CO2 - CO-processador Gráfico COncorrente

Rui Guerreiro
Florival Janeiro
João Pereira

INESC - Rua Alves Redol, 9 - 1000 Lisboa
IST - Lisboa

E-mail: rmg@eniac.inesc.pt

Resumo

O Co-processador Gráfico Concorrente é um projecto de software e de hardware, tendo em vista a obtenção de um sistema gráfico para visualização de imagens 3D de alta resolução e com elevado grau de realismo, em tempo real.

O sistema tem uma resolução máxima de 1280 x 1024 pixels, com 24 bits/pixel. Esta resolução é configurável por software, o que permite a compatibilidade com outras normas, nomeadamente1024 x 768 e 640 x 480.

Os elementos processadores do sistema são Transputers T800. Cada Transputer tem a sua própria memória local, sendo a comunicação inter-processadores feita exclusivamente por mensagens através de portos série de alta velocidade, permitindo que o hardware seja construído de forma modular.

É utilizado um pipeline de visualização 3D que executa as transformações geométricas e processamento de rendering necessário à obtenção de imagens realistas a partir de uma descrição hierárquica da cena a visualizar.

A remoção de faces ocultas é feita através de um Z-buffer. É também executada uma pré-remoção de faces ocultas. Espera-se, com esta pré-remoção, obter uma melhoria de desempenho em cenas complexas.

O software é desenvolvido no sistema operativo Helios sob a forma de pipeline de processos programados sequencialmente segundo o binding ANSI C e que comunicam por mensagens. O paralelismo é expresso através da linguagem CDL (Component Distribution Language), que especifica as ligações inter-processos e a distribuição desses processos pelos processadores do sistema.

Software

O software do Co-processador Gráfico Concorrente foi desenvolvido sobre o Sistema Operativo Helios [HELIOS 89], por forma a utilizar racionalmente os recursos do hardware, nomeadamente os Transputers. É utilizada a estrutura de pipeline de processos comunicantes por mensagens, e programados segundo o binding C ANSI.

Estrutura dos Processos

O software deste co-processador gráfico está organizado num pipeline de processos, onde

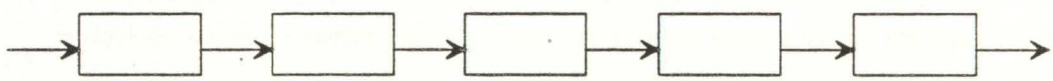

Fig 1 - Pipeline de processos

cada um executa uma função muito específica e, na medida do possível, com o menor esforço computacional possível. Esta organização dos processos foi escolhida devido à fácil adaptação de um pipeline de visualização 3D. O uso de comunicação por mensagens é o indicado para o hardware utilizado, uma vez que aproveita os portos série de alta velocidade dos Transputers, evitando assim esquemas de partilha de memória mais complexos e que necessitam de hardware adicional.

A especificação das ligações entre processos é feita através da linguagem CDL (Component Distribution Language), onde de uma forma elegante e de alto nível se pode especificar o grau de paralelismo existente. É este o mecanismo de construção das aplicações paralelas no Sistema Operativo Helios.

Podemos então considerar o pipeline de visualização como um pipeline de processos comunicantes por mensagens, onde cada um é visto como uma "caixa preta". As primitivas de comunicação utilizadas entre processos em diferentes processadores serão as de mais baixo nível que são fornecidos pelo sistema operativo.

As mensagens são, na sua maioria, constituídas por dois blocos, que passamos a descrever. O primeiro bloco é constituído apenas por uma palavra (32 bits) por ser o tamanho mínimo de uma mensagem num Transputer [HELIOS 89]. Nesta palavra nos 8 bits menos significativos é enviado o tipo de mensagem, e nos restantes o comprimento, em bytes, do segundo bloco. Caso este segundo bloco não exista, esses bits estarão a zero. As mensagens cuja dimensão é fixa não têm estes cálculos de empacotamento, sendo também

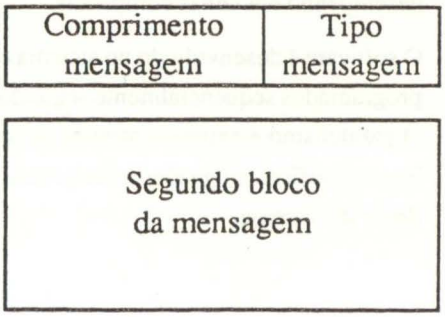

Fig 2 - Uma mensagem de dois blocos

desnecessário o envio desse comprimento, poupando assim algumas instruções redundantes.

