LSI-SO.01 - Um Sistema Operacional Multiprocessador

Edson Toshimi Midorikawa, Felipe Knop,
Roberto Diniz Branco, Wang Kuei Yu

Laboratório de Sistemas Integráveis
Escola Politécnica da Universidadade de São Paulo
Av. Prof. Luciano Gualberto, 158 - Trav.03
Cep.: 05508 Caixa Postal 8174 - São Paulo
tels.: (011) 815-9322 (R.270) 211-4574

RESUMO

Este trabalho descreve o sistema operacional multiprocessador LSI-SO.01, compatível com o UNIX System V.3, em desenvolvimento no LSI-USP. Este sistema fornece meios para a obtenção de processamento paralelo através da utilização dos recursos da arquitetura multiprocessadora da máquina. Descrevem-se as principais características do sistema, tais como, seu sistema de arquivos, memória virtual e aquelas referentes a processamento paralelo e multiprocessamento.

ABSTRACT

This paper describes the multiprocessor operating system LSI-SO.01, which is compatible with UNIX System V.3, under development at LSI-USP. This system provides means to obtain parallel processing by taking advantage of the multiprocessor architecture. The main features of the system are described, such as its file system, virtual memory and its parallel processing supports.

1. INTRODUÇAO:

Este trabalho faz uma breve apresentação do sistema operacional LSI-SO.01, atualmente em desenvolvimento no Laboratório de Sistemas Integráveis (LSI) da Escola Politécnica da USP. Este sistema operacional será inicialmente instalado no Minissupercomputador MS8701, também em desenvolvimento no LSI.

Na descrição do software, apresentamos suas principais características, bem como alguns aspectos de sua implementação. Dá-se uma visão geral do sistema de arquivos e da memória virtual. Daí passamos a detalhar as características referentes a processamento paralelo e multiprocessamento.

Façamos inicialmente uma breve descrição da arquitetura do MS8701, que foi desenvolvida sinergicamente com este sistema operacional.

2. DESCRIÇAO DA CIRCUITARIA (HARDWARE):

Na figura 1 a seguir, encontra-se um diagrama de blocos da organização do sistema MS8701.

Cada um dos componentes deste diagrama é descrito de maneira breve a seguir:

1 - Placa de Processamento Geral, PPG

É constituída por quatro módulos de processamento (MP's), sendo cada um deles constituído por: um processador 68020 (16.67MHz), um coprocessador aritmético MC68881 (16.67 MHz), um gerenciador de memória paginada (PMMU) MC68851, uma memória local de 256kb acessada em modo supervisor e uma memória global com extensão de 4Mbytes. A PPG é composta por 4 processadores, cada um dos quais com uma cópia do sistema operacional em memória própria, e uma via local para acesso à memória compartilhada.

Dessa maneira, a memória compartilhada local contém os processos a serem executados juntamente com as estruturas de dados do sistema compartilhadas pelos 4 processadores.

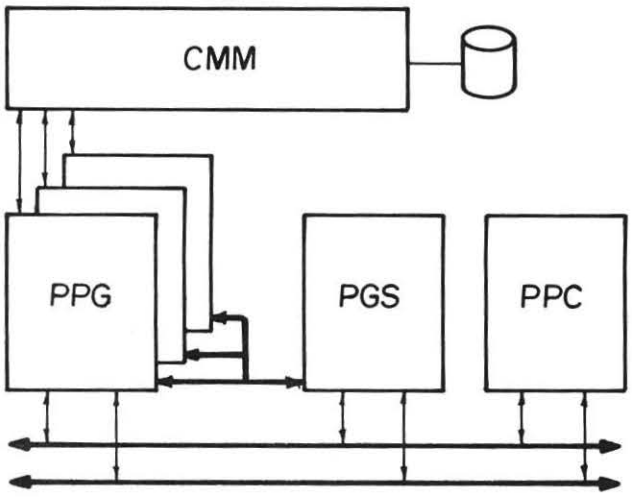

fig.1 - Arquitetura do Sistema.

2 - Computador de Memória de Massa, CMM.

É composto por um processador que executa os programas de gerência de discos e escalonamento de requisições.

3 - Subsistema de Comunicações de Caracteres, SCC.

