

Um Sistema de E/S para uma Arquitetura Matricial

Rafael R. dos Santos¹
Gerson G. H. Cavalheiro²
Philippe O. A. Navaux³

Universidade Federal do Rio Grande do Sul - UFRGS
Curso de Pós-Graduação em Ciência da Computação - CPGCC
Av. Bento Gonçalves, 9500
Cx.P.15064 - CEP:91501-970
Porto Alegre - RS - Brasil

Resumo

Este trabalho analisa aspectos de desempenho de um processador matricial e busca uma forma mais eficiente de realizar o processo de E/S de dados nesta arquitetura. A placa NCR-GAPP é composta por um conjunto de 144 elementos de processamento, com memória local, conectados entre si sob topologia de interconexão do tipo *mesh* 12x12, com *wraparound* cilíndrico. Experimentos realizados comprovaram que o processo de E/S de dados consome grande parte do tempo de processamento de uma aplicação e prejudica o desempenho do sistema como um todo. Um novo sistema de E/S é proposto e os resultados esperados são confrontados frente aos índices de desempenho obtidos através do sistema original.

Abstract

This work evaluates performance features of an array processor and seeks for a more efficient way of performing data I/O processing in this architecture. The NCR-GAPP board is composed of a set of 144 processing elements, with local memory, connected in a 12x12 mesh interconnection topology, with cylindrical wraparound. Performed experiments led to the conclusion that I/O processing spends a major portion of application processing time and degrades the overall system performance. A new I/O system is proposed and expected results are compared to the performance indexes obtained with the original system.

¹B.Sc.; Mestrando - Áreas de interesse: Arquitetura de Computadores, Processamento Paralelo, Avaliação de Desempenho. E-mail: rrsantos@inf.ufrgs.br

²M.Sc.; Pesquisador Associado - Áreas de interesse: Linguagens de Programação, Processamento Distribuído, Avaliação de Desempenho. E-mail: gersonc@inf.ufrgs.br

³Prof. Dr.; Inst. Informática - Áreas de interesse: Arquitetura de Computadores, Processamento Paralelo, Avaliação de Desempenho. E-mail: navaux@inf.ufrgs.br

1 Introdução

As técnicas que outrora permitiram velocidades de processamento muito elevadas, e foram sem dúvida pontos marcantes para o avanço tecnológico, tendem hoje a estagnar [KUC94]. A exploração do paralelismo é hoje o principal campo de estudo para a melhora de desempenho nas arquiteturas de computadores. Através da replicação de unidades funcionais extremamente rápidas consegue-se uma performance cada vez mais elevada a medida em que os recursos são eficientemente aproveitados.

As arquiteturas matriciais executam uma única instrução, em um determinado instante de tempo, sobre um conjunto de dados explorando o paralelismo espacial síncrono de dados. Isto é possível devido as arquiteturas matriciais, também denominadas SIMD (*Single Instruction Stream - Multiple Data Stream*) [FLY72], possuírem diversos processadores com memória local (neste caso específico), interligados por uma malha de comunicação. Os dados a serem processados devem estar dispostos na matriz de processamento antes que uma instrução seja executada. Após a execução, os dados já processados devem ser retirados do local de processamento.

Em [CAV92, DER92, NAV95] foram relatados experimentos realizados com uma placa de processadores matriciais GAPP, seção 2, e analisados os resultados de desempenho para a execução da aplicação de convolução de imagens. Várias limitações foram encontradas em relação à forma com que a placa comunica-se com o hospedeiro: a existência de apenas um registrador de entrada de dados (8 bits), a necessidade de converter os dados de bit-paralelo para bit-serial e a impossibilidade de mascarar alguns EPs durante a execução de alguma tarefa.

Devido ao alto desempenho do circuito matricial NCR-GAPP, como constatado em [WON90], a placa GAPP pode vir a funcionar efetivamente como uma placa aceleradora da máquina hospedeira, nos casos em que especificamente o paralelismo de dados é explorado. No entanto, observa-se nitidamente nos resultados obtidos por [CAV92, NAV95] que as operações de E/S, envolvidas na aplicação objeto daquele estudo, correspondem a aproximadamente 50% do tempo total de processamento. Resultados melhores foram obtidos ao aumentar-se a complexidade do problema exigindo uma carga de processamento maior, o que, de certa forma, justificou o tempo de E/S envolvido no processo.

Neste trabalho pretende-se mostrar a complexidade do processo de E/S de dados na placa GAPP e, a partir desta constatação, apresentar uma alternativa eficiente para realizar este processo sem afetar por demais o tempo de processamento total de uma aplicação que deseje beneficiar-se das características paralelas deste tipo de equipamento.

