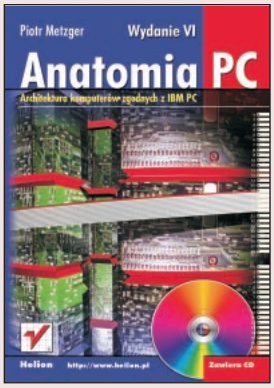
Anatomia PC

Wydanie VI



Autor: Piotr Metzger
ISBN: 83-7197-343-8
Format: B5, stron: 952
oprawa twarda
Zawiera CD-ROM

Książka stanowi próbę całościowego ujęcia architektury komputerów zgodnych z IBM PC. Czytelnik znajdzie tu omówienie następujących tematów:

- podstawowe elementy składowe IBM PC takie jak procesor, pamięć, kanały DMA, mechanizm przerwań, układy odmierzające czas, systemy obsługi urządzeń pamięci masowej, magistralę ISA, interfejs szeregowy i równoległy oraz pamięć konfiguracyjną;
- rozszerzenia architektury w kierunku systemów wieloprocesorowych;
- organizację pamięci operacyjnej i techniki stosowane w pamięciach podręcznych;
- pracę magistral PCI, AGP, SCSI, IDE, USB i IrDA;
- wykorzystanie platformy PC dla przetwarzania obiektów 3D i sygnałów wideo;
- konfigurację systemu poprzez właciwe korzystanie z programu BIOS-SETUP.

Osobny rozdział poświęcony został układom otoczenia procesora (chipset). Przedstawiono nowe typy pamięci (Double Data Rate, High Speed, Virtual Channel i Rambus) oraz system ograniczania zużycia energii (ACPI).
Wydanie szóste uzupełnione zostało o dane aktualnych procesorów, uwzględniono również szereg rozszerzeń wprowadzanych do magistral. W dodatku zebrane zostały adresy internetowe mające związek z zagadnieniami poruszanymi w tej publikacji. Dołączona do książki płytka CD wypełniona jest różnymi ciekawymi programami diagnostycznymi i użytkowymi.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Rozdział 15.
Grafika 3D

Przetwarzanie obrazów 3D przy pomocy komputerów klasy PC stało się jedną z najbardziej dynamicznie rozwijających się dziedzin techniki komputerowej ostatnich czasów. Jego siłą napędową jest spory rynek stosunkowo zasobnych konsumentów, którzy komputer PC wykorzystują głównie do rozrywki (przy założeniu, że gry komputerowe można nazwać rozrywką).

Twórcy podstaw architektury PC nie myśleli chyba o takiej przyszłości dla IBM-PC, który we wstępnych założeniach miał być raczej wysokiej klasy programowanym kalkulatorem z pamięcią. Na szczęście komputer PC daje się rozbudowywać. Braki w zakresie dźwięku i grafiki (a już na pewno tej trójwymiarowej) rekompensuje się przy pomocy różnych kart rozszerzających, na których dominuje architektura właściwa dla danego zastosowania.

Przetwarzanie syntetycznych obrazów 3D, nawet na poziomie jakościowym, sporo oddalonym od sygnału dostarczanego z popularnej kamery VHS, nie jest możliwe bez dodatkowych układów wspomagających. To zadanie spełniają właśnie akceleratory 3D. Czas życia produktów w tej branży jest krótszy niż gdziekolwiek indziej, a konkurencja rozliczna i bezlitosna. Firmy, które opracowują nowe i naprawdę dobre produkty mogą cieszyć się dominacją na rynku zwykle przez krótki okres czasu. Wiele produktów nie może przeżyć fazy beta swoich własnych sterowników programowych. Zanim jeden produkt ugruntuje swoją pozycję, jest wypierany przez inny, często tej samej firmy.

Same parametry techniczne nowych opracowań nie decydują o powodzeniu produktu na rynku. Konieczne jest uwzględnienie aspektów ekonomicznych, bo one znajdują swoje odbicie w cenie produktu. Z tego względu, mimo iż aktualny stan technologii leży często znacznie wyżej, producenci zmuszeni są do wprowadzania różnorodnych ograniczeń i wybierania kompromisów. Część z tych uproszczeń nie ma w istocie dużego wpływu na jakość postrzeganych obrazów, bowiem zmysł wzroku człowieka nie jest kamerą cyfrową i reaguje na barwne obrazy w bardzo specyficzny sposób. Istniejące naturalne bariery (szczególnie te dotyczące szybkości obróbki i prezentacji obiektów trójwymiarowych) stanowią nadal gwarancję, iż komputer klasy PC długo jeszcze nie będzie roboczą stacją graficzną.

Spory wkład w usytuowanie produktów na rynku mają działy marketingu. Wymyślne nazwy handlowe akceleratorów 3D mają zachęcić potencjalnych klientów do kupna. Ulotki reklamowe nafaszerowane są dobrze brzmiącymi pojęciami, które mało kto rozumie. Wiele z nich zostanie wyjaśnionych w tym rozdziale.

Schemat przetwarzania obiektów 3D

Możliwość podziwiania na ekranie monitora obiektów w formie 3D była do niedawna zarezerwowana wyłącznie dla użytkowników programów typu CAD, uruchamianych na silnych stacjach graficznych. Potrzeba symulacji płynnego ruchu nie jest w tym przypadku zbyt paląca. Dla potrzeb CAD wystarczyło zwykle, jeśli obiekt można było dowolnie obracać i przesuwać. Częstotliwość projekcji leży w takim wypadku zakresie 5 fps. Dużo ważniejsza jest natomiast wysoka dokładność, na potrzeby której przeznacza się znaczną część mocy obliczeniowej. Bryły 3D przedstawiane były w programach tego typu na ogół jako modele siatkowe, co potęgowało wrażenie 3D. Powierzchnia zewnętrzna bryły była jednorodna, a starsze wersje programów nie zawsze uwzględniały nawet oświetlenie[1].

Wśród profesjonalnych użytkowników CAD utrwalił się pogląd, iż próby przeniesienia tych aplikacji na platformę komputerów PC oznaczają w najlepszym razie inwestycję w kartę graficzną droższą od całej reszty zestawu. Ogromny postęp, który dokonał się w ostatnim okresie w dziedzinie akceleratorów graficznych i procesorów, oraz stała modernizacja samej architektury PC pozwalają jednak na weryfikację takich poglądów. Wiecznie nienasycony rynek użytkowników gier komputerowych stanowi główny motor postępu w tej dziedzinie. Ostatnio dołączają do nich pierwsze nieśmiałe adaptacje profesjonalnych symulatorów, osadzające użytkownika w mniej lub bardziej wiarygodnym sztucznym świecie (*Virtual Reality*). Zapotrzebowanie na moc obliczeniową przewyższa tu jeszcze nadal możliwości oferowane przez technikę. Wymagana prędkość wyświetlania sięga w profesjonalnych systemach VR zakresu 100 fps. Ponieważ projekcja odbywa się tutaj z dwóch projektorów (na lewe i prawe oko oddzielnie), faktyczna prędkość przetwarzania jest dwukrotnie większa.

Dla potrzeb zwykłego użytkownika wystarcza na razie animacja z prędkością powyżej 25 fps, poniżej której zatraca się powoli subiektywne odczucie płynności ruchów. Szybkie gry interakcyjne podnoszą poprzeczkę do wysokości około 60 fps, bowiem odgrywa w nich rolę czas reakcji użytkownika mierzony, jak by nie było, w ułamkach sekund. Rozdzielczość obrazu 3D przekroczyła już barierę 800 × 600 i nadal rośnie. Powyższe parametry projekcji nie stanowią żadnego problemu dla współczesnych akceleratorów, pod warunkiem, iż wyświetlana kompozycja obiektów 3D nie jest zbyt szczegółowa, a horyzont nie sięga zbyt daleko (głębia scenerii). To, co różni produkty dobre od złych, to możliwości dodatkowe, poprawiające stopień wiarygodności scenerii: uwzględnienie oświetlenia, efektów specjalnych (rozbłysków, odblasków) i atmosferycznych (mgła), praca z cieniem, bogactwo tekstur o wysokiej rozdzielczości oraz skuteczne eliminacje

[1] Aktualne wersje oszałamiają użytkownika bogactwem funkcji: cieniowanie, mapowanie, różne modele oświetlenia

zniekształceń perspektywicznych i innych. Niektóre akceleratory najnowszej generacji pracują w systemach równoległych, dublując cały potok przetwarzający (*Dual 3D- Pipeline*) lub przynajmniej procesor teksturujący (*Dual Texture Engine*). Dla zrozumienia tych i innych, często tajemniczo brzmiących pojęć, wymagana jest pewna znajomość zasad konstrukcji i przetwarzania obiektów 3D. Tym właśnie zagadnieniom poświęcono dalszą część tego rozdziału.

Obiekty 3D stanowią pewien fragment przestrzeni ograniczonej powierzchniami o różnym stopniu komplikacji. Jakkolwiek każda taka powierzchnia da się, bez względu na stopień komplikacji, jednoznacznie zdefiniować przy pomocy równań matematycznych odpowiedniego rzędu, ich przetwarzanie pochłonęłoby całą dostępną moc obliczeniową.

Podobnie jak wiele innych problemów i ten daje się uprościć poprzez przybliżenie, którego dokładność można regulować w miarę potrzeb i możliwości. Uproszczenie polega na rozłożeniu każdej z takich powierzchni na odpowiednio dużą (zależną od stopnia dokładności) liczbę wielokątów płaskich. Najprostszym z wielokątów jest oczywiście trójkąt i ten właśnie używany do wspomnianej aproksymacji.

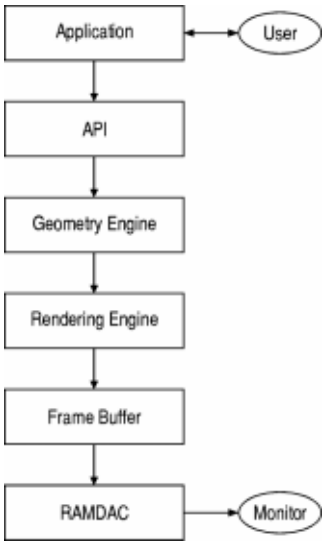
Dowolnie skomplikowana bryła da się zawsze[2] rozłożyć na takie trójkąty. W zależności od stopnia komplikacji i wymaganej dokładności trójkątów takich jest mniej lub więcej: sześcian ma ich 12, a bardzo skomplikowany obiekt w grach komputerowych nawet wiele tysięcy.

Przedstawienie takie ma ogromną zaletę: każdy z wierzchołków trójkąta jest jednoznacznie zawieszony w przestrzeni przy pomocy trójki współrzędnych (x, y, z). Cały obiekt 3D przechowywany jest w spójnym fragmencie pamięci, a wszelkie na nim operacje (przemieszczenie, obrót, skalowanie itp.) sprowadzają się do rachunku macierzowego. Algorytm przesunięcia takiego obiektu jest stosunkowo prosty i da się zapisać w kilku linijkach kodu źródłowego. Prostota algorytmu nie oznacza bynajmniej szybkości działania. W przełożeniu na język maszynowy rachunki takie są bardzo pracochłonne, bowiem mnożenie macierzy to cały szereg operacji mnożenia i dodawania.

Manipulacja obiektami 3D to tylko skromny wycinek pracy, jaką trzeba włożyć w to, by widziany na ekranie obraz mógł zostać być uznany za zbliżony do rzeczywistości Rozłożona na elementarne trójkąty bryła przechodzi skomplikowany proces przekształceń nazywany potokiem przetwarzania 3D.

Ogólna koncepcja scenerii narzucana jest przez aplikację i ewentualnie modelowana zachowaniem użytkownika. Przełożenie tych żądań na język niższego poziomu odbywa się w warstwie API, która stanowi pewien pomost pomiędzy oprogramowaniem użytkowym a sprzętem. W następnym etapie do akcji przystępuje procesor geometryczny (*Geometry Engine*), który rozmieszcza obiekty na ekranie oraz dba o ich aktualne położenie i rozmiary. Po rozłożeniu na wspomniane wcześniej trójkąty elementarne, oddaje je do dalszego przetwarzania już na poziomie punktów stanowiących wnętrze trójkąta (pikseli) do procesora rasteryzującego (*Rendering Engine*). Obliczone punkty obrazu umieszczane są w pamięci karty graficznej (*Frame Buffer*), skąd odczytywane są przez przetwornik (RAM-DAC) i wyświetlane na monitorze.

[2] Aby być w zgodzie z teorią, należy pominąć w tym momencie tzw. powierzchnie fraktalne.



Rysunek 15.1. *Potok przetwarzania 3D*

Poszczególne etapy przetwarzania omówione zostaną oddzielnie w kolejnych punktach.

API

Aplikacje przetwarzające obiekty 3D odwołują się do sprzętu najczęściej poprzez zestaw pewnych funkcji ujętych w biblioteki. Funkcje biblioteczne to metody operujące na obiektach 3D. Typowe operacje to formowanie pojedynczych obiektów, łączenie ich w grupy, manipulacje w przestrzeni oraz zagadnienia związane z samą wizualizacją (np. oświetlenie).

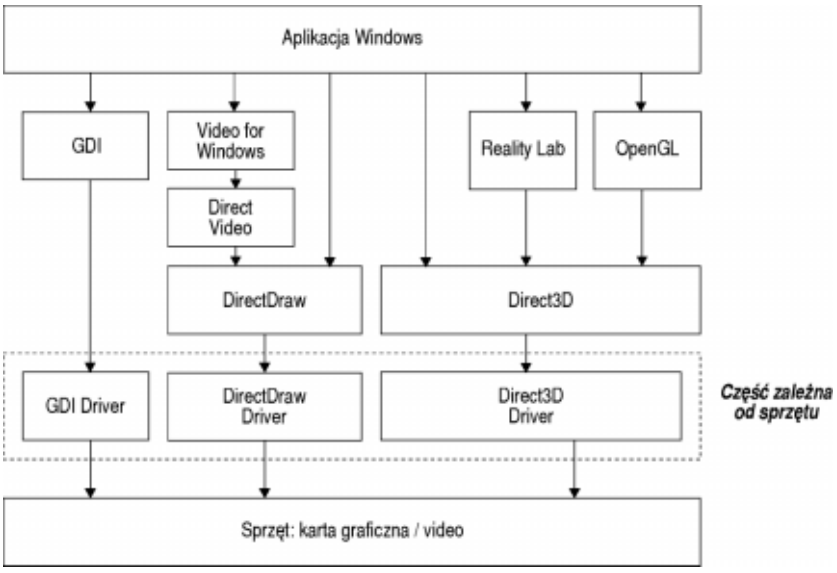
Powyższy zestaw funkcji wraz z definicjami formatów danych stanowi interfejs API (*Application Programming Interface*), oszczędzający programistom znaczną część pracy. Odsunięcie spraw sprzętowych od programowych sprzyja swobodzie myślenia. Producent karty graficznej (akceleratora) wyposaża ją w biblioteki. Są one dostępne dla różnych platform sprzętowych i umożliwiają przez to łatwe przenoszenie oprogramowania. Oto przykłady klasycznych API:

- *GKS (Grafisches Kernsystem).* Wprowadzona do niemieckich norm ISO i DIN w roku 1986, zaprojektowana początkowo dla potrzeb obiektów 2D została poszerzona o funkcje trójwymiarowe (GKS-3D).
- *PHIGS (Programmers Hierarchical Interactive Graphics System).* Stanowi rozwinięcie GKS o interakcyjne funkcje GUI. W dalszym etapie uzupełniona do postaci PHIGS-PLUS (*PHIGS-Plus Lumiere und Surfaces*) oferuje optymalizację cieniowania.
- *GL (Graphics Library).* Niezależnie od rozwoju na rynku europejskim w USA zostaje opracowana dla potrzeb najpopularniejszych tam silnych stacji graficznych (Silicon Graphics Workstations) biblioteka GL. Zalety tej biblioteki zo-

stały docenione również przez wyznawców innych platform sprzętowych, co doprowadziło do uzgodnienia szerszego standardu OpenGL (obecnie już wkomponowany w struktury Windows NT). W ślad za tym rozwinęły się dodatkowe narzędzia wspomagające (*Open Inventory*) wraz z własnym językiem programowania VRML (*Virtual Reality Modeling Language*). W chwili obecnej sterowniki bazujące na interfejsie OIL (*Open Inventor Library*) dołączane są przeważnie do wysokiej klasy akceleratorów 3D.

W dobie dominacji Internetu język VRML stał się nieformalnym standardem w dziedzinie przekazu ruchomych obiektów 3D.

