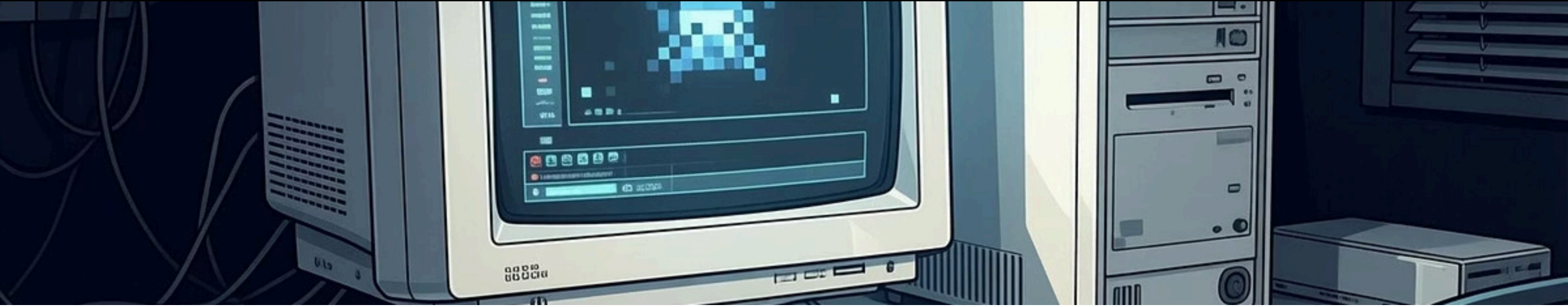


Wolfenstein 3D

A Engenharia por Trás de um Clássico

CAPÍTULO 1 — INTRODUÇÃO (21-26)

CAPÍTULO 2 — HARDWARE (27-39)

Game Engine Black Book: Wolfenstein 3D - Fabien Sanglard

INTRODUÇÃO

Lançamento e Código Aberto

🏆 Fundador de um Gênero

O **Wolfenstein 3D** além de vender milhões de cópias, **definiu o gênero FPS** e demonstrou que o PC era uma plataforma legítima para games de alto impacto visual.

🎮 Sucesso

O Jogo vendeu cerca de 100.000 cópias em um ano e ganhou vários *mods* de fãs, que os fizeram por meio da **engenharia reversa**, sem conhecer os 'segredos' da velocidade e engenharia 3D.

🔓 Código Aberto em 1995

Ao contrário da prática da indústria de guardar motores gráficos a sete chaves, a **id Software de John Carmack** liberou o código-fonte do motor em 1995, permitindo adaptações para novos hardwares e sistemas operacionais ao longo das décadas.

CONTEXTO TÉCNICO

PC vs. Consoles em 1991

Em 1991, o **PC** era primariamente uma máquina de escritório. Consoles como o **Super NES** e o **Sega Genesis** dominavam os jogos com motores de sprites altamente eficientes e animações fluidas. Por que, então, apostar no PC?

O Problema dos Sprites

Para criar um mundo 3D imersivo, **sprites eram insuficientes**. A id Software precisava desenhar a tela *pixel a pixel* usando um **framebuffer**, algo que os consoles da época não conseguiam fazer.

O Poder da CPU do PC

A solução foi usar o **alto poder de processamento da CPU do PC**, que superava o dos consoles da época e era capaz de calcular cada frame em tempo real, a base do motor 3D da id Software.

DESAFIOS

Os Grandes Obstáculos Técnicos

A equipe da **id Software** enfrentou uma série de limitações brutais do hardware da época. Cada componente do PC apresentava um desafio único que precisava ser superado de forma criativa.

🖥️ Vídeo VGA

O sistema de vídeo **não suportava double buffering**, causando falhas visuais (screen tearing) durante a renderização de cenas em movimento.

⚙️ CPU Inteira

A CPU **só calculava números inteiros**, tornando operações com frações e trigonometria — essenciais para 3D — extremamente custosas em ciclos de clock.

📣 PC Speaker

O alto-falante padrão do PC **só emitia bipes rudimentares**, sem suporte a samples de áudio, tornando o som um desafio à parte.