Para tanto, a placa GAPP passou por uma fase de avaliação de desempenho através de monitoração, onde foram modeladas fórmulas matemáticas para a obtenção do número de ciclos de CPU e E/S necessários à carga ou à descarga de dados.

Analisando-se os fatores que degradam o desempenho, relativos ao processo de carga e descarga de dados, projetou-se uma alternativa em hardware eficiente o bastante para diminuir significativamente o número de ciclos necessários à este processo. Outros benefícios foram também alcançados, como: possibilidade de transferências concorrentes à programação de instruções sobre dados já carregados e diminuição do número de registra-

dores necessários.

Os resultados em termos de desempenho são comparados à versão existente deste sistema, o qual apresenta uma deficiência no processo de E/S, prejudicando a obtenção de taxas melhores de desempenho no processo computacional paralelo.

A placa NCR-GAPP é composta por um conjunto de 144 elementos de processamento conectados entre si conforme uma malha SIMD *mesh 12x12* com *wraparound* cilíndrico. A ausência de uma unidade de controle faz com seja necessária a presença de um hospedeiro que realize todo tipo de controle sobre a unidade de processamento.

2 Entrada e Saída no GAPP

O GAPP é um circuito CMOS com 2 processadores dispostos em uma matriz bidimensional com capacidade de execução em paralelo. Cada circuito GAPP está conectado conforme uma malha *mesh* [HWA84] 6×12 de elementos de processamento de um 1 bit, organizados como uma arquitetura SIMD. Cada elemento de processamento pode se comunicar com quatro vizinhos: norte (N), oeste (W), sul (S) e leste (E). Este é composto de uma ULA bit serial, uma RAM 128×1 e quatro registradores de 1 bit. Três registradores mantêm entradas para a ULA e o quarto registrador habilita operações de E/S através da célula sem interromper a ULA, isto é, operações de E/S dos EPs podem ser sobrepostas à computação.

Neste trabalho, quando forem analisados aspectos de desempenho sobre o circuito de processadores GAPP, serão usados dados obtidos através de programas executados no *PC Development System Kit* [NCR86], que consiste em uma placa composta por dois circuitos *NCR GAPP NCR45CG72* (total 144 EPs), conectados segundo uma malha *mesh 12x12* EPs com *wraparound* cilíndrico Norte/Sul, Leste/Oeste.

A placa, por não possuir unidade de controle, necessita de um hospedeiro para a realização desta tarefa. Para tanto, o hardware da placa foi projetado com uma interface para o padrão IBM-PC, o qual serve de hospedeiro. Conectada ao barramento de E/S de tal equipamento, a placa funciona como um dispositivo periférico, mapeado em memória de E/S.

A comunicação da placa NCR-GAPP com o computador hospedeiro se dá através de 8 registradores denominados P0, P1, P2, P3, P4, P5, P6 e P7. Estes registradores são mapeados em 8 endereços consecutivos do espaço de endereçamento de E/S, com endereço inicial configurável através de *dip-switchs* existentes na placa.

2.1 O Problema de Entrada e Saída da Placa NCR-GAPP

Os dados de entrada para um sistema composto por processadores GAPP, usualmente não estão no formato requerido pela matriz de processadores GAPP. Os dados de um conversor analógico-digital, ou de um barramento de sistema, normalmente estão arranjados sob a forma palavra serial, bit paralelo, enquanto que o processador GAPP só aceita dados no formato palavra paralela, bit serial.

Por esta razão, algum dispositivo de hardware, externo a matriz de processadores, deve ser usado para armazenar uma linha de valores de dados e serialmente apresentar cada bit para a entrada da matriz. O dado deve, fisicamente, fazer uma troca de direção de 90 graus para ser reformatado de bit paralelo para bit serial, por isso o processo de reformatação é chamado de *corner turning*, conforme [NCR87b].

A placa NCR-GAPP possui como dispositivo reformatador de dados uma matriz de processadores GAPP idêntica à matriz de processamento. Durante a carga de dados, estes passam primeiramente pela matriz reformatadora antes de serem armazenados definitivamente na matriz de processamento, o mesmo acontece quando enviados ao hospedeiro, durante a descarga de dados. Este dispositivo externo é chamado de *Corner Turner Line Buffer*, ou simplesmente *CTLB*.

