Uma Breve Descrição sobre Cell Processor

Yang Yun Ju
Engenharia de Computação
Instituto de Computação
yyunju@gmail.com

1.Introdução

Apesar de Cell foi inicialmente projetado apenas para agir como "core" de estação console de PayStation III, e com a idéia de superar a velocidade de PlayStation II em 100 vezes, a tecnologia foi abilitada a atenter muito mais que o objetivo inicial de projeto.

Além de conseguir poupar grande parte de custo fabricação de componentes eletrônicos por parte de Toshiba e Sony, o Cell vem com o inituito de servir não só como um simples processador de consumo geral, mas a aliança (IBM, SCE, Toshiba) vem o objetivo de criar uma nova arquitetura para áreas de entreitenimento digital, amplamente voltado para aplicativos de multimida, que exigia maior capacidade de processamento de pontos flutuantes, e instruões de múltiplo-paralelismo, e ao mesmo tempo, pretende obter maior versatilidade na integração com diferentes dispositivos, tais como celular, PDA, HDTV, Web-Server, caracteristicas de processamento em tempo real, que tem gigante fluxo de dados em alta velocidade. Ainda dentro desse visão inivador, e pricipalmente por parte de IBM pretende projetar um processador de alto grau de processamento paralelo, que atende principalmente necessidades de aplicações da área científica, militar, e aeroespaciais.

Nesse trabalho, pretendemos comentar brevemente a arquietura de plataforma Cell e enfatizamos um pouco sobre o desenvolvimento de aplicativos no cell, modelos de programação e com o isso tentamos entender, na visão teória, como exploramos o alto desempenho de arquitetura cell.

2. História de Plataforma Cell

O cenceito de Cell foi inicialmente concebido pelo pai de PalyStation, **Ken Kurtaragi**, projetista de Sony Computer Entertainment inc.do Japan (S.C.E), e posteriormente em ano 2000, foi patenteado conjuntamente por **Masakazu Suzuoki** e **Takeshi Yamazaki** da mesma empresa. Desde então, essa arquitetura foi implementado pelo **S.C.E**, juntamente com a **Toshiba Corporation** do Japão, co-desenvolvedor de PlayStation II, e finalmente com o grande e famoso **IBM** (International Business Machine Corporation).

Apesar da existência de especulação sobre a possível frustração da liga, em que os sócios apresentam atuações e interesses de mercado totalmente distintas, Toshiba Corporation, líder mudial de equipamentos eletrodomésticos, que requer aplicações de baixo de consumo de energia e alta credibilidade do sistema. O SEC, atual líder de área de entreterimento, que exige dos aplicativos em processar imagnes e sons eficientemente. IBM, tradicional fabricante de chip de alto desempenho, e requer o poder de multi-processamento e autovetorização dos CPUs, as três empresas optaram pelo método "full-custom" em desenvolver desse ambicioso projeto (que pretende alcançar 100 vezes a velocidade de PlayStation II), ao mesmo tempo, possuiam perspectiva de atender satisfatoriamente a necessidades de cada um dos seus membros.

Por um outro ânglo, podemos dizer que o projeto foi resultado de várias fatores técnicas e estratégias comerciais entre as mega-coporações internacionais. De um lado, podemos lembrar que a tecnologia atual de "clock-cycle" de CPU bateu o limite de 4,3 Ghz, e que enfrenta gravemente o problema de dissipação de calor nos chips devido à existência de milhões de transistores em executar instruções complexas e técnicas especiais de brench prediction. Por tratar de guerras comerciais, percebe-se que o projeto foi uma resposta clara dada por SEC a Microsoft pela sua entrada no mercado de consoles de vídeo games com seu X-BOX. Enquanto a IBM, por sua vez, ambiciou em disafiar a maior produtora de processador do mundo, a Intel.