- *SGL biblioteka graficzna PowerVR.* Biblioteka dopasowana jest do układów PowerVR. Opiera się na modelu matematycznym, określanym mianem „infinite planes", gdzie obiekty modeluje się jako wycinki przestrzeni ograniczone półprzestrzeniami i płaszczyznami.
- *Direct3D.* Stanowi część ogólnego systemu DirectX autorstwa firmy Microsoft, która sukcesywnie wprowadza na rynek kolejne odmiany tego interfejsu. W strukturach systemu operacyjnego Windows 2000 zintegrowana została wersja 7.0.
- *Inne biblioteki.* Do grupy tej zaliczyć można opracowania innych firm mocno zaangażowanych w zastosowania 3D. Pod dachem firmy Autodesk powstała biblioteka HOOPS (*Hierarchical Object Oriented Picture System*). Również Spea stworzyła swój własny standard nazwany SP3D. Nowością jest też – opracowywana wspólnie przez SGI i Sun – biblioteka CosmoGL (głównie dla potrzeb 3D-Web)



Rysunek 15.2. *Struktura biblioteki DirectX*

Geometry Engine

Znaczną część potoku przetwarzającego 3D stanowią operacje geometryczne i określenie oświetlenia T&L (*Transformation & Lighting*). Elementy wirtualnej scenerii muszą zostać usytuowane na właściwym miejscu w trójwymiarowej przestrzeni a cały obraz musi zostać rzucony na dwuwymiarową płaszczyznę ekranu. Uwzględnia się przy tym naturalnie położenie obserwatora oraz źródeł światła, co pociąga za sobą całą maszynerię obliczeniową związaną z oświetleniem. Z punktu widzenia matematyki, operacje geometryczne sprowadzają się do różnorodnych manipulacji na współrzędnych punktów.

Jakkolwiek przyjęło się w tym momencie mówić o przetwarzaniu przez procesor geometryczny (*Geometry Engine*), trzeba wyraźnie sprecyzować, który z procesorów ma się na myśli. Do niedawna obliczenia te dokonywała wyłącznie jednostka centralna komputera (CPU). Akceleratory 3D troszczyły się jedynie o rasteryzację pojedynczych trójkątów i umieszczenie ich w pamięci karty graficznej. Procesor musiał więc przygotować parametry wierzchołków trójkątów i przekazać je do dalszego przetwarzania (już w obrębie akceleratora 3D na karcie graficznej). Dopiero akceleratory najnowszej generacji biorą na siebie obliczenia geometryczne oraz część obliczeń przypadających na określenie oświetlenia obiektu.

Pierwsze przeznaczone na rynek masowy akceleratory z funkcją T&L wprowadziła firma nVidia (układy o nazwie G-Force). Wcześniej też istniały akceleratory 3D, ale do zastosowań profesjonalnych, a ich cena skutecznie odstraszała przeciętnego użytkownika. Procesory (a właściwie z punktu widzenia architektury PC – koprocesory) te przejmowały od jednostki centralnej obliczenia geometrii. Pracowały w arytmetyce zmiennoprzecinkowej, a dokładność ich obliczeń przewyższała możliwości oferowanie przez CPU.

Do obowiązków procesora geometrycznego należy pięć podstawowych operacji omówionych w kolejnych punktach.

Tłumaczenie opisu środowiska

W pierwszym kroku procesor analizuje opis scenerii 3D. Sceneria to zestaw obiektów 3D, takich jak budynki, pojazdy itd. podawany w języku wyższego poziomu. Efektem końcowym tej fazy przetwarzania jest rozbicie obiektów na pojedyncze trójkąty elementarne.

Oświetlenie i tekstura

Aby sceneria sprawiała wrażenie naturalnej, trzeba uwzględniać wpływ źródeł światła. Dla każdego z trójkątów określa się więc kierunek i typ oświetlenia oraz przypisuje pewną teksturę. Obliczenia barw obiektów oświetlanych jest skomplikowane bowiem wypadkowa zależy od barwy własnej podłoża i od barwy światła padającego na nią. Zwykle przyjmuje się pewne modele matematyczne (np. model Phonga) upraszające

nieco to zagadnienie. Dokładne określanie barwy wypadkowej dla każdego piksela oddzielnie pochłaniałoby zbyt wiele mocy obliczeniowej. Dla uproszczenia obliczeń dokonuje się dla narożników trójkątów.

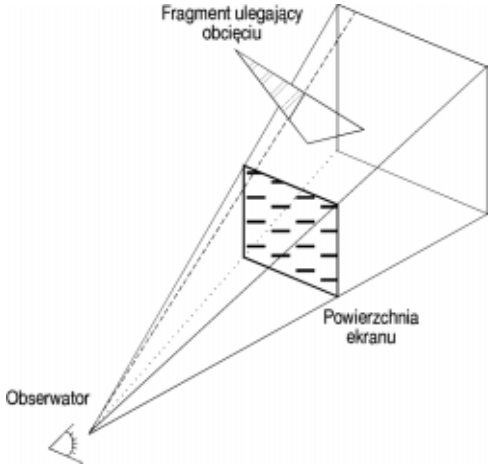
Przekształcenia geometryczne

Przekształcenia geometryczne to wszelkie zmiany w układzie scenerii (przesunięcia i obroty obiektów oraz zmiany ich rozmiarów) obserwowane z punku widzenia domniemanego obserwatora. Zmianie może ulegać zarówno sama scena, jak i również położenie obserwatora. Operacje te sprowadzają się do rachunków macierzowych i wektorowych. Każdy narożnik trójkąta trzeba przemnożyć przez macierz [4 × 4] i to w arytmetyce zmiennoprzecinkowej (FP). Rachunek wektorowy nie jest skomplikowany, ale czasochłonny. Składa się na niego szereg następujących po sobie mnożeń i dodawań (w tym wypadku 16). Aby przekonać się, jak łatwo przy tym sięgnąć kresu rezerw mocy obliczeniowej, wystarczy przeanalizować prosty przykład. Sceneria rozłożona na 2 000 trójkątów to 6 000 wierzchołków. Aby zagwarantować prędkość animacji powyżej 30fps, należy przeprowadzić co najmniej 2 880 000 operacji zmiennoprzecinkowych na sekundę. Dla porównania, wydajność jednostki FP procesora Pentium wynosiła około 4 milionów mnożeń na sekundę.

Strefa widoczności

W wyniku transformacji scenerii (przesunięcia obiektów) część trójkątów (lub ich fragmentów) znika z pola widzenia. Pole widzenia wyznaczane jest przez ściany boczne, tzw. piramidy widoczności, u wierzchołka której umieszczony jest obserwator.

Rysunek 15.3.
Piramida widoczności



Procesor geometryczny nie musi się zajmować trójkątami, które w całości znalazły się poza piramidą. Kłopot sprawiają te, wychodzące na zewnątrz trójkąty, których fragmenty znajdują się w jej wnętrzu. Muszą one zostać obcięte (*Clipping*) na granicy ścian piramidy. W tym celu procesor geometryczny oblicza najpierw współrzędne punktów

przecięcia trójkąta z płaszczyznami ścian. Atrybuty tych nowych punktów (tekstura, kolor) przejmowane są dla uproszczenia od starych wierzchołków, leżących już poza piramidą. Mamy teraz do czynienia ze złożonym wielokątem (rozpiętym pomiędzy czterema punktami), który musi zostać rozłożony na trójkąty elementarne.

Przekazanie parametrów do jednostki rasteryzującej

Gdy wszystkie powyższe kroki zostały przeprowadzone, procesor przechodzi do projekcji trójwymiarowej przestrzeni na płaszczyznę dwuwymiarową. Uwzględniane są przy tym niezbędne skróty perspektywiczne oraz rozmiary obrazu na ekranie. Dalszy etap przetwarzania odbywa się już w procesorze rasteryzującym (*Rendering Engine*) i odnosi się do poszczególnych punktów we wnętrzu trójkąta. W danym cyklu opracowywany jest tylko jeden trójkąt, a procesor rasteryzujący nie dysponuje żadnymi informacjami odnoszącymi się do całej scenerii. Procesor geometryczny przekazuje wyniki swoich obliczeń w formie pakietu zawierającego współrzędne i parametry wierzchołków jednego z trójkątów.

Rendering Engine

Dalszy ciąg potoku przetwarzania 3D odbywa się pod kontrolą procesora rasteryzującego. Rasteryzacja (*Polygon Rendering*) polega na obliczeniu (zwykle w formie składowych RGB) ostatecznego koloru każdego z punktów wewnętrznych przetwarzanego w danym momencie trójkąta. Zanim zostaną ustalone definitywne wartości tych składowych, punkt przechodzi przez szereg etapów uwzględniających różne czynniki modyfikujące, takie jak np. teksturowanie lub efekty atmosferyczne i specjalne. W końcowej fazie, po przejściu całego cyklu przetwarzania przez procesor rasteryzujący, do pamięci obrazu zapisywany jest pojedynczy punkt (piksel) o pewnych wypadkowych składowych kolorów (RGB). W momencie skompletowania wszystkich pikseli, pamięć obrazu wyświetlana jest na ekranie.

Procesor rasteryzujący przejmuje od procesora geometrycznego jako dane wstępne zestaw parametrów, przypisywany każdemu z wierzchołków trójkąta. Poniższy zestaw parametrów przekazywany jest więc trzykrotnie, dla każdego wierzchołka oddzielnie:

- Współrzędne położenia wierzchołka w przestrzeni trójwymiarowej (*x, y, z*), gdzie *z* oznacza odległość punktu od płaszczyzny obserwacji, mierzoną w kierunku od obserwatora.
- Odniesienie do współrzędnych tekstury (*u, v*), której wzór pokrywać ma dany trójkąt.
- Barwa wierzchołka rozpisana na elementy składowe R, G, B.
- Informacja o stopniu przezroczystości powierzchni (tzw. współczynnik α). Ze względów oszczędnościowych α jest zwykle wspólne dla wszystkich trzech wierzchołków. Stopniowa zmiana tego współczynnika (po każdym obliczonym obrazie) pozwala na delikatne wymiksowanie trójkąta ze scenerii.

- Parametr korekcyjny skrótu perspektywicznego (*w*). Korekcja taka staje się niezbędna dla niektórych trójkątów, które leżą w płaszczyznach znacznie odchylonych od powierzchni ekranu. Współczynnik *w* jest poza tym często używany w zastępstwie współrzędnej *z*, bowiem gwarantuje (mimo uwzględnienia nieliniowego skrótu perspektywicznego) liniowy rozkład wartości w buforach, przechowujących informację o odległości punku od obserwatora.

Zakres zadań spoczywających na procesorze rasteryzującym jest stosunkowo rozległy. Ilość niezbędnych do przeprowadzenia obliczeń jest ogromna i zależy od stopnia komplikacji algorytmów. Niektóre zagadnienia dają się co prawda uprościć poprzez przybliżenia i interpolacje, ale i tak obowiązuje generalna zasada, że im bardziej skomplikowane modele matematyczne, tym lepsze wyniki końcowe. Bogate efekty świetlne, pogłębiające wierne odczucie rzeczywistości, wymagają potężnej mocy obliczeniowej, którą nowoczesne akceleratory 3D dostarczają coraz łatwiej.

Zaoszczędzić można już na etapie przekazywania danych pomiędzy procesorami geometrycznym i rasteryzującym. Transmitowanie wszystkich trzech wierzchołków każdego trójkąta oddzielnie jest stosunkowo nieekonomiczne. Niektóre trójkąty przylegają przecież do siebie i mają wspólne wierzchołki. Sześć przylegających do siebie trójkątów (TS, *Triangle Stripes*) ma tylko 6 niezależnych wierzchołków i nie trzeba przekazywać aż 18 punktów. Elastyczne przełączanie trybu pracy na TS i z powrotem możliwe jest dzięki temu, iż nowoczesne akceleratory dysponują uniwersalnym protokołem wymiany danych, który przypomina system pakietów informacyjnych o różnej zawartości.

W dalszym etapie przetwarzania następuje zwykle konwersja formatów. Procesor geometryczny wydaje dane w formacie zmiennoprzecinkowym, a akcelerator pracuje w arytmetyce stałoprzecinkowej. Sam ten krok zająłby procesorowi Pentium 90 co najmniej 15 cykli zegarowych (dla każdego przekazywanego parametru oddzielnie!).

Istota rasteryzacji polega, jak już wcześniej wspomniano, na obliczeniu kolorów wszystkich wewnętrznych pikseli trójkąta. Procesor dysponuje na wstępie wartościami składowych kolorowych jedynie dla wierzchołków. Barwy wierzchołków przejmowane są od procesora geometrycznego jako wynik przyjętych założeń odnośnie barwy własnej oraz oświetlenia obiektu. Nie wolno przecież zapominać, że widzimy tylko to, co jest oświetlone. Jest to zagadnienie o kapitalnym znaczeniu, jeśli chcemy, by nasz wirtualny obiekt 3D przypominał coś w miarę rzeczywistego. Dokładne odwzorowanie rzeczywistości jest z matematycznego punktu widzenia stosunkowo skomplikowane. Dla ułatwienia obliczeń przyjęło się posługiwać jednym z modeli uproszczonych (lub kombinacją kilku z nich). Zagadnienia dotyczące modelowania oświetlenia wykraczają zdecydowanie poza zakres tej publikacji.

Najbardziej prymitywny (i w dzisiejszych czasach już wcale niestosowany) tryb pracy polega na obliczeniu pewnej wartości średniej z trzech punktów wierzchołkowych i skopiowaniu jej na wszystkie punkty wewnętrzne (*Flat Shading*). Modelowana w tym stylu sceneria nabiera wprawdzie pewnego wrażenia „3D", ale powierzchnie obiektów są nienaturalnie kanciaste, bowiem bardzo wyraźnie widać pojedyncze trójkąty. Nawet mimo rozbicia na dużą liczbę elementów oko ludzkie dostrzega „szlify" na powierzchni.

Bardziej zaawansowaną technikę stanowi cieniowanie Gouranda[3] (*Gourand Shading*). Dla zwiększenia złudzenia, iż mamy do czynienia z równomiernie zakrzywioną powierzchnią, modyfikuje się w pewien określony sposób składowe kolorów dla poszczególnych punktów wewnętrznych trójkątów.

Algorytm określa najpierw kolor wypadkowy punktów przecięcia P_1 i P_2 linii skanowania z ramionami trójkąta. Wypadkową stanowi średnia ważona barwy wierzchołków:

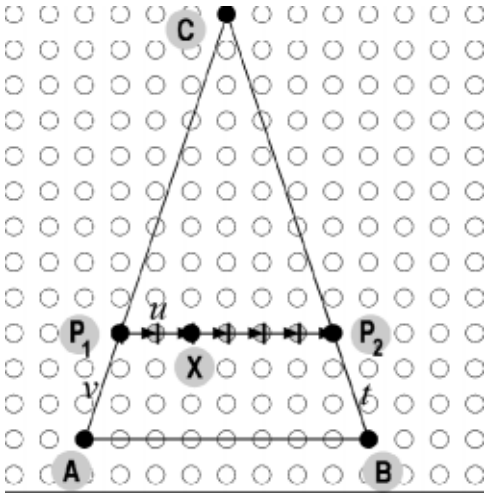
- Dla P_1 wynik uśrednia się na podstawie A i C:

  $I_{P1} = v \times I_A + (1 - v) \times I_C \qquad v \in \{0...1\}$

- Barwa P_2 zależna jest od B i C:

  $I_{P2} = t \times I_B + (1 - t) \times I_C \qquad t \in \{0...1\}$

Rysunek 15.4. *Zasada cieniowania wg Gouranda*



Każda ze zmiennych I_X symbolizuje barwę punktu z uwzględnieniem rozbicia na składowe elementarne. Oznacza to, że każde obliczenie przeprowadza się trzykrotnie dla wyodrębnienia składowych R, G i B. Współczynniki v i t stanowią odpowiednio wagi średniej, tj. odzwierciedlają stopień oddalenia punktów P_1 i P_2 od wierzchołków (im większy od P_1, tym mniejszy od P_2). Dla każdego z punktów X aktualnie skanowanej linijki interpoluje się barwę wypadkową w sposób bardzo podobny. Budowana jest średnia ważona rozpinana pomiędzy punktami P_1 i P_2 :

$$I_X = u \times I_{P1} + (1 - u) \times I_{P2} \quad u \in \{0...1\}$$

Obliczenia I_X (tzn, R_{Xi}, B_{Xi} i G_{Xi})wykonuje się oczywiście dla każdego punktu oddzielnie. Ponieważ mnożenie zajmuje nieporównywalnie więcej czasu od kilku nawet operacji dodawania, algorytmy omijają powyższe równanie i zamiast tego dodaje się (określane tylko jeden raz dla danej linii) przyrosty Δ_i :

[3] Henri Gourand: Continous Shading of Curved Surfaces, IEEE Transactions on Computers, C-20(6): 623 – 628, Juni 1971.

$$I_{Xi+1} = I_{Xi} + \Delta_i$$

gdzie:

$$\Delta_i = I_{P2} - I_{P1} / (n - 1) \qquad n \in \{1...n - 1\}$$

Cieniowanie Gouranda, mimo swej prostoty, daje bardzo dobre efekty. Powierzchnie modelowanych w ten sposób brył sprawiają miłe dla oka wrażenie ciągłości. Metoda ta nadaje się jednak wyłącznie do obrazowania prymitywnych scenerii złożonych ze sztucznych obiektów, pozbawionych wszelkich szczegółów powierzchniowych i faktur, bez charakterystycznej dla świata rzeczywistego gry świateł, cieni i odblasków.