Pode-se subdividir o processo de E/S em duas etapas: a transferência e a reformatação. A transferência corresponde à etapa de comunicação entre *host* e placa, ou seja, ao período em que os dados são movimentados através do barramento de dados do *host* até o registrador de entrada da placa e posteriormente para dentro do *CTLB*. A reformatação é responsável pela retirada dos dados do *CTLB* e armazenamento na matriz de processamento. A descarga de dados executa os mesmos passos citados acima porém em sentido inverso.

Dois são os motivos que podem atenuar o desempenho do procedimento de E/S: o elevado número de instruções necessárias à E/S e a complexidade da etapa de reformatação. Afim de comprovar esta hipótese, serão apresentadas duas fórmulas extraídas dos algoritmos de carga e descarga de dados de/para a placa NCR-GAPP. Estes algoritmos foram escritos em linguagem C++ para fins de avaliação de desempenho e foram apresentados por [SAN94].

A fórmula 1 produz o número de ciclos de CPU ou E/S necessários à carga de dados na matriz de processadores GAPP. As constantes K_1 , K_2 e K_3 devem possuir respectivamente os seguintes valores: 20, 10 e 5. Estes valores foram calculados durante a fase de monitoração da programação da placa GAPP e correspondem ao número de ciclos de CPU ou E/S gastos durante a fase de carga de dados. As variáveis X e Y correspondem ao número de colunas e linhas da matriz de dados a ser carregada na matriz de processamento.

$$NC_{TransfCargaAtual} = Y * (K_1 + K_2X + K_3) \quad (1)$$

A fórmula 2 produz o número de ciclos de CPU gastos na descarga de dados quando as constantes K_1 , K_2 e K_3 possuírem os valores: 20, 5 e 8, respectivamente. E produzirá o número de ciclos de E/S quando as mesmas constantes possuírem valores iguais a: 20, 5 e 5 respectivamente. Analogamente à fórmula anterior, fórmula 1, os valores destas constantes também foram obtidos através da análise do monitoramento de programação da placa GAPP. E as variáveis X e Y correspondem ao número de colunas e linhas da matriz de dados a ser descarregada da matriz de processamento.

$$NC_{TransfDescargaAtual} = Y * (K_1 + K_2X + K_3X) \quad (2)$$

As constantes K_i representam o número de ciclos gastos em etapas da transferência de dados de/para a placa NCR-GAPP. As instruções de IN ocupam 5 ciclos de barramento de E/S para enviarem um dado a um dispositivo periférico enquanto que na CPU ocupam 8 ciclos. Já as instruções de OUT ocupam 5 ciclos tanto nos barramentos de E/S quanto na CPU [INT87]. Dados comparativos serão mostrados na seção 4.

3 Proposta do Sistema de E/S - SISIO

Partindo do princípio de que as arquiteturas matriciais operam sempre sobre um conjunto grande de dados optou-se por usar o *DMA* para fazer a transferência dos dados do *host* para a placa e vice-versa. Para tanto, foi necessário projetar um sistema em hardware que gerasse os sinais de controle para recepção, envio, armazenamento e recuperação de dados no *CTLB*.

O *CTLB* é um dispositivo de armazenamento temporário de dados, onde é feita a reformatção dos mesmos. A matriz de dados a ser processada obrigatoriamente passa pelo *CTLB* antes de ser efetivamente carregada na matriz de processamento. O mesmo ocorre durante a descarga desta matriz, ela obrigatoriamente passará pelo *CTLB* antes de ser enviada de volta ao hospedeiro.

A idéia de usar o *DMA* para executar o processo de transferência dos dados da RAM do hospedeiro até o *CTLB* e vice-versa, advém de dois fatores principais: os dados são transferidos da memória do hospedeiro para um dispositivo periférico e o *DMA* pode executar esta operação de transferência concorrentemente ao funcionamento da CPU. Como se pode notar, as características necessárias a este processo se adaptam exatamente ao tipo de processamento feito por um *DMA Controller*.

3.1 Descrição do Sistema SISIO

O *SISIO* é um sistema em hardware que executa a tarefa de detectar uma solicitação de envio/recepção de dados feita pelo *DMA* e gera todos os sinais de controle para que o GAPP forneça os dados ao barramento de E/S (descarga) ou os retire deste (carga). As figuras 1 e 2 apresentam o esquema projetado para execução desta tarefa.