É o elo de ligação entre o sistema e o usuário, o qual pode estar localizado em um terminal ou local, ou remoto em algum ponto de rede ou entre programas rodando em alguma outra máquina.

4 - Placa de Gerenciamento de Sistema, PGS.

É responsável pelo gerenciamento e atendimento de interrupções. É constituída por um processador MC68020, uma memória local onde devem residir tabelas do sistema e outras estruturas globais do S.O.

Cada PPG tem capacidade de enviar uma mensagem de interrupção à PGS através de escrita na região da memória compartilhada destinada às mensagens. A interface com a SCC se processa da mesma forma.

A PGS, por sua vez, pode interromper ou enviar uma mensagem à PPG ou a um módulo de processamento MP através de uma via especial destinada a enviar mensagens e interrupções, DINT.

A comunicação entre PGS e CMM é feita por um duto de mensagens, pelo qual o CMM informa o atendimento de um comando encerrado e sinaliza uma interrupção.

3. DESCRIÇÃO DA LOGICIONARIA (SOFTWARE):

O LSI-SO.01 é um sistema operacional multiprocessador atualmente em desenvolvimento no Laboratório de Sistemas Integráveis (LSI) da Escola Politécnica da Universidade de São Paulo. Sua principal característica é o suporte ao multiprocessamento e processamento paralelo, ou seja, o sistema suporta a existência de mais de um processador que tenha a capacidade de executar processos, bem como, possui características que propiciam a distribuição da execução de um programa entre vários processos. O usuário possui meios de determinar a maneira pela qual os processadores recebem tarefas a serem executadas.

Como outras características do sistema podemos citar:

- sistema compatível com o UNIX System V Release 3;
- modularidade;
- portabilidade;
- sistema de arquivos rápido;
- sistema de arquivos distribuído;
- gerenciamento de memória virtual por demanda de páginas;
- suporte a processos de tempo real.

Das características de interface definidas no documento editado pela AT&T ("System V Interface Definition" [SVID86]), o sistema LSI-SO.01 implementa todas as chamadas ao supervisor, inclusive as consideradas como extensões, as rotinas de biblioteca e os utilitários básicos e de suporte mínimo para a utilização do sistema. Além disto, possui algumas chamadas ao supervisor especiais para o controle de multiprocessamento e processamento paralelo.

4. SISTEMA DE ARQUIVOS:

O sistema de arquivos do LSI-SO.01 apresenta algumas diferenças em relação ao sistema de arquivos tradicional do

UNIX [RITC74]. Estas mudanças visam proporcionar ao usuário um ambiente mais eficiente e poderoso.

4.1 - Sistema de Arquivos Rápido

No sistema LSI-SO.01, é implementada uma nova organização de dados no disco, visando minimizar a ineficiência na recuperação de dados existentes nos dispositivos de armazenamento de massa, sem perder a compatibilidade com UNIX tradicional [McKU84].

A organização de um disco lógico (partição) é a seguinte: para manter a "compatibilidade", o bloco lógico número 0 é reservado para "boot" e o bloco número 1 para o super-bloco; o restante do disco lógico é organizado em algumas subpartições (ou regiões) consecutivas para facilitar a alocação dos blocos de um mesmo arquivo de modo a estarem fisicamente próximos - uma tentativa de diminuir o atraso causado pelos acessos aleatórios no disco.

Cada subpartição consiste de um grupo de blocos pertencentes àquela região do disco e um grupo de descritores de arquivos reservados aos arquivos daquela região; desta forma, garante-se a proximidade física entre o descritor de arquivo e os blocos de dados.

Os blocos lógicos (normalmente grandes) podem ser fragmentados em unidades menores para o melhor aproveitamento do espaço disponível - os blocos lógicos grandes têm a vantagem de serem mais eficientes no acesso, porém acarretam desperdício de espaço para arquivos pequenos -.

A política de alocação de blocos determina que os blocos de um mesmo arquivo e os arquivos de um mesmo diretório sejam localizados na mesma subpartição; desta forma, estão fisicamente próximos, aumentando, assim, a eficiência de acessos sequenciais aos arquivos.