Dalszy poważny krok w kierunku upodobnienia syntetycznych obiektów 3D do świata naturalnego umożliwiają zaprezentowane w następnym punkcie techniki mapowania.

Teksturowanie

Technika mapowania przy pomocy tekstur (*Texture Mapped*) była jeszcze do niedawna wyłączną domeną drogich i profesjonalnych systemów graficznych, wykorzystywanych w symulacjach i projektowaniu CAD. Mapowanie w czasie rzeczywistym dokonywane było na stacjach graficznych o specjalnie zaprojektowanej architekturze dostosowanej do przetwarzania grafiki. Moc obliczeniowa zwykłego komputera klasy PC sięgnęła jednak w międzyczasie tak wysoko, iż można się było pokusić o przeniesienie tych technik również na jego platformę.

Mapowanie nie służy niczemu innemu, jak oszukaniu zmysłu wzroku i ułatwieniu życia ludziom projektującym obiekty trójwymiarowe. Nie ma tu mowy o wzroście dokładności lub wierności przedstawienia modelu. Stanowi natomiast ostateczny i bardzo efektowny szlif nadawany wymodelowanym bryłom.

Mapowanie związane jest nierozerwalnie z pojęciem tekstury. Tekstura jest zwykłą mapą bitową (zeskanowaną, sfotografowaną lub wytworzoną sztucznie), naklejaną elektronicznie na wybrane miejsca obiektu 3D. Pojedyncze punkty tekstury nazywane są tekslami (dla odróżnienia od punktów obrazu, na które są naklejane, czyli pikseli). W ten oto prosty sposób jednorodny prostopadłościan staje się klocem lub drewnianą deską, a statuetka może być w szybkim tempie przerobiona z marmurowej na pokrytą błyszczącym chromem. Proszę zwrócić uwagę na niewspółmiernie mały nakład pracy potrzebnej do wymodelowania prostopadłościanu i pokrycia go teksturą: ten sam obiekt składa się w rzeczywistości z nierównomiernie zachodzących na siebie warstw drewna (słojów i poziomic), które należałoby oddzielnie modelować i łączyć.

Komputer operuje stosunkowo małą, jak na ten stopień skomplikowania obiektu, ilością danych. W pamięci przechowywane są współrzędne dwunastu trójkątów na które rozbite zostają płaszczyzny prostopadłościanu i kilka dodatkowych danych o sposobie „nalepiania" tekstury. Ten sam obiekt traktowany „na poważnie", musiałby zostać zamodelowany milionami trójkątów elementarnych.

Tekstury, jak już wspomniano, są obiektami płaskimi. Obraz taki łatwo jest nakleić na ścianę prostopadłościanu, gdyż jest ona tak samo płaska. Nałożenie mapy bitowej na bardziej skomplikowane (a w szczególności nieregularne) obiekty wymaga doboru pew-

nego przekształcenia przestrzennego. Jednym z nich może być wpisanie bryły w dodatkową kulę. Teksturowanie przebiega wtedy w dwóch etapach. W pierwszej fazie pokrywa się teksturą samą kulę, co jest o tyle łatwiejsze, że stanowi ona bryłę regularną. Wystarczy tu prosta funkcja transformująca współrzędne kuliste (długość i szerokość geograficzną) na współrzędne (x,y) płaskiej mapy bitowej. Jeżeli mapa bitowa o rozmiarach 180×360 umieszczona zostanie w układzie współrzędnych prostokątnych, to dla każdego punktu położonego na kuli można podać parę liczb:

- $x \in (0 \ldots 360)$,
- $y \in (-90 \ldots +90)$ lub po prostej zmianie $(0 \ldots 180)$.

W drugim etapie przetwarzania, zamkniętą w kulę bryłę przecina się siecią promieni wychodzących z jakiegoś umownego punktu środkowego. Każdy z promieni przecina powierzchnię bryły i powierzchnię kuli, sprowadzając poszukiwany teksel.

Powyższy prosty model nie bierze jednak pod uwagę, iż zarówno mapowany, obiekt jak i sama tekstura składają się z pewnej skończonej liczby pikseli i tekseli (konieczność uwzględniania rozdzielczości, czyli rastra). Teksturowanie ma miejsce w punktach świetlnych obiektu (pikselach) rozmieszczonych na rastrze. Pomiędzy nimi nie ma żadnych punktów pośrednich. Algorytm skanujący teksturę musi zwrócić dla każdego piksela jakąś określoną wartość składników RGB i tylko jedną. Jeśli w wyniku obliczeń skanera przeprowadzającego odwzorowanie trafienie ma miejsce w przestrzeń pomiędzy tekselami, konieczne jest pewne przybliżenie wartości (z sąsiednich tekseli). Nie można liczyć na to, iż każdy piksel obrazu trafia dokładnie w jakiś teksel mapy bitowej, tym bardziej, iż rozmiary tekstury są inne niż obiektu. Dodatkowym źródłem zniekształceń jest również wynik oddziaływania skrótu perspektywicznego.

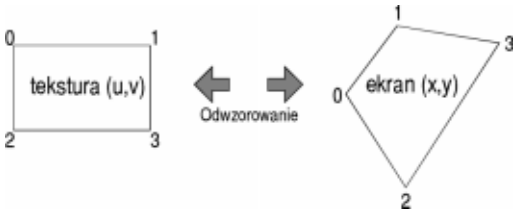
Mnogość różnych czynników zniekształcających i konieczność ich korekcji sprawia, iż teksturowanie jest jednym z najbardziej skomplikowanych i czasochłonnych fragmentów potoku przetwarzającego 3D. Wprowadzanie tych poprawek oraz różnorodne aproksymacje należą oczywiście do obowiązków procesora rasteryzującego. Najpierw przeprowadzana jest korekcja perspektywy, a następnie przystępuje się do nakładania mapy. Akceleratory aktualnych generacji potrafią nakładać mapy wielokrotnie i symulować różnorodne efekty specjalne oraz atmosferyczne.

Korekcja perspektywy

Konieczność korekcji perspektywy wynika z faktu rzutowania teksturowanego obrazu 3D na płaszczyznę ekranu. Skrót perspektywiczny oddziaływuje nie tylko na geometrię obiektów (a właściwie pojedynczych trójkątów elementarnych), ale dodatkowo zakłóca samo mapowanie. Prostokątna mapa bitowa musi zostać rozciągnięta od strony obserwatora i ściśnięta w miarę posuwania się w głąb przestrzeni 3D. Istotę omawianego zniekształcenia przedstawia rysunek 15.5, na którego prawej części naniesiony jest prostokąt zawieszony w przestrzeni 3D. Prostokąt ten teksturowany jest mapą bitową przedstawioną symbolicznie w lewej części rysunku. Wyraźnie widać, że trudno jest o proste odwzorowanie tych dwóch obiektów. Transformacja przestrzeni $u,v \rightarrow x,y$ jest

dosyć skomplikowana i jeśli nie uwzględnia się szeregu poprawek, prowadzi do nienaturalnych zniekształceń (*Visual Artifacts*). Różnorodne procedury korekcyjne występują często pod wspólnym terminem Anti-Aliasing.

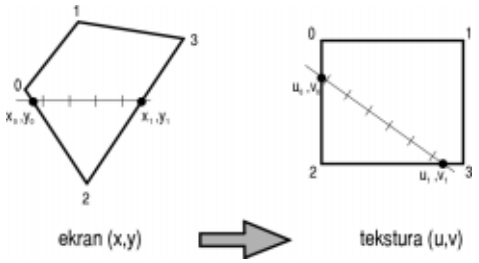
Rysunek 15.5.
Zjawisko skrótu perspektywicznego



Zagadnienie skrótów perspektywicznych doczekało się bardzo dokładnego opisu matematycznego. Głębsza analiza problemu wykazuje, że dla zniwelowania skutków zniekształceń tekstur, którymi pokrywane są obiekty 3D, nie wystarczy interpolacja liniowa, a w równaniach pojawiają się składniki wyższego rzędu oraz operacje dzielenia.

Procesor rasteryzujący musi skanować w dwóch przestrzeniach: obrazu i tekstury. Obraz skanowany jest w naturalny sposób wzdłuż linii poziomych. W układzie współrzędnych tekstury analogiczna linia przechodzi przez punkty (u_0, v_0) i (u_1, v_1), jednakże na skutek omawianego zniekształcenia perspektywy jest linią ukośną.

Rysunek 15.6.
Odwzorowanie punktu przestrzeni ekranu na punkt w przestrzeni tekstury



Algorytm przemierza przestrzeń (U, V), poruszając się równymi krokami od punktu (u_i, v_i) do punktu (u_{i+1}, v_{i+1}):

$$u_{i+1} = u_i + d_u$$
$$v_{i+1} = v_i + d_v$$

Elementarne przyrosty w kierunku u i v wynikają z następującego wzoru:

$$d_u = (u_1 - u_0) / (x_1 - x_0)$$
$$d_v = (v_1 - v_0) / (x_1 - x_0)$$

Dzielenie liczb (i to w formacie zmiennoprzecinkowym) należy do najbardziej czasochłonnych operacji. Każda z nich zajmuje do kilkudziesięciu cykli zegarowych. Często newralgiczne czynniki typu 1/w aproksymuje się składowymi wyższego rzędu (ddu i ddv). Wielokrotne dodawania są i tak zawsze szybciej wykonywane niż dzielenie.

Nakładanie mapy

Procesor rasteryzujący oblicza dla każdego piksela obrazu (przestrzeń x,y) współrzędne w przestrzeni mapy bitowej (u,v). W tak określonym punkcie niekoniecznie należy się spodziewać jednego z tekseli. Algorytm musi jednak zwrócić jakąś wartość składowych RGB. Wybór sposobu postępowania w tym właśnie momencie decyduje w znacznym stopniu o końcowym wyglądzie scenerii. Warto może przyjrzeć się drodze, jaką odbyły akceleratory 3D: od najprostszego przybliżenia punktu sąsiedniego do skomplikowanych interpolacji tekstur o różnych skalach.

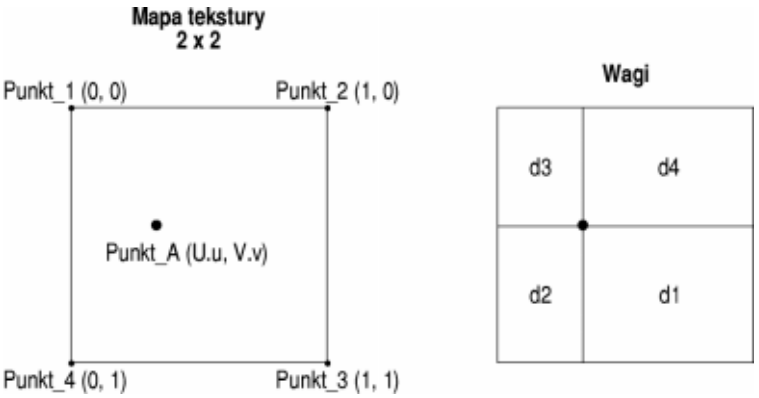
Przyporządkowanie najbliższego punktu (Peak Nearest)

Ten prosty algorytm pobiera parametry tekstury z punktu leżącego najbliżej wyliczonych współrzędnych. Powierzchnia obiektu pokrywanego tą techniką nie daje zadowalających efektów. Płaszczyzny sprawiają nienaturalne wrażenie struktury gruboziarnistej, szczególnie, gdy leżą blisko obserwatora (silne powiększenie). Zniekształcenia dotyczą również linii ukośnych, które kreślone są jako schodkowe.

Mapowanie w tym trybie zostało porzucone (skoro tylko pozwoliła na to nadwyżka mocy obliczeniowej) na korzyść bardziej złożonych algorytmów, które dają zdecydowanie lepsze efekty. Uśrednienie z kilku teksli jest cechą wspólną bardziej rozbudowanych metod. Należą do nich: filtracja bilinearna, MIP-Mapping oraz filtracja trilinearna.

Filtracja bilinearna (Bilinear Interpolation)

W metodzie tej analizie podlega otoczenie punktu. Parametry RGB uśrednia się w oparciu o czterech sąsiadów, przy czym budowana wartość średnia ważona jest stopniem oddalenia od punktu trafienia. Teksle leżące bliżej mają większy wpływ na wynik.



Rysunek 15.7. *Odwzorowanie filtracji bilinearnej*

Dla pewnego punktu obrazu A(x,y) otrzymujemy parę współrzędnych (U.u, V.v), lokalizującą punkt w przestrzeni tekstury. Punkt ten otoczony jest czterema sąsiadami:

Punkt_1 (U, V)
Punkt_2 (U+1, V)

Punkt_3 (U+1, V+1)
Punkt_4 (U, V+1)

Składowe ułamkowe 0.u i 0.v współrzędnych punktu (U.u, V.v) reprezentują właśnie położenie we wnętrzu kwadratu ograniczonego punktami 1, 2, 3 i 4.

Każdy z narożnych punktów ma jasno zdefiniowane parametry R, G, B (R_1, G_1, B_1, R_2, G_2, B_2 ...). Poszukiwane składniki barwy dla punktu A stanowią wypadkową kolorów punktów 1 – 4, ale z uwzględnieniem ich odległości od punktu. Im bliżej narożnika usytuowany jest punkt A tym większy jest jego wpływ na kolor wypadkowy. Budowana jest średnia ważona, a współczynniki wagowe $d_1 – d_4$ odpowiadają powierzchni prostokątów zaznaczonych na rysunku:

$$d_1 = (1 – 0.u) \times (1 – 0.v)$$
$$d_2 = 0.u \times (1 – 0.v)$$
$$d_3 = 0.u \times 0.v$$
$$d_4 = (1 – 0.u) \times 0.v$$

Teraz można już wyznaczyć parametry R, G, B punktu A:

$$R = d_1 \times R_1 + d_2 \times R_2 + d_3 \times R_3 + d_4 \times R_4$$
$$G = d_1 \times G_1 + d_2 \times G_2 + d_3 \times G_3 + d_4 \times G_4$$
$$B = d_1 \times B_1 + d_2 \times B_2 + d_3 \times B_3 + d_4 \times B_4$$

W praktyce spotkać można było również różne formy pośrednie filtracji bilinearnej, wprowadzane zwykle ze względów oszczędnościowych. I tak np. kontroler graficzny z akceleratorem 3D typu GD5464 firmy Cirrus Logic uśredniał na podstawie sąsiedztwa dwóch punktów, a nie czterech.

Algorytm filtracji bilinearnej stanowi rozsądny kompromis pomiędzy osiąganymi wynikami a stopniem komplikacji i w chwili obecnej należy do standardowego wyposażenia każdego szanującego się akceleratora 3D. Warto jednak wiedzieć, iż nie usuwa on wszelkich zniekształceń, a tylko niektóre. Nadal pozostaje w mocy szkodliwy wpływ skrótów perspektywicznych (*Depth Aliasing*). Wynika to z nieuchronnego faktu, iż teksturowany obiekt, który oddala się od obserwatora (poruszający się obiekt lub fragment scenerii – typu droga biegnąca w stronę horyzontu), ściska w coraz to mniejszych ramach stale tę samą mapę. W celu skorygowania tego czynnika powstały algorytmy wyższej generacji, które uwzględniają w odpowiednim momencie zmianę rozdzielczości mapy bitowej.

MIP-Mapping

Tekstury są, jak już wspomniano, mapami bitowymi o ustalonej rozdzielczości (np. 256 × 256), a co za tym idzie, ich rozmiar jest stały. Poddawane mapowaniu obiekty 3Dmogą mieć różne rozmiary.

Weźmy za przykład grę wykorzystującą naturalną scenerię powierzchni ziemi. Obszar dzieli się na kwadraty 1 × 1 km, z których każdy przechowywany jest jako mapa bitowa

1000×1000 punktów. Obraz taki widziany z kabiny symulatora lotniczego ma rozdzielczość graniczną 1 m^2. Jeśli akcja przeniesie się na ziemię, gracz nie rozróżnia szczegółów poniżej 1 m.

Optymalne dopasowanie następuje jedynie wtedy, gdy rozmiar mapy bitowej zgodny jest wymiarami pokrywanej powierzchni. W przypadkach obiektów, których rozmiary ulegają zmianie (bo np. oddalają się lub przybliżają do obserwatora), rodzi się problem dopasowania tekstury do mapowanej powierzchni. Jeśli wymiar tekstury osiąga wartość dwukrotnie większą od długości lub szerokości obiektu, procesor rasteryzujący powinien przeprowadzić interpolację bilinearną i zredukować powierzchnię mapy o jedną czwartą (z każdych czterech tekseli mapy poziomu wyższego otrzymuje się jeden teksel mapy poziomu niższego). Ponieważ wielokrotne przeprowadzanie takich interpolacji w czasie rzeczywistym jest czystą stratą mocy obliczeniowej (jeśli wyniki za każdym razem są porzucane), przefiltrowane mapy bitowe zapamiętuje się na stałe. Dla każdej tekstury przygotowuje się obszerny zestaw[4] map bitowych o różnych stopniach rozdzielczości. Przechowywane mapy (nazywane też Mipami) mają rozmiary, będące kolejnymi potęgami liczby 2. Lista zamykana jest zwykle w okolicach 500, gdyż potrzeba mapowania tak dużych obiektów w standardowych ekranowych trybach rozdzielczości jest znikoma. Do dyspozycji stoją więc zwykle mapy o następujących rozmiarach:

$1 \times 1, 2 \times 2, 4 \times 4, 8 \times 8, 16 \times 16, ... 512 \times 512$

Procesor rasteryzujący określa, który Mip jest w danym momencie optymalny. W tym celu obliczany jest wyznacznik LOD (*Level of Detail*). LOD = 0 oznacza mapę o najwyższej rozdzielczości, która odpowiada stosunkowi 1 teksel na 1 piksel. Mapa o poziomie LOD = 1 przefiltrowana jest już w stosunku 4 tekseli na 1 piksel, itd.