O surgimento desse mega-projeto, em tese, foi uma tendências natural, e não se deve estranhar tanto sobre a formação de seus membros, como sobre o uso de "novos conceitos" de processador RISC, que foi pesadamente aplicado na plataforma.

A fim de concretizar o projeto, o desenvolvimento foi iniciado com a construção de um centro de pesquisa em Austin em ano 2000, e outro em Texas no março de 2001, reunindo os engenheiros provenientes de três companhias, e que posteriormente foi completados consecutivamentes pelos 10 centros de desenvolvimento espalhados pelo mundo inteiro. Somando, em total, a participação de em torno de 400 técnicos especializados em área.

A quantidade de investimentos relacionados ao processo de desenvolvimento, sem sobra de dúvida, foi imensa. Além do investimento de bilões de dólares em construir 2 faundery de chips de 65 nm, SEC tinha pago IBM umas centenas de milhões de dólares em levantar uma fábrica de linha de produção em leste de Fishkill e Nova York, juntamente com poucas centenas milhões de dólares em desenvolvimento de chip. Porém, tudo isso, deprezando a parte de marketing e salário dos especialistas e as licenças relacionadas às tennologias, que não são nativas dessa união, como por exemplo. Memória XDR RAM e Flex I/O da empresa Rambus, os primeiros chips e simuladores de cell só serão lançados no início de ano 2006.

3. Arquitetura de Cell (PPE, SPE, I/O, EIB) , e desempenho teórico.

O cell foi desenvolvimento na metodologia "full-custom",e em relação aos outros projetos que adotaram mesmo método, podemos destacar algumas detalhes interessantes:

2.0 Tecnologia de Chips

* Chips **CMOS-SOI** (Silicon-on-Insulator) – difere da tecnologia tradicional da fabricação de CMOS-Chip, isto é, transistor CMOS-SOI são construídos em uma camada muito fina de silicio, que por sua vez é depositado sobre uma camada muita grossa de de isolante de SiO2. Usando essa tecnologia reduziria transistores parasitárias e melhorá o caráter condutor-isolante dos transistores. Em sumo, esse método promoverá um melhoramento de qualidade em quase 23% em relação aos chips de CMOS convencionais.

* Tecnologia de múltiplo grid de clock, apesar de ser comumente usados em tecnologia de fabricação de ASIC SoC, em cell possui múltiplos clocks com desvio de clock controlado em torno de 10 ps.

* uso de várias famílias de latches, envolvendo desde S-Latch até D-Latch que é capaz de minimizar o atraso pela inerção (insertion delay), em efeito disso, aumenta-se a velocidade dos letches no processador.

2.1 Resumo básico da arquitetura Cell

Segundo a apresentação que a liga SIT (Sony, IBM, Toshiba) forneceram no congrsso internacional de estado sólido (**ISSCC 2005**), no fevereiro de ano 2005, uma plataforma Cell deve ser capaz de ter seguintes caracteristicas básicas:

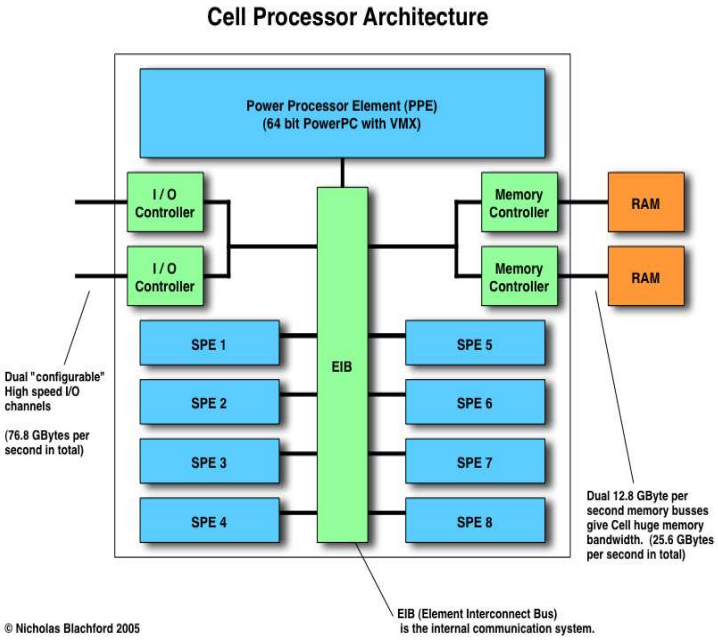

figura 1

1 Elemento Processador Power PC (PPC)

8 Elemento Processador Sinérgico(SPE)

1 Elemento de barramento de Interconexão(EIB)

Controlador de DMA (DMAC)

2 controlador de memória de XDR da empresa Rambus

1 interface (I/O) FlexIcO da mesma empresa

assim, na teoria, essa plataforma sera capaz de ter frequência de clock-cycle igual a 4 Ghz, largura de banda de memória igual 25,6 Gbyte/s, capacidade de processar 256 GFLOPS (pontos flutuantes na frequencia de 4 Ghz) e 256 GOPS (inteiros em frequência de 4GHz) e 25 GFLOPS (pontos flutuantes de precisão dupla em frequnência de 4 GHz) e tem tamanho igual 235 minímetro ao quadrado, e usam 235 milhões de transistores. O consumo de energia, pela estimativa inicial, é em torno de 60-80 Watts em velocidade de 4GHz. Em sumo, pode-se obsevado pela figura de cima.

2.2 PPE (Power-PC Processor element)

uma unidade PPE é um processador convencional que aloca tarefas para cada unidade de PPE. Basicamete é um procesador basedo na arquitetura de de Power-PC de 64 bits, e apresentam 512 K bytes de memória Cache . Devido à esse razão, plataforma Cell, em tese, não apresentará grande dificuldade em termo de compatibilidade com os softwares binários que estão presentes na arquitetura Apple e Power-PC. Apesar de um PPE foi projetado a base de um Power-PC, ele foi feito de uma forma extremamente simplificado, isto é, ele não possuiem complexos ISA e técnicas especias de execução de CPU, que estão presentes num processador como 970/G5. Um PPE em si, é dupla-thread (**SMT**), ou seja, é capaz de executar dois thread simultaneamente,e de processamento ordenado (diferente dos grandes partes de processador CISC, que são essencialmente de processamento fora da ordenamento (OOO), usando intensivo de tecnicas de brenche prediction e escalonamento e paralelismo no nível de instrução). Assim como todo processador in-ordem, um PPE será capaz de executar uma instrução por vez, e com isso, entretanto, cetamente vai afetar o desempenho de PPE, comparado com uma arquitetura convencional de CISC.

A razão pela essa escolha foi na verdade uma solução dada pela desenvolvedores de arquitetura RISC quando os técnicos de Intel bateram o limite de frequência de clock de 4,3 Ghz. Com a simplificação de arquitetura, evitamos o uso de grande quantidade de transistores na composição de processador, e comsequentemente abaixamos tanto o comsumo de energia como na dissipação de calor.

Um outro ponto interessante é sobre a capacidade de suportar instruções de padão SIMD, espcialmente no caso de PPE, ele suporta instruções vetoriais de VMX.(também conhecidos como AltiVec).

