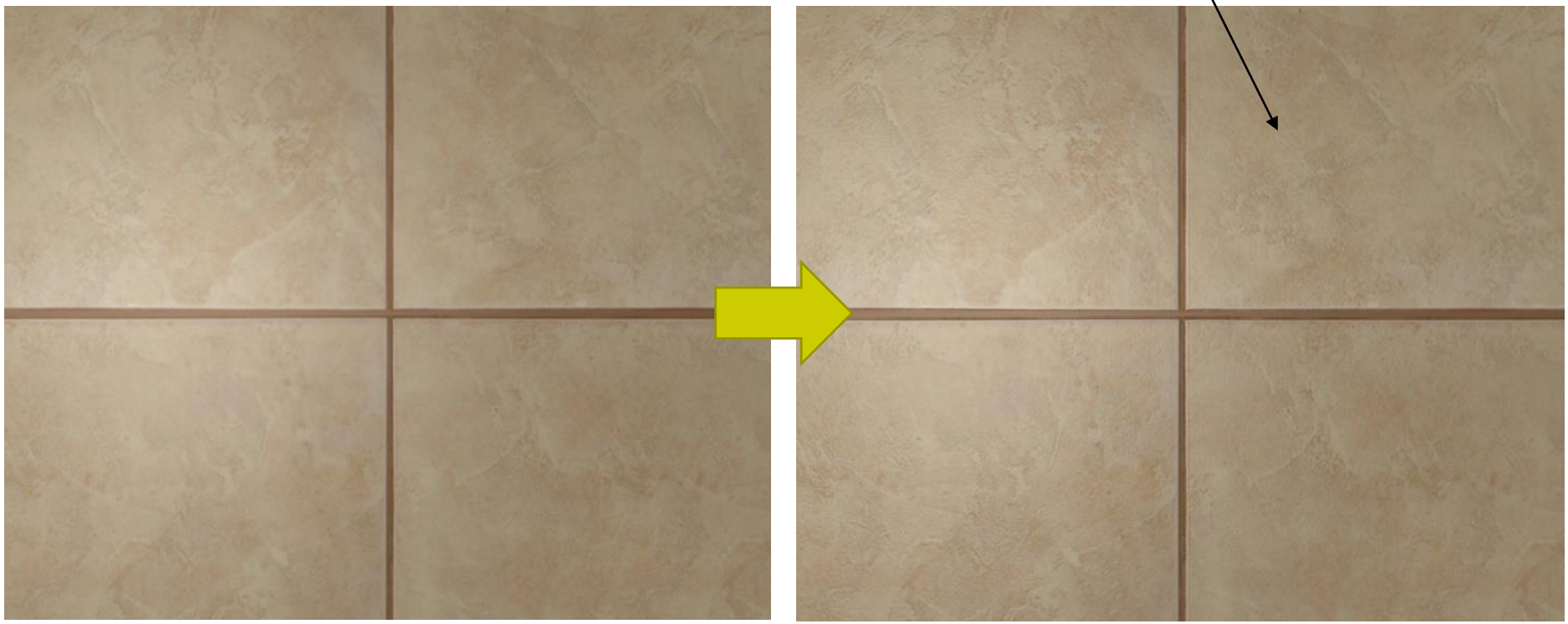
Texture

=

Texture Mapped Object

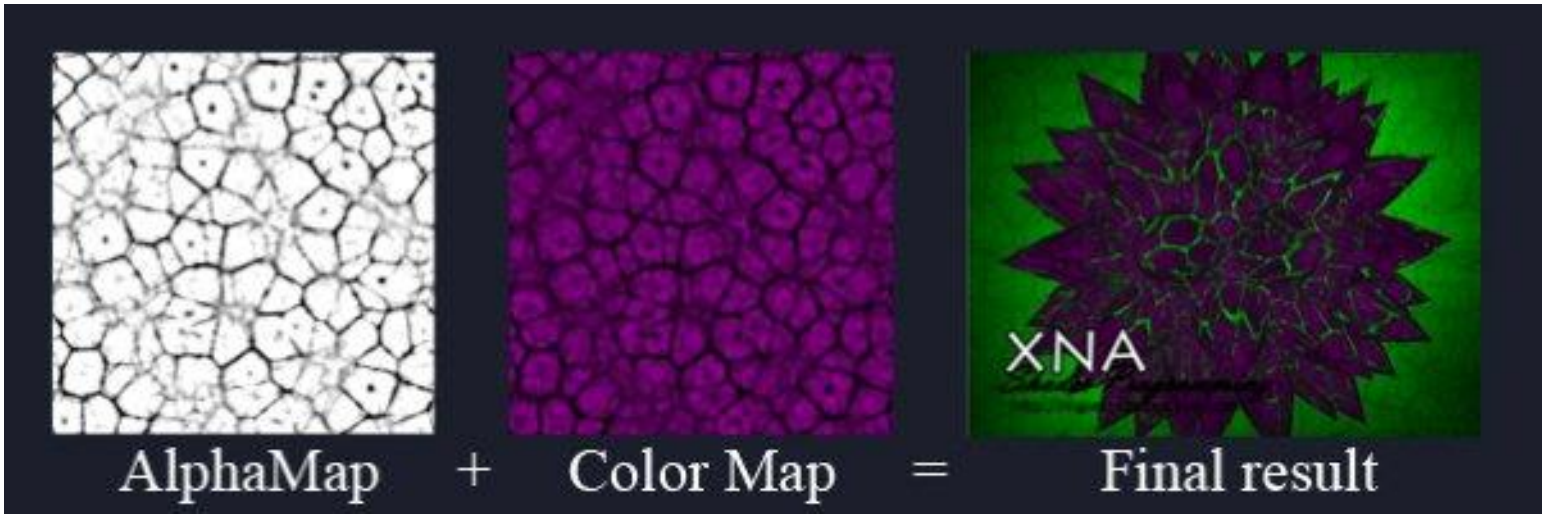
3D akcelerace

- **Bump mapping** je technika texturování, která vytváří iluzi nerovnosti povrchu bez změny jeho geometrie
- Pro uložení informace o hrbolatosti povrchu se používá **bump mapa**