Współczynnik LOD powinien być w zasadzie obliczany dla każdego piksela trójkąta oddzielnie (*Per Pixel MIP-Mapping*), bowiem taka procedura daje najlepsze wrażenia optyczne. W formie oszczędnościowej LOD określa się raz na cały trójkąt (*Per Triangle MIP-Mapping*), ale postępowanie takie prowadzi często do nieciągłości obrazu w narożnikach trójkąta.

MIP-Mapping, mimo iż jest bardzo zaawansowaną i skuteczną techniką nakładania tekstur, rozwiązuje jedne problemy, ale w zamian za to wywołuje inne skutki uboczne. W pewnych niekorzystnych warunkach widoczny jest mianowicie moment przełączeń na kolejne stopnie LOD (zjawisko określane jako MIP-Banding). Obserwator dostrzega wówczas nieistniejące w naturze linie wzdłuż granic dzielących Mipy o różnym poziomie LOD. Dla zniwelowania tego efektu opracowany został jeszcze bardziej rozbudowany algorytm teksturowania, nazywany filtracją trilinearną.

Filtracja trilinearna

Algorytm ten daje bardzo dobre wyniki, ale pochłania wiele mocy obliczeniowej, bowiem stanowi kombinację filtracji bilinearnej i średniej ważonej dwóch sąsiednich Mipów. Metoda rozmywa przejścia pomiędzy stopniami LOD, co odbierane jest przez oko ludzkie zdecydowanie lepiej, niż ostro zarysowane punkty przełączeń. Filtracja triline-

[4] Stąd nazwa Mip-Mapping (Mip = **M**ultum **i**n **p**arwo, dużo w jednym).

arna bierze swoją nazwę stąd, iż stosuje w sumie 3 filtry, aby wydzielić jeden uśredniony teksel. Procesor wybiera z danej mapy bitowej dwa Mipy o rozdzielczościach leżących najbliżej rozmiarów mapowanego obiektu. Filtr pierwszy dokonuje filtracji bilinearnej mapy Mip jednego poziomu. Filtr drugi opracowuje mapę bitową poziomu drugiego. Filtr trzeci uśrednia wyniki z powyższych dwóch operacji.

Jednoznaczny dobór dwóch Mipów (o rozdzielczościach zamykających obiekt od góry i od dołu) nie zawsze jest możliwy. Na skutek działania skrótu perspektywicznego może się zdarzyć, iż różne obszary tego samego obiektu pasują lepiej do różnych Mipów. Proszę wyobrazić sobie trójkąt mocno odchylony w kierunku horyzontu. Długość jego podstawy jest niewielka (optymalny Mip 8 × 8), natomiast długość ramion wielokrotnie większa (wymagany Mip 128 × 128). Nowoczesne akceleratory 3D przechodzą wówczas do innego trybu pracy, określanego jako filtracja anizotropowa (*Anisotrophic Filtering*). Algorytm ten posługuje się mapami bitowymi o odmiennych rodzielczościach w kierunku X i Y.

Filtracja anizotropowa stanowi zaawansowaną technikę z grupy tzw. *Oversampling* (inaczej *Sub-Pixel Interpolation*). Termin ten oznacza próbkowanie ze zwiększoną rozdzielczością (większą niż odstęp pomiędzy tekselami i pikselami) i uśrednianie wyników. Algorytmy tego typu dają doskonałe efekty, ale pochłaniają ogromną ilość mocy obliczeniowej. 8-krotny oversampling redukuje wartość fps w stosunku 1:8. Techniki te są obecnie przedmiotem intensywnych badań i nie ma jak na razie rozwiązań praktycznych, nadających się do powszechnego zastosowania.

Omawiane tu funkcje filtrujące redukują efekt schodkowy, powstający na zewnętrznych krawędziach obiektów w stosunku do tła (*Edge Anti-Aliasing*). Bardziej nowoczesny system (*Scene Anti-Aliasing*) może redukować schodki tworzące się na liniach przecięć wielokątów we wnętrzu obiektów. Algorytmy tego rodzaju przetwarzają obraz jako całość i pochłaniają naturalnie ogromną ilość mocy obliczeniowej. Producenci akceleratorów inwestują wiele w prace nad optymalizacją takich algorytmów.

Mieszanie kolorów

Procesor rasteryzujący może uwzględniać fakt mieszania się kolorów (*Blending*) podłoża mapowanego obiektu i nakładanej tekstury. Ta część procesu przetwarzania ma miejsce w specjalnej programowanej jednostce mieszającej barwy (*Blending Engine*). W zależności od zaprogramowanej operacji i użytych kolorów uzyskuje się efekty przyciemnienia, rozjaśnienia lub połysku (*Specular Lighting*). Jednostka mieszająca może zsyntetyzować dowolne dwa obrazy w jeden, posuwając się piksel po pikselu. Najprostsze algorytmy to modulacja i przesłanianie.

- *Modulacja.* W modelu tym zakłada się, iż trójwymiarowa bryła przed oklejeniem jej teksturą była idealnie biała, tzn. odbijałaby całkowicie padające na nią światło. W takim razie każdy z barwnych elementów tekstury stanowi filtr tłumiący światło odbite w tym kolorze.

Przykład:

Jeden z narożników idealnie białego trójkąta oświetlony zostaje zielonym światłem. Punkt ten ma teraz parametry RGB (0,1 ; 0,9 ; 0,1). Naklejona w tym miejscu tekstura cechuje się zabarwieniem RGB (0,2 ; 0,3 ; 0,4). Jako barwa wynikowa przyjęta zostanie trójka:

$R=0,1 \times 0,2=0,02$

$G=0,9 \times 0,3=0,27$

$B=0,1 \times 0,4=0,04$

- *Przesłanianie*. Zamiast prostego mnożenia składników RGB dla podłoża i tekstury można przyjąć bardziej elastyczne rozwiązanie. Dla każdego ze składowych wynikowych z osobna wprowadza się funkcję postaci:

  $C_X = a \times P_X + (1 - a) \times T_X$

  gdzie:

  C_X – jest składnikiem wynikowym (C_R , C_G lub C_B),

  P_X – składnik koloru podłoża,

  T_X – składnik koloru tekstury,

  a – (0 ... 1) współczynnik przesłaniania.

  Model ten pozwala na swobodne określenie stopnia dominacji nakładających się na siebie barw: jeśli a = 0, tekstura przesłania całkowicie podłoże; dla a = 1 widzimy tylko podłoże, a tekstura jest przeźroczysta. Liczby z przedziału 0 – 1 wyobrażają wszelkie stany pośrednie.

  Z punktu widzenia komputera obliczenia przesłaniania są bardziej pracochłonne od modulacji. Oprócz niezbędnych w pierwszym modelu trzech mnożeń trzeba jeszcze przeprowadzić dodatkowo trzy dodawania. Sprawę można uprościć, jeżeli współczynnik przenikania a jest stały[5] w ramach całej tekstury. Dla wiernego odtworzenia rzeczywistości założenie to nie zawsze jednak może być przyjęte.

Efekty specjalne

Gry i symulacje komputerowe ostatniej generacji wprowadzają do swojego repertuaru szereg efektów specjalnych, które mają pogłębić u obserwatora wrażenie naturalności scenerii. W ich realizacji bierze intensywny udział jednostka mieszająca (*Blending Engine*). Większość z nich wymaga kolejnego nałożenia dwóch tekstur. Oto kilka przykładów:

- Mgła (*Fog*). Należy do grupy efektów atmosferycznych (*Atmospheric Effects*) i polega na sterowaniu współczynnikiem przejrzystości tekstur.
- Przejrzystość obiektów (*Alpha Blending*). Efekt wywoływany jest przez maszynę mieszającą barwy. Istota tej symulacji polega na manipulacji barwy piksela już obliczonego i spoczywającego w pamięci obrazu. Jego parametry pobiera się i miesza z innym kolorem, a zaktualizowany piksel wynikowy jest ponownie zapisywany w pamięci.

[5] Jeżeli a jest zmienne, to przechowywane jest wraz z mapą bitową tekstury.

- *Depth Cueing*. Stopniowe obniżanie jasności obiektów w miarę oddalania się od obserwatora.
- *Bump Map*. Ta sama mapa nakładana jest dwukrotnie, z tym że za drugim razem wprowadzane jest lekkie przesunięcie. Algorytm daje zadziwiająco dobre wyniki w symulowaniu powierzchni o „tłoczonej" fakturze. Wrażenia takiego nie da się uzyskać przy pomocy pojedynczej tekstury.
- *Environment Map*. Symulacja błyszczących powierzchni, w których w naturalnych warunkach odbijają się przedmioty otoczenia. Trudność polega na tym, iż algorytm pochłania ogromną ilość mocy obliczeniowej, bowiem współrzędne tekstur (u;v) nie są na stałe przypisane do wierzchołków trójkąta, a muszą być dynamicznie określane dla każdej rasteryzowanej sceny oddzielnie.
- *Lighting Map*. Symuluje odblaski rzucane na otoczenie przez silne źródła światła (eksplozje, rozbłyski laserów, itp)
- Cienie (*Shadow*). Symulacja cieni rzucanych przez przedmioty wymaga również teksturowania dwukrotnego.

Spora część wymienionych tu efektów (*Bump Map*, *Environment Map*, *Lighting Map*, *Shadowing*) wymaga teksturowania wielokrotnego (*Multi Texturing*). Obróbka scenerii bogatych w efekty tego rodzaju jest coraz bardziej pracochłonna. W jednym błyszczącym obiekcie odbija się otoczenie, ale z kolei sam obiekt rzuca odblaski na otoczenie. Aby przyspieszyć takie obliczenia, najnowocześniejsze akceleratory 3D wyposażane są dwa równoległe potoki przetwarzające (*Dual 3D Pipe Line*) lub przynajmniej w podwójną jednostkę teksturującą (*Dual Texture Unit*). Procesory tego typu pobierają w jednym cyklu przetwarzania po dwa teksele i to z dwóch różnych tekstur.

Teksturowanie wielokrotne stanowi jeden ze znaków szczególnych DirectX w wersji 6.0 i zwrócone jest wyraźnie w stronę układów najnowszej generacji, posiadających dublowane procesory teksturujące (Voodoo-2, RivaTNT czy też Permedia-3). Jeśli kontroler nie należy do wspomnianej klasy, funkcje wewnętrzne DirectX 6.0 mogą symulować programowo brakujące fragmenty sprzętu. Kolejne rozszerzenia interfejsu DirectX implementowane są w wersji 7.0, która dołączona jest do systemu operacyjnego Windows 2000.

Pamięć lokalna akceleratora

Lokalna pamięć akceleratora 3D (fizycznie obecna na karcie kontrolera, w odróżnieniu od wirtualnej pamięci udostępnianej przez mechanizm AGP) podzielona jest na kilka obszarów funkcjonalnych.

Obszar najbardziej istotny to pamięć obrazowa (*Frame Buffer*) przechowująca składowe kolorowe (zwykle w formacie RGB lub YUV) wszystkich X × Y pikseli wyświetlanego obrazu. W obrębie pamięci obrazowej obowiązuje adresowanie współrzędnymi ekranu (x i y, gdzie x = 1 ... X oraz y = 1 ... Y), zgodnie z aktualnie ustawionym trybem rozdzielczości (X × Y).

Obszar drugi tworzy tzw. bufor Z (lub w nowszych modelach bufor W). Do grupy obszarów adresowanych współrzędnymi (x,y) zaliczyć należy również tzw. *Stencil Buffer*, który jest implementowany w akceleratorach ostatniej generacji.

W czasach przed wprowadzeniem AGP kontroler 3D musiał przeznaczać część swej lokalnej pamięci na bitowe mapy tekstur. Mimo iż mechanizmy oferowane przez magistralę AGP pozwalają na przetwarzanie tekstur wprost w pamięci operacyjnej, część kontrolerów wybiera obróbkę w pamięci lokalnej (jednak większa szybkość), zwłaszcza iż możliwa stała się silna kompresja map bitowych bez szczególnej utraty rozdzielczości. W dalszej części rozdziału omówimy pokrótce wspomniane obszary pamięci.

Frame Buffer

Pamięć obrazu pracuje w bardzo specyficznych warunkach. Podlega ona jednoczesnemu zapisywaniu (przez kontroler graficzny) i odczytywaniu (przez przetwornik RAM-DAC). Gdyby wszystko miało się odbywać w obrębie tego samego bloku pamięciowego, przetwornik musiałby czekać na ostateczne zakończenie procesu zapisu. Do momentu odczytania ostatniego piksela nie wolno byłoby z kolei nic zapisywać (wówczas oczekuje kontroler). Aby wymienione dwa procesy nie zakłócały się wzajemnie, wprowadzono podwójne buforowanie (*Dual Buffering*). Do dyspozycji kontrolera graficznego stawia się dwa jednakowe bufory pamięciowe. Jeśli jeden z nich wypełniany jest świeżą treścią (*Back Buffer*), drugi można oddać do dyspozycji RAM-DAC, by przekazał kompletną zawartość na ekran (*Front Buffer*). W ten sposób zawsze jakiś bufor jest pokazywany a jakiś stoi gotów do zapisu i wspomniane dwa procesy nie zakłócają się wzajemnie.

Moment przełączenia buforów musi wypadać w chwili, gdy generowany jest kolejny impuls synchronizujący w pionie, w przeciwnym razie obraz podzielony zostanie poziomą linią. Może się jednak zdarzyć, iż mimo nadejścia kolejnego impulsu V_{SYNC} akcelerator nie zdążył z kompletnym wypełnieniem bufora.

Rozważmy prosty przykład obliczeniowy. Pracujemy w trybie 60 Hz i RAM-DAC generuje co 1/60 sekund impuls synchronizacji pionowej. Jeżeli czas obliczeń jednej sceny wyniesie tylko nieco dłużej niż 1/60 sekundy (na przykład 1/59 sekundy), to kontroler 3D przegapia o ułamek sekundy moment przełączenia i musi czekać na nadejście następnego. W tym czasie procesor graficzny stoi bezczynnie. Do jednego z buforów nie może jeszcze pisać (bo nie został do tej pory wyświetlony), a drugi bufor jest aktualnie pokazywany i też nie można go zamazywać. Rzeczywista częstotliwość odtwarzania obrazu wynosi w tym przypadku już tylko 30 fps.

Rozwiązanie tego problemu możliwe jest jedynie poprzez wprowadzenie trzeciego bufora (*Triple Buffering*). Przy podziale na 3 bufory odpada czas oczekiwania na impuls synchronizacyjny, bowiem kontroler zawsze dysponuje jakimś obszarem pamięci, do którego może zapisywać wyniki swoich obliczeń. Wadą tego rozwiązania jest zmienna częstotliwość odtwarzania [fps], bowiem nie ma już mechanizmu wyzwalającego w równych odstępach czasowych moment przełączania buforów. Połączenie zalet sztywnej synchronizacji ze stałą możliwością zapisu do jednego z buforów możliwe jest w układach implementujących bufory poczwórne (*Quad Buffering*).

Warto nadmienić, iż zapis do pamięci obrazu odbywać się może zarówno przy załączonej, jak i wyłączonej synchronizacji z impulsami V_{SYNC}. Synchronizację wyłącza się zwykle w celach pomiarowych, np. uruchamiając programy typu Bench-Mark. Kontroler może wtedy pisać do bufora obrazu z narzucaną przez siebie prędkością i nie musi czekać na moment powrotu wiązki elektronów w cyklu odchylania pionowego. Osiągane w tych anormalnych okolicznościach wartości fps sięgają w chwili obecnej powyżej 100 obrazów na sekundę. W warunkach rzeczywistych, gdy odtwarzana jest jakaś aplikacja, a nie program testujący, kontroler graficzny, wypełniwszy w całości bufor obrazu musi odczekać nadejścia kolejnego impulsu synchronizacyjnego, jeśli chce pokazać kompletny obraz. W takim uzależnionym układzie częstotliwość powtarzania obrazu 3D nie leży nigdy powyżej częstotliwości odchylania pionowego, z którą pracuje monitor.