4.2 - Sistema de Arquivos Distribuído

O Sistema de Arquivos Distribuído (SAD) é uma das características inovadoras oferecidas pelo LSI-SO.01. O SAD acrescenta uma nova dimensão no ambiente de computação do usuário oferecendo acesso transparente aos arquivos localizados em outras máquinas independentes (remotas). ([RIFK86], [GOUL86])

A intenção é proporcionar ao usuário uma visão de sistema de arquivos extendido através de redes e discos remotos.

Além de prover o mecanismo das chamadas de sistema mount/umount (acessos aos arquivos pertencentes aos discos removíveis ou partições dos discos da própria máquina), o objetivo principal do SAD é fornecer aos usuários e aplicativos um meio de acessar arquivos remotos, levando em consideração os seguintes pontos:

- acesso transparente - para o nível usuário, acessos a arquivos locais ou remotos são indistinguíveis;
- preservação de semântica - todos os tipos de arquivos são acessíveis, inclusive os arquivos especiais (dispositivos) e os "pipes" nomeados. Provê suporte para travamento de arquivos e registros;
- compatibilidade binária;
- independência do tipo de rede;
- portabilidade para diferentes implementações de circuitaria ("hardware").

Extendendo a noção de mount/umount do UNIX, o SAD permite que uma sub-árvore de uma máquina servidora seja logicamente conectada à árvore de arquivos local de uma máquina cliente.

Estabelecendo a conexão lógica entre duas máquinas da rede, através dos pontos de entrada e ligação (diretório de ligação), o usuário "caminha" de uma estrutura de árvore a outra sem tomar conhecimento do desvio (a localização física do arquivo é irrelevante para o usuário).

O sistema operacional da máquina cliente mantém dados dos diretórios de ligação de modo a possibilitar a localização da máquina e o acesso aos arquivos remotos.

A comunicação entre duas máquinas é feita através da troca de mensagens pela rede. Os comandos enviados à máquina servidora são pedidos de dados (blocos) ou informações. As informações são necessárias para se determinar o percurso para atingir um arquivo ou a localização dos dados.

O uso de um protocolo de comunicação modular e da interface no nível de transporte definido pelo UNIX, permite que a implementação seja independente da rede de comunicação utilizada.

O SAD fornece os seguintes mecanismos de proteção:

- autenticação - identificação da máquina;

- autorização - permissão de acesso dado a um grupo de máquinas;
- permissão do usuário - mapeamento da identificação do usuário;
- permissão normal do UNIX - leitura/escrita/execução para usuário, grupo e outros;
- travamento de registros.

5. MEMORIA VIRTUAL:

O projeto de memória virtual no LSI-SO.01 proporciona para os processos um espaço de endereçamento total de 8 Gbytes, dividido em espaços de 4 Gbytes, para o código e dados.

Ele permite compartilhamento de memória entre os processos com cópia em escrita ("copy-on-write") ou com permissão de leitura e escrita.

A implementação da memória virtual é semelhante à do UNIX da AT&T [BACH86]. A política de substituição de página adotada é o "working-set", proposta por P. J. Denning [DENN68], que procura manter na memória apenas as páginas referenciadas recentemente. Existe um mecanismo de "cache" de páginas que visa minimizar a frequência de acessos ao disco mantendo informações sobre as páginas livres. O uso deste mecanismo pelo sistema operacional resulta numa melhora do desempenho do sistema.

Quando um processo é executado, as páginas das suas regiões de texto e de dados são carregadas sob demanda ("demand-paging") do seu arquivo objeto a cada falta de página ("page fault").

Esta política de carga de páginas apresenta um custo muito elevado para programas que são frequentemente chamados. Para reduzir este custo, o LSI-SO.01 pode manter as páginas de códigos na memória mesmo depois do processo terminar sua execução. Este mecanismo é chamado "stick regions".

Uma das vantagens de memória virtual no LSI-SO.01 se refere ao tempo necessário para iniciar a execução de um programa. Ao invés de realizar uma cópia física do processo pai, o núcleo apenas duplica as tabelas de páginas e outras estruturas de dados, de modo que as páginas físicas ficam compartilhadas entre os processos, até que um deles escreva nelas. Isto faz com que a duplicação dos processos (fork) seja muito rápida.

Devido à arquitetura do minissuper MS8701, o gerenciamento da memória de cada PPG é feito de modo independente em relação às demais, ou seja, processos em execução numa PPG utilizam páginas de sua placa, exceto aquelas compartilhadas com processos de outras placas.