2.3 SPE (Sinergistic Processor Element)

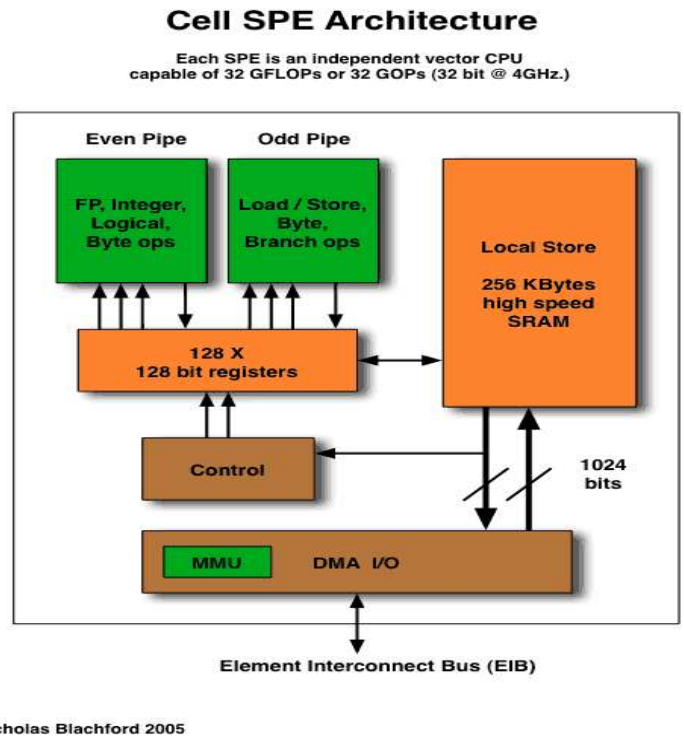

figura 2

podemos dizer que o grande secreto de alto desempenho da plataforma de Cell provém da existência de 8 elemtentos de processamento sinérgico. Basicamente os SPEs são processadores vetorias auto-contidos, e que cada um deles são capaz de processaor indepedentemente. Cada um deles apresentam 128 registradores de 128 bits, 4 unidades de processar pontos flutuantes de 32 GFLOPS e 4 unidades de aritméticas de inteiros em 32 GO, ambos na frequência de 4 GHz. Um SPE também inclui uma pequena memória local em vez de usar um cache para execução de tarefas. São compostos pelas unidades: MFC (Memory Flow Controller), MMU(memory, management unit) ,SPU (synergistic processor unit – unidade de processamento principal de cada SPE), LS (Local Storage de 256 kbytes), 128 registradores de 128 bits cada um, DMA local.

Assim como um PPE, um SPE não tem capacidade de executar instruções complexas e são processadores in-ordem. Entretanto, eles possuem suporte ao procesamento vetorial de instruções de padrão SIMD, especialmete AltiVec. Isto significa que são capaz de exercer múltiplas operações com uma única instrução. Apesar de a idéia dessa técnica já foi amplamente conhecido desde a década de 70, mas, hoje em dia, ainda é amplamente aplicado em aceleradores de aplicativos de multimídia, tais como MMX, SSE, VMX/AltiVec...etc. Assim, cada programa a ser executados dentro da unidade de SPE deve ser, na teoria vetoriariza seus conjuntos de instruções, e com essa técnica permite que o Cell seja um canditado ideal para aplicações na área de processamento de image 3D, audio, produção de video-geme, cáculos científicos.

2.4 Questão de Local Store de SPE:

em vez de usar um cache local como a maiorias dos processadores de dual-core, o cada elemento SPE possuem uma memória local. Em um CPU convencional as tarefas são realizados em registradores e que leiam e esrevam dados diretamente na memória cache, e apresentam velocidade muito mais lentos que a velocidade de operação dos registradores de CPU. Em caso de que os dado que o CPU precisa para completar a execução de cálculos, provocará um “miss” de dado, em consequência disso, fará com que o CPU cai em estado de Stall, esperando a busca de dados na memóia pricipal. Para solucionar os eventuais problemas relacionados ao desempenho e detalhes de implementação de cache, que exige muito tradeoff na escolha de aplicação de princípio de localidade e temporal, o SPE simplemente usam uma memória local(hight speed SRAM) de 256 kBytes, e totalizando em 8 X 256 Kbytes. Cada uma das unidade de memória local, na verdade, são verdadeiros bancos de registradores secundários e apresentam velocidade de acesso tanto na leitura como na escrita próximos a daos registradores de prórpio SPU. De uma visão mais global, um elemento SPE recebe dados de toda a plataforma numa taxa de 64 GigaBytes por secundo, e cada unidade de memória local também é capaz de emanar dados na mesma frequência para o próprio unidade SPE sem a necessidade de acessar a a RAM.