Bufor Z/W

Położenie dowolnego punktu w przestrzeni 3D jest jednoznacznie zdefiniowane przez trójkę liczb (*x, y, z*), będących jego współrzędnymi. Z punktów takich składają się trójkąty elementarne, a z nich z kolei modelowane właściwe elementy scenerii. Oś Z przebiega w kierunku od płaszczyzny ekranu w głąb przestrzeni 3D. Wartość współrzędnej *z* rośnie w miarę oddalania się od obserwatora.

Mimo iż rzecz dotyczy przestrzeni 3D, cały proces przetwarzania ma miejsce w płaszczyźnie ekranu, adresowanej parą współrzędnych płaskich (x, y).

Mogłoby się wydawać, że trzecia współrzędna *z* reprezentująca odległość punktu od ekranu staje się zbędna. Tak jednak nie jest, bowiem skutkiem projekcji na ekran trójwymiarowe elementy przedstawianej scenerii mogą się wzajemnie przesłaniać. Zjawisko przesłaniania się obiektów 3D sprowadza się do przesłaniania się trójkątów elementarnych, a w gruncie rzeczy do przesłaniania się pikseli. Analizę przesłaniania przeprowadza się właśnie na poziomie pojedynczych punktów obrazu.

Na ekranie kreślone będą więc tylko te punkty bryły, które są widoczne dla obserwatora. O tym, czy punkt jest widoczny, czy nie, decyduje właśnie jego współrzędna z. W tym celu utworzony zostaje w pamięci karty graficznej tzw. bufor Z. Stanowi on tablicę Z[x,y] o rozmiarach ekranu (X × Y). Elementy tablicy Z są zwykle dwubajtowe. Przed przystąpieniem do formowania nowego obrazu ustawiana jest wartość początkowa, dla każdego z[x,y]=0xFFFFh, co w ramach dokładności oferowanej przez zakres 0 – 216 odpowiada $+\infty$ na osi Z.

W miarę budowania trójwymiarowej scenerii, rozpisanej na elementarne trójkąty, rozpatruje się współrzędne *z* dla każdego z aktualnie obliczanych punktów składowych i porównuje je z dotychczasową wartością zapamiętaną w buforze Z (*Z-Buffering*). Punkt o współrzędnych (*x*, *y*, *z*) zostaje uznany za widoczny, jeśli spełnia warunek:

$$z < Z[x,y]$$

Obliczone składowe RGB dla punktu uznanego za widoczny wędrują do bufora obrazowego i umieszczane są pod adresem [x,y]. Współrzędna ***z*** takiego nowego punktu zamazuje napotkaną w buforze Z wartość:

Z[x,y]=z

Tak zmodyfikowany element tablicy Z pozostaje niezmieniony do momentu przetwarzania punktu obiektu leżącego jeszcze bliżej obserwatora.

Mechanizm bezbłędnego określania wzajemnego przesłaniania się pikseli funkcjonuje teoretycznie jedynie przy założeniu nieskończenie wielkiej dokładności obliczeń. Odstęp pomiędzy blisko położonymi obiektami 3D (np. obraz wiszący na ścianie) może w pewnych warunkach wymykać się spod kontroli, jeśli dokładność obliczeń (kwantyzacja) nie gwarantuje możliwości odróżnienia obiektów od siebie. Kluczową rolę odgrywa tutaj głębia przedstawianej sceny, na nią bowiem rozkłada się stojąca do dyspozycji dokładność mechanizmu kontroli głębokości.

Im większy jest zakres widoczności obserwatora, tym większe wymagania spoczywają na dokładności pomiaru odległości na osi Z. Organizacja bufora Z stawia do dyspozycji określoną ilość bitów. Bufor Z może pracować na liczbach 16-, 24- lub 32-bitowych. Akceleratory powszechnego użytku zadowalają się zwykle organizacją 16-bitową, modele nowszych generacji, szczególnie, te które zostały bogato wyposażone w pamięć, pracują w trybie 32-bitowym. Stojący do dyspozycji rozmiar bufora Z można różnie wykorzystywać. Zwykle pracuje się w arytmetyce liczb stałoprzecinkowych (skalowane liczby całkowite). Obiekty położone blisko (o małej współrzędnej Z) posiadają przez to większą dokładność oceny głębokości. Przejście na arytmetykę zmiennoprzecinkową gwarantują, jednakową dokładność w całym zakresie widzenia, ale spowalnia cały proces obliczeniowy.

Nie bez znaczenia jest znowu wpływ skrótu perspektywicznego, skutkiem czego rozkład wartości *z* w buforze nie jest liniowy. Obiekty znacznie oddalone od obserwatora mają coraz to bardziej zbliżone wartości współrzędnej *z*, przez co rośnie niebezpieczeństwo przekroczenia wartości minimalnej poniżej granicy dokładności. Jeśli zamiast współrzędnej *z* używać skorygowanej współrzędnej *w* (która również wyraża odległość od obserwatora, ale w innej skali), rozkład wartości w buforze (nazywanym teraz buforem W) staje się liniowy.

Z-Buffering (lub W-Buffering) obciążają bardzo mocno pamięć lokalną akceleratora poprzez szereg odwołań typu Read/Write-Modify. Odwołania tego typu są szczególnie uciążliwe, bowiem wymagają nieustannych przełączeń z zapisu na odczyt i odwrotnie. Nowoczesne pamięci dynamiczne osiągają swoje maksymalne parametry jedynie w trakcie szeregu następujących po sobie cykli dostępu tego samego rodzaju.

Warto wspomnieć w tym miejscu o istnieniu ciekawego algorytmu, pozwalającego na ominięcie całego procesu porównywania współrzędnych *z*, a w dodatku niewymagającego odwołań do pamięci. Mowa tu o sortowaniu trójkątów (*Z-Sort*). Przed przystąpieniem do przetwarzania wszystkie trójkąty elementarne ustawiane są w szereg (odzwierciedlający wzajemne przesłanianie). Gdy lista jest gotowa, procesor przystępuje do przetwarzania w kolejności „od tyłu do przodu". Jeżeli przetwarzany będzie każdy trójkąt, obiekty przesłonięte zostaną w naturalny sposób zamazane przez obiekty leżące bliżej. W ten sposób omija się zagadnienie niedokładności oceny odległości na osi Z. Bez względu na to, jak blisko od siebie spoczywają dwa trójkąty, zawsze jeden z nich można uznać za bliższy (poprzez pozycję w kolejce).

Pamięć tekstur

Jednym z czynników silnie obciążających lokalną magistralę pamięciową akceleratora jest dostęp do tekstur. Są to co prawda wyłącznie cykle odczytu, ale za to w dużej liczbie: przy aktywnej filtracji trilinearnej potrzeba wczytywać parametry 8 tekseli na każdy piksel obrazu. Procesory wyposażone w podwójne potoki przetwarzania (kompletne lub jedynie jednostki teksturujące) generują oczywiście dodatkowe obciążenie. Projektanci akceleratorów od dawna łamią sobie głowę nad różnymi sposobami optymalizacji architektury i zwiększenia wydajności pamięci.

Spore rezerwy drzemią w samej organizacji pamięci, a zwłaszcza wyrafinowanym rozlokowaniu map bitowych. Tekstury układane są często w pozornie nielogicznej formie, która gwarantuje jednak szybszy dostęp do kolejnych tekseli bez konieczności zmiany wiersza. Duże, kwadratowe obszary tekstury składające się z większej liczby logicznych linii, złożone są fizycznie w jednym wierszu pamięci. Cykl dostępu ulega znacznemu skróceniu, bowiem pamięć może przejść w tryb pracy *Burst,* a przygotowanie kolejnego adresu obejmuje jedynie modyfikację kolumny.

Profesjonalne akceleratory 3D dysponują często lokalną pamięcią pośrednią typu *Cache*, w której przechowuje się spory fragment aktualnie przetwarzanej mapy bitowej. Duże oszczędności przynosi sam sposób kodowania danych, np. przejście z przestrzeni RGB na YUV. W zależności od wymagań odnośnie jakości można pozwolić sobie na ograniczenie formatu danych: zamiast przeznaczać po jednym bajcie na każdą składową koloru można przejść do jednego z trybów oszczędnościowych: RGB-565[6] , RGBA-4444 lub RGBA5551. Procesor teksturujący tłumaczy formaty tego typu na swój wewnętrzny format RGBA8888, w którym dokonuje wszelkich operacji.

Aspekty ekonomiczne zadecydowały, iż akceleratory wcześniejszych generacji powszechnie wyposażane były w 64-bitową szynę komunikacyjną. Połowa jej szerokości przeznaczana była na komunikację z pamięcią obrazu, a reszta łączyła kontroler z pamięcią tekstur. Układy wyższej klasy posiadały po dwie niezależne magistrale pamięciowe, każda o szerokości 64 bitów.

Ostatnio do użytku wchodzi również inne rozwiązanie, pozwalające na bardziej efektywne wykorzystanie pamięci. Prezentowany tu system kompresji tekstur (S3TC) zaproponowany został przez firmę S3 i wprowadzony przez Microsoft do biblioteki DirectX 6.0.

Żelazna reguła mówiąca, iż pamięci jest zawsze za mało, obowiązuje naturalnie i w tym względzie. Osiągane efekty wizualne są tym lepsze, im większa jest rozdzielczość map bitowych, którymi posługuje się procesor tekstur. Obiekt teksturowany mapą o rozmiarach 512 × 512 oddziaływuje znacznie bardziej realistycznie na obserwatora, pokrywany tym samym wzorem ale o rozdzielczości 64 × 64.

Zapotrzebowanie na pamięć rośnie naturalnie dodatkowo w systemach posługujących się mapowaniem MIP. Każda z map bitowych przechowywana musi być w kilku lub kilkunastu wstępnie przefiltrowanych rozmiarach.

[6] RGB-565 oznacza rozpisanie na składniki 16-bitowej komórki pamięci obrazu: dla składnika zielonego 6 bitów, a dla składników czerwonego i niebieskiego po 5 bitów.

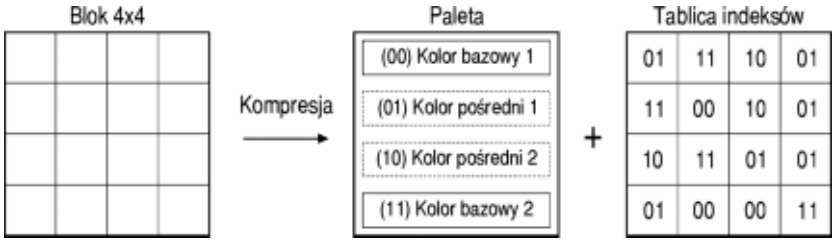
Właśnie temu stale rosnącemu zapotrzebowaniu na tekstury ma przyjść z pomocą system S3TC. Stosowany tu mechanizm pozwala na osiągnięcie sześciokrotnego współczynnika kompresji (przy założeniu, że produktem wejściowym była mapa bitowa przeznaczająca 24 bity na piksel, czyli w formacie RGB888). Algorytm oferuje 2 modele kompresji. W trybie podstawowym przeznaczane są po 4 bity na każdy kodowany teksel. W modelu drugim generowany jest strumień wyjściowy o podwójnej szerokości.

Algorytm kompresji nie koduje całej tekstury na raz, lecz dzieli ją wstępnie na bloki o rozmiarach 4 × 4 i w takich porcjach przetwarza. W obrębie bloku analizowany jest kolor każdego teksla w celu wydzielenia dwóch wiodących kolorów bazowych.

Produktem wyjściowym mechanizmu kompresji są dwie struktury danych: kodowana (wyjściowa) mapa bitowa oraz paleta kolorów (*Texture Palette Lookup*).

Przetworzony blok wejściowy stanowi teraz zestaw indeksów do palety kolorów. Paleta jest czterowierszową tablicą zawierającą w sumie cztery kolory: dwa bazowe i dwa pośrednie, wyznaczane przez interpolację kolorów bazowych. Kolory pośrednie generowane są automatycznie w fazie dekodowania i nie przechodzą do strumienia wyjściowego (nie są zapisywane). Paletę koduje się w formie RGB565, tj. po 16 bitów na kolor.

W ten sposób kodowany blok 4 × 4 zamieniany jest na zestaw dwubitowych indeksów i zajmuje teraz rozmiar 32 bitów (4 × 4 × 2). Możliwe wartości indeksów to oczywiście 00, 01, 10 lub 11.



Rysunek 15.8. *System kompresji tekstur S3TC*

W wyniku kompresji bloku 4 × 4 zapisywane są 64 bity (32 bity – zestaw indeksów i 32 – bity paleta). Algorytm w tej wersji produkuje więc średnio po 4 bity na teksel.

Rozmiar pamięci i organizacja

Faktyczne zapotrzebowanie na pamięć graficzną wynika w głównej mierze z aktualnego trybu pracy sterownika graficznego. Pełna strona obrazu zajmuje:

$$P = p \times l \times c$$

gdzie:

P – ilość pamięci w bajtach,

p – ilość punktów w linii,

l – ilość linii na ekranie (p × l opisuje rozdzielczość obrazu w pikselach, np. 800 × 600),

c – tryb kolorowy, tj. liczba bajtów odwzorowujących barwę piksla.

Jeden bajt daje możliwość rozróżnienia 256 kolorów. Dwa bajty mogą zakodować do 2^{16} różnych kolorów (tryb *High Color*, w zależności od przyjętej umowy, oznacza 32 768 lub 65 535 barw). Praca w trybie *True Color* wymaga poświęcenia trzech bajtów na piksel. Pozwala to na przedstawienie ponad 16 milionów barw. 24-bitowe cykle dostępu do pamięci graficznej nie są najmocniejszą stroną kontrolerów graficznych z architekturą 64-bitową. Niektóre z nich poświęcają więc 4 bajty pamięci w celu przyspieszenia dostępu. Tryb 3-bajtowy można niekiedy wymusić. Możliwość ta ukrywa się w oknach konfiguracyjnych pod nazwą *Packed Pixel Mode*. Bez tego nie byłyby możliwe niektóre z trybów np. 1280 × 1024[7] i *True Color* w kartach dysponujących 4 MB pamięci graficznej.

Zgodnie z powyższym praca z rozdzielczością 1024 × 768 punktów w trybie *High Color* (16 bitów na piksel, c = 2) pochłania 1024 × 768 × 2=1 572 864 bajty pamięci. Jeśli uwzględniać wyłącznie zapotrzebowanie wynikające z pracy sterownika graficznego (bez podwójnego buforowania, tekstur i bufora-Z), już nawet niewielka ilość pamięci pozwala na osiągnięcie rozsądnego trybu pracy:

Tabela 15.1.
Zapotrzebowanie na pamięć karty graficznej

Ilość pamięci	RAM-DAC	Maksymalnie możliwy tryb
1 MB	135 MHz	1280 × 1024 / 16 col. / 75 Hz
		1024 × 768 / 256 col. / 75 Hz
		800 × 600 / HiColor / 75 Hz
2 MB	135 MHz	1280 × 1024 / 256 col. / 75 Hz
		1152 × 864 / HiColor / 80 Hz
		1024 × 768 / HiColor / 80 Hz
		800 × 600 / TrueColor / 75 Hz
4 MB	175 MHz	1280 × 1024 / TrueColor / 75 Hz
		1600 × 1200 / HiColor / 65 Hz
	220 MHz	1280 × 1024 / HiColor / 80 Hz

Powyższa kalkulacja traci sens w przypadku akceleratora 3D, który bardzo intensywnie korzysta z pamięci graficznej. Już w trybie 640 × 480 i High Color może powstać deficyt nawet przy obsadzie 4 MB. Na samą pamięć obrazu przypada zgodnie z powyższymi obliczeniami, około 600 kB. Drugie tyle trzeba przeznaczyć na dodatkowy bufor (*Back Buffer*, w którym przygotowywany jest obraz i jeszcze raz tyle samo na potrzeby bufora-Z. Razem już prawie 2 MB. Do tego doliczyć musimy pamięć tekstur. Jeśli włą-

[7] 1280 × 1024 × 4 = 5 MB

czona jest filtracja trilinearna, w pamięci schronienie musi znaleźć cała rodzina map bitowych: oryginał 256 × 256 plus pomniejszenia 128 × 128, 64 × 64, aż do 1 × 1. Na całość potrzeba zarezerwować prawie 90 kB, a to dopiero jedna tekstura. Widać wyraźnie, jak niezmiernie pamięciożerne są funkcje 3D.

Typowy dla Windows 16-bitowy (2^{16} barw) obraz kolorowy w formacie 1024 × 768 zajmuje około 1,6 MB pamięci graficznej. Aby go kompletnie odnowić w stosunkowo krótkim czasie, np. 1/10s, procesor musi przesłać strumień danych 16 MB/s. Nie stanowi to jeszcze przeciążenia dla magistrali PCI. Lepsze karty graficzne mogą przejąć z magistrali do 60 MB/s. Ponadto w praktyce prawie nigdy nie odbudowuje się całego obrazu, a jedynie pewne jego fragmenty (np. przesłonięte lub przesunięte okna).