A dimensão da mensagem e a informação nela contida são escolhidas de maneira a minimizar o processamento entre andares adjacentes no pipeline. Assim, não existe um formato único de mensagem, nem o formato das mensagens de um determinado tipo se mantém ao longo do pipeline. Por exemplo, as mensagens referentes à informação dos polígonos mudam de formato e dimensão ao longo do pipeline, acompanhando as transformações aplicadas aos dados, incluindo informação já calculada anteriormente e que seja necessária mais adiante, ou descartando informação que já não é necessária para o processamento.

Descrição dos Processos

O pipeline de processos pode ser dividido em dois grandes blocos, com funcionalidades distintas: o primeiro faz a interface com o sistema "host" e mantém a estrutura hierárquica de dados que descreve a cena a visualizar. O segundo bloco é um pipeline de visualização 3D, com processamento geométrico e rasterização.

Gestor da Estrutura Hierárquica

A descrição da cena a visualizar é feita através de uma árvore. Cada "nó" da árvore representa um objecto da cena (podendo ser um objecto composto) e tem uma transformação associada. Cada "folha" descreve uma faceta triangular, com a respectiva lista de vértices. Para evitar duplicação de informação, tanto os vértices como as características de reflexão da luz são mantidas em listas separadas, e cada folha da árvore tem apenas apontadores para essas estruturas.

A árvore comporta-se como uma árvore n-ária, mas a sua construção é a de uma lista de listas. Esta opção foi tomada com o objectivo de diminuír o número de chamadas à rotina

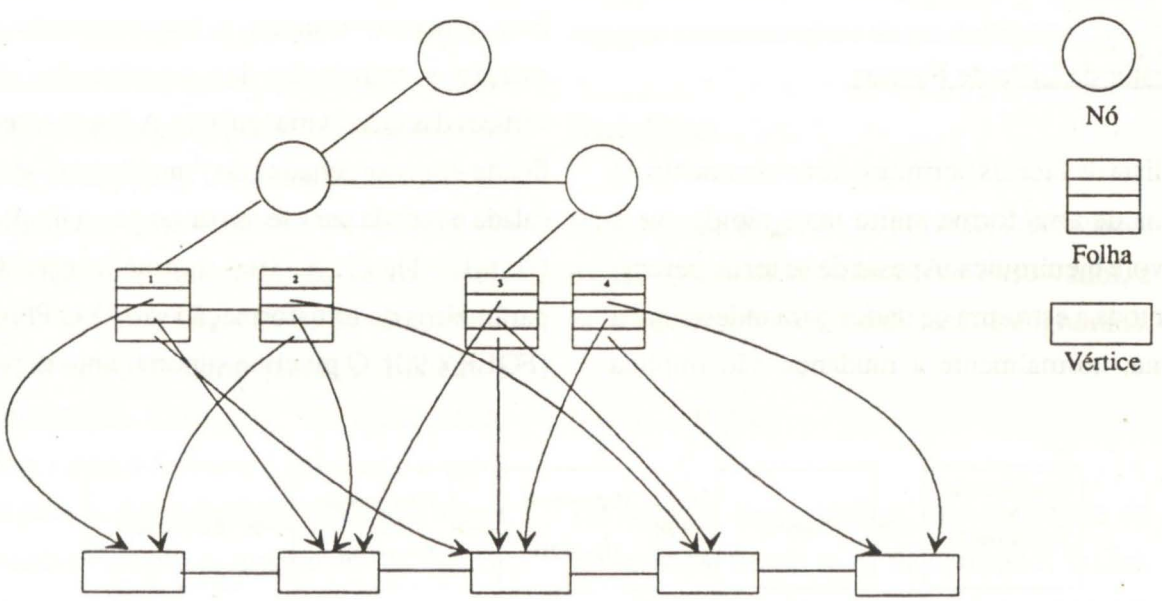

Fig 3 - A Árvore Hierárquica

recursiva para efectuar o atravessamento. De notar que o espaço ocupado pelo stack não diminui, mas apenas o número de "call" efectuados, e cuja execução no Transputer é relativamente lenta [INMOS 87][INMOS 89]. A estrutura completa da hierarquia é descrita na figura 3. Qualquer alteração feita na cena a visualizar será, numa primeira instância, executada sobre esta árvore.

uma modificação na maior parte das coordenadas das facetas que a compõem. Assim, a árvore só é percorrida a partir do ponto alterado, procedendo-se à actualização da lista de facetas, o que significa a modificação de apenas uma fracção do número total de facetas. O preço a pagar por este andar, na tentativa de explorar a coerência entre imagens (em animação é normalmente este o caso), é uma grande utilização de memória.