No sistema original de E/S da placa NCR, para cada dado da matriz de dados a ser carregada é necessário um elevado número de instruções pois o dado deve ser enviado primeiramente até o registrador de entrada da placa, depois, deve ser enviado o conjunto de bits de controle, (estes bits de controle correspondem a microinstrução que os elementos de processamento deverão realizar) e, para finalizar a ação, mais uma instrução de OUT indicando que a placa deve executar a microinstrução enviada ou que estiver previamente contida nos registradores de controle. As microinstruções são compostas por 13 bits e o endereço usado nas operações de acesso à memória local dos EPs contém 7 bits, somando um total de 20 bits que serão necessariamente enviados à placa durante a programação da mesma.

Com o sistema *SISIO*, somente são necessárias instruções para programação do *DMA*.

Depois de programado, o DMA transfere os dados até a placa e esta dá o tratamento adequado sem a interferência da CPU, que poderá estar executando qualquer outra tarefa.

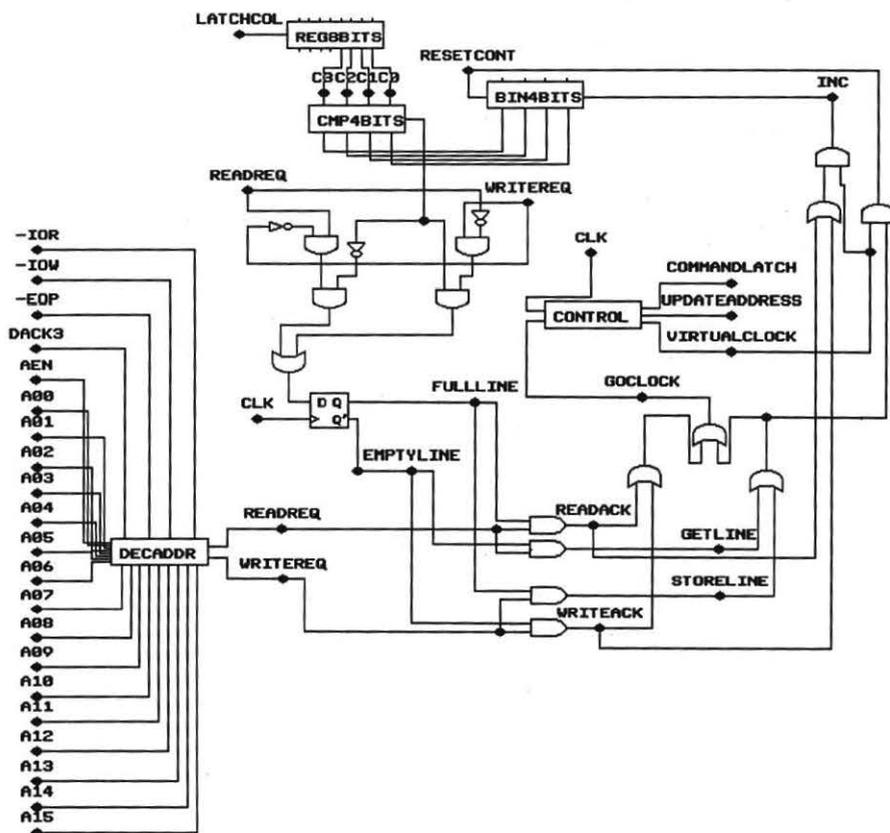


Figura 1 Diagrama lógico do circuito SISIO, Parte 1

As figuras 1 e 2 fazem parte do mesmo projeto, foram separadas para que fosse mantida a clareza. Subtende-se que as linhas *COMMANDLATCH*, *UPDATEADDRESS*, *VIRTUALCLOCK*, *READACK*, *GETLINE*, *STORELINE* e *WRITEACK* sejam as mesmas em ambas as figuras.

O módulo *DECADDR* é responsável pela detecção das solicitações feitas pelo *DMA* e gera os sinais de requisição de leitura/escrita de dados no *CTLB*.