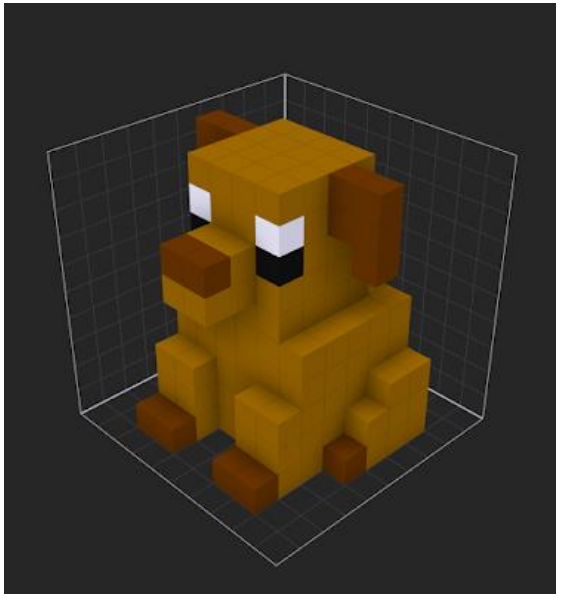
3D akcelerace

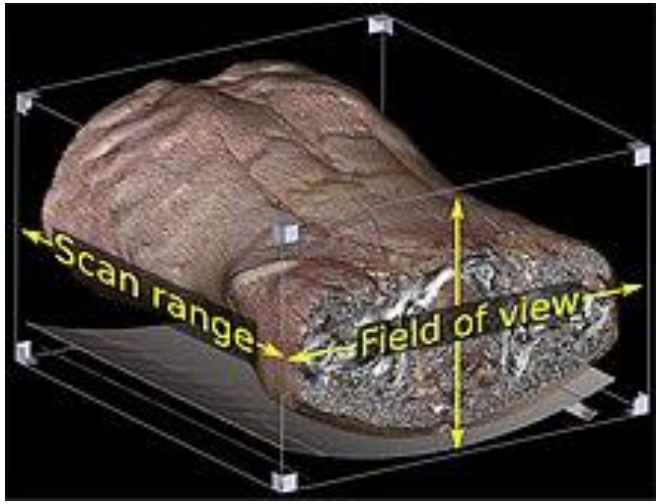
- **Alpha mapping**– K informacím o barvě pixelu (RGB) je přidán ještě A - „alfa kanál“ určující hodnotu průhlednosti. Nižší hodnota znamená vyšší průhlednost. Textura tedy může být v různých místech různě průhledná



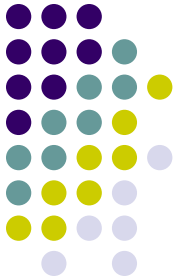
Základní pojmy

- 3D grafiku lze dělat také úplně jiným způsobem
- Objekty lze poskládat z „kostiček“ – něco jako Minecraft
- **Voxel** – objemový element – v podstatě prostorový pixel, který má objem
- Tímto způsobem se pracuje například v lekářské 3D grafice
- Magnetická rezonance nebo jiné zobrazovací metody nepoužívají vertexy a polygony, ale voxely
- Použití voxelu má smysl tam, kde potřebujeme vidět „dovnitř“
- Voxely mohou mít různé vlastnosti. Nejen barvu, ale například mohou reprezentovat hustotu tkání.





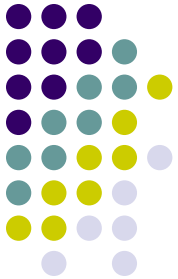
3D akcelerátor



- Činnost jednoduchých 3D akcelerátoru lze ve zkratce popsat takto
  - Vypočítají obrazová data snímku jako promítnutí plošných trojúhelníků (polygonů)
  - Tyto trojúhelníky mohou být vyplněné buď konstantní barvou, barevným přechodem nebo texturou
  - Scénu je možné nasvítit více zdroji světla

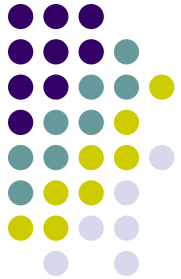
  - Příklad prvních jednoduchých 3D akcelerátorů
    - **3Dfx Voodoo Graphics**
    - **Matrox Mystique**
    - **nVidia Riva 128**

3D akcelerace



- Pokročilejšími operacemi jsou
  - **Osvětlení** několika zdroji světla z různých směrů
  - **Fog** – postupné skrývání objektů v mlze
  - **AntiAliasing** – odstranění artefaktů textury při zmenšování a naklánění
  - Výpočty **trajektorií** a kolizí – jednotlivým objektům lze přiřadit nejen polohu, ale také směr a rychlost pohybu. Grafická karta si pak sama pro jednotlivé snímky počítá novou polohu pohybujících se objektů.