2.4 EIB (Element Interconnection Bus)

O barramento de interconexão inteno de Cell (EIB), como pode ser observado na figura 1, é formado por 4 aneis de 16 bytes que circulam na meia frequência de meia CPU clock. E permite até três transações simultâneos de dados. Ele desempenha o papel de interconetar cada unidade de processamento de Cell (que em teoria, será capaz de tarnsmitir 384 Gbytes de dados por secundo só em cada SPE) e em sumo, apresemtam a bandawidth de em torno de 76,8 Gbytes/s de transação de dados. Particularmente associado ao questão de EIB, dizemos que é essencial a exitência de DMAC para ajudar no alto desempenho de SPE. Foram feitas várias mudanças significativas no design de DMAS desde a época de patenteamento de 2000. Hoje, na plataforma atual de Cell, o controlador de DMA continua presente em Cell e somente controla o acesso a memória principal mas não carrega o papel de proteção de dados. veja a figura abaixo:

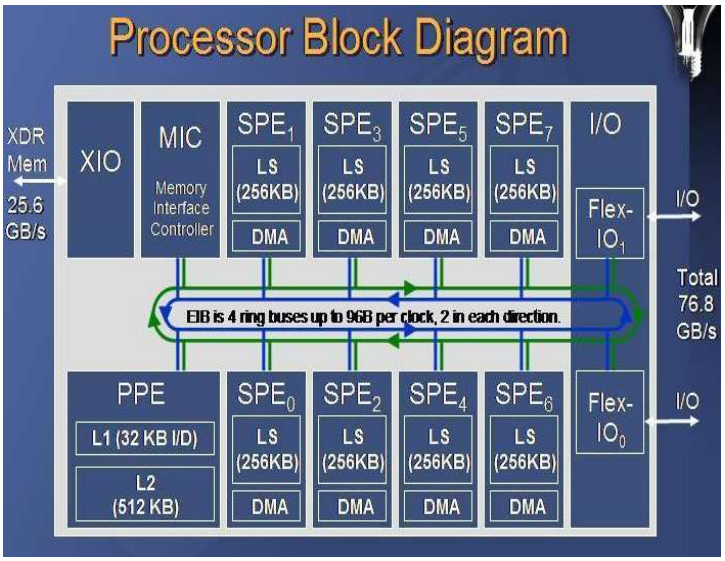

figura 3

I/O e memória

Como todas as unidade internas de processamento precisam ser alimentados com grande velocidade em acesso aos dados, um sistemas de I/O de alta velocidade é absolutamente necessários. Para atenter essa necessidade, STI licenciou as tecnologia de empresa Rambus, os conhecidos "Yellowstone" e "Redwood" que foram usados na construção de XDR RAM e FlexIO do plataforma.

Tanto um como outro são tecnologias interessantes não somente por causa das grandes velocidades que eles oferecem, mas também pela simplicidade de layout que eles apresentam na hora de integralização de plataforma Cell.

4. Modelos de Programação

Se quisermos mapear o modelo de programação de Cell no mundo real, podemos imaginar que os SPE são orquestras, e cada SPE tem maior bandwidth e desempenho que o próprio PPE, e usamos SPE em instruções frequentemente repetidos(aproveitando principio de localidade, e todos eles usam o paradigma de fluxo de dado para executar tarefas)

PPE é o maestro de orquestra que contrói e controla a sincronização global de system. Especialmete Sistema Operacinal que roda em PPE será resposável pela alocação de recursos, controle de dispositivos e responder system call do software. Assim, usamos o PPE para tarefas menos frequentes e menos repetidos com maior grau de acesso aleatório aos dados.

