Arquitetura IBM Blue Gene para Supercomputadores

Carlos Alberto Petry
Unicamp – IC – PPGCC
MO801 Arquitetura
de Computadores I
Campinas, SP – Brasil
ra095345@students.ic.unicamp.br

Resumo

O presente artigo apresenta uma visão geral da arquitetura IBM Blue Gene/L para supercomputadores. São abordados dois aspectos: a arquitetura do hardware do sistema e da arquitetura de software, esta em nível mais introdutório. O sistema Blue Gene/L corresponde a um supercomputador destinado a executar aplicações altamente paralelizáveis, sobre tudo aplicações científicas, podendo conter mais de 65 mil nodos de computação. O objetivo principal do projeto foi atingir a marca de 360 TeraFLOPS em desempenho utilizando uma relação custo x benefício e consumo x desempenho não obtidas em arquiteturas tradicionalmente usadas em supercomputadores construídos anteriormente ao projeto Blue Gene.

Categoria e descrição do assunto

C.5.1 [**Computer System Implementation**]: Large and Medium ("Mainframe") Computers – Super (verylarge) computers.

Termos Gerais

Desempenho, Projeto de hardware, Experimentos.

Palavras Chaves

Arquitetura de Computadores, Supercomputadores, Hardware, Alto Desempenho.

1. INTRODUÇÃO

Blue Gene constitui um projeto de pesquisa da IBM com o objetivo de explorar os limite da supercomputação em termos de: arquitetura de computadores, requisitos de software para programar e controlar sistemas massivamente paralelos e uso de computação avançada para ampliar o entendimento de processos biológicos como desdobramento de proteínas. Novas aplicações estão sendo exploradas com os sistemas Blue Gene como: hidrodinâmica, química quântica, dinâmica molecular, modelagem climatológica e financeira [2].

Blue Gene/L é um sistema baseado no processador embarcado, PowerPC modificado e acrescido em suas funcionalidades em relação ao original, suportando um grande espaço de endereçamento de memória, fazendo uso de compiladores padrão e de um ambiente de passagem de mensagens para a comunicação [1], além de ser um supercomputador cujo sistema pode chegar a ter até 65536 nodos de computação utilizando grande escala de paralelismo. Através da integração de sistemas em um único chip (SoC) [6] [7] juntamente com uma arquitetura celular escalável, o sistema Blue Gene/L foi projetado para atingir um desempenho de pico de 180 ou 360 TeraFLOPS [1] [2], dependendo de como o sistema for configurado.

BlueGene/L foi projetado e construído em colaboração com o Departamento de Energia norte-americano o NNSA/Lawrence Livermore National Laboratory (LLNL), devido a uma parceria estabelecida em Novembro de 2001, o qual atingiu o pico de desempenho de 596,38 TeraFLOPS na unidade localizada no LLNL [2], alcançado em 2007 utilizando 212992 núcleos de computação [4].

A versão Blue Gene/L Beta System DD2 chegou ao 1º lugar entre os supercomputadores em Novembro de 2004, atingindo desempenho de pico de 91,75 TeraFLOPS conforme publicado em Top500 [3]. Atualmente ainda ocupa o 5º lugar na classificação deste mesmo site [4], conforme levantamento anunciado em Junho de 2009. O objetivo do projeto foi disponibilizar um sistema cuja relação custo x desempenho fosse a mais alta possível, além de atingir altas taxas de desempenho em relação ao consumo de potência e área ocupada pelo sistema. Este objetivo foi, em grande parte, viabilizados pelo uso dos processadores embarcados IBM PowerPC (PPC) de baixo consumo e baixa freqüência, tendo sido um dos primeiros projetos de supercomputadores a usarem o paradigma *low power*. O sucessor da versão /L, o modelo Blue Gene /P, instalado no Argonne National Laboratory, atingiu o 4º lugar entre os computadores mais eficientes em relação ao desempenho por watt confirme publicado em Green500 [5] (371,75MFlops/Watt). A medição para o modelo /L não está disponível uma vez que esta avaliação iniciou apenas em Novembro de 2007.

O restante deste trabalho está organizado da seguinte forma: a seção 2 apresenta as características da arquitetura de hardware do sistema Blue Gene/L; a seção 3 apresenta as características básicas da arquitetura de software utilizada neste sistema; por fim a seção 4 apresenta a conclusão e considerações finais.