💾 Memória

O **modelo de memória segmentada** do MS-DOS, a limitação de apenas **1 MB de RAM** e os barramentos de dados lentos completavam o quadro de dificuldades.

HARDWARE

A Pipeline do PC em 1991

O PC de 1991 pode ser dividido em **cinco subsistemas** que formam uma linha de montagem para o processamento de um jogo. A qualidade de cada componente variava enormemente — de aceitável a praticamente impossível de lidar.

✅ Aceitável

RAM e **Inputs** eram gerenciáveis, com limitações contornáveis.

⚠️ Problemático

Áudio era o maior pesadelo nessa categoria uma limitação difícil de contornar.

❌ Impossível

CPU e **Vídeo** eram os maiores pesadelos limitações que desafiariam qualquer programador da época.

CPU

Intel 80386: O Rei do Mercado

O desempenho de um PC na época era definido pelo seu processador. O **Intel 80386** era o grande rei do mercado, mas com peculiaridades importantes para os desenvolvedores de jogos.

1 386-DX vs. 386-SX

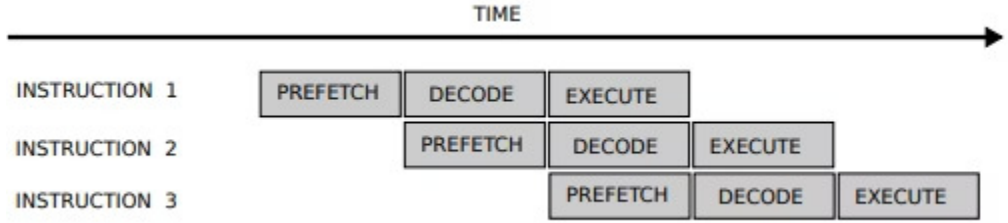
A Intel lançou duas versões: o **386-DX** (barramento de dados de **32 bits**) e o **386-SX**, internamente idêntico, mas com barramento de **16 bits** para baratear custos e reaproveitar placas-mãe do modelo **286**.

2 Compatibilidade com o 286

O sucesso do 386 se deu por manter **compatibilidade com programas antigos** do **286** e por adicionar um modo operacional de 32 bits.

3 Pipeline de 3 Estágios

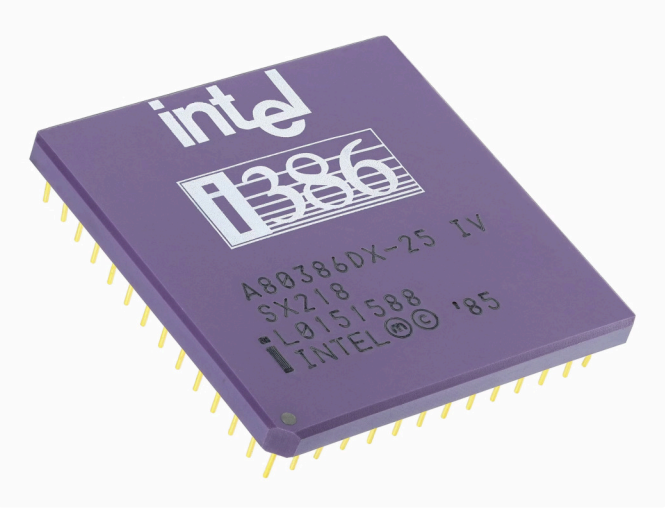
A CPU possuía um pipeline de **Busca (Prefetch) → Decodificação (Decode) → Execução (Execute)**. Sem cache SRAM interna e com decodificação lenta, mesmo instruções simples levavam **no mínimo 2 ciclos de clock**.