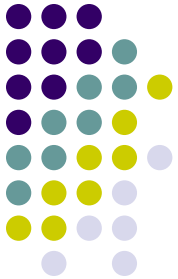
Grafický adapter a základní pojmy



- **GPU** – Graphics processing unit – grafický procesor – výpočetní jednotka grafické karty
- Moderní grafický adapter je v podstatě samostatnou výpočetní jednotkou, která zpracovává obrazová data a k tomuto zpracování má přizpůsobenou **speciální architekturu**
- Výpočet obrazových dat grafickým adapterem je mnohonásobně rychlejší, než kdyby ho prováděl klasický univerzální mikroprocesor

- **ROP** – Render output unit – jednotka odpovědná za generování signálu pro monitor. Zde již neprobíhají žádné výpočty, pouze se grafická data uložená ve videopaměti převádějí na výstupní elektrický signál pro monitor. Nejstarší grafické karty vlastně fungovaly jen jako ROP.

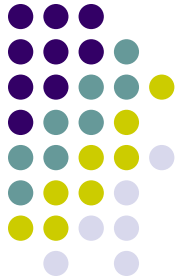
3D akcelerace



- Několik dalších důležitých pojmů

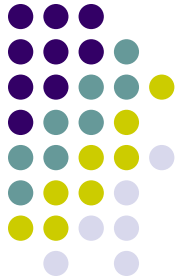
- **Z-buffering**
  Souřadnice **z** udává vzdálenost objektu. Na základě toho jsou vykreslovány jen ty nejbližší objekty. Výpočet pouze blízkých objektů výrazně snižuje náročnost a zvýší FPS
- **Fill rate -** určuje, kolik je karta schopna vygenerovat pixelů za sekundu. Udává se v miliónech za sekundu (Mpx/s)
- **Polygon rate** - určuje, kolik je karta schopna propočítat a vykreslit na obrazovku trojúhelníků (polygonů).

3D akcelerátory



- Všechny funkce, které jsou dnes implementovány v grafických akcelerátorech, je možné vyjádřit či popsat pomocí funkcí dvou nejpoužívanějších rozhraní pro 3D grafiku
  - **OpenGL** spravuje konsorcium *ARB* (*Architecture Review Board*), jehož členy jsou firmy NVIDIA, SGI, Microsoft, AMD
    - Existuje pro všechny počítačové platformy
    - Definuje okolo 250 funkcí a procedur pro vykreslování objektů
    - založeno na architektuře klient-server – program (klient) vydává příkazy, které grafický adaptér (server) vykonává.
  - **DirectX** – vytvořeno firmou Microsoft (jeho součástí jsou i knihovny pro ovládání zvuku)
- To je výhodné pro programátory, kteří mohou jednotně vytvářet aplikace funkční na všech grafických akcelerátorech. Programátor nemusí vytvářet několik různých verzí hry nebo virtuální reality – vše popíše jen jednou.
- Na každé grafické kartě, pak ale může být 3D obraz vygenerován trochu jinak – v jiném rozlišení a jinak rychle podle výkonu a pro výpočty mohou být použity různé algoritmy a snímky se mohou v detailech lišit, některé efekty nemusí být podporovány vůbec
- Pokud není nějaký grafický efekt a výpočet přímo podporován na akcelerátoru, je zvolen jeho pomalejší programový výpočet mikroprocesorem nebo se efekt vůbec nepoužije

3D akcelerátory



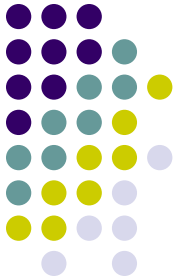
- **Vulkan** je nový standard (používá se od roku 2016) pro 3D grafiku vyvíjený skupinou Khronos Group jako nástupce OpenGL, se kterým není zpětně kompatibilní
- Vulkan dokáže lépe využít moderní vícejádrové grafické čipy než OpenGL
- OpenGL zatím nekončí, ale moc nových vlastností už by přibývat nemělo.
- Hry a aplikace v OpenGL pravděpodobně budou programovány dále, protože s Vulkanem si poradí jen nový hardware.
- Lze předpokládat, že přechod z OpenGL na Vulkan bude pozvolný.

3D akcelerátor

- Další generace 3D akcelerátoru přidávaly další nové možnosti
  - **Zvýšení výpočetního výkonu**
  - **Multitexturování** – kombinace více textur na stejné ploše
  - Zavedení **TMU** (jednotka pro uložení textur a výpočet jejich mapování)

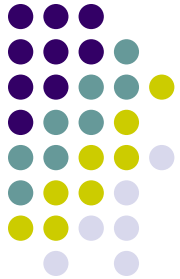
- Příklad „druhé generace“ 3D akcelerátorů
  - Riva TNT
  - ATI Rage 128
  - Matrox G400

3D akcelerátor



- Další generace akcelerátoru přicházejí se stále složitějšími algoritmy a obrazovými efekty
  - **paralelizaci** co největšího množství algoritmů
  - **fog** – skrývání objektů v mlze
  - **Per pixel lighting** – výpočet osvětlení každého jednotlivého pixelu
  - **shadow mapping** - výpočet vržených stínů
  - **accumulation buffer** – výpočet „mezisnímků“ pro podporu efektů **Motion Blur** (pohybující se předměty jsou rozmazané ve směru pohybu) a **Depth o field** (popředí nebo pozadí, na které není zaostřen pohled, je rozmazané)
  - **environment mapping** – výpočet vzhledu reflexivního povrchu (např. na lesklé kapotě auta se zrcadlí okolní prostředí)
  - **display list** – zavedení „podprogramů“, které se vykreslují opakující se části scény, ale při opakovaném volání je není třeba znovu celé provést a vypočítat.
  - Podpora **komprimace a dekomprimace textur** v reálném čase. Umožňuje za cenu ušetřit až 75 % kapacity texturovací paměti (za cenu drobné ztráty kvality textury)
  - Dva nové funkční bloky **T&L** (**transform and lighting**)