Os módulos *REG8BITS*, *BIN4BITS* e *CMP4BITS* são responsáveis pela lógica que

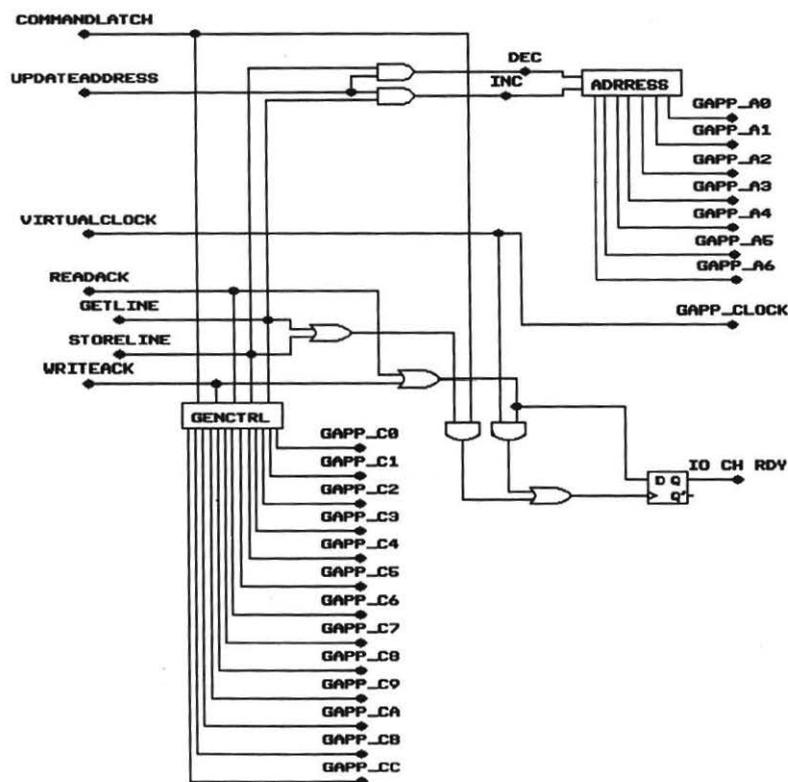


Figura 2 Diagrama lógico do circuito SISIO, Parte 2

controla o número de dados que pode ser recebido pelo *CTLB* ou então o número de dados que ainda podem ser retirados deste elemento.

Quando há uma requisição de escrita de dados no *CTLB* (carga de dados no GAPP) podem ocorrer duas situações: ainda existe espaço no *CTLB* para receber os dados (*WRITEACK*), ou então, este se encontra cheio e os dados ali presentes devem ser armazenados (*STORELINE*).

Simetricamente, quando o *DMA* faz uma requisição de leitura (descarga de dados no GAPP) o *CTLB* pode possuir estes dados (*READACK*), ou então, pode precisar recuperá-los na memória (*GETLINE*).

Sempre que um dos sinais (*READACK*, *WRITEACK*, *STORELINE* ou *GETLINE*) for gerado, o módulo *CONTROL* gera três sinais em ciclos consecutivos de *clock*, na respectiva ordem: *COMMANDLATCH*, *UPDATEADDRESS* e *VIRTUALCLOCK*.

Analisando a composição dos sinais gerados pelos outros módulos, o módulo *GENCTRL* gera os sinais de controle (microinstrução) que farão com que o GAPP execute alguma tarefa quando o sinal *VIRTUALCLOCK* for ativado.

O módulo *ADDRESS* é responsável pela manutenção do endereço corrente de busca/armazenamento na memória do *CTLB*. Esta estrutura funciona como uma pilha. Cada linha de dados da matriz de carga/descarga é armazenada/recuperada da memória do *CTLB* em posições consecutivas de memória.

O sinal *IO CH RDY* faz com que o *DMA* pare de executar uma tarefa. Isto é útil quando se faz necessário armazenar/recuperar dados no *CTLB*. Para isto, são necessários apenas 3 pulsos de *clock*: um para gerar os sinais de controle, outro para atualizar os endereços correntes e o último para habilitar um pulso de *clock* para que o GAPP execute a microinstrução.

O projeto lógico dos circuitos apresentados nas figuras 1 e 2 foi executado com a utilização da ferramenta *Electronics Workbench 2.0a* onde foram simulados sob modelo de tempo de atraso zero.

3.2 Tempos de Transferência no Sistema SISIO

O GAPP tem um tempo de ciclo de 100 ns a 10 MHz [NCR87a], enquanto que o barramento de E/S tem um tempo de ciclo de 210 ns. Todas as transferências que ocorrem via barramento de E/S levam aproximadamente 5 ciclos, ou 1.05 μ s/byte [COF84, IBM83].

Como visto na seção 3.1, são necessários 3 ciclos apenas para que o *CTLB* possa ler ou escrever no barramento de E/S. Logo, a transferência após ter começado ocupa 5 ciclos de E/S para cada dado transferido mais 3 ciclos adicionais quando se faz necessário armazenar/recuperar alguma linha de dados.

Propõe-se duas fórmulas, fórmulas 3 e 4, para o cálculo do número de ciclos de CPU e E/S, respectivamente, necessários à uma transferência. Não são computados o número de ciclos necessários à reformatação pois esta etapa não foi abordada pelo projeto por estar intimamente ligada à arquitetura do circuito GAPP.