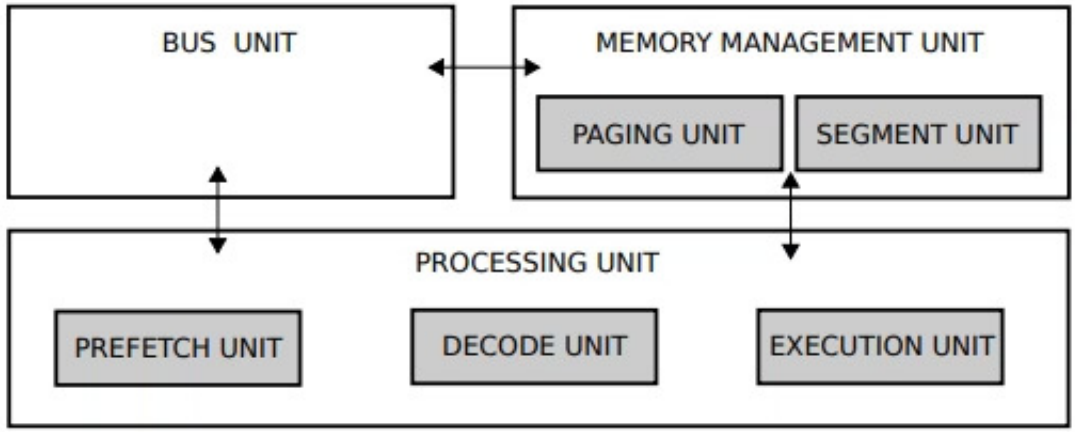


CPU 80386

Arquitetura Interna: Os Três Sistemas

A arquitetura interna do **Intel 80386** se organiza em três sistemas principais que atuam em conjunto. Eles controlam a comunicação externa, o gerenciamento de memória e a execução das instruções.





\

CPU

Intel 80386-DX 16 MHz die layout

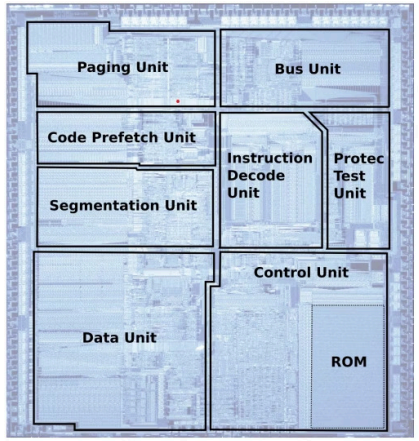


Figure 2.4: Intel 80386-DX 16 MHz die layout[8]

Os blocos visíveis incluem a **Data Unit**, **ROM**, **Page Unit**, **Bus Unit**, **Code Prefetch Unit**, **Segmentation Unit**, **Control Unit**, **Instruction Decode Unit** e **Protection Test Unit**. Em conjunto, esses componentes coordenam a busca, decodificação, proteção e execução das instruções.

Data Unit
Processamento de dados

ROM
Instruções e microcódigo

Page Unit
Paginação e memória

Bus Unit
Interface externa

Code Prefetch Unit
Busca antecipada

Segmentation Unit
Segmentação de endereços

Control Unit
Coordenação interna

Instruction Decode Unit
Decodificação

Protection Test Unit
Verificação de proteção

CPU 80386

O Custo das Instruções

Mesmo com pipeline, o Intel 80386 precisava de pelo menos 2 ciclos de clock por instrução.

Instrução	Ciclos de Clock
ADD reg16, reg16 (Addition)	2 ciclos
INC reg16 (Increment)	2 ciclos
IMUL reg16, reg16 (Integer Multiplication)	12-25 ciclos (depende da posição do bit mais significativo)
IDIV reg16, reg16 (Integer Division)	27 ciclos
MOV [reg16], reg16 (Move)	4 ciclos
OUT [reg16], reg16 (Output to Port)	25 ciclos
IN [reg16], reg16 ((Input from Port)	26 ciclos

PONTO FLUTUANTE

O Ponto Flutuante

O ponto flutuante é uma forma de representar números reais em **binário**. Ele divide os **32 bits** em partes para **sinal**, **expoente** e **mantissa**. Isso permite trabalhar com **frações** e valores muito **grandes** ou muito **pequenos**.

1 bit	8 bits	23 bits
Sinal	Expoente	Mantissa

PONTO FLUTANTE

O Dilema do Ponto Flutuante

A matemática de ponto flutuante era essencial para a **precisão** exigida pelos gráficos 3D, mas representava um desafio enorme no hardware de **1991**.