3D akcelerace

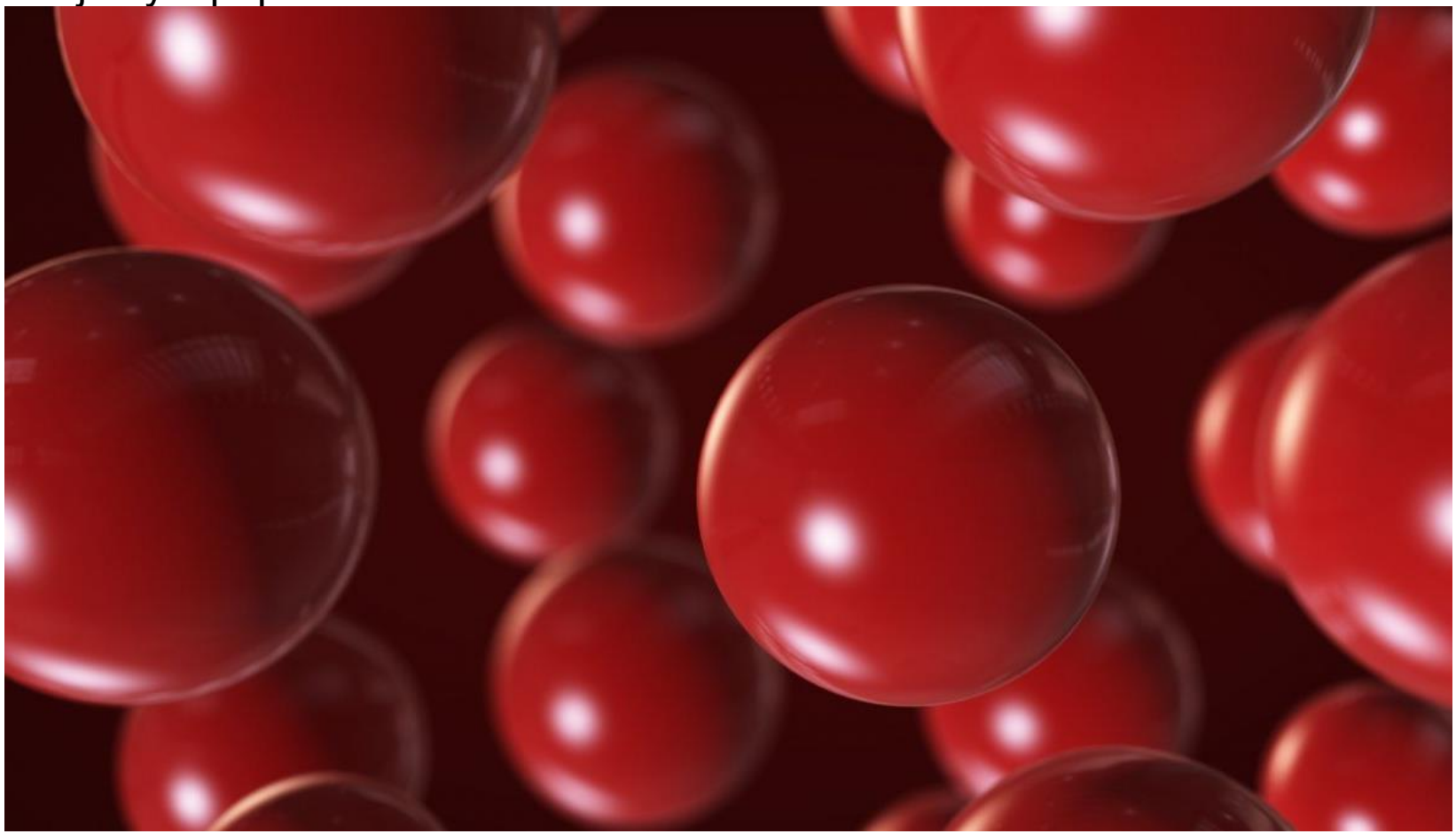


- ukázka efektu motion blur
- https://www.youtube.com/watch?v=6zhTNYY8ehM

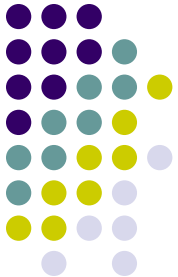
- Pokud jsou pohybující se objekty rozmazané, působí jejich pohyb plynulejším dojmem i při nižší snímkové frekvenci

3D akcelerace

- Ukázka „depth of field“ efektu – snížená hloubka ostrosti. Ostré jsou pouze objekty v popředí

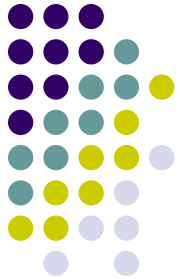


Blok transform



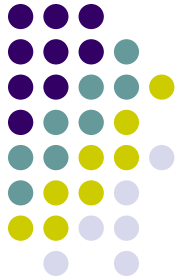
- V bloku **Transform** jsou prováděny transformace vertexů a hran a odstranění (**culling** a **clipping**) neviditelných nebo odvrácených polygonů před jejich vykreslením
- Nejdříve je třeba transformovat polygony ze systému světových souřadnic (**world coordinates**) do souřadnic pozorovatele – tato transformace se nazývá pohledová transformace (**view transformation**)
- Dále se provádí transformace ze souřadnic pozorovatele do 2D souřadnic na obrazovce (**screen coordinates**) - této transformaci se říká projekce nebo promítání (**projection**).
- Všechny popsané transformace lze převést na poměrně jednoduše realizovatelné operace násobení dvojice matic
- Prováděné operace jsou poměrně jednoduše realizovatelné a dají se velmi dobře provádět proudově (pipelining) a paralelně s větším objemem nezávislých dat
- Výpočty neobsahují **žádné větvení** (žádné podmíněné skoky)
- Tyto transformace je možné provádět také přímo pomocí hlavního procesoru počítače s využitím specializovaných instrukcí MMX, 3D Now! a SSE, ale vzhledem k velkému množství vrcholů u běžných 3D scén by se jednalo o značné zatížení
- Proto se většina výpočtů, kde se neřeší složité rozhodování a neprovádějí cykly, realizuje přímo na grafickém akcelerátoru.