Problem rodzi się w momencie wyprowadzania danych na monitor. Chcąc zachować ergonomiczną częstotliwość odświeżania, należy przesłać w ciągu jednej sekundy 75 bloków danych po 1,6 MB. Wymagana szerokość pasma wynosi więc 120 MB/s. Podobne obliczenie przeprowadzone dla rozdzielczości 1280 × 1024 da wynik 200 MB/s, a w formacie 1600 × 1200 aż 288 MB/s. Sytuację można trochę podratować, redukując liczbę kolorów: 1024 × 768 przy 256 kolorach i 70 Hz daje już tylko 55 MB/s.

W toku rozwoju kart graficznych projektanci już dawno stanęli przed problemem ograniczeń w paśmie (dla standardowego trybu VGA mamy około 10 MB/s). Pierwszy poważny przyrost zapotrzebowania z 10 na 55 MB/s był możliwy do zaspokojenia początkowo jedynie poprzez zastosowanie drogich układów pamięci V-RAM.

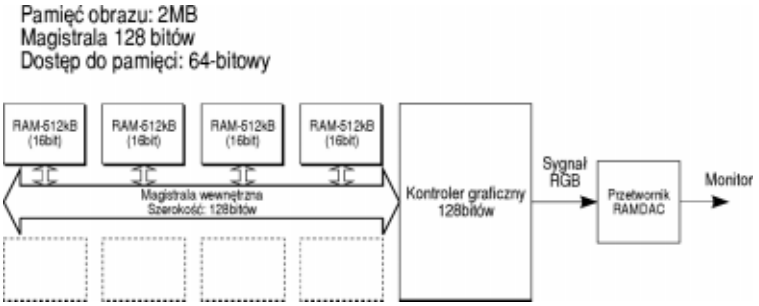
Brak innych rozwiązań spowodowany był dodatkowo małą szerokością magistral systemowych. Architektura ISA zadowalała się przez długie lata szynami 16-bitowymi, nawet w obrębie sterowników graficznych. Przyrost szerokości tych magistral umożliwił zaspokojenie głodu pasma już przy użyciu tanich układów pamięci. Bardzo szybko dokonał się skok z 16 na 32 bity, 64 bity należą do standardu, a coraz częściej pojawiają się już modele wyposażone w szyny 128-bitowe[8].

Niestety, nie wszystkie produkty przyozdobione naklejkami „128“ i „64“ mogły w istocie rzeczy poszczycić się adekwatnymi parametrami. W zaciszu kolorowych pudełek czekały na użytkownika często niemiłe niespodzianki. Ich przyczyna leżała w konfiguracji z niepełną obsadą pamięci. Względy ekonomiczne decydowały o tym, iż szerokość portu danych pojedynczych chipów pamięciowych wynosi zwykle 16 bitów. W zależności od szerokości magistrali wewnętrznej sterownika i aktualnej obsady pamięci mogła wystąpić następująca sytuacja:

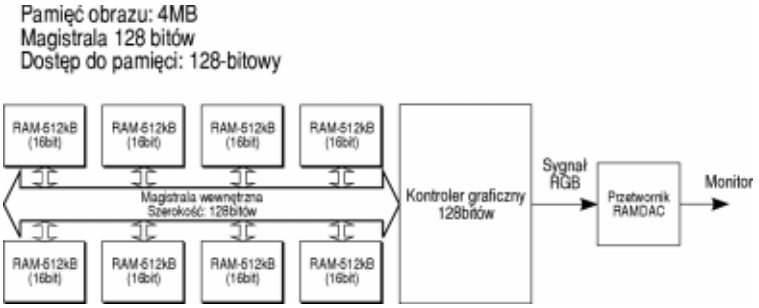
Cztery układy pamięciowe, każdy po 512 kB, tworzą wprawdzie w sumie blok 2 MB ale dostęp do pamięci odbywa się w trybie 64-bitowym. Jeśli nawet kontroler wyprowadza szynę podwójnej szerokości, nie może ona być w pełni wykorzystana. Produkt w tej konfiguracji nie powinien być reklamowany jako 128-bitowy. Dopiero uzupełnienie do 4 MB uprawnia do posługiwania się takim określeniem.

[8] Jeszcze raz warto podkreślić, że mowa o wewnętrznych magistralach kart graficznych, łączących ich procesory z blokami pamięci. Komunikacja sterownika ze światem zewnętrznym przebiega nadal poprzez 32-bitową szynę PCI lub AGP.

Rysunek 15.9. *Niepełna obsada pamięci może powodować zawężenie szerokości magistrali*



Rysunek 15.10. *Pełna obsada pamięci gwarantuje wykorzystanie pełnej szerokości magistrali*



Redukcja obciążenia magistrali pamięciowej karty graficznej możliwa jest do osiągnięcia na drodze obniżenia częstotliwości odchylania pionowego. Większość ludzi zadowoli się z pewnością częstotliwością 75 Hz i nie zauważy różnicy między 90 a 120 Hz. Spory fragment populacji odbiera wręcz negatywnie częstotliwości powyżej 100 Hz.

Wzrost zapotrzebowania na szybką pamięć graficzną i konieczność racjonalnego zagospodarowania pasma przepustowego wynika z wielu dodatkowych powodów. Techniki multimedialne narzucają nowe wymagania i stanowią dodatkowe obciążenia. Pamięć graficzna służy przecież nie tylko jako prosta matryca pikseli, z których buduje się obraz. Jest również miejscem przechowywania bitowych map tekstur oraz buforem Z. Głębokość buforu Z jest różna i waha się w granicach 16 (Matrox Millennium) do 32 bitów (Spea Fire GL). Filtracja bilinearna tekstur pracuje zwykle w trybie uśredniania z czterech otaczających punktów (cztery cykle odczytu). Jeszcze gorszy bilans prezentuje Mip-Mapping. W generowaną średnią zaangażowane są dwie mapy bitowe o różnej rozdzielczości (8 cykli odczytu). Skalowanie w pionie to często ekstrapolacja z dwóch linii. W trybie pracy z przełączanymi buforami (*Double Buffering*) zarządza się ponadto dwoma blokami obrazowymi jednocześnie.

Realizacja powyższych funkcji mnoży ogromną ilość dodatkowych cykli dostępu, a przeznaczana na nie ilość czasu pozostaje bez zmian. Kontrolery graficzne profesjonalnych systemów multimedialnych (np. symulatorów) wyposaża się więc w oddzielne szyny pamięciowe kierowane do niezależnych bloków. Jednej z magistral przydziela się część pamięci dla potrzeb bufora Z i tekstur, a druga obsługuje blok pamięci obrazowej.

Nawet pozornie duża ilość zainstalowanej na karcie pamięci nie zawsze może sprostać pokładanym w niej zadaniom. Łatwo obliczyć, że w trybie True-Color (32 bity na piksel) i rozdzielczości 1280 × 1024 nawet 8 MB nie wystarcza do realizacji funkcji po-

dwójnego buforowania: na karcie mieści się tylko jedna strona obrazu (5 MB). Kontroler Glint radzi sobie w tym wypadku, zrzucając drugi bufor do pamięci operacyjnej PC, co skutecznie blokuje magistralę PCI.

Kilka przykładów wykorzystania pamięci na kartach z akceleratorami 3D przedstawia tabela 15.2.

Tabela 15.2.
Typowe wykorzystanie pamięci w akceleratorach 3D

Pamięć na karcie	Rozdzielczość	Liczba kolorów	Bufory obrazowe (Front/Back)	Bufor Z (16-bitowy)	Pamięć dla tekstur
1 MB	320 × 200	2^{16}	0,12 MB / 0,12 MB	0,12 MB	0,63 MB
2 MB	640 × 480	2^{16}	0,59 MB / 0,59 MB	0,59 MB	0,24 MB
	800 × 600	2^{8}	0,46 MB / 0,46 MB	0,92 MB	0,17 MB
4 MB	640 × 480	2^{16}	0,59 MB / 0,59 MB	0,59 MB	2,24 MB
	800 × 600	2^{16}	0,92 MB / 0,92 MB	0,92 MB	1,25 MB

Rodzaje pamięci graficznych

Technika podwójnego buforowania (*Double Buffering*) umożliwia wprawdzie bardziej racjonalne wykorzystanie mocy obliczeniowej akceleratora i wpływa na płynną prezentacje obrazu, ale stanowi też dodatkowe obciążenie dla magistrali pamięciowej (przeplatające się cykle zapisu i odczytu). Źródłem naprzemiennych cykli R/W jest również konieczność komunikacji z buforem Z. Dostępu tego typu nie lubią w szczególności pamięci SDRAM i SGRAM, które osiągają swoje szczytowe parametry w długich sekwencyjnych cyklach. Każde przełączenie buforów (Back/Front) to szereg operacji zapisu, wymuszających zerowanie pamięci obrazu i inicjujących bufor Z wartością $+\infty$.

Jeśli maksymalna przepustowość magistrali wynika z przyjętej architektury oraz typu pamięci, to w danym przypadku nie da jej się przekroczyć. Wartość ta rzutuje w pewnym sensie na ograniczenie stopnia złożoności przedstawianych scen, ilości trójkątów elementarnych, rodzaju efektów świetlnych, sposobu teksturowania, aktywowanych filtrów itp. Podwyższenie dowolnego z tych czynników musi owocować obniżeniem częstotliwości odtwarzania obrazu. Jedyne wyjście z tego impasu oferują (przynajmniej na jakiś czas) nowe i coraz to szybsze typy pamięci, oszczędna nią gospodarka, a często nawet pewne sztuczki ograniczające pasmo przenoszenia.

Do jednej z nich należy tzw. rasteryzacja. Jądro akceleratora 3D przeprowadza obliczenia kolorów z dokładnością do 32 bitów. Aby zredukować szerokość zajmowanego pasma, zapis obliczonych wartości (do pamięci graficznej) odbywa się jednak w trybie 16-bitowym.

Oszczędności mają naturalnie swoje ujemne strony, które są niestety natychmiast wykrywane przez nasz zmysł wzroku. Oko ludzkie wyłapuje przejścia między kolorami i widzi w tym miejscu nieistniejące w gruncie rzeczy linie podziału (*Mach-Banding*).

W takich krytycznych obszarach procesor musi rozmazywać nieco obraz (*Dithering*), bowiem oko ludzkie jest mniej czułe na barwne szumy, niż na wyraźne granice międzykolorowe. Rozmycie linii uzyskuje się poprzez nieregularne domieszki interpolowanych kolorów pośrednich.

Pamięci graficzne przeszły dosyć długą drogę rozwojową, a prace nad ich udoskonalaniem trwają nieprzerwanie. Oto krótki przegląd powszechnie stosowanych typów.

DRAM

Elementarna komórka pamięci realizowana jest za pomocą pary złożonej z jednego tranzystora i kondensatora. Komórki zgrupowane są w pola, a dostęp do pojedynczego bitu odbywa się poprzez wybranie adresu wiersza i kolumny. W trybie przyspieszonym (*Fast Page Mode*) wystarczy podawać adresy kolumn przy niezmiennym adresie wiersza, co daje skrócony o połowę czas dostępu w stosunku do pełnego adresowania.

DRAM jest typem-prekursorem w grupie pamięci dynamicznych i wywodzi się z czasów pierwszych komputerów PC. Stosowana jest (z małymi modyfikacjami) do dnia dzisiejszego jako pamięć operacyjna, o czym decydują jednak inne kryteria. Ten standardowy typ pamięci nie jest obecnie w stanie sprostać wymaganiom zastosowań multimedialnych.

EDO i BEDO DRAM

Stanowi odmianę pamięci DRAM i udostępnia pasmo nieco powyżej 200 MB/s. Podczas odczytu dane utrzymywane są na wyjściu aż do momentu, gdy pole pamięci gotowe jest do przekazania następnego słowa. W ten sposób kontroler graficzny może przygotowywać się do następnego cyklu odczytu, będąc jeszcze w trakcie przejmowania danych z cyklu poprzedniego. Zachodzenie na siebie kolejnych cykli (technika *pipeline*) jest podstawą pracy w stylu *Burst* (stąd wywodzi się nazwa Burst EDO, BEDO DRAM). Dopiero ten zysk na czasie pozwolił na taktowanie pamięci bez cykli oczekiwania, tzn. każdy cykl zegara pamięci wyzwala 1 cykl odczytu. Cztery równolegle połączone układy pamięci (pracujące w konfiguracji $4 \times 16 = 64$bity) taktowane zegarem 50 MHz osiągają maksymalny transfer 400 MB/s (8×50). Średnia prędkość transmisji jest naturalnie mniejsza, bowiem kontroler musi kiedyś wreszcie zmienić adres wiersza i opuścić tryb Fast-Page.

O ile pamięci EDO stosowane były bardzo powszechnie, typ BEDO umarł śmiercią naturalną, nie doczekawszy się powszechnej akceptacji.

SDRAM

Synchroniczna odmiana pamięci DRAM jest wytwarzana przez wszystkich liczących się producentów. Typowa częstotliwość taktowania sięga 100 MHz. Szybkie odmiany tego typu przystosowane są do zegara 125 MHz, co pozwala osiągnąć transfer 640 MB/s (przy szerokości magistrali: 64 bity).

W przeciwieństwie do klasycznych układów pamięci DRAM, które wymagają precyzyjnie uformowanych sygnałów RAS i CAS (*Row Address Strobe*, *Column Address Strobe*), pamięci synchroniczne mają własny kontroler przetwarzający impulsy zegarowe na niezbędne sygnały sterujące. Metoda taka zmniejsza wymogi nakładane na dokładność wykonania ścieżek na płytkach drukowanych i gwarantuje zwiększenie prędkości taktowania.

Bloki pamięci SDRAM i SGRAM organizowane są zwykle w dwa banki obsługiwane naprzemiennie (technika *interleave*) co pozwala na nakładanie się w czasie kolejnych cykli dostępu.

SGRAM

Odmiana synchronicznej pamięci SDRAM, cechująca się dodatkowym trybem pracy blokowej przy zapisie (*Block-Write*). Pojedyncze układy pamięci mają szerokość 32 bitów, co stanowi korzystny czynnik przy obsadzie 2 – 4 MB. Częstotliwość zegara taktującego osiąga również zakres 100 MHz.

MDRAM

Multibank-DRAM należy do grupy synchronicznych pamięci DRAM. W odróżnieniu od SDRAM i SGRAM (w trybie *burst* współpracują ze sobą 2 banki) mamy tu do czynienia z przypadkiem nakładania się na siebie 8 banków. Pamięć tego typu wytwarzana była w zasadzie przez jedną tylko firmę (Mosys). Bazowała na logicznych jednostkach pamięci 256 kB, każda po 8 banków 32 kB. Na rynku powszechne były układy 1 MB, integrujące w sobie cztery jednostki pamięci.

Technologia ta umożliwiała (w skali laboratoryjnej) taktowanie zegarem 100 – 125 MHz, ale rozrzut parametrów i niedoskonałość wykonania elementów w produkcji masowej ograniczał częstotliwość do 85 MHz. Z pamięcią MDRAM współpracował kontroler ET-6000, będący dziełem firmy Tseng.

Fizyczna szerokość magistrali wewnętrznej kontrolera ET-6000 wynosi w zasadzie 32 bity, ale jej organizacja jest dość nietypowa. Sterownik wymienia dane z pamięcią przez jeden z dwóch portów o szerokości 16 bitów. Każdy z nich wyprowadza multipleksowaną szynę komunikacyjną, którą przesyłane są adresy i dane. Szyny taktowane są sygnałem zegarowym pamięci (ok. 100 MHz), a przesyłanie odbywa się zarówno podczas narastającego, jak i opadającego zbocza zegara. Szczytowa przepustowość takiej szyny sięga więc (w przeliczeniu na jeden układ MDRAM) ok. 400 MB/s. Dwa bloki pamięci cechują się wydajnością wykraczającą poza parametry typowe dla pamięci V-RAM. Pamięć MDRAM, podobnie jak wiele innych rozwiązań, przeszła już do historii.

RDRAM

Mianem tym określana jest pamięć Rambus. Zapewnia ona stosunkowo duży transfer (po 500 MB/s na każdy układ), zawdzięczany głównie taktowaniu bardzo szybkim zegarem 250 MHz. Dane przenoszone są podczas obydwu zboczy zegara. Tak szybka technika stwarza oczywiście szereg problemów. Rosną wymagania odnośnie stosowanych materiałów i precyzji wykonania.

Szerokość magistrali pojedynczego układu ograniczona została do 8 bitów, co bynajmniej nie ułatwia integracji w strukturach aktualnie rozwijanych sterowników. RDRAM wymaga specjalnego sterownika pamięci, co naturalnie podnosi koszty produktów.

Sterowniki łączące dwa kanały RDRAM (każdy po 667 MB/s) pozwoliły na przełamanie po raz pierwszy bariery 1 GB/s.