1

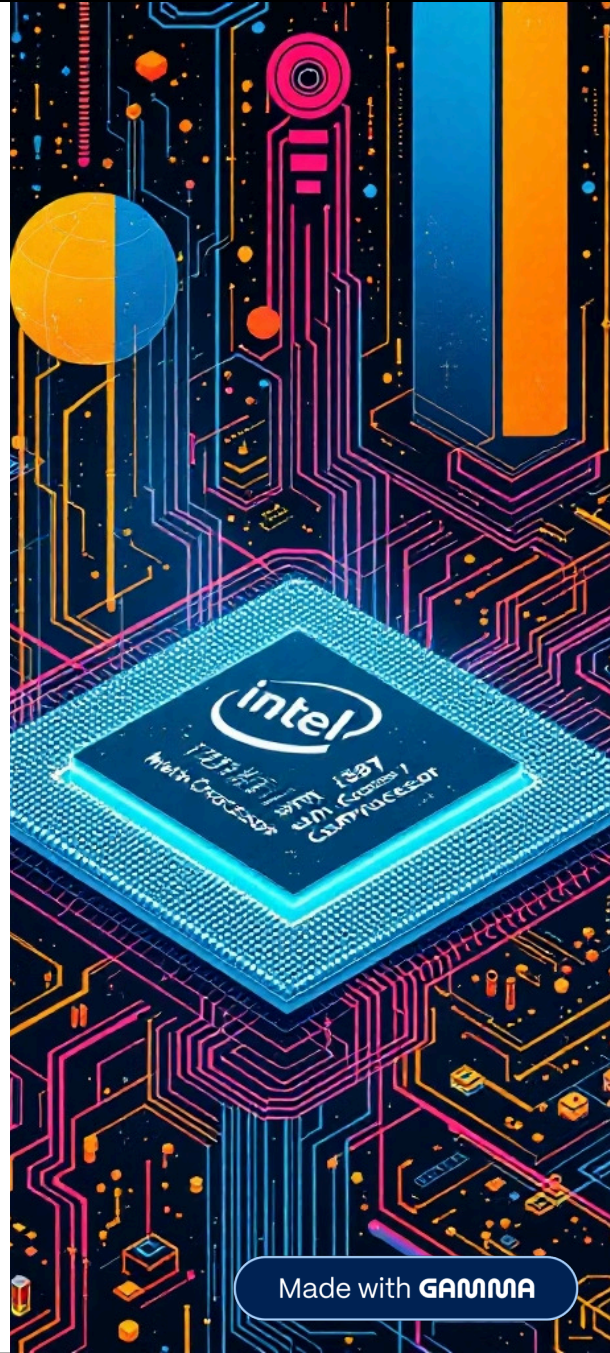
A Origem do Float em C

Mesmo com operações de ponto flutuante sendo lentas, **Dennis Ritchie** e **Ken Thompson**, entusiastas da astronomia, adicionaram os tipos `float` e `double` à linguagem C, antecipando que o hardware futuro (como o PDP-11) teria unidades de ponto flutuante dedicadas.

2

O Coprocessador i387

Para quem necessitava de processamento de ponto flutuante via hardware, a Intel oferecia o **i387 "Math CoProcessor"**. Apesar do preço exorbitante na época e desempenho apenas mediano, era uma peça de hardware cobiçada por cientistas e engenheiros.



PONTO FLUTANTE

O Dilema do Ponto Flutuante

Uma solução alternativa para lidar com **frações** usando apenas inteiros — multiplicar por uma constante (100 ou 1000) e dividir no final — era impraticável. No **Intel 386**, uma instrução de multiplicação levava de **12 a 25 ciclos**, e uma divisão custava impressionantes **27 ciclos**.

O resultado para desenvolvedores de jogos era um beco sem saída: inteiros eram rápidos, mas imprecisos para a complexidade 3D, enquanto o ponto flutuante era preciso, mas inaceitavelmente lento para a **renderização** em tempo real.