Blok lighting



- Funkční blok provádějící výpočet osvětlení jednotlivých polygonů
- Výpočty se podle požadované kvality zobrazení provádějí buď pro každý vrchol (**Per-Vertex Lighting**) polygonu, nebo pro každý vykreslovaný pixel (**Per-Pixel Lighting**)
- Základní **vlastnosti světelných zdrojů**:
  - pozice v prostoru
  - orientace světelného kuželu(pouze u směrového a reflektorového světla)
  - koeficienty ambientní, difúzní a odrazové složky světla
- Základní **vlastnosti povrchu** (materiálů, textury) jsou popsány koeficienty ambientní, difúzní a odrazové složky
- Výpočet osvětlení je založen na **skalárním součinů vektorů** (vektor směřující ke světelnému zdroji s normálovým vektorem a vektor ideálně odraženého světelného paprsku s vektorem orientovaným směrem k pozorovateli)

3D akcelerace

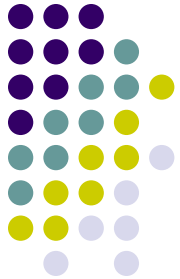


- Dalším stupněm vývoje grafických karet s 3D akcelerací jsou **programovatelné akcelerátory**
- První generace programovatelných grafických procesorů obsahovala operace pro úpravu vykreslovaných pixelů načítaných z textury
- čipy **nVidia GeForce 1** a **nVidia GeForce 2** obsahovaly takzvané **register-combiners**, pomocí nichž bylo možné naprogramovat jednoduché rastrové operace

- Funkce moderního grafického akcelerátoru, který je programovatelný, je možné rozdělit na dvě části.
  - jednotka, která se zabývá transformací geometrických (vektorových) souřadnic.
  - jednotka provádějící rasterizaci – převod na snímek složený z pixelů

- Možnosti vektorových i rastrových bloků grafických karet byly postupně rozšířeny k plné programovatelnosti pomocí programů, které lze do jejího konkrétního grafického procesoru nahrávat

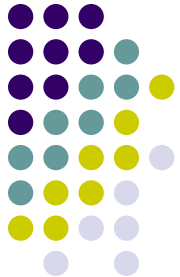
Programovatelné akcelerátory



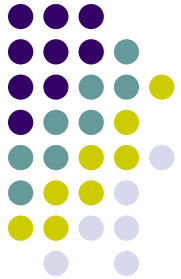
- Všechny dříve popsané typy grafických akcelerátorů mají jednu společnou vlastnost: ke komunikaci s nimi sloužila určitá programátorská rozhraní (**API**) poskytující neměnnou sadu funkcí, které lze využívat
- Nebylo však možné funkčnost grafických akcelerátorů libovolně programově měnit nebo rozšiřovat jejich schopnosti (přidávat nové vykreslovací funkce, které si popíšete vlastním programem)
- Moderní grafické karty již obsahují programovatelné jednotky, do nichž lze zadat vlastní programy
- Podle typu zpracovávaných dat je možné rozlišit dvě programovatelné jednotky grafických karet
  - **vertex shader -** vertexový procesor, se zabývá transformací geometrických (vektorových) souřadnic
  - **pixel shader -** rasterizační procesor, provádí vlastní vykreslování fragmentů

- Jednotka může pracovat ve dvou režimech:
  - **Programovatelný režim** - grafický procesor se ovládá pomocí externě nahraného programu
  - **Pevný režim** - předem nastavený výrobcem grafického akcelerátoru.

Shader

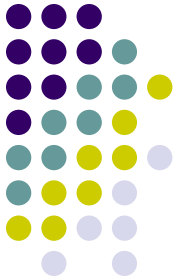


- **shadery** - programy, které slouží k ovlivnění vykreslování scény úpravou základních vykreslovacích algoritmů
- Shadery se v renderovacích programech zapisují pomocí assembleru nebo vyššího jazyka (Cg, HLSL, GL2), který je podobný jazyku C
- Obvyklejší je programování v některém z vyšších jazyků, což zaručuje lepši přenositelnost na různé typy grafických akcelerátorů
- Program napsaný v některém z vyšších jazyků kompilátor převede do „strojového kódu“, který je posléze uložen do paměti grafického akcelerátoru a spuštěn.