V-RAM

Specjalny typ pamięci opracowany przez firmę Texas Instruments. Skrót V-RAM (Video-RAM) nie oddaje w pełni istoty sprawy. Z punktu widzenia kontrolera graficznego układy pamięci V-RAM zachowują się jak normalne pamięci DRAM. Osiągane pasmo przenoszenia przy zapisie magistralą 64-bitową nie przekracza 200 MB/s. Cechą szczególną jest niezależny port[9] wyjściowy prowadzący do przetwornika RAM-DAC. Transfer na tym odcinku sięga 360 MB/s, a kontroler nie musi tracić cykli zegarowych na wytworzenie sygnału dla monitora. Nie występuje tu, charakterystyczne dla wszystkich innych typów pamięci zjawisko stopniowego blokowania ograniczonego przecież pasma przepustowego magistrali, w miarę wzrostu rozdzielczości i ilości odtwarzanych kolorów. Można więc zachować wysoką (lub co najmniej ergonomiczną) częstotliwość odświeżania ekranu przy pracy w trybach o wysokiej rozdzielczości rzędu 1600 × 1280.

Wbrew powszechnie panującej opinii, pobór danych z portu wyjściowego nie jest całkowicie niezależny od cykli zapisu. Odczyt pamięci V-RAM przebiega według następującego algorytmu:

- Cykl inicjowany jest przez kontroler, który adresuje punkt początkowy żądanego obszaru.
- Układ V-RAM wyprowadza automatycznie bit po bicie zawartość kolejnych komórek tego obszaru do specjalnego rejestru przesuwnego.
- Stojące do dyspozycji w powyższym rejestrze dane mogą być pobierane przez przetwornik RAM-DAC.
- Do wyczerpania zawartości rejestru pamięć może być zapisywana. W tym sensie możliwy jest jednoczesny zapis i odczyt.

Pamięci V-RAM obciążone są opłatami licencyjnymi na rzecz wynalazcy, firmy Texas Intstruments, przez co zbyt drogie dla rynku powszechnego konsumenta.

WRAM

WRAM (Window RAM) stanowi wariant V-RAM, poszerzony o pewne dodatkowe, realizowane sprzętowo funkcje blokowe (*Aligned Move* i *Fill*). Układy tego typu wytwarzane były wyłącznie przez firmę Samsung i montowane na wczesnych wersjach kart (Millenium-1) firmy Matrox.

[9] Stąd też pochodzi bardziej precyzyjna nazwa tego typu pamięci: Dual Ported RAM.

RAM-DAC

Układ RAM-DAC[10] (*Random Access Memory-Digital/Analog Converter*) stanowi stopień wyjściowy karty graficznej i przetwarza zakodowany cyfrowo obraz pobierany z pamięci na analogowe sygnały RGB sterujące tor wizyjny monitora.

Prędkość maksymalna, z jaką może przebiegać ta konwersja (*Pixel Clock*), leży w zakresie od około 130 MHz (tanie produkty powszechnego użytku) do prawie 300 MHz (akceleratory najnowszej generacji). Rozwój technologii ostatnich lat pozwolił również na stopniowe integrowanie przetworników w obrębie struktury scalonej samego akceleratora, co wcześniej było niemożliwe. Szybkie przetworniki (powyżej 200 MHz) wytwarzane były wyłącznie jako niezależne układy scalone. Wymagana częstotliwość taktowania wynika z aktualnie ustawionej rozdzielczości i częstotliwości odświeżania obrazu, a nie zależy od ilości dostępnych w danym trybie barw. Sam fakt obecności na karcie przetwornika klasy 220 MHz nie oznacza bynajmniej, że pracuje on stale z tą częstotliwością.

Szerokość pasma magistrali pamięciowej prowadzącej do przetwornika jest wykładnikiem częstotliwości odświeżania. Zależy ona oczywiście od parametrów pamięci, a konkretnie szybkości, z jaką można ją odczytywać. Na nic przecież nie zda się tryb pracy w wysokiej rozdzielczości z ogromną ilością dostępnych barw, jeżeli obraz na ekranie drga[11] i oczy odmawiają posłuszeństwa.

Może się zdarzyć, że ze względu na posiadany monitor nigdy nie wykorzystamy pełnych możliwości oferowanych przez RAM-DAC. Aktualna częstotliwość pracy wynika z następującego obliczenia:

$$f_{RAMDAC} = k \times p \times l \times f_H$$

gdzie:

k – współczynnik bezpieczeństwa, zapewnia rezerwę na okresy powrotów linii i ramki,

p – ilość punktów w linii,

l – ilość linii na ekranie ($p \times l$ opisuje rozdzielczość obrazu w pikslach, np. 800×600),

f_H – częstotliwość odświeżania obrazu.

Łatwo teraz obliczyć, do czego naprawdę potrzebny jest RAM-DAC klasy 220 MHz. W trybie 1024×768 przy 75 Hz wystarczy przecież częstotliwość 79 MHz. Dopiero dla 1600×1200 i 85 Hz zegar taktujący musi przełączyć się na najwyższe obroty. Genero-

[10] W zasadzie są to trzy niezależne przetworniki, po jednym dla każdego z kolorów R, G, B. Czasami można spotkać bardziej rozbudowane rozwiązania, jak np. te z wyjściami na dwa monitory. Karta posiada wtedy 2 niezależne zestawy po 3 przetworniki RAM-DAC.

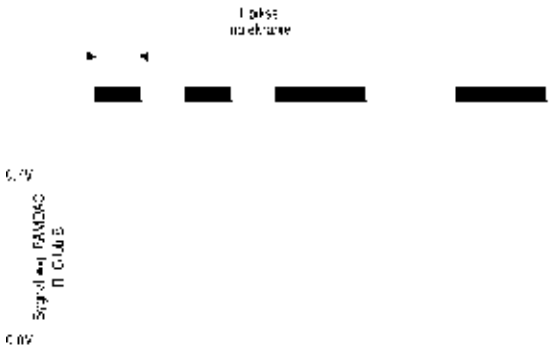
[11] Za dolną granicę postrzegania takich drgań przyjmuje się (nie wnikając w obowiązujące w tej dziedzinie przeróżne normy instytucji ochrony pracy) w praktyce 72 – 75 Hz. Ocena jest często mocno subiektywna, zależy od treści obrazu i konkretnego człowieka. Jeszcze większe rozbieżności panują przy określaniu górnej granicy. Twierdzenie, że im więcej, tym lepiej, jest w ostatnim czasie poddawane ostrej krytyce.

wane przez RAM-DAC przebiegi powinny być impulsami prostokątnymi o możliwie stromych zboczach. Stopień wyjściowy przetwornika to 3 źródła prądowe (dla każdego z kolorów RGB oddzielnie), tak dobrane, by wytwarzały na rezystancji 37,5 Ω [12] następujące poziomy napięcia:

- 0 V dla punktu czarnego,
- 0,7 V dla punktu o maksymalnej jaskrawości (biały kolor gdy R=G=B= 0,7 V).

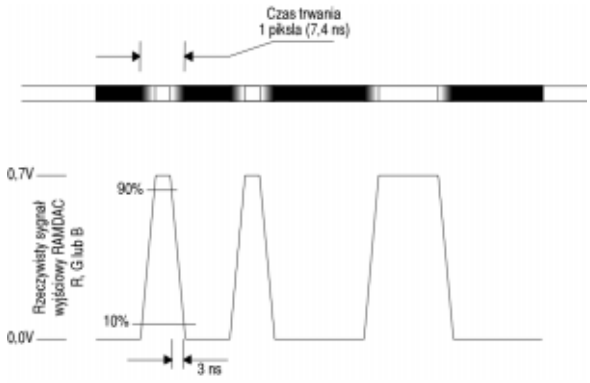
W stanie idealnym przetwornik wytwarza przebieg prostokątny o tak wąskich impulsach, by można było zapalać i gasić nawet pojedyncze piksele ekranu.

Rysunek 15.11. *Idealny przebieg prostokątny*



Każdy z punktów może wówczas świecić pełnym blaskiem przez cały czas trwania impulsu. Prawa fizyki ograniczają niestety stromość zboczy. Sygnał nie może narastać nieskończenie szybko, mamy wyraźnie zaznaczone fazy narastania i opadania. Faza pełnej jasności trwa nieco krócej, niż szerokość impulsu mierzona u podstawy.

Rysunek 15.12. *Rzeczywisty sygnał generowany przez układ RAM-DAC*



Na ekranie o wysokiej rozdzielczości pojedynczy biały punkt na czarnym tle w pierwszej połowie swego życia znajduje się w fazie gdzieś między czarnym a białym chociaż taka barwa wcale nie jest pożądana. To samo ma miejsce przy gaszeniu piksela. W trybie

[12] Wartość stanowi wypadkową otrzymaną z równoległego połączenia impedancji obciążenia obydwu końców linii: dwa razy po 75 Ω. Niedopasowanie impedancji może być przyczyną szkodliwych zjawisk o charakterze falowym (odbicia i interferencje). Ich obecność objawia się na ekranie w formie uciążliwych efektów (mory, zwielokrotnianie krawędzi), psujących jakość obrazu i męczących oczy.

1 280 i częstotliwości RAM-DAC 135 MHz czas życia punktu można określić na 7,4 ns (1/135 MHz). Firma S3 wymienia na liście parametrów dla swoich zintegrowanych przetworników czasy narastania i opadania 3 ns mierzone na poziomach 10 i 90%.

Powyższe liczby unaoczniają obecność fizycznych granic ostrości obrazu. Sytuacji nie polepszają również często stosowane filtry wyjściowe typu Π, instalowane w celu ograniczenia skutków szkodliwego promieniowania.

Rysunek 15.13. *Filtr pomiędzy układem RAM-DAC a monitorem*

Strome zbocza produkują przebiegi harmoniczne sięgające zakresu GHz. Promieniowanie tego typu trzeba wyeliminować, tak by produkty mogły uzyskać stosowne certyfikaty (w Europie CE, w USA FCC). W tyle nie pozostają również producenci monitorów, którzy w trosce o redukcję zakłóceń emitowanych przez swoje wyroby, zaopatrują wszelkie przewody sygnałowe w opaski z pierścieni ferrytowych.

Dopasowanie monitora do karty

Kombinacja monitora z kartą graficzną stanowi o jakości, z jaką prezentuje się cały komputer wraz ze swoją mocą obliczeniową. Moment zakupu karty graficznej powinien być poprzedzony solidnym przemyśleniem. Decyzja jest tym bardziej trudna, im szerszy jest wybór. Specjaliści od marketingu starają się zasypywać potencjalnego klienta ogromem dobrze brzmiących pojęć, które często nie stanowią same w sobie nic szczególnego. Testy porównawcze w gazetach komputerowych i Internecie pozwalają na udowodnienie każdej tezy, bowiem zawsze można dobrać taki program testujący, by wykazać to co się zamierza.

Dobrze jest wiedzieć, które z głównych parametrów karty i monitora odgrywają decydującą rolę i jakimi kryteriami należy się kierować przy wyborze określonego produktu. Rzadko kto nie musi uwzględniać czynników ekonomicznych, ale nawet ktoś mogący sobie pozwolić na wszystko powinien sobie zdawać sprawę z tego, co dany produkt może mu zaoferować i czy jest w ogóle w stanie wykorzystać jego zalety.

Parametry karty

Przy obecnej sytuacji cenowej stanowczo należy odradzić nabywanie jakichkolwiek kart z pamięcią poniżej 4 MB, bowiem są one tylko minimalnie tańsze od obsady 8 MB. Koszt ewentualnego późniejszego rozszerzenia pamięci jest zawsze większy i też pod warunkiem, że dana karta w ogóle przewiduje taką możliwość, a potrzebne „kostki" są jeszcze gdzieś do zdobycia. Wszystkie nowoczesne akceleratory 3D mają co najmniej 16 MB, a niektóre 32 MB. Warto sobie jednak zdawać sprawę, iż tak duża ilość pamięci niczego nie przyspiesza, jak długo nie wchodzimy w zakres zastosowań 3D.

2 MB EDO-RAM wystarczy w zupełności do zwykłych zastosowań komputera klasy PC pracującego pod kontrolą systemu operacyjnego z interfejsem graficznym. Karty tej klasy obrazują tekst, tabele i proste rysunki przy rozdzielczości 800 × 600 w trybie True-Color. Zakres High-Color pokrywany jest do rozmiarów 1024 × 768, a jeśli zadowala nas 256 kolorów, można sięgnąć po raster 1280 × 1024 jeszcze z ergonomiczną częstotliwością odchylania 75 Hz. Cieszący się dużą popularnością, ze względu na lepsze proporcje format 1152 × 1024, nakłada ostrzejsze wymagania na monitor i nie zawsze jest wspomagany przez sterowniki graficzne. Praca w trybach powyżej 800 × 600 może być uciążliwa, jeśli przewody połączeniowe nie są dobrej jakości (uwaga na wszelkie przełączniki i przedłużacze). Często zmusza to do wyboru czcionki ekranowej o większych wymiarach.

W zastosowaniach DTP i podobnych sięgnąć trzeba wyżej. Praca w trybie True-Color przy rozdzielczości 1280 × 1024 lub ergonomiczne 1600 × 1200 w High-Color wymaga obsady 4 MB (V-RAM lub WRAM) przy współudziale przetwornika RAM-DAC co najmniej 220 MHz. Osiągnięcie częstotliwości odświeżania powyżej 69 Hz w obecności przetwornika klasy 175 MHz nie jest w tych trybach możliwe.

Dzisiaj już prawie każda karta VGA ma wbudowane funkcje akceleratora wideo, tzn. jest zdolna do odtwarzania sekwencji wideo z dysku kompaktowego lub twardego i prezentowania ich na ekranie monitora z całkiem przyzwoitą ilością klatek na sekundę (tego typu przyspieszanie funkcji wideo nie ma nic wspólnego z digitalizacją lub obróbką materiału filmowego przejmowanego ze źródeł zewnętrznych).

Sekwencje wideo w formacie AVI nie stanowią nadmiernego obciążenia dla komputera klasy PC. Z kolei MPEG-1, szyty na miarę procesora Pentium 100, nawet przy pracy pełnoekranowej nie wymaga żadnej specjalnej karty graficznej. Większość znajdujących się na rynku produktów osiągnęła pewien stosunkowo wysoki poziom rozwoju, a niewielkie między nimi różnice wychwycić można jedynie przy pomocy specjalnych programów testujących. Jeśli zadowala nas projekcja z prędkością 10 fps, przy niezbyt wyszukanej jakości dźwięku, wystarczy nawet maszyna 486 i odtwarzacz software'owy Xing. MPEG-I wspomagany sprzętowo gwarantuje pełne 25 fps i stereofoniczny dźwięk nawet na komputerze 486.

To co w istotny sposób odróżnia od siebie poszczególne chipy graficzne, to podejście do skalowania okna wideo (*zoom*). Kontroler typu Virge interpoluje w poziomie (wytwarza dodatkowe piksle w obrębie linii), w pionie natomiast dubluje ślepo linie, na czym cierpi jakość obrazu.

Karty z wyjściem TV (SCART, S-Video), często wyposażone jeszcze w modulator HF, są przeznaczone raczej do prezentacji obrazów niż do ich opracowywania i tworzenia. Czcionki ekranowe w tym formacie są trudno czytelne. Ograniczenia techniczne wbudowane w standardy telewizyjne (pasmo toru wizji i zależności czasowe toru odchylania) nie pozwalają na wyjście poza rozdzielczość 800 × 600.

Tuner telewizyjny realizowany jest najczęściej w formie dodatkowego modułu nasadzanego na przystosowaną do tego celu kartę graficzną. Połączenie odbywa się własną niestandardową magistralą lub ujednoliconym złączem *Scenic Highway* wprowadzonym przez firmę S3. Można też spotkać rozwiązania kombinowane, np. All-in-Wonder firmy

ATI (SVGA, niezły akcelerator 3D, wejście i wyjście wideo oraz tuner telewizyjny). Komputer PC wyposażony w tuner z dekoderem teletekstu ma szerokie pole do popisu przy automatycznym przetwarzaniu szerokich strumieni informacji, np. notowań giełdowych. Wybrane strony można wiązać w sekwencje i zapisywać do plików w formacie AVI (np. 320 × 240 / 25 fps).

Miłośnicy gier komputerowych z pewnością nie potrzebują drogich zestawów opracowanych dla potrzeb CAD. Surowy i co najwyżej cieniowany obraz jest nie do zaakceptowania, tu oczekuje się realizmu w możliwie najwyższej formie, a nie precyzji modeli siatkowych.

Użytkownik PC, który posługuje się komputerem jako rozbudowaną maszyną do pisania nie zauważy różnicy między drogim produktem markowym z 32 MB pamięci SGRAM, a tanim egzemplarzem no-name z obsadą 2 MB RAM, jeśli tylko częstotliwość odświeżania obrazu przewyższa nieco 70 Hz. Rozsądnym kompromisem na co dzień jest kompatybilność SVGA, a w miarę potrzeb akcelerator 2D lub 3D.

Najważniejsze parametry karty graficznej skupiają się nadal w obrębie pamięci graficznej: jej rozmiaru, typu i szybkości. Czynniki te decydują o rozdzielczości, ilości odtwarzanych barw oraz o częstotliwości odświeżania obrazu.