6. MULTIPROCESSAMENTO E PROCESSAMENTO PARALELO:

A implementação de um sistema operacional multiprocessador envolve as seguintes questões não presentes em sistemas monoprocessadores: como gerenciar os processadores de modo a equilibrar a carga entre eles, como aproveitar a existência de vários processadores para melhorar o desempenho de um programa e como resolver problemas de concorrência e sincronismo.

6.1 - Paralelismo

O usuário poderá obter uma melhora no desempenho de seus programas dividindo-os em tarefas a serem executadas por diversos processadores de forma paralela.

O paralelismo é conseguido através de uma biblioteca de rotinas que suportam "microtasking" e "multitasking", permitindo que o usuário explicite construções paralelas em seus programas.

Os serviços fornecidos por estas rotinas são:

- criação, disparo e destruição de processos;
- sincronização de processos: semáforos, barreiras e seções críticas;
- chaveamento entre execução paralela e sequencial.

Estas rotinas são implementadas utilizando-se as chamadas de sistema fork, wait, kill e o mecanismo de semáforos do UNIX System V. Deve-se notar que a implementação do fork no LSI-SO.01, como descrito na seção anterior, apresenta uma baixa sobrecarga ("overhead"), diminuindo o tempo de ativação dos processos.

O compilador e o sistema operacional são conectados de maneira que os dados globais do processo "pai" são alocados em uma região de memória compartilhada com os processos "filhos", de forma a facilitar a sincronização e a manipulação das variáveis globais.

A partir da biblioteca de processamento paralelo, serão construídas rotinas matemáticas que efetuam cálculos com

vetores e matrizes, resolução de equações diferenciais etc., utilizando vários processadores paralelamente.

Outra maneira de se melhorar o desempenho de programa é a sua divisão em rotinas executadas em processos diferentes de forma paralela ("multitasking"). Estes processos podem comunicar-se através do mecanismo de mensagens ou memória compartilhada fornecido pelo sistema operacional.

Um possível uso para as rotinas de "microtasking" é a execução paralela de comandos de laços do tipo "do" do Fortran.

6.2 - Multiprocessamento

Cada um dos processadores pode executar tanto os programas de usuário quanto o sistema operacional de modo simétrico. Uma configuração mestre-escravo ("master-slave"), onde apenas um processador executa um sistema operacional, apesar de ter uma implementação mais simples [BACH84], foi descartada por ser pouco eficiente (existem estatísticas que mostram que um processo no UNIX executa em média quase que 50% do tempo em modo supervisor). As dificuldades de implementação de um sistema simétrico serão detalhadas no ítem "sincronismo".

Para se evitar a sobrecarga do barramento global, os processos serão escalonados somente em um dos quatro processadores pertencentes ao mesmo conglomerado ("cluster"). Assim sendo, inicialmente o balanceamento de carga será estático e feito no instante de criação dos processos. Desse modo a PGS, além de gerenciar as interrupções, monitora a carga em cada conglomerado e escolhe o menos carregado para a atribuição ao processo que acaba de ser criado. A medida de carga de um conglomerado baseia-se no número de processos em execução ou prontos para correr e na quantidade de memória ocupada.

Estão sendo realizados estudos para a implementação de um mecanismo de balanceamento dinâmico de carga, cuja principal desvantagem é a contenção do barramento global causada pela migração dos processos. No entanto, essa desvantagem pode ser compensada nos casos onde o sistema apresenta grande disparidade de carga entre os diferentes conglomerados.

6.3 - Sincronismo

Toda vez que estruturas de dados compartilhadas são manipuladas concorrentemente, podem ocorrer inconsistências de dados ou mesmo a corrupção das estruturas, como no caso de estruturas com ponteiros. Isto pode ocorrer em sistemas multiprocessadores, se tivermos mais de um processador manipulando a mesma estrutura concorrentemente ou em sistemas monoprocessadores, se ocorrer uma interrupção durante a manipulação (um acesso a estrutura feito durante a rotina de atendimento de interrupção poderá encontrá-la em um estado inconsistente).

No sistema UNIX monoprocessador, o código assume que o núcleo nunca é interrompido, ou seja: uma interrupção não poderá causar reescalonamento