2. ARQUITETURA DE HARDWARE

Esta seção apresenta um estudo da arquitetura de hardware do supercomputador Blue Gene/L.

A abordagem de construção do Blue Gene/L (BG/L) representou uma inovação para a arquitetura de supercomputadores devido ao uso em grande escala dos chamados nodos de computação, detalhados posteriormente, operando a uma freqüência de clock de 700MHz, baixa para os padrões da época.

O componente básico da arquitetura BG/L se constitui de um SoC baseado em processadores embarcados PowerPC 440 CMOS incorporando todas as funcionalidades necessárias, tais como: processador para computação, processador para comunicação e controle, três níveis de memória cache, memória DRam embarcada (EDRam) e múltiplas interconexões de rede de alta velocidade usando um sofisticado roteamento de dados, sobre um único ASIC (Application Specific Integrated Circuit) [1]. Devido a freqüência da memória Ram ser próxima a do processador, foi possível construir nodos de computação com baixo consumo de energia e alta densidade de agrupamento, podendo chegar a conter 1024 nodos de computação inseridos em um único *cabinet* (detalhado da próxima seção). A comunicação entre nodos está embutida na lógica do ASIC dispensando o uso de Switches externos.

2.1 Estrutura modular do Hardware BG/L

O sistema BG/L é escalável, podendo atingir um máximo de 2^{16} (65536) nodos de computação, configurados como uma rede *Torus 3D* com dimensões de 64x32x32. Sua arquitetura física é caracterizado por uma estrutura modular composta de 5 níveis, conforme apresenta a Figura 1. O componente básico é um chip ASIC contendo dois processadores e denominado nodo de computação; um compute card que é composto de dois chips ASIC; um node board que contém 16 compute card; 32 node board que formam um cabinet ou rack, o qual é composto por dois midplanes com 16 node board cada um; e por fim, podendo existir até 64 cabinets agrupados formando um System. Na especificação completa de um *System*, cada componente modular possuem as seguintes quantidade de nodos de computação: *compute card* → 2, *node board* → 32, *cabinet/rack* → 1024 e *system* → 65536.

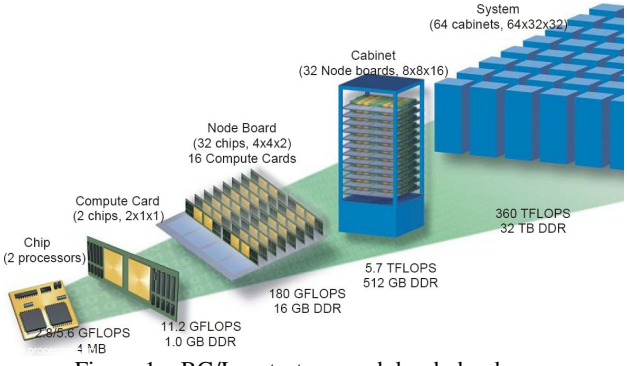

Figura 1 – BG/L: estrutura modular do hardware.

Cada processador do nodo de computação pode realizar quatro operações de ponto flutuante na forma de duas adições /multiplicações de 64 bits por ciclo, o que permite atingir um desempenho máximo (pico) de 1.4 GigaFLOPS, considerando uma freqüência de operação de 700MHz. Os nodos de computação do BG/L podem operar em dois modos distintos [1] [7]: (i) os dois processadores são usadas para computação, operações sobre a aplicação; (ii) um processador é usado para computação e outro para controle da comunicação MPI e controle, sendo que o modo de operação é definido em função das características da aplicação a ser executada. O nodo de computação atinge um pico de desempenho de 5,6 ou 2,8 GFLOPS dependendo do modo de operação configurado. Como existem 1024 nodos de computação em um *cabinet*, o seu desempenho máximo atinge 5,6 ou 2,8 TeraFLOPS. Por fim na configuração completa, 64 *cabinets*, um *System* atinge o desempenho de pico aproximado de 360 ou 180 TeraFLOPS, conforme o modo de operação em uso.