- Ukázka programu v assembleru pro vertex shader

```
DP3 R0, c[11].xyzx, c[11].xyzx;
RSQ R0, R0.x;
MUL R0, R0.x, c[11].xyzx;
MOV R1, c[3];
MUL R1, R1.x, c[0].xyzx;
DP3 R2, R1.xyzx, R1.xyzx;
RSQ R2, R2.x;
MUL R1, R2.x, R1.xyzx;
ADD R2, R0.xyzx, R1.xyzx;
```

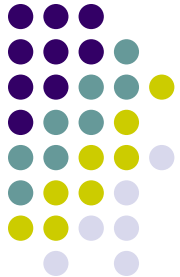


- Ukázka programu v jazyce Cg pro vertex shader

```
struct appdata { float4 position : POSITION;
float3 normal : NORMAL;
float3 color : DIFFUSE;
float3 VertexColor : SPECULAR; };
struct vfconn
{ float4 HPOS : POSITION; float4 COL0 : COLOR0; };

vfconn main(appdata IN, uniform float4 Kd, uniform
float4x4 ModelViewProj)
{ vfconn OUT;
mul OUT.HPOS = mul(ModelViewProj, IN.position);
OUT.COL0.xyz = Kd.xyz * IN.VertexColor.xyz;
OUT.COL0.w = 1.0; return OUT; }
```

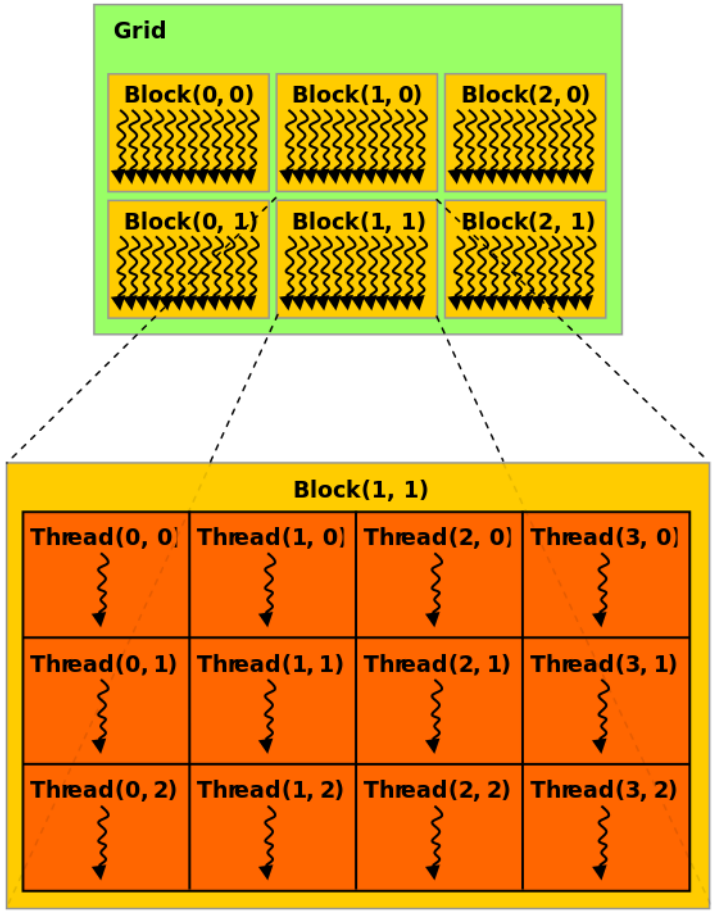
GPGPU



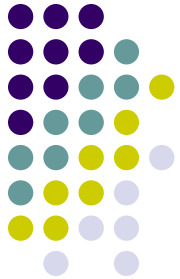
- **General-purpose computing on graphics processing units** je způsob využití paralelizace na grafické kartě k výpočtu libovolných algoritmů
- S příchodem programovatelných akcelerátorů se vyskytla možnost běžných výpočtů prováděných grafickou kartou.
- Výpočty na grafické kartě jsou vhodné u algoritmů, které mají podobný formát jako transformace grafických dat a obsahují mezi sebou minimální nebo žádné vazby
- Grafický procesor (GPU) má značně rozdílnou architekturu na rozdíl od klasického mikroprocesoru, který má rozsáhlou instrukční sadu uzpůsobenou k realizaci univerzálních výpočtů.
- Architektura grafického čipu byla vyvíjena s myšlenkou realizace malého množství specifických instrukcí paralelně s velkou datovou sadou
- Grafické karty mohu použity k urychlení obecných masivně paralelních algoritmů.
- Příklady vhodných aplikací pro urychlování na GPU mohou být algoritmy:
  - Počítačového vidění, analýza obrazu umělou inteligencí
  - Simulace neuronové sítě
  - Fyzikální simulace (simulace chování částic, proudění kapaliny, plynu apod.)
  - Kryptografické výpočty

CUDA

- **C**ompute **U**nified **D**evice **A**rchitecture
- umožňuje na GPU od firmy nVidia spouštět programy napsané v jazycích C/C++, Fortran nebo programy postavené na technologiích OpenCL
- Technologii CUDA představila společnost nVIDIA v roce 2006
- Drtivou většinu plochy čipu grafického akcelerátoru od nVidie zabírá velké množství relativně jednoduchých skalárních procesorů, které jsou organizovány do větších celků zvaných streaming multiprocesory
- jedná o SIMT architekturu (Single instruction multiple threads) – určité množství výpočetní jednotek vykonává mnohem větší počet vláken, která jsou tvořena sledem SIMD instrukcí
- Vlákna jsou organizována do 1D, 2D nebo 3D **bloků**, kde vlákna ve stejném bloku mohou sdílet data a lze synchronizovat jejich běh. Počet vláken na jeden blok je závislý na výpočetních možnostech zařízení
- Každý blok vláken musí být schopen pracovat nezávisle na ostatních, aby byla umožněna škálovatelnost systému (na GPU s více jádry půjde spustit více bloků paralelně oproti GPU s méně jádry, kde bloky poběží v sérii)
- **Bloky** jsou organizovány do 1D, 2D nebo 3D **mřížky**.
- Balík vláken zpracovávaných v jednom okamžiku se nazývá **warp**. Jeho velikost je závislá na počtu výpočetních jednotek.