Jakość monitora

Ostatnim ogniwem w łańcuchu przetwarzania informacji w formie graficznej jest monitor. Pada on najczęściej ofiarą cięć oszczędnościowych, szczególnie przy zakupie kompletnych systemów przez początkujących niedoświadczonych użytkowników. Publikowane czasem dane statystyczne potwierdzają smutną prawdę, że dużo łatwiej przekonać jest klienta do zakupu doskonałej karty graficznej, niż doinwestowania do monitora. Tymczasem dopasowanie parametrów zainstalowanej graficznej do posiadanego monitora pełni kluczową rolę w optymalnym wykorzystaniu całości dostępnych zasobów.

Poziom techniczny obrazu na ekranie zależy nie tylko od klasy monitora, ale również od czystości sygnału wytwarzanego przez kartę. Podobnie jak większość członów składowych systemu PC, ich jakość może być różna. Uważać należy szczególnie przy zakupie zestawów komputerowych. Ostra kalkulacja cenowa stanowi dużą szansę trafienia na oba składniki o stosunkowo słabej jakości.

Optymalne dopasowanie parametrów monitora i karty odgrywa kluczową rolę. Słaby monitor nie wykorzysta wysokorozdzielczych trybów oferowanych przez sterownik. Dobry monitor marnuje się, współpracując z kartą o marnej jakości. W łańcuchu połączonych ze sobą elementów decyduje najsłabsze ogniwo.

Będące w powszechnym użyciu monitory dają się zaliczyć do jednej z grup. Podstawą do takiej klasyfikacji jest dopuszczalna częstotliwość odchylania poziomego. Aby monitor mógł pracować w określonym trybie, musi w pierwszym rzędzie sprostać wymogom częstotliwości odchylania poziomego. Ergonomiczna częstotliwość odświeżania leży nieco powyżej 72 Hz i ten warunek spełnia prawie każdy monitor. Program konfi-

guracyjny sterownika karty graficznej proponuje często w wysokich trybach pracy częstotliwości odchylania dochodzące do 200 Hz. Zawsze jednak można zażądać redukcji do 75 Hz, z korzyścią dla zmniejszenia obciążenia magistrali pamięciowej karty.

Tabela 15.3.
Ergonomiczny tryb pracy monitora, w zależności od maksymalnej częstotliwości odchylania pionowego

f_H max [kHz]	Ergonomiczne tryby pracy
38	640 × 480 / 72 Hz
48	800 × 600 / 72 Hz
64	1024 × 768 / 80 Hz
	1152 × 864 / 71 Hz
78	800 × 600 / 124 Hz
	1024 × 768 / 97 Hz
	1280 × 1024 / 73Hz
	1152 × 864 / 86 Hz
85	1024 × 768 / 106 Hz
	1152 × 864 / 94 Hz
	1280 × 1024 / 80 Hz
	1536 × 1152 / 71 Hz
102	1280 × 1024 / 94 Hz
	1536 × 1152 / 83 Hz
	1600 × 1200 / 80 Hz

Dany tryb pracy nakłada określone wymogi na pasmo toru wizyjnego zainstalowanego monitora. Wartości minimalne zebrane zostały w tabeli 15.4.

Tabela 15.4.
Minimalne wymagania toru wizyjnego monitora

Rozdzielczość	f_V [Hz]	f_H [kHz]	Pasmo wideo [MHz]
640 × 480	75	38	25 – 30
800 × 600	75	48	45 – 50
1024 × 768	75	62	70 – 80
1280 × 1024	75	80	120 – 135
1600 × 1200	75	95	200 – 220

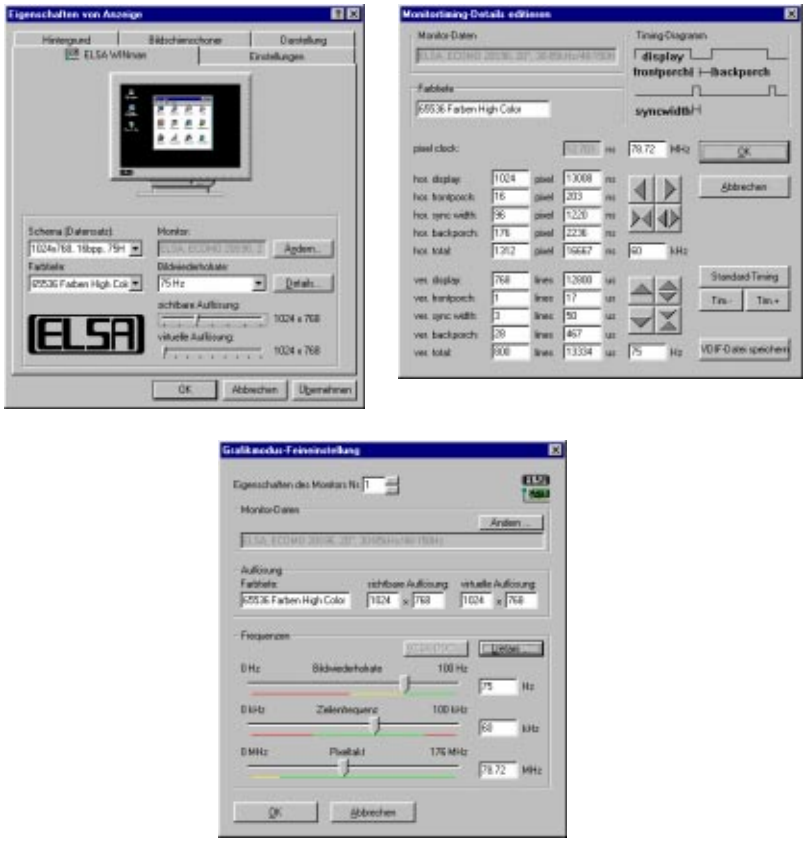
Przekroczenie określonej dla danego monitora maksymalnej częstotliwości odchylania poziomego o więcej niż 5% stanowi w pierwszej mierze niebezpieczeństwo zerwania synchronizacji, a teoretycznie nawet i uszkodzenia, chociaż nowe modele bronią się przed takim traktowaniem przy pomocy wbudowanych mechanizmów ograniczających.

Wykroczenie poza pasmo toru wizyjnego nie stanowi natomiast żadnego zagrożenia. Po prostu sygnał o wysokiej jakości nie będzie odtwarzany wraz z wszystkimi szczegółami (nieostry obraz).

Programy instalacyjne

Procedury konfiguracyjne dołączane do karty graficznej próbują wydobyć to, co najlepsze z oferowanych im zasobów. Dobre programy instalacyjne potrafią rozpoznać monitor, jeśli naturalnie jest to możliwe (DDC). Często typ monitora można wyłowić z bazy danych programu. Jeśli nie można go tam znaleźć, pozostaje przestudiowanie instrukcji obsługi i ręczne wprowadzanie danych.

Ilość możliwych do ustawiania opcji jest różna. Mało który z programów prezentuje ich tak wiele, jak instalator produktów Elsa:



Rysunek 15.14. *Możliwości konfiguracji oferowane przez program obsługi karty Elsa*

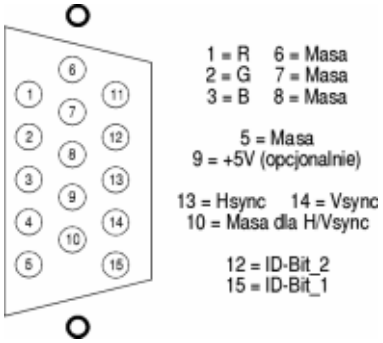
Nieumiejętne posługiwanie się tak potężnymi narzędziami może spowodować uszkodzenie mało odpornych monitorów.

Kanał informacyjny VESA DDC

DDC (*Display Data Channel*) jest znormalizowanym przez VESA złączem komunikacyjnym pomiędzy monitorem a kartą graficzną. Wymiana danych na tym odcinku umożliwia wprowadzenie w życie idei Plug&Play w stosunku do monitora, który może przekazać swoje preferencje sterownikowi graficznemu.

Twórcom systemu przyświecał na szczęście cel nadrzędny, jakim było maksymalne uproszczenie całego przedsięwzięcia, między innymi poprzez wykorzystanie istniejącego okablowania. Typowy przewód łączący analogowy monitor ze sterownikiem VGA nie jest bowiem w pełni wykorzystany[13].

Rysunek 15.15. *Rozmieszczenie sygnałów w złączu monitor-karta graficzna*



Kanał DDC posługuje się liniami 12 i 15. W zależności od ich znaczenia wyróżniamy trzy odmiany systemu, zaprezentowane poniżej.

DDC1

DDC1 jest typowym kanałem jednokierunkowym, w którym to jedynie monitor przekazuje informacje do karty sterownika. Transmisja ma charakter synchroniczny, odbywa się linią 12. Sygnał zegarowy pobierany jest z końcówki 14 (synchronizacja pozioma).

W każdym takcie zegara monitor wysyła 128-bitowy blok informacyjny EDID (*Extended Display Identification*). Blok taki zawiera, oprócz funkcji identyfikacyjnych (typ urządzenia, producent, wymiar przekątnej ekranu itp.), również istotne dane techniczne, takie jak maksymalnie dozwolone częstotliwości odchylania, zalecane parametry czasowe i amplitudowe przebiegów synchronizujących oraz poziomy sygnałów wideo. Monitor może też przekazać współczynnik nieliniowości swojego toru wideo (funkcja Gamma) oraz uznane przez siebie tryby DPMS. Znajomość funkcji Gamma monitora upraszcza czasochłonny proces kalibracji barw.

[13] Uwaga na tanie przedłużacze i przełączniki (*Switch-Box*) z niepełną obsadą kontaktów.

DDC2B

DDC2 jest – w przeciwieństwie do szeregowego przekazu złączem DDC1 – magistralą pracującą w oparciu o protokół I^2C. Kanał tego typu umożliwia już obustronną wymianę informacji. Potrzebny jest jednak drugi przewód łączący. Linia 15 przejmuje funkcję sygnału SCL, a impulsy SDA przesyłane są końcówką 12. DDC2B ma okrojoną listę dostępnych operacji. Sterownik może w zasadzie tylko zażądać od monitora przesłania bloku EDID lub obszernego zestawu danych skupionych w bloku VDIF (*VESA Display Interface File*)

DDC2AB

Jest rozszerzeniem DDC2B o funkcje znane z systemu Access.Bus. Możliwe jest więc przekazywanie rozkazów sterujących do monitora, np. dla skorygowania położenia obrazu lub regulacji jaskrawości.

W chwili obecnej na rynku obecne są niektóre urządzenia zgodne ze standardem DDC1 i ewentualnie DDC2B. Należy jednak nadmienić, że nie brak innych rozwiązań, choćby z zastosowaniem klasycznego złącza szeregowego RS-232, magistrali Access.Bus, a ostatnio ponownie odkrytego USB.

Podział mocy obliczeniowej

Akceleratory 2D ograniczały się głównie do wyzwalanego jednym rozkazem CPU przesuwania i wypełniania obrazów prostokątnych.

Komplikacja zagadnień związanych z realistycznym przedstawianiem obiektów 3D przewyższa do dnia dzisiejszego stopień rozwoju techniki komputerowej. Nieodzowne stało się więc sprzętowe wspomaganie jednostki centralnej przy wykonywaniu funkcji graficznych. Budowa obrazu 3D w czasie rzeczywistym stanowi, mimo stosowania rozbudowanych akceleratorów, nadal stosunkowo duże obciążenie dla CPU.

Skuteczność działania procesorów wspomagających zależy oczywiście w dużej mierze od obranej strategii działania, tj. określenia grupy zadań przejmowanych od centralnego procesora systemowego. W akceleratorach nierealizujących funkcji T&L (patrz punkt: „Geometry Engine”), kluczową rolę pełniło dopasowanie mocy obliczeniowej współpracujących ze sobą jednostek (z uwzględnieniem przepustowości łączących je magistral). Najnowsza generacja chipów graficznych przejmuje obliczenia geometryczne i minimalizuje udział CPU w procesie przetwarzania, dlatego dalsza część rozdziału odnosi się wyłącznie do układów pozbawionych T&L.

Jakkolwiek akcelerator 3D można pod wieloma względami uznać za wyspecjalizowany i niezależny procesor graficzny, przetwarza on dane podsuwane mu przez CPU. Pełne wykorzystanie potencjału obliczeniowego takiego podsystemu możliwe jest jedynie w warunkach gwarantujących stały dopływ materiału wejściowego. Ten optymalny stan ma rzadko miejsce i ograniczany jest zarówno przez wydajność centralnego procesora

systemowego, jak i przepustowość magistrali wewnętrznej, doprowadzającej dane do akceleratora. Akceleratory starszych generacji zdane były początkowo wyłącznie na dominującą powszechnie magistralę PCI. Jej specyfikacja określa szczytową prędkość przesyłania danych na 133 MB/s. Wprowadzony później standard AGP (*Accelerated Graphics Port*) dopuszczał w swym podstawowym trybie 1X maksymalny transfer do wysokości 264 MB/s.

Powróćmy do podziału zadań. CPU, posługując się swoją jednostką zmiennoprzecinkową, przygotowuje parametry trójkątów elementarnych i przesyła je do akceleratora, a ściśle mówiąc do procesora rasteryzującego. Przekaz odbywa się magistralą (PCI, ew. AGP) w postaci specjalnie uformowanych pakietów danych. Spróbujmy oszacować szerokość pasma, które trzeba poświęcić na powyższą komunikację. Jeśli przyjąć, że każdy narożnik definiowany jest przy pomocy 120 bajtów[14], średnio skomplikowana sceneria, rozłożona na 10 000 trójkątów, stanowi blok danych o rozmiarach 1,2 MB. Przy założeniu, że akcelerator ma nieograniczoną wydajność (i przyjmuje oraz przetwarza dowolną ilość informacji), a wymagana szybkość odtwarzania obrazu 3D ma wynosić 25 klatek na sekundę, generowany jest strumień o szerokości 30 MB/s.

Miarą rzeczywistej wydajności akceleratorów 3D (określanych w tym kontekście często jako 3D-Engine) jest naturalnie zdolność do przetwarzania pewnej liczby trójkątów w jednostce czasu. Wielkość tę wyraża się w jednostkach tps (*Triangles per Second*) lub Pol/s (*Polygons per Second*), uzupełnianych stosownymi przedrostkami k lub M. Podawane przez producentów liczby towarzyszące oferowanym na rynku produktom należy przyjmować ze stosowną rozwagą. Stanowią one zwykle wymiar szczytowych możliwości samego akceleratora, które trudno osiągnąć w warunkach realnych. Już samo rozciągnięcie okna na cały ekran może położyć kres optymistycznej kalkulacji.

Maksymalna częstotliwość odbudowywania obrazu 3D powiązana jest ścisłą zależnością z zapasem mocy obliczeniowej akceleratora i stopniem komplikacji obiektów (sumaryczną liczbą trójkątów). Karta o wydajności 200 000 tps może budować i wyświetlać obraz 3D z prędkością 10 fps tylko pod warunkiem ograniczonego stopnia komplikacji sceny: tutaj nie więcej niż 20 000 trójkątów.

Kres możliwości magistrali PCI leży, jak wiadomo, w okolicach 133 MB/s. Oddając całą przepustowość szyny na potrzeby grafiki 3D, można przesłać około 1,1 miliona trójkątów, ale oczywiście przy założeniu, że ilość taką jest w stanie przygotować centralny procesor. Moc obliczeniowa procesorów rośnie w takim tempie, że warunek ten jest w zasadzie zawsze możliwy do spełnienia, pod warunkiem pominięcia nadmiernie skomplikowanych zagadnień związanych z oświetleniem.

Koprocesor graficzny średniej klasy (*Voodoo*) rysuje przy włączonej filtracji bilinearnej około 45 MTex/s (45 milionów teksturowanych pikseli na sekundę). Z ograniczeń magistrali PCI wynika maksymalny rozmiar kreślonego trójkąta:

45 milionów pikseli/1,1 miliona trójkątów = 41 pikseli/trójkąt

[14] 3 narożniki po 10 parametrów (xyz, uvw, rgba) dają razem 30 stałych. Jeżeli przeznaczyć po 4 bajty (najprostszy format zmiennoprzecinkowy) na każdą z nich, otrzymujemy 120 bajtów danych.

Scenerie typowych gier zbudowane są z dużo większych trójkątów, tak że czynnikiem hamującym byłby w tym przypadku akcelerator 3D, a nie CPU. Przedstawiona kalkulacja dowodzi, że wąskie gardło na linii wymiany danych może się przemieszczać w zależności od stopnia komplikacji sceny. Przy dużych trójkątach oczekuje CPU, a przy małych akcelerator.

Trzeba pamiętać, iż akceleratory 3D wymagają przydziału przerwania sprzętowego. Dlatego w BIOS-SETUP należy zawsze sprawdzić położenie opcji ***Assign IRQ to PCI VGA*** (prawidłowo: *Enabled*). Jeżeli przydział jest zablokowany, prowadzi to do sytuacji, w której co prawda sterowniki programowe dla akceleratora dają się zainstalować (i brak jest konfliktów), ale w oknie menażera sprzętowego pojawia się charakterystyczny żółty wykrzyknik.