Os nodos são interconectados através de 5 redes [1]: (i) rede torus 3D usada para comunicação ponto-a-ponto de mensagens entre nodos de computação; (ii) árvore global combining /broadcast para operações coletivas através de toda a aplicação, como por exemplo MPI_Allreduce; (iii) rede global barrier e interrupt; (iv) duas redes Ethernet gigabit, uma para operações de controle via conexão JTAG e outra para conexão com outros sistemas como hosts e file system. Por motivos de custo e eficiência, nodos de computação não são diretamente conectados à rede Ethernet, usando a árvore global para realizar a comunicação com nodos de I/O. Além dos 65536 possíveis nodos de computação, o BG/L contém 1024 nodos de I/O que são idênticos aos nodos de computação porém dedicados a operações de comunicação com os dispositivos externos como *hosts* e *file system*.

2.2 Estrutura do nodo de computação

O componente básico de computação, nodo de computação, consiste de dois processadores, três níveis de memória cache, dispositivos de suporte à conexões externas e memória Ram DDR ECC cuja capacidade máxima é de 2GB, entretanto são usados apenas 256MB devido a definição de projeto. A Figura 2 apresenta o diagrama de blocos simplificado do SoC contido no chip ASIC, exceto a memória Ram DDR que está contida no *compute card*, comunicando com o ASIC via barramento local.

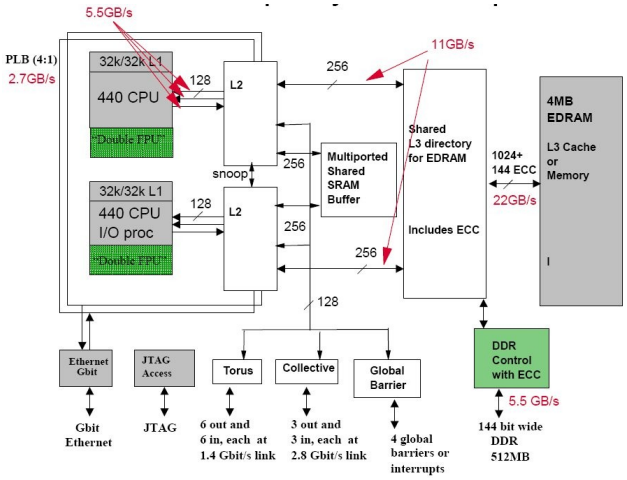

Figura 2 –Diagrama de blocos simplificado do SoC ASIC BG/L.

O nodo de computação é composto por dois processadores PPC 440, cada um dispondo de uma unidade de ponto flutuante (FPU2) dupla que corresponde a uma unidade de 128 bits melhorada. O processador PPC 440 se caracteriza por uma arquitetura superescalar de 32 bits, usado em sistemas embarcados e produzido pela IBM, não dispondo de hardware para suporte a multiprocessamento simétrico (SMP). A FPU2 é

composta por duas FPUs convencionais de 64 bits unidas, entretanto as unidades foram estendidas para suportar instruções do PPC estilo SIMD, além do conjunto de instruções original [7].

Existem três níveis de memória cache: (i) L1, uma cache para cada núcleo, não coerente e não dispondo de operações atômicas em memória, porém estas deficiências foram contornadas pelo uso de dispositivos de sincronização no chip, como mecanismo *lockbox, L3*-scratchpad, *blind device* e SRam compartilhada; (ii) um segundo nível de cache (L2) é implementado para cada processador, possui capacidade de 2KB e é controlada por um mecanismo *data pre-fetch*, que corresponde a um *SRam array* para acesso a comunicação entre os dois núcleos; (iii) o projeto disponibiliza ainda um terceiro nível de cache (L3) produzidas com memórias *Embedded DRam* (ERam) com capacidade de 4MB. Complementando a arquitetura do ASIC, existem ainda interfaces usadas para comunicação com dispositivos externos: Ethernet, JTAG, operações de roteamento e controladora de memória Ram DDR externa. Os níveis de cache L2 e L3 são coerentes entre os dois processadores PPC440.

Em modo de operação normal, um nodo de computação é usado para realizar o processamento da aplicação, enquanto o outro serve às funções de comunicação e controle de rede fazendo uso de troca de mensagens MPI. Entretanto, não existe impedimento do ponto de visto do hardware para usar os dois nodos para computação. Este uso é determinado por aplicações que se caracterizam por possuir alta taxa de computação se comparado ao processamento de mensagens.