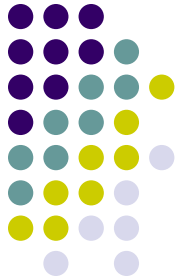
CUDA



- Díky CUDA lze výpočetní výkon grafické karty využít i k jiným než grafickým výpočtům
- Grafická karta může dnes provádět paralelně tisíce vláken, přičemž v každém z nich se provádí SIMD instrukce s mnoha daty současně
- Pokud jde o výpočet, který lze dobře rozdělit na vlákna, ve kterém se shora dolů (bez větvení) řeší velké množství stejných výpočtů s různými daty, může být výpočetní výkon grafické karty mnohonásobně vyšší než výkon běžného mikroprocesoru
- Tohoto se dnes využívá například při těžbě kryptoměn

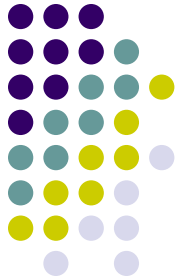
- CUDA nefunguje na GPU firmy AMD, které jsou tvořeny mnoha VLIW SIMD jednotkami (stream procesory)
- Špičkové grafické karty mají tisíce stream procesorů

OpenCL



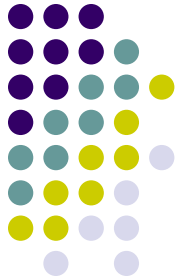
- **OpenCL** (**Open C**omputing **L**anguage) je standard pro paralelní programování heterogenních počítačových systémů (systému s různými paralelně pracujícími výpočetními jednotkami – např CPU + GPU)
- Vývoj spravuje Khronos group
- **OpenCL Framework** poskytuje aplikacím možnost využívat hostitelský systém a jeho zařízení jako heterogenní paralelní stroj.

Paměť a uložení grafických dat v současnosti



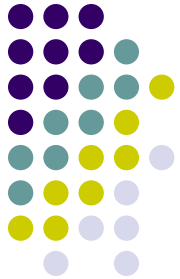
- **System memory –** systémová paměť počítače PC. Tato paměť představuje primární zdroj všech dat a programůpřed přenesením do grafického akcelerátoru. Grafický akcelerátor do ní může přistupovat přes DMA
- **Video memory** – paměť, která je umístěna přímo na grafickém akcelerátoru a obsahuje:
  - **Geometry –** v tomto bloku jsou geometrická data těles. Nazývá se také vertex-buffer. Z tohoto bloku se při vykreslování posílají geometrické informace do grafického čipu, kde jsou dále transformovány a rasterizovány.
  - **Textures** – uložení rastrových textur použitých ve scéně. Pokud je kapacita video paměti pro uložení textur nedostatečná, je možné textury při vykreslování postupně přenášet z operační paměti počítače, ale dojde ke ztrátě rychlosti
  - **Framebuffer** – sem ukládá grafický procesor jednotlivé vykreslované fragmenty
    - **color buffer –** část paměti, která může být vykreslena na obrazovku.
    - **depth buffer, stencil buffer** - nejsou pro uživatele přímo viditelné, ale slouží k realizaci různých grafických algoritmů
  - **Commands** – v tomto bloku video paměti jsou uloženy programy pro **vertex shader** a **pixel shader**.
    - Programů zde může být uloženo více a grafický procesor se mezi nimi může při vykreslování různých grafických efektů přepínat.
    - Délka jednotlivých programů dosahuje až stovky instrukcí.

Grafická karta - paměť



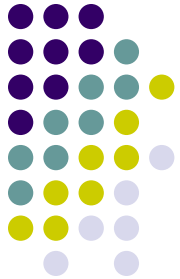
- Grafická karta využívá svou vlastní paměť, jejíž rychlost zásadně ovlivňuje její výkon.
- Používají se paměti typu **GDDR** = Graphics DDR s kapacitou v řádu jednotek GB (obvykle 2 -16 GB)
- Čím větší je kapacita GDDR paměti, tím komplexnější grafická data do ní lze uložit (více podrobných textur, více připravených snímků, více geometrických dat objektů)
- Vyšší frekvence paměti umožňuje rychlejší přesun grafických dat
- Novou kategorii tvoří vrstvené paměti HBM, které jsou integrovány přímo do grafického chipu. Zatím jsou velmi drahé

Grafická karta - paměť



Typ	Frekvence	Přístupy/s	Propustnost	
GDDR2	500 MHz	2 GT/s	128 Gbit/s	16.0 GB/s
GDDR3	625 MHz	2.5 GT/s	159 Gbit/s	19.9 GB/s
GDDR4	275 MHz	2.2 GT/s	140.8 Gbit/s	17.6 GB/s
GDDR5	625–1000 MHz	5–8 GT/s	320–512 Gbit/s	40–64 GB/s
GDDR5X	625–875 MHz	10–12 GT/s	640–896 Gbit/s	80–112 GB/s
GDDR6	875–1000 MHz	14–16 GT/s	896–1024 Gbit/s	112–128 GB/s

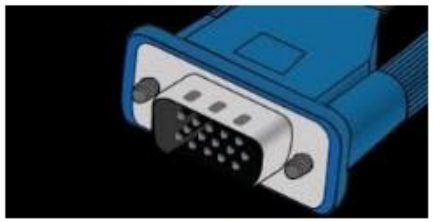
Grafická karta – přetaktování a chlazení