A fórmula 3 apresenta o número de ciclos de CPU necessários as transferências de carga/descarga de dados no *CTLB*. O número de instruções gastas para programar o *DMA*

fornece o número de ciclos gastos nesta tarefa. A partir deste momento não é necessária a intervenção da CPU no processo de transferência, ficando esta liberada para outras tarefas. A constante P_{dmac} contém o valor 43, correspondendo ao número de ciclos de CPU gastos na programação do DMA. Este valor foi obtido através de experimentos de programação do DMA.

$$NC_{TransfSISIO} = P_{dmac} \quad (3)$$

A fórmula 4 produz o número de ciclos que são gastos no barramento de E/S quando ocorre a carga ou descarga de dados do CTLB. Neste caso as constantes P_{dmac} , K_1 e K_2 devem conter os valores 40, 5 e 3 ciclos de E/S respectivamente. O número de ciclos gastos na programação do DMA, neste caso, é menor que no caso anterior, pois no barramento de E/S as instruções de IN ocupam 5 ciclos enquanto que na CPU ocupam 8 ciclos [IBM83, INT87]. Já os valores das constantes K_i foram obtidos através da simulação dos circuitos lógicos projetados.

$$NC_{TransfSISIO} = P_{dmac} + ((Y - 1) * (K_1X + K_2)) + K_1X \quad (4)$$

Analogamente aos casos anteriores, fórmulas 1 e 2, as variáveis X, Y correspondem respectivamente ao número de colunas e linhas da matriz de dados a ser carregada/descarregada.

4 Avaliação Comparativa de Desempenho

Nesta seção será apresentada a comparação do desempenho obtido com o sistema proposto *SISIO* frente ao desempenho do sistema de E/S original da placa NCR.

Nas seções 2.1 e 3.2 foram mostradas quatro fórmulas para o cálculo do número de ciclos de CPU e E/S necessários à uma transferência. Nestas fórmulas, as variáveis X, Y correspondem respectivamente ao número de colunas e linhas de cada matriz a ser carregada/descarregada do *CTLB*.

Tabela 1 Número de ciclos de CPU e E/S

	Carga-NCR	Descarga-NCR	<i>SISIO</i>
Ciclos de E/S	1740	1680	793
Ciclos de CPU	1740	2112	43

A tabela 1 mostra o número de ciclos de CPU e E/S necessários para carga/descarga de matriz de dados de 8 bits e dimensão 12×12 . As colunas Carga-NCR e Descarga-NCR agrupam valores correspondentes aos testes feitos com a placa NCR-GAPP e, a coluna *SISIO* agrupa valores simulados do desempenho do sistema *SISIO*, obtidos pela aplicação das fórmulas 3 e 4.

Tabela 2 Comparação de desempenho

SISIO	Carga-NCR	Descarga-NCR
Ganho em ciclos de E/S	54,43 %	52,80 %
<i>Speed-up</i> em ciclos de E/S	2,119	2,118
Ganho em ciclos de CPU	97,52 %	97,96 %
<i>Speed-up</i> em ciclos de CPU	40,465	49,116

$$Speed - up = \frac{NCiclosNCR - GAPP}{NCiclosSISIO} \quad (5)$$

A fórmula 5 fornece o índice de *Speed-up* do sistema *SISIO* em relação ao sistema de E/S original da placa NCR. A variável $NCiclosNCR - GAPP$ corresponde ao número de ciclos gastos com a placa NCR-GAPP usando-se o sistema original de E/S, e a variável $NCiclosSISIO$ ao número de ciclos gastos usando-se o sistema *SISIO*.

Observa-se nas tabelas 1 e 2 o ganho promovido pelo sistema *SISIO*.

O número de ciclos de CPU gastos para efetuar uma carga ou descarga com o sistema *SISIO* é computado em função da programação do DMA. A partir deste momento, o processo de transferência ocorre de maneira concorrente ao funcionamento da CPU que, poderá estar executando qualquer espécie de processamento, inclusive programando a própria matriz de processamento da placa. Com isto, é possível que uma transferência esteja em andamento enquanto a CPU envia sinais de controle (programas) à matriz de processadores GAPP.

No gráfico comparativo apresentado na figura 3, mostra-se as curvas de desempenho dos sistemas de E/S. As curvas legendadas por *Carga* e *Descarga* correspondem aos índices obtidos para o sistema original de E/S da placa NCR. A curva *SISIO* mostra os índices de desempenho estimados para o sistema *SISIO*. Este gráfico apresenta a comparação destes índices em função do número de ciclos de E/S, variando o tamanho da matriz de dados.