Como o caminho de dados entre a cache de dados e a FPU possui 128 bits, é possível armazenar ou recuperar dois elementos de dados, usando precisão simples ou dupla, a cada ciclo de clock, implementado também um mecanismo para transferências de dados entre posições de memória de alta velocidade, útil no processamento de mensagens. Todas as instruções de computação, exceto divisão e operações sobre valores não normalizados, são executadas em cinco ciclos de clock, mantendo uma vazão de de um ciclo por instrução. Divisões são executadas de forma iterativa produzindo dois bits a cada ciclo. A FPU2 também pode ser usada pelo processador para otimizar operações de comunicação, uma vez que ela permite ampla largura de banda para acesso ao hardware da rede [1].

A característica de baixo consumo do nodo de computação é obtida pela implementação de técnicas como uso de transistores com baixa corrente de fuga, *clock gate* local e habilidade de desativar dispositivos como CPU/FPU, além do uso de métodos que permitem reduzir o chaveamento do circuito.

O sistema de memória foi projetado de forma a oferecer alta largura de banda e baixa latência tanto para memória como cache, por exemplos *hits* em cache consomem 6 a 10 ciclos de CPU para L2 e 25 ciclos para L3 e um *misses* sobre a cache L3 consome cerca de 75 ciclos.

2.3 Hardware de rede

Toda a comunicação entre nodos de computação é realizada através da troca de mensagens MPI. Entre as cinco redes do BG/L (torus, árvore, Ethernet, JTAG e interrupção global) a principal comunicação ponto-a-ponto na rede para mensagens é realizada pela rede torus 3D [7]. Cabe salientar que o nodo de I/O não faz parte da rede torus 3D.

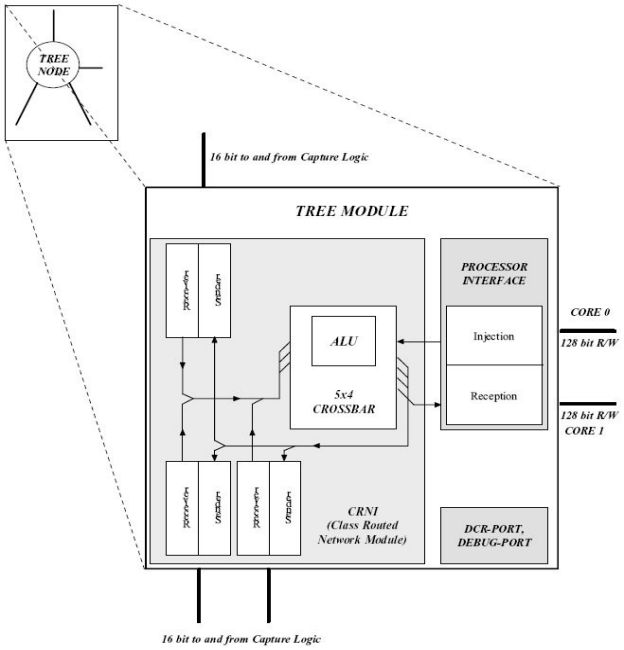

Figura 3 – Módulo de controle da rede em árvore.

A rede em árvore suporta comunicação ponto-a-ponto usando mensagens de tamanho fixo de 256 bytes, bem como *broadcast*, operando com uma latência de 1,5ms para um sistema composto por 65536 nodos de computação. Tanto nodos de computação como de I/O compartilham a rede em árvore, constituindo esta rede o principal mecanismo para comunicação entre as duas classes de nodos (computação e I/O). Para suportar a rede em árvore existe um módulo que opera a funcionalidade necessária, o módulo árvore, apresentado na figura 3. Nele, existe uma ALU de inteiros e *bitwise que* combina pacotes que chegam e encaminha os pacotes resultantes ao longo da árvore. Além da ALU, podem ser visto à esquerda (acima e abaixo) as três conexões bidirecionais (send, receive) com a rede torus 3D e à direita aparece a interface local com os processadores PPC. A rede em árvore possui 2 canais virtuais (*virtual channels*–VC) e faz uso de pacotes (componente atômico que trafegam na rede) para viabilizar o fluxo de comunicação, sendo o controle realizado pelo uso de *tokens*. O fluxo de dados, que flui através dos VCs, é realizado de forma não bloqueante. Através de registradores, existentes no hardware da rede em árvore, é possível selecionar um modo de comunicação. entre duas formas existentes: *combining* e *point-to-point*. O modo *point-to-point* é usado para nodos de computação que precisam se comunicar com os nodos de I/O. O modo *combining* opera agrupando pacotes a serem transmitidos, sendo usado para suportar operações MPI coletivas, como por exemplo MPI_Allreduced, através de todos os nodos conectados na rede em árvore.

Tanto a rede em árvore como a torus são acessadas através de endereços mapeados em memória, ou seja, o processador envia uma mensagem para um endereço especial que é redirecionado pelo hardware para a rede e implementado como uma FIFO usando registradores SIMD de 128 bits.

O BG/L possui agrupamentos computacionais denominados *partições* que são conjuntos de nodos de computação eletricamente isoladas constituindo subconjuntos auto-contidos dentro de um *System*. Cada *partição* é dedicada a executar uma aplicação ou conjunto comum de aplicações que podem ser distribuídas em tarefa,. O menor dimensionamento configurável corresponde a um *midplane* (meio *cabinet*), que constitui uma rede torus com 512 nodos de computação. É possível criar redes menores, até o limite inferior de 128 nodo de computação. Isto é feito desativando-se as FIFOs nos chips usados para controle de limites do *midplane*.

Cada SoC ASIC no BG/L possui ainda uma rede Ethernet utilizada para conectividade externa e uma rede serial JTAG para operações de boot, controle e monitoração do sistema.

Cada nodo da rede torus 3D possui seis conexões bidirecionais com os nodos vizinhos (adjacentes). Os 65536 nodos de computação, previstos no projeto, são organizados numa distribuição de rede cuja dimensão é 64x32x32, conforme já mencionado. Os dispositivos de rede disponíveis no ASIC garantem a entrega de pacotes contendo até 256 bytes de forma confiável, não ordenada e livre de *dead-lock*, usando algoritmo de roteamento adaptativo mínimo. O algoritmo de roteamento selecionado para uso no BG/L foi o *virtual cut-through* (VCT) [8] de forma a aperfeiçoar a capacidade vazão e latência da rede. As mensagem podem ser compostas por mais de um pacote, podendo ser enviadas numa ordem e recebidas em ordem diferente, variando seu tamanho entre 32 e 256 bytes com granularidade de 32 bytes, portanto foram criados mecanismos para recompor a ordem original da mensagem. A implementação de canais virtuais ajuda a melhorar a vazão, existindo na configuração original do sistema 4 CVs, dois dedicados a roteamento adaptativo e dois para roteamento determinístico. O controle de fluxo entre roteadores é realizado pelo uso de *tokens*. A rede torus é usada para operações de propósito geral, passagem de mensagem ponto-a-ponto e operações de *multicast* dentro de uma *partição*.

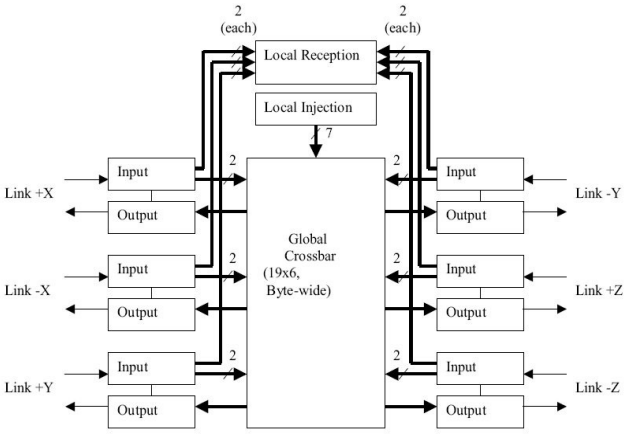

Figura 4 – Estrutura básica da rede torus inserida nos nodos.

A estrutura básica da rede torus inserida em cada nodo de computação pode ser vista na figura 4, onde podem ser observados as seis conexões bidirecionais (x, y e z), a conexão local (reception e injection) e o *crossbar* que realiza a interconexão entre as portas de duas conexões. Cada conexão bidirecional possui duas FIFOs usadas para recepção de dados direcionados ao nodo e para repasse de dados destinados a outros nodos. Existe um coprocessador de mensagens que é responsável pelo controle de recebimento e envio de mensagens contidas nas duas FIFOs. Pacotes recebidos de outros nodos são imediatamente repassados caso não haja contenção na rede, caso exista contenção eles são armazenados na FIFOs de entrada e aguardam até que possam ser enviados. Existe capacidade suficiente nos dispositivos FIFO de forma a manter o fluxo de dados na rede, desde que não inexista contenção [1]. O controle de arbitragem é altamente *pipelinizado* e distribuído, existindo um árbitro em cada unidade de entrada e saída.

Baseado em experimentos e simuladores da rede, os projetistas determinaram os valores de dimensionamento dos dispositivos existentes no módulo (VC, FIFO size, etc), para otimizar e atender o fluxo de rede de foram adequada. Experimentos realizados em um S*ystem* contendo 32768 nodos de computação, usando comunicação intensiva através da chamada MPI_Alltoall, demonstraram que o uso de roteamento dinâmico demonstrou ser mais eficaz que o estático devido à característica *pipelinizada* da rede, além de demonstrar que usando apenas dois VC dinâmicos as conexões se mantém ocupadas (utilizadas) praticamente todo o tempo, conforme observado na tabela 1 [1].

Tabela 1 – Experimento de comunicação contendo 32768 nodos de computação utilizando roteamento estático x dinâmico.

	Média da utilização da conexão (Pacote)	Média da utilização da conexão (Payload)
Roteamento estático	76%	66%
Roteamento dinâmico 1 VC	95%	83%
Roteamento dinâmico 2 VC	99%	87%

3. ARQUITETURA DE SOTEWARE

De forma a atender os requisitos de escalabilidade e complexidade e disponibilizar um ambiente de desenvolvimento confiável, mantendo os padrões de software o máximo possível, os projetistas de software do BG/L implementaram a arquitetura apresentada na figura 5.

Aplicações destinadas a executarem no BG/L são organizadas como uma coleção de processos computacionais[1], cada um sendo executado em seu próprio nodo de computação a partir de uma *partição* no sistema, auxiliados pelos nodos de I/O que oferecem serviços de suporte à execução das aplicações.

A preocupação principal durante o projeto do sistema de software foi disponibilizar um ambiente com alto nível de desempenho, oferecendo facilidades de desenvolvimento familiar, onde as aplicações executassem sob um sistema operacional (SO) unix-

like, porém mantendo capacidade suficiente para atender aos requisitos do projeto. A abordagem adotada foi dividir as funcionalidades do SO em dois grupos: computação e I/O. Nodos de computação executam exclusivamente aplicações de usuário, enquanto nodos de I/O executam operações de sistema, proporcionando acesso ao sistema de arquivos e *host*, além de dispor de facilidades como operações de controle, disparo de tarefas, boot e depuração do software. Esta abordagem manteve a arquitetura do hardware e software o mais simples possível [1].

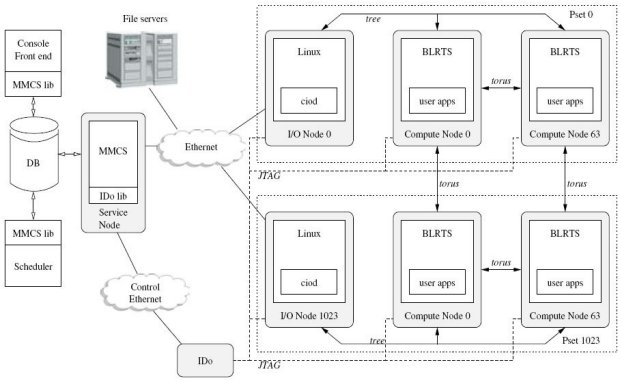

Figura 5 – Estrutura básica do sistema de software BG/L.

3.1 Software para nodos de computação

O BG/L executa uma SO personalizado nos nodos de computação denominado BLRTS (BG/L Run Time Supervisor). Este SO oferece um espaço de endereçamento contínuo de 256MB sem suporte a paginação, compartilhando este endereçamento entre o kernel e a aplicação sendo executada. O kernel fica localizado na parte inicial da memória (a partir do endereço 0), esta área está protegida através da programação da MMU realizada durante o processo de boot. Acima do SO fica localizada a aplicação a ser executada pelo processador seguida pelas suas áreas de pilha e *heap*.

Os recursos físicos, como rede em árvore e torus, são particionados entre aplicação e kernel, sendo que toda a rede torus é mapeada dentro do espaço de usuário, de forma a se obter melhor desempenho da aplicação, além de ser disponibilizado para kernel e aplicações de usuário um dos dois canais da rede em árvore.

O BLRTS segue o padrão Posix dispondo da biblioteca GNU/Glibc que foi portada para este SO, de forma a prover o suporte básico para operações de I/O sobre o sistema de arquivos. O suporte a multitarefa é desnecessário uma vez que o SO é monousuário executando uma única aplicação com suporte *dual-thread*, assim não foi incluída na biblioteca o suporte multiusuário. Operações de carga, finalização e acesso à arquivos para aplicações são realizadas via passagem de mensagens entre os nodos de computação e de I/O, através da rede em árvore. Sobre o nodo de I/O é executado o *deamon* CIOD (Console I/O deamon) que provê as tarefas de controle e gerenciamento de entrada e saída de dados para todos os nodos da *partição*. Kernel e biblioteca implementam a comunicação entre nodos de computação através da rede torus. Devido à natureza monousuária o SO não necessita realizar trocas de contexto e operações de paginação.

3.2 Software para nodos de I/O

Sobre os nodos de I/O é executado o SO GNU/Linux versão 2.4.19 contendo suporte para execução multitarefa. Para suportar os requisitos de projeto foram necessárias alterações na seqüencia de boot, gerenciamento de interrupções, layout de memória, suporte à FPU e drivers de dispositivos, em relação ao kernel original. Interrupções e exceções forma adequados ao BIC (BG/L custom interrupt controller); a implementação do kernel para MMU remapeia as redes em árvore e torus para o espaço de usuário; foi inserido suporte para controlador Ethernet Gigabit; também foi disponibilizado suporte para operações *save/restore* sobre registradores de ponto flutuante de precisão dupla para a FPU2 e suporte a trocas de contexto, uma vez que neste nodo o SO é multitarefa. No nodo de I/O é executado apenas o sistema operacional, ou seja, nenhuma aplicação esta presente ou em execução. O objetivo do processamento realizado neste nodo é dispor de serviços não suportados pelo SO do nodo de computação como acesso a sistemas de arquivos e *sokets* de conexões com processos de outros sistemas. Quando uma operação é requisitada pela aplicação a rede em árvore transporta esta solicitação para ser processada no nodo de I/O. Entre as operações suportadas estão: autenticação, contabilização e autorizações em nome de seus nodos de computação. Ainda está disponível no nodo de I/O facilidades para depuração de aplicações de usuário executadas nos nodos de computação. O fluxo de dados relativo à depuração também faz uso da rede em árvore.

Como os nodos do BG/L são *diskless* o sistema de arquivos raiz é fornecido por um *ramdisk* inserido no kernel do linux. O *ramdisk* do nodo de I/O contém interfaces de shell, utilitários básicos, bibliotecas compartilhamento e clientes de rede como ftp e nfs.

3.3 Gerenciamento e controle do sistema

A infra-estrutura de controle provê uma separação entre o mecanismo de execução da aplicação e as políticas de decisões em nodos externos à partição. O SO dos nodos locais são responsáveis pelas decisão locais. O SO "global" [7] toma todas as decisões coletivas e atua como interface com os módulos externos, realizando diversos serviços globais (boot, monitoramento, etc). O SO "global" executa em nodos de serviços externos (*hosts*), sendo que cada *midplane* é controlado por um MMCS (Midplane Management and Control System).

O processo de boot realiza a carga de um pequeno código (boot loader) inserido na memória do nodo (de computação e I/O) fazendo uso dos nodos de serviço através da rede JTAG. Este pequeno código carrega na seqüência a imagem do boot principal que então passa a controlar o processamento. Como os SOs dos nodos de computação e I/O são diferentes cada nodo recebe sua imagem correspondente. A imagem para computação possui cerca de 64KB e a I/O aproximadamente 2MB, já incluso o sistema de arquivos e *ramdisk*. Uma vez que existem diversos nodos é necessário carregar também configurações adicionais

que individualizam os nodos, este procedimento foi denominado de *personalidade* do nodo [7].

O sistema é monitorado pelos nodos de serviço externos ao sistema principal. Estes serviços são implementados através do software contido nestes nodos. Informações tais como velocidade dos ventiladores e voltagem da fonte de energia são obtidos através da rede de controle, sendo registradas (log) pelos serviços destes nodos.

A execução de tarefas é viabilizada conjuntamente pelos nodos de computação e de I/O. Quando uma aplicação é submetida a execução o usuário especifica a forma e tamanho da *partição* desejada, estalecida no sistema pelo escalonador que é uma facilidade fornecido pelos nodos de serviço.

A arquitetura de comunicação no BG/L é implementada por três camadas. A primeira corresponde a camada de pacotes (*packet layer*) constituída de uma biblioteca que permite acesso ao hardware da rede; a segunda é a camada de mensagem (*message layer*) que provê um sistema para entrega de mensagens de baixa latência e alta largura de banda, cuja comunicação ocorre ponto-a-ponto; a última camada é a MPI que corresponde a uma biblioteca de comunicação em nível de usuário (aplicação), designada basicamente para executar as facilidades da especificação MPI [9].

3.4 Modelo de programação

A passagem de mensagens foi elegido como o modelo de programação adotado como principal abordagem para uso na implementação da programação paralela existente no BG/L Este modelo é suportado através da implementação da biblioteca MPI message-passing, sobre a qual os projetistas tiveram especial atenção quanto a eficiência no mapeamento de operações para as redes em árvore e torus.

4. CONCLUSÃO

O projeto IBM Blue Gene introduziu novos paradigmas para a construção de supercomputadores, sobre tudo o uso de componentes de baixo consumo de energia como processadores embarcados além do emprego de SoCs. Atingiu um baixo consumo de energia em relação ao alto desempenho disponibilizado pelo sistema.

O projeto foi construído e testado fazendo uso de aplicações altamente exigentes em termos de recursos computacionais, caracterizando uma abordagem massivamente paralelizável.

A arquitetura IBM Blue Gene introduziu novos paradigmas no projeto e construção de supercomputadores, levando ao desenvolvimento de sistemas de alta capacidade computacional, cujo custo, alto desempenho e baixo consumo forneceram referenciais atualmente utilizados em outros projetos de supercomputadores não só da IBM mas também de outros fabricantes.

5. REFERÊNCIAS

[1] Adiga, N. R.; Almasi, G.; et. al. “An Overview of the BlueGene/L Supercomputer”. Proceedings of the IEEE/ACM SC2002, pp. 60-75, Nov 2002.

[2] IBM – Blue Gene [online]. http://domino.research.ibm.com/comm/research_projects.nsf/pages/bluegene.index.html, acessado em Junho de 2009.

[3] TOP500 Supercomputer Sites – Top List November 2004 [online].http://www.top500.org/list/2004/11/100, acessado em Junho de 2009.

[4] TOP500 Supercomputer Sites – Top List June 2009 [online]. http://www.top500.org/list/2009/06/100, acessado em Junho de 2009.

[5] The GREEN500 List – June 2008 [online]. http://www.green500.org/lists/2008/06/list.php, acessado em Junho de 2009.

[6] Almasi, G.; Almasi, G.S.; et. al. “Cellular supercomputing with system-on-a-chip”. Proceedings of the IEEE International ISSCC 2002, vol1, pp. 196-197, Feb 2002.

[7] Almási, G.; Bellofatto, R.; et. al. “An Overview of the Blue Gene/L System Software Organization”. Lecture Notes in Computer Science, Euro-Par 2003, vol 2790, pp. 543-555, Jun 2004.

[8] Kermani, P.; Kleinrock, L. “Virtual Cut-Through: A New Computer Communication Switching Technique” Computer Networks, vol. 3, pp. 267-286, 1979.

[9] Snir, M.; Otto, S.; et. al. "MPI – The Complete Reference. Volume 1 – The MPI Core, 2nd edition". The MIT Press, 448 pp, Sep 1998.