Nesse contexto de mundo musical, o programador desempenha papel de compositor, que se encarrega de escolher e atribuir tarafas ou ao PPE ou ao SPE, considerando o problema de sincronização de sistema e secção crítica.

As preocupações dos projetistas de plataforma nesse estágio inicial são essencialmente ligadas a questão de programabilidade, que evnetualmente decidirá o alto desempenho de systema como todo.Essa preocupação não se deve a uma mudança radial de estilo convencional de programação, mas se deve principalmente pelo esforço em tornar o sistema mais facilemnte programável e mais acessível possível.

O primeiro grande desafio que os programadores em geral defrontará com a arquitetura Cell é a existência de LS (local Storage), e que de fato, exige dos partes de programadores a desenvolver códigos que serão executados dentro da propria memória local de SPE. O secundo gargalo é sobre a questão de natureza de instruções de SIMD, que está presentes em ttempo todo no fluxo geral de dados do sistema. Apesar de os programadores podem simplesmente ignorar esse fato, mas certamente vai afetar o performace de software de uma forma brusca.

Cada elemento de **SPE** diferem de processadores convencionais em diversas formas, entre eles o mais relevante certamente é a existência de 128 registradores de 128 bites. As maneira de executar instruções de brench são evitadas, e juntamento com a necessidade de hardware, são acrescentados algumas instruções que exercem o papel de arbitragem entre memória local e conjunto de instruções. O último diferencial que os SPEs apresentam está no fato de que eles só suportam o único contexto durante cada execução. Isto é, ou estamos no contexto user, ou estamos no contexto kernel, mas nunca simultaneamente.

Um outro aspecto relevante é sobre a existência de um grande conjunto de intruções que são responsáveis na comunicação e transferência de dados entre vários SPEs, e até entre o PPEs e diversos SPEs simultaneamente.

Além de existem muitos modelos que permitam a virtualização de elementos internos de SPE e PPE na hora que tentemos desenvolver aplicativos para a plataforma, o próprio plataforma fornece diversos interfaces de progrmação para simplificar a vida dos programadores, e normalmente os aplicativos são caracterizados pelo compartilhamento de controle de elemento SPE, permitindo a maior grau de uso de recursos constantes, e exigindo do programador a abilidade de desenvolver códigos que trata problema de gerenciamento de SPEs dentro do próprio aplicativo.

Como o um propósito originais de cell, que é ser versátil, o modelo de progrmação também é muito flexível e apresentam vários modelos diferentes de programação, e que muitas instruções que nao faziam parte da arquitetura de 2000, foram introduzidas no modelo atual, justamente para garantir essa flexibilidade aos desenvolvedores de aplicativos.

Os software desenvolvidos no padrão Cell em geral apresenta características que exploram funcionalidades como: facilidade de acessar multiplo- paralelismo, pipeline de Multi-SPEs, SPE atuando como um recurso de rede, massiço fluxo de dados, exploração recursos como SIMD (em especial, os códigos de VMX).

Em seguida, apresentamos alguns desses modelos de programação que foram pensamos em possibilidade de explorar o total potencial de alto desempenho de Cell:

(1) Function Offload model

nesse modelo de programação, os SPEs são usados para acelerar certos tipos de funções que possivelmente afetará o performance de sistema. A idéia básica é alocar as funções complexas presentes em certas bibliotecas de convencionais, que eventualmente são envocados pela rotina principal do programa, para os elementos de SPEs. As funções especiais são otimizados e recompilados para ambiente de SPE, e um objetos executável, compilados em SPEs são injetados em PPC como um módulo objeto numa secção de leitura única de memória principal. Assim, quando um aplicativo que roda em PPE e envoca a chamada de funções presentes em biblioteca, as funções de chamada remota de stub, presentes em PPE fará uma chamada aos procesimentos de subsistema de SPE. Em sumo, o maior desafio por parte de programadores será em decidir quais rotinas, que possivelmente afeta o desempenho de sistema, devem ser alocados ao SPE, e quais subrotinas será de resposabilidade de PPE.