Estes índices foram produzidos a partir da aplicação das fórmulas 1, 2, 3 e 4 vistas em seções anteriores. Quatro situações foram propostas para montagem deste gráfico variando-se as dimensões das matrizes de dados simuladas, 12x12, 48x48, 96x96 e 128x128.

Deve-se notar que o ganho de desempenho da utilização da placa GAPP com o *SISIO* aumenta consideravelmente em função do aumento da massa de dados processada, para ciclos de E/S. O número de ciclos de CPU com o sistema *SISIO* teoricamente permanece constante (43 ciclos) para qualquer situação.

5 Conclusão

As arquiteturas matriciais, em geral, apresentam problemas relacionados a E/S de dados pois, processam uma grande quantidade de dados de forma idêntica, exigindo um fluxo muito alto de transferências entre a matriz de processamento e o dispositivo de armazenamento dos dados.

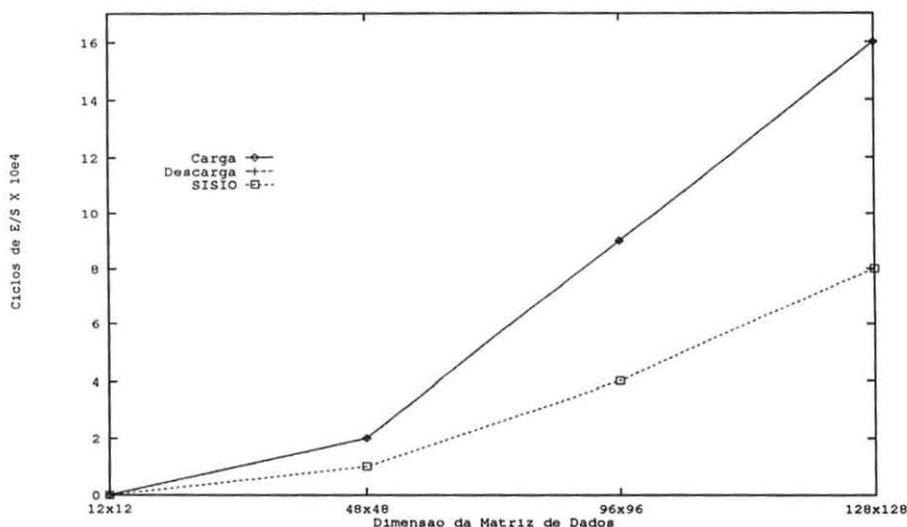


Figura 3 Gráfico do número de Ciclos de E/S

Em relação a placa NCR-GAPP o problema está ligado ao fato de que existe uma diferença entre o formato dos dados do computador hospedeiro e o formato trabalhado pelo GAPP e a maneira como a comunicação ocorre entre *host* e placa.

Existem duas etapas que ocorrem durante o processo de E/S de dados: a transferência e a reformatação destes, quando chegam à placa.

A transferência compreende o fluxo de dados através do barramento de E/S do *host*, e a reformatação, a tarefa de inverter o formato dos dados de bit paralelo-palavra serial para bit serial-palavra paralela, formato aceito pelo GAPP.

Outro problema ocorre devido a maneira como a comunicação é feita entre o *host* e a placa. Esta não possui uma unidade local de controle delegando, portanto, à CPU do *host* todo o controle, seja de processamento de E/S ou seja de programação (processamento efetivo).

O objetivo específico deste trabalho foi otimizar o processo de E/S sem alterar a arquitetura do circuito GAPP e trazer assim, um ganho de desempenho em relação a situação atual. Outros benefícios foram alcançados e também podem proporcionar ganho de desempenho, não só em relação ao processo de E/S, mas também em relação ao processamento total da aplicação que irá usar a máquina GAPP.

Como resultado principal, foi projetada a lógica de um circuito digital que executa a tarefa de gerar sinais controle ao *Corner Turner Line Buffer*, de uma placa hipotética de processadores GAPP, durante a transferência dos dados através de um DMAC (DMA

Controller).

O *SISIO* apresentou resultados melhores, do que os obtidos com o sistema de E/S da placa NCR-GAPP, quando avaliado em função do número de ciclos de E/S e CPU usados na transferência de dados. Observando-se a lógica de funcionamento do sistema, modelou-se uma fórmula matemática que fornece o número de ciclos de E/S e CPU gastos nas transferências de dados, em função do número de linhas e colunas da matriz a ser trabalhada. E, através dos algoritmos usados na aplicação de teste da placa NCR-GAPP, extrairam-se outras duas fórmulas com o mesmo objetivo.