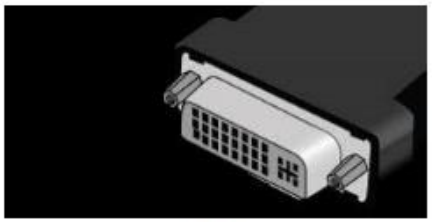


- GPU grafické karty má výrobcem nastavenou nominální taktovací frekvenci
- Tato frekvence je nastavena tak, aby instalované chlazení zvládalo s rezervou kartu chladit
- Pokud se provádí pouze 2D akcelerace (např. přehrávání videa, posuv obrazu apod.), snižují grafické karty frekvenci a napětí GPU i pamětí a automaticky se podtaktují
- **Automatické přetaktování (Boost)** se používá při požadavku na maximální výkon, kdy si grafická karta navyšuje frekvenci jádra sama.
- Většinu grafických karet lze přetaktovat pomocí aplikací přímo od výrobce nebo jiných specializovaných aplikací (ty obvykle dovolí více, ale nemusí to již být stabilní a vyžadují od uživatele odborné znalosti)
- **TDP – Thermal design power** je maximální spotřeba karty při provozu v reálných podmínkách, například při hraní her, zpracování 3D grafiky a podobně. Podle ní výrobci dimenzují chlazení.
- Méně výkonné grafické karty vystačí s pasivním chlazením (bez větráčku), které je tiché
- Středně výkonné grafické karty mají aktivní chlazení vlastním větráčkem. Ten obvykle není zapnutý stále, ale jen při zátěži a otáčky jsou obvykle regulovány podle teploty a hlučnost je pak kolísavá.
- Nejnáročnější grafické karty mohou vyžadovat vodní chlazení. Voda odvádí teplo 23x lépe než vzduch, ale takové chlazení je drahé, prostorově náročné a komplikované na instalaci.

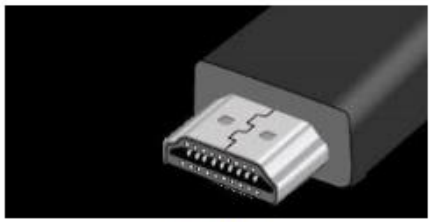
Grafické karty - připojení

- Moderní grafické karty se v počítači připojují do slotu sběrnice **PCI-E x16** – většina dnes již vyžaduje **PCI-E 3.0**, se starší verzí sběrnice karta obvykle funguje také, ale může podstatně klesnout její výkon
- Méně výkonné karty se napájí přímo přes tuto sběrnici
- Výkonnější karty většinou vyžadují přídavné napájení (12V) přímo ze zdroje.
- V některých méně výkonných počítačových sestavách se s instalací grafické karty nepočítá a nebude možná z prostorových důvodů, zdroj nebude disponovat potřebnou napájecí linkou, nebo na zákl. desce zcela chybí slot PCI-E
- Konektory a linky pro připojení monitoru a přenos obrazového signálu jsme již probrali dříve

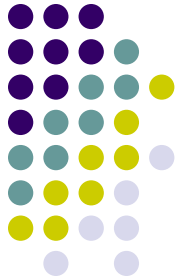

VGA


DVI


DisplayPort


HDMI

Výpočetní výkon GPU



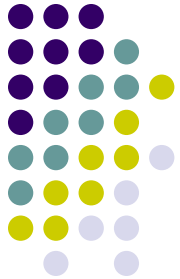
- Výpočetní výkon grafických karet se značně liší
- Porovnávat výpočetní výkon podle udávaných technických parametrů (např. velikost paměti, počet stream procesorů) je nemožné
- Jediným smysluplným způsobem, jak vyjádřit a porovnat výkon grafického adapteru, je použití benchmarku
- **Benchmark** je v podstatě srovnávací zátěžový test
- Grafická karta se podrobí sadě standardizovaných náročných úkolů a měří se při tom, jak rychle je zvládá, za což jí je přiděleno skóre
- Čím vyšší **skóre**, tím vyššího výkonu karta dosahuje
- Nejčastěji používaným nástrojem pro měření výpočetního výkonu GPU je dnes **3DMark**
- Výsledné skóre nezáleží jen na GPU, ale je ovlivněno celkovou konfigurací počítačové sestavy

Výpočetní výkon

- Ukázka – srovnání **3D Mark Score** a cen několika běžných středně výkonných karet

Name	MSRP Price	3DMark Score
AMD Radeon R9 390X	$429	4251
AMD Radeon HD 7870	$412	1658
AMD Radeon RX 5700 XT	**$399**	9536
NVIDIA GeForce RTX 3060 Ti	$399	11872
NVIDIA GeForce RTX 2060 SUPER	$399	8828

Další vývoj?



- Moderní vícejádrové mikroprocesory (architektura Intel Core počínaje SandyBridge) obsahují integrovanou jednotku **GPU**
- Smysl má tam, kde nejde o grafický výkon, ale o **efektivitu a nízkou spotřebu a cenu**
- Umístění GPU do mikroprocesoru přineslo snížení spotřeby energie a rozměrů a zefektivnění komunikace (mezi CPU a GPU není pomalá sběrnice)
- Díky technologii **Turbo Boost** lze v případě potřeby vyššího výkonu grafiku přetaktovat na úkor procesoru a naopak

- Počítače IBM PC neměli vysoký výkon, ani výhodnou cenu, ale byl to ale stroj-stavebnice, který umožnil upravit konfiguraci podle potřeb zákazníka. Proto také zvítězil ve světě, který do té doby okupovaly stroje s pevně danou konfigurací
- Dnes se pomalu děje pravý opak
- do procesoru se nastěhovala část chipsetu a teď už i grafická „karta“. Časem se tam určitě objeví i paměťové čipy (nikoliv pouze cache) a třeba i veškerá rozhraní.
- SOC (System-on-a-Chip) bude mít určitě spoustu výhod: menší výrobní náklady, menší latence, menší spotřeba… Jen to už nebude konfigurovatelné zařízení (což byla řadu let základní vlastnost počítačů řady PC)