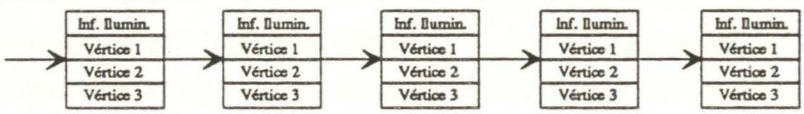

Fig 4 - Lista de facetas

Com este tipo de organização dos dados, existe uma utilização eficiente da memória, mas como contrapartida o atravessamento desta árvore torna-se bastante lento. Assim, utilizamos uma segunda estrutura de dados, sob a forma de lista, onde são mantidas todas as facetas que compõem a cena, depois de executadas todas as transformações de modelação.

Gestor da Lista de Facetas

A lista de facetas permite o atravessamento da cena de uma forma muito mais rápida que a árvore hierárquica. Apesar de se ter de percorrer toda a estrutura de dados para redesenhar a cena, normalmente a mudança não implica

Descrição do Pipeline de Visualização

O pipeline de visualização tem a organização apresentada na figura 5. Descrevem-se de seguida em pormenor todos os andares que o constituem.

Transformação de Normalização

Este processo executa a transformação de rotação e translação das coordenadas dos vértices das facetas triangulares. A janela especificada em coordenadas do "mundo real" é escalada e rodada para se tornar no paralelipípedo (-1, -1, -1), (1, 1, 0) - normalização. Os parâmetros da transformação são "à la Phigs" [FOLEY 90]. O pipeline suporta tanto as pro-

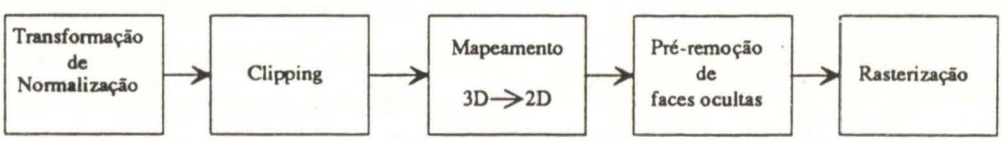

Fig 5 - Pipeline de visualização

jecções paralelas como projecções perspectivas.A matriz de transformação é calculada por ordem explícita dos andares acima, e não quando qualquer parâmetro é alterado.

Todos os cálculos são efectuados em vírgula flutuante.

Clipping

O clipping é executado por sete processos, e o algoritmo utilizado é o de Cohen-Sutherland para três dimensões [PLASTOCK 86].Este algoritmo consiste no cálculo de um vector de seis bits para cada vértice. Cada um desses bits está associado a uma fronteira de clipping, e quando a "1" indicam que o vértice está fora da região de clipping na coordenada correspondente ao bit. A existência deste vector de bits permite-nos fazer testes muito simples para eliminar facetas que estejam completamente fora do volume de visualização. O teste é um simples "and" bit a bit dos vectores de todos os vértices da faceta. Se resultar um vector com pelo menos um bit a "1", a faceta é descartada. Estas operações acima descritas são executadas pelo primeiro processo do pipeline de clipping. Os restantes seis processos vão, para cada fronteira do volume de visualização, verificar o bit do vector resultante do "or" bit a bit de todos os vectores da faceta bit esse que corresponde à fronteira a ser processada. Caso seja "1", é executado clipping sobre a faceta e é calculado o novo vector para as fronteiras que ainda não foram processadas. Este processamento do novo vector é feito porque existirão casos onde serão evitados cálculos desnecessários.

Em todos estes processos os cálculos são efectuados em vírgula flutuante.

Mapeamento 3D para 2D

Depois de executado o clipping, as facetas anteriormente triangulares podem ter até 9 vértices, expressos em coordenadas homogéneas. Há que fazer então o mapeamento destes para o viewport. Os vértices resultantes vêm expressos em vírgula fixa na notação Q12 (12 bits de parte fraccionária), escalados para coordenadas de ecrã.

Pré-remoção de Faces Ocultas

A pré-remoção de faces ocultas visa "aliviar" o trabalho a ser feito pelo andar de rasterização eliminando as facetas que sejam "back-faces" ou que sejam facilmente identificadas como ocultas por outras. Os cálculos efectuados são muito simples, e visam obter uma melhoria de desempenho da placa para cenas complexas. Esta pré-remoção tem dois processos que executam dois testes distintos.

Back-Faces

É executado um teste às normais das facetas, e caso a normal se "afaste" do observador, a faceta é considerada "back-face" e é ignorada.

Macro Z-Buffer

Tal como o nome indica, este algoritmo de pré-remoção baseia-se na filosofia do Z-Buffer. A superfície de visualização é dividida em quadrados, e para cada um é guardado o valor

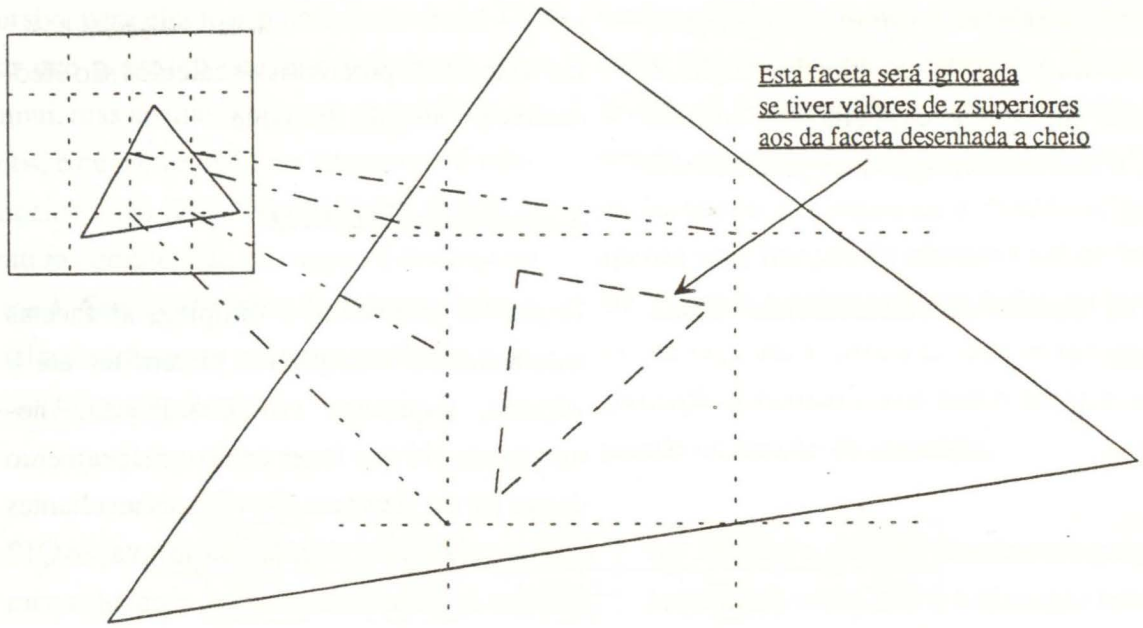

Fig 6 - Funcionamento do Macro Z Buffer

da maior profundidade aí existente. Esse valor só é modificado quando existe uma faceta que "ocupe" todo o quadrado. Facetas que fiquem completamente dentro desse quadrado e que tenham sempre valores de *z* maiores que o guardado para esse quadrado (ou vários, se estiverem na mesma situação) são descartadas. Na figura 6 está representada essa situação.

A determinação dos quadrados abrangidos pela faceta é feita através do cálculo do paralelipípedo envolvente da faceta. Isto pode levar a testes desnecessários ou mesmo a que uma faceta que seria "reprovada" passe por a sua envolvente ocupar uma superfície muito maior. Mas estes cálculos da envolvente são simples e são aproveitados para a rasterização, e por isso preferidos em detrimento de outros porventura mais exactos.

Remoção de Faces Ocultas

A remoção de faces ocultas é feita recorrendo ao método do Z Buffer, que tem uma profundidade de 16 bits.

Rasterização

O andar de rasterização é composto por um único processo que executa o enchimento de polígonos. Não é previsto o caso especial de linhas, porque qualquer objecto que inicie o atravessamento do pipeline deve ter três vértices. O caso de linhas implica que um dos extremos seja duplo, i.e., que exista uma repetição de um dos vértices na descrição da faceta.

O algoritmo de sombreamento implementado é o de Gouraud. A rasterização é executada recorrendo ao método de varrimento linha a linha, o que significa que depois de determinado o segmento de recta horizontal pertencente ao interior da faceta, basta exe-

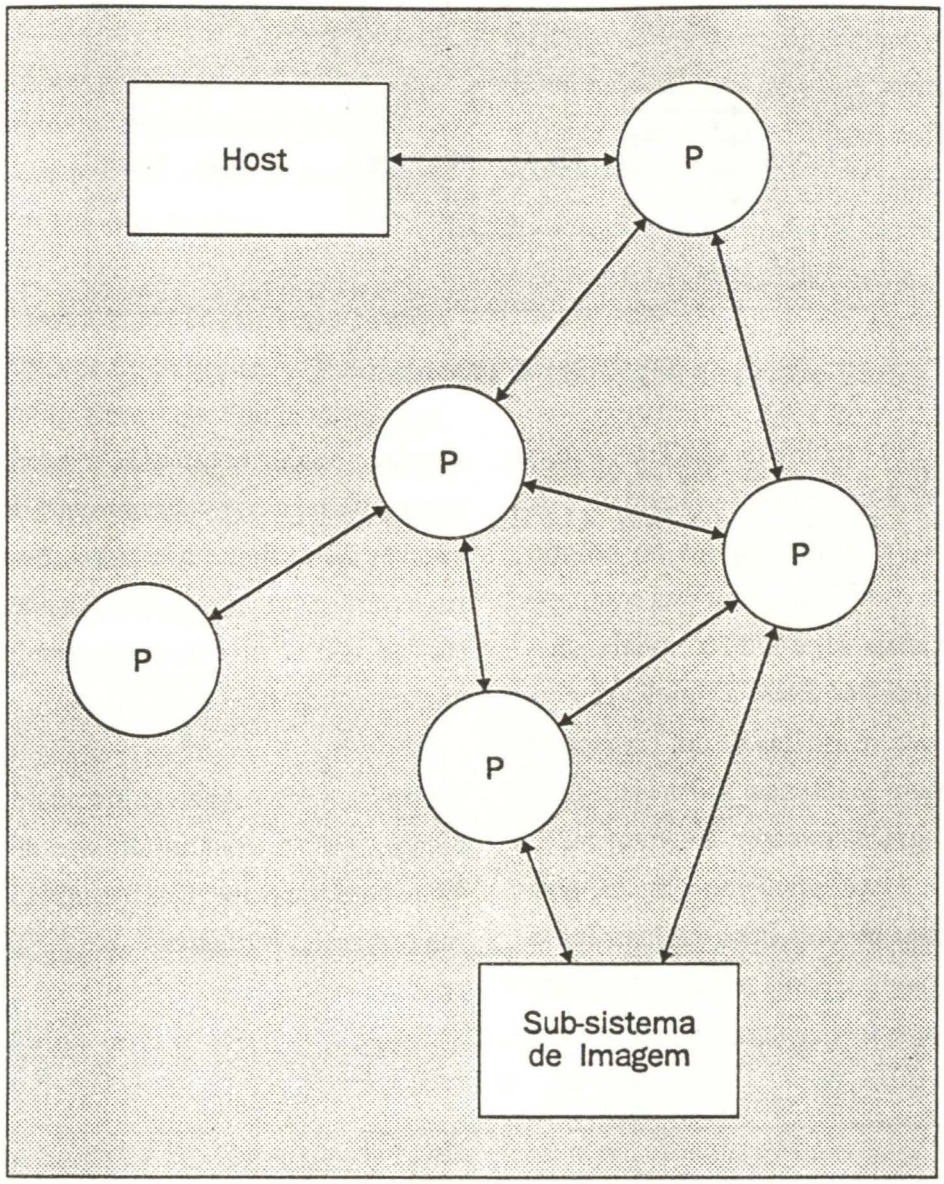

Fig 7 - O sistema gráfico numa rede de Transputers

cutar a interpolação linear dos valores da profundidade (z) e de côr (r, g e b). Estes valores para a côr foram, entretanto, calculados utilizando o modelo de iluminação de Phong [FOLEY 90].

Hardware

Ao nível do hardware, o paralelismo do Sistema Gráfico Concorrente é explorado usando uma rede de nós de processamento independentes que comunicam entre si. A figura 7 mostra uma configuração possível para esta rede, devendo notar-se que a topologia geral das ligações entre os diversos nós não precisa de obedecer a nenhum esquema pré-determinado. Ao contrário, as ligações podem ser estabelecidas arbitrariamente, de acordo com as necessidades. O número de nós da rede pode variar desde um até infinito e é escolhido de acordo com a necessidade de processamento da aplicação.

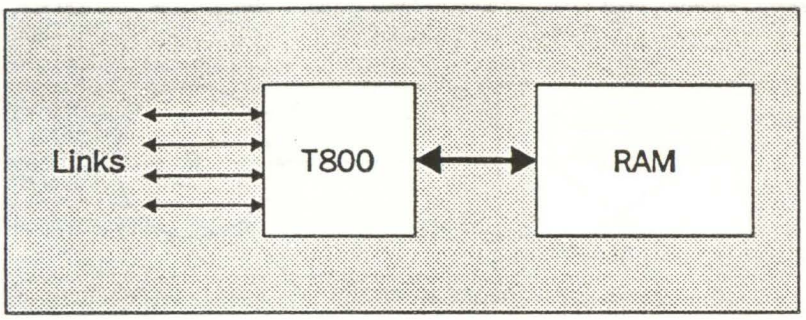

Fig 8 - Nó de processamento

Os processadores escolhidos são Transputers T800 [INMOS 89], devido às facilidades de comunicação que têm incluídas no próprio chip. De facto, cada Transputer está equipado com quatro canais série bidireccionais, chamados links, que permitem a ligação directa ponto a ponto, entre um link de um Transputer e um link de outro Transputer, a uma velocidade de 20 Mbit/seg. Deste modo, a comunicação entre dois nós é extremamente simples, pois necessita apenas de dois fios (ou duas linhas de PCB) a ligarem os dois processadores em causa. O tempo que a informação enviada demora a percorrer o canal é indiferente, o que torna possível a comunicação entre dois processadores físicamente afastados. Neste caso é necessária a utilização de linhas de transmissão de 100 Ω ou, alternativamente, a utilização de buffers.

Na figura 8 representa-se um nó de processamento, que é constituído apenas por um Transputer e a respectiva memória RAM para sua utilização exclusiva.

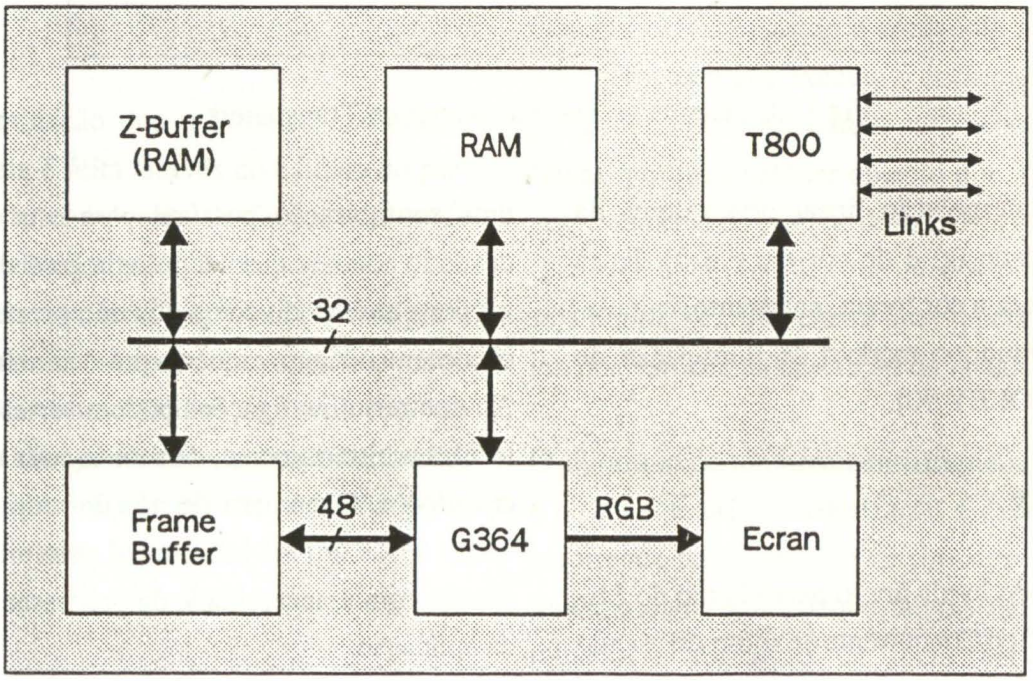

Fig 9 - Arquitectura do Sub-sistema de Imagem

A comunicação entre o host e o co-processador gráfico é assegurada através dum adaptador série/paralelo bidireccional (C012 [INMOS 89]), que permite a ligação dum link de um Transputer ao bus do Host.

A figura 7 mostra ainda outro elemento obrigatório na rede, que é o Sub-sistema de Imagem. Este sub-sistema é pormenorizado na figura 9, e incui um Transputer T800, o frame-buffer, o z-buffer e o controlador de vídeo. O Transputer do sub-sistema executa o último andar do pipeline de processos, cuja tarefa consiste em calcular as várias componentes de côr e profundidade ao nível do pixel, armazenando os valores calculados no frame-buffer. A componente de profundidade é guardada no z-buffer, que implementa o último estágio de eliminação de superfícies ocultas.