Nesse contexo de modelo de offload, ainda exite um caso especial em que cada SPE forneça funções similares ao um funcionamento de um dipositivo, e que usam registradores especias de Cell, tais como, "on-chip mail-box register", memory maped SPE acessible register.

(2) Job Queue Model

mútiplos SPEs são alocados para execução de tarefas, e PPE mantém uma fila de execução de tarefas na memória de sistema, e PPE pode escalonar tarefas para SPE e monitorar processos alocados. Tarefas de SPE são programas auto-contidos, SPE sozinho jah consegue executar programas, capaz de ter input e gerar output, ter capacidade de endereçamento na memoria de sistema.SPE usam próprio local storage unit para armazenar dados durante o processamento, gerenciador de tarefas cuida de DMA e sincronização de sistema.

esse método oferece uma interface fácil de software, abstração de DMA e sincronização, operaçoes de alta capacidade sincronizador, bom metodo para compartilhar SPE e para muitos outras tarefas independentes.

(3) Self-multi-tasking of SPE

permite que cada SPE exerce processos multi-tarefados de forma cooperativo. Múltiplos SPEs usam memória compartilhada para conjuntos de tarefas, SPE sincroniza com cada um dos outros para executar tarefas ded task-queue , usuário poder livremente programar tarefas de sincronização usando metodos como semáfro, mutexes, livre para definir dara_flow, usando operaçoes de DMA arbitrário. com esse modelo de programar cell, seremos capaz de explorar o máximo capacidade de processamento de cell, maximinizando o uso de múltiplo SPE e superar o dilema de limite de tamanho de Local Store Unit.

(4) (4)Stream Processing

consiste de um stream de input e um stream de output, todo input são vetorisados em um stream de entrads, e essa entradas são ,em pacotes ,atribuidos a unidades de entrada de cada SPE, construindo um armazenando em buffer-local e são processados localmente pelo cada SPE e geram saídas exclusivas referentes à cada entrada e usam LS para processamento desses dados e em seguida passam saídas para stream de out-put.

esse modelo permite que o programador explora o máximo potencial de processamento de SPEs, e com a característica de multi-buffering seriam capaz de esconder o acesso de dado durante a execução, ou seja, maior proteção de dados em runtime. Fluxo de dado são escolhidos para ser naturalmente compatível com o tamanho de local storage model, e com tudo isso, não tem intervenção de PPE.

(5) Cell in Linux

PPE /SPE são programáveis em languagem C/C++, com isso jah ganhamos milhares bibliotecas padrões prontos a ser usados, com isso, basta criar um novo API que adapta unidades PPE e SPE, seremos capazes de rodar programas escritos em c/c++ diretamento em uma arquitetura de padrão cell. Além disso, não temos necessidade de trabalhar diretamente em código assembly de Cell por existir uma função interna reservado para compiladores que abilita as funções de acesso ao SIMD.

Uma outra questão interessante, e possivelmete será diretriz pricipal para a nova arquitetura Cell ser aceito pelo grande mercado dominado pelo intel é o fato da compatibilidade do sistema cell com a linux. Segundo a atual mantedor de kernel de projeto "Linux on Cell" da prórpia IBM, **Arnd Bergmann**, portar o sistema linux para o elemnte PPE é uma tarefa relativamente fácil devido à grande semalhança entre PPE com a convencional arquitetura Power PC e Apple Power Machintosh, porém essa forma, seremos incapaz de revelar todos potencial que o Cell eventualmente fornce para nós. Como somente o Kernel de sistema será capaz de comunicar com cada elemento SPE, então a grande dificuldade se volta para que o Kernel tenha possibilidade de fornecer uma interface de hardware que atende as chamada de sistema e driver de dispositivos para acessar os SPEs. Então a mais importante de uma interface de SPE entre usuário e hardware deve ser capaz de: carregar o binário de porgrams em um SPU (unidade de processamento de uma unidade SPE), transferir dados de memória entre um programa de SPU e o espaço de usuário de aplicativos de linux, e finalmente tarefas de sincronizar fluxo de execução de tarefas.