Uma situação foi proposta para que fossem analisadas as diferenças entre os dois sistemas. Os valores resultantes da análise desta situação foram confrontados com valores extraídos da monitoração do funcionamento da placa NCR-GAPP, definindo índices de desempenho elevado em relação ao processo de E/S original.

Observou-se através do uso de DMA, a possibilidade de se fazer transferência de dados entre memória e dispositivos de E/S, concorrentemente ao processamento executado pela CPU.

Com o sistema *SISIO* foi viabilizada a exploração efetiva do paralelismo à nível de dados e a nível de tarefas. Está é uma vantagem importante em relação à organização e arquitetura atual da placa NCR-GAPP.

Como produto deste trabalho, obteve-se então, a lógica de funcionamento de um circuito digital que fará o controle da recepção e envio de dados do CTLB para com o barramento de E/S do computador hospedeiro. Estas transferências poderão ocorrer concorrentemente ao funcionamento da CPU liberando-a da atividade de controlar o fluxo de dados entre memória RAM e placa NCR.

Com esta alternativa o número de registradores presentes na placa pode ser diminuído, já que não é mais necessário apresentar sinais de *clock* externos à placa, pois estes serão gerados automaticamente, assim como, não são mais necessários registradores distintos para entrada e saída de dados.

A concorrência permitida entre E/S e programação da placa pode promover índices altos de desempenho de aplicações matriciais na placa GAPP com o sistema *SISIO*. A análise deste aspecto não foi computada neste experimento pois exige uma simulação mais completa e dependente da aplicação que se tem em vista. A continuidade dos estudos sobre a placa GAPP poderão fazer desta alternativa um fator muito importante no uso deste equipamento.

Referências

- [CAV92] CAVALHEIRO, Gerson G. H.; DE ROSE, César; NAVAU, Philippe O.A. *A Operação de Convolução de Imagens em uma Arquitetura Matricial*. São Paulo. IV SBAC-PAD. Novembro. 1992.
- [COF84] COFFRON, James W. *The IBM PC Connection*. Berkeley: SYBEX. 1984.

- [DER92] DE ROSE, César; CAVALHEIRO, Gerson G. H.; MENNA BARRETO, Ricardo.
Rotinas de Comunicação para a Placa GAPP. Porto Alegre, CPGCC da UFRGS, 1992. Relatório de Pesquisa.
- [FLY72] FLYNN, M. J. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*. New York, v. C-21, n. 9 pp.948-160. Sept. 1972.
- [HWA84] HWANG, K.; BRIGGS, F. *Computer Architecture and Parallel Processing*. New York: McGraw Hill. 1984.
- [IBM83] International Business Machines Corporation. *Technical Reference*. Personal Computer. Hardware Reference Library. Boca Raton, Florida. 1983.
- [INT87] Intel Corporation. *Microprocessor and Peripheral Handbook*. Microprocessor, v.1. 1987.
- [KUC94] KUCK, David J. What Do Users of Parallel Computer Systems Really Need ? *International Journal of Parallel Programming*. New York and London: Plenum Publishing Corporation, v. 22, n. 1, p. 99-127, February 1994.
- [NAV95] NAVAUX, Philippe O. A.; CAVALHEIRO, Gerson G. H.; DE ROSE, César. *Performance Evaluation of a GAPP Processor in Image Processing*. *Microprocessing and Microprogramming* 41 (1995) 71-82.
- [NCR86] NCR Microelectronics. *GAPP Personal Computer Development System User's Manual*. Fort Collins: NCR Microelectronics, 1986. 99P. IL.
- [NCR87a] NCR Microelectronics. *Geometric Arithmetic Parallel Processor*. Data Sheet NCR45CG72. 1987.
- [NCR87b] NCR Microelectronics. *Data Input/Output Techniques for the Geometric Arithmetic Parallel Processor (GAPP)*. Data Sheet NCR45CG72. 1987.
- [SAN94] SANTOS, Rafael R. dos. *Projeto de um Sistema de Entrada e Saída para uma Arquitetura Matricial*. Porto Alegre, PUCRS, 1994. (Trabalho de Conclusão)
- [WON90] WONG, W. F.; LUA, K. T. A Preliminary Evaluation of a Massively Parallel Processor : GAPP. *Microprocessing and Microprogramming*, North Holland, v.29, n.1, p.53-61, July 1990.