Organização do Frame-Buffer

A resolução escolhida para o monitor é de 1280 x 1024 pixels. Por cada pixel são armazenados 24 bits de côr, sendo 8 bits para cada uma das componentes RGB. Deste modo

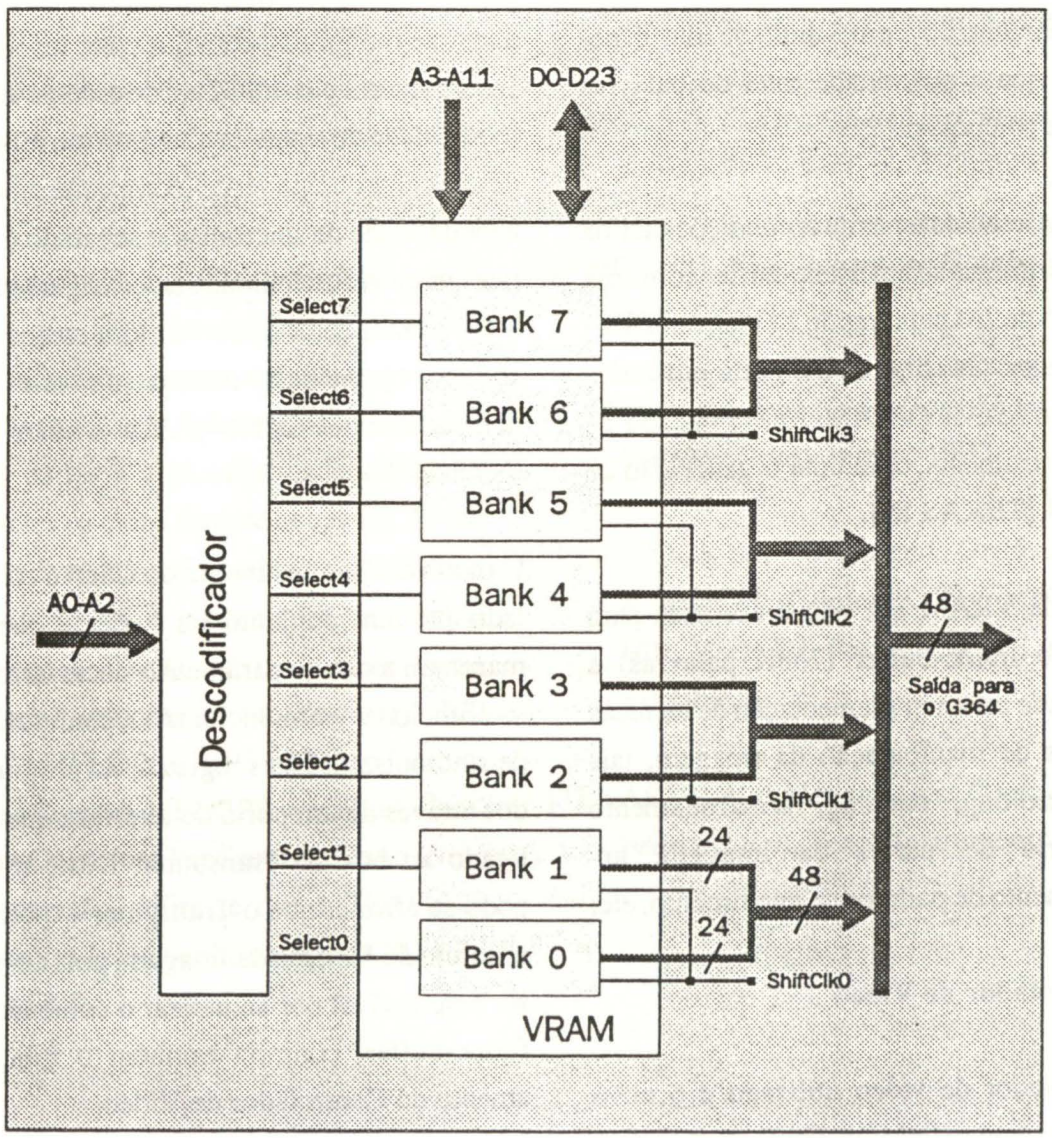

Fig 10 - Frame - Buffer

são necessárias 1,25M palavras de 24 bits para guardar o conteúdo do frame-buffer.

Por outro lado, a elevada resolução do sistema implica um fluxo de pixels do frame-buffer para o monitor da ordem dos 100 Mpixel/seg, sendo esta frequência muito superior à frequência máxima de acesso suportada pelos chips de memória que compõem o frame-buffer. A solução encontrada consiste em entrelaçar a memória (*memory interleaving*) de modo a que pixels sucessivos sejam armazenados em diferentes chips de memória (ver fig 10). Para que o entrelaçamento possa ser implementado com simplicidade, é necessário qua a quantidade total de palavras seja uma potência de 2.

Por forma a satisfazer as duas condições acima descritas, utiliza-se um frame-buffer com uma dimensão de 2M palavras de 24 bits, num total de 6 Mbytes. Os chips de memória utilizados são Video RAMs (VRAMs) TMS44C251 da Texas Instruments, com uma organização de 256K x 4 [TEXAS 89].

A zona da VRAM que não é utilizada para guardar o frame-buffer (768K palavras) é aproveitada para a implementação de técnicas eficientes de manipulação da imagem, tais como scrolling, panning, armazenamento temporário de partes da imagem, armazenamento de padrões de enchimento, etc.

O Controlador de Vídeo

O controlador de vídeo utilizado é o novo G364 da Inmos [INMOS G90], que integra num único chip um gerador de sincronismo de vídeo, uma memória de cursor (overlay cursor store), uma palette de 256 côres de 24 bits, e ainda três conversores digital-analógico (DAC) que geram os sinais RGB analógicos para o monitor. Quando o controlador funciona em modo True-Color, a memória interna que serve de palette não é desperdiçada, pois neste caso pode ser utilizada como tabela de mapeamento para correcção gamma em tempo real (esta funcionalidade é realmente usada pelo Sistema Gráfico Concorrente). O gerador de sincronismo de vídeo (abreviado VTG, de *Video Timing Generator*) é configurável por software, permitindo a adaptação a uma vasta gama de monitores, com qualquer resolução. Existe ainda a possibilidade de escolha entre a modalidade de sinais de sincronismo misturados na componente verde (composite sync) ou a modalidade de sincronismo separado (separate sync). A flexibilidade de configuração do VTG permite gerar saídas de vídeo compatíveis com as principais normas de televisão, nomeadamente CCIR (PAL) e EIA 343 (NTSC).

O modo de funcionamento do G364 é controlado por um conjunto de registos que armazenam todos os parâmetros necessários ao seu funcionamento, incluindo a configuração de sincronismo. Estes registos são programados através de um porto de interface que está ligado ao bus do Transputer. Deste modo, pode-se afirmar que o Transputer é o senhor absoluto do Sistema de Imagem, pois além de ser responsável por actualizar o conteúdo do frame-buffer, controla também o funcionamento do Controlador de Vídeo.

Melhoramentos Futuros

Está a ser desenvolvido um chip interpolador linear com a finalidade de excutar a última função do processo de rasterização, que é a que consome a maior parte do tempo neste andar do pipeline. Este chip terá a capacidade de calcular uma componente de côr ou profundidade de um pixel em cada ciclo de relógio. Um conjunto de 4 destes chips funcionando em paralelo poderá calcular várias dezenas de Mpixel/seg, dependendo da frequência de relógio máxima que fôr possível atingir. A função de interpolação linear é a que consome quase todo o tempo no processo de rasterização, por isso prevê-se uma melhoria substancial no desempenho do sistema com a introdução deste ASIC.

Com a introdução dos chips interpoladores, é provável que alguns processos do sistema, correspondentes a andares superiores do pipeline de visualização, não sejam capazes de fornecer dados a uma velocidade suficiente para manter os chips sempre ocupados. Neste caso, prevê-se para estes andares do pipeline a utilização do novo processador da família Transputer, o T9000, capaz de atingir uma velocidade de processamento (de pico) de 200 MIPS / 25 MFLOPS (MIPS = Million Instructions Per Second, MFLOPS = Million Floating point Operations Per Second). Como termo de comparação, refira-se que o T800 a 20 MHz tem um desempenho de pico de 20 MIPS e 3 MFLOPS.

Está ainda a ser estudada a possibilidade de implementação de uma arquitectura de Frame-Buffer distribuído, que poderá vir a ser usada em conjunto com vários grupos de quatro Interpoladores, ou alternativamente com vários Transputers ligados ao frame-buffer.

Referências

[FOLEY 90] *Computer Graphics - Principles and Practice*, Foley, van Dam, Feiner, Hughes, Wiley, 1990, 2ª Edição

[HELIOS 89] *The Helios Operating System*, Perihelion Software, Prentice-Hall

[INMOS 87] *The transputer instruction set - a compiler writers' guide*, inmos, 1987

[INMOS 89] *The Transputer Databook*, inmos, 1989

[INMOS G90] *The G364 Data Sheet (Preliminary Data)*, inmos, 1990

[PLASTOCK 86] *Computação Gráfica*, Roy A. Plastock, Gordon Kalley, McGraw-Hill, 1991, Ed. Portuguesa (Ed. Original 1986)

[TEXAS 89] *Mos Memory Data Book - European Edition*, Texas Instruments, 1989