Segundo **Arnd**, a integração de suporte ao PPE do cell em kernel 2.6.13 será capaz de usar um único imagem kernel de 2.6.13 em plataformas de PowerPc-64 bits, isso inclue PPE de Cell, Apple, Power Mac. Poém ainda falta uma certa distência para surgir um a interface automatica de acesso ao SPE. Atualmente esse problema é contornado pela abortagens de tratar cada elemento SPU com uma partição de systema de arquivo virtual, e que são acessíveis mediante ao system call feito pelo de usuário e atendido pela intromissão de kernel. Mais detalhamento sobre esse abortagem consulte site: **www-128.ibm.com/developerworks/power/library/pa-cell/.**

5. Conclusão e Persperctivas dos Fabricantes

Assim como foi dito pelo prórprio criador de PlayStaion

" ***Though sold as a game console, what will in fact enter the home is a Cell-based computer***. " - Ken Kutaragi

o Cell foi ambiciosamente projetado pela liga STI e com o intuito de desobedecer a famosa lei de mundo computacional "**evolution not revolution"**. Ou seja, em vez de aproveitar e usar os conceitos convencionais de design para promover um melhoramento de desempenho, eles simplesmente revolucionou o conceito de processamento.
Como está sendo produzido em larga escala, abrirá a possibilidade e fornecer um Hardware de baixo preço mas de alto performance e alto grau de vesatilidade em desenvolver aplicações comuns. O mais entre esses benefícios é capaz a satisfazer a perspectiva da STI, que é rápido espalhamento de produtos embarcados (é muito mais amplo e versátil que o mundo de PC) que usam Cell processor como core de processamento principal.

A plataforma Cell é uma nova arquitetura e pela informaçõe até agora nós dispõe, aprentemente ela é forte e ambiciosa, e está indo em sentido de conqistar o convencional mercado de desktop, ocupdo pelo Microsoft e Intel. Porém, antes de concretizar o seu primeiro passo de caminhada, eventualmente que ela precisam solucionar os problemas que são levantados dentro do próprio contexto da mudança de conceito de procesamento. Terá que enfrentar novs problemas e descobrir novas saídas para continuar a corrida. Apesar de como uma ameaça para algumas corporações tradicionais, para alguns, ela representa a possibilidade de novas mudanças e novas oportunidade de mercado.

6 . Refrências

[1] **J. A. Kahle, M. N. Day, H. P. Hofstee, C. R. Johns, T. R. Maeurer, D. Shippy: Introdution to the Cell Multiproceessor.**

[2] Jan M. Rabaey, Ananthan Chandrakasan, Borivoje Nikolic: Digital Integrated Circuits, a design perspective, 2 ED.

[3] **B. Flachs, S Asano, S.H Dhong, P Hofstee. Streaming Processing Unit for a Cell Processor. ISSCC 2005 / setion7/ 7.4**

[4] D. Pham, S Asano, M. Bolliger, M. N. Day, H. P Hofstee. The Design and implementation id a first-generation Cell processor.

[5] **Cell Arquitecture Explained, site : http://www.blachford.info/computer /Cell/**

[6] **Dominic Mllinson, Mark DeLoura, Cell: a new platform for digital entertaiment, site: www.research.scea.com/research/html/CellGDC05/**

[7] Arnd Bergmann, Spufs: The Cell synergistic Processing Unit as a virtual file system,

site: www-128.ibm.com/developerworks/power/library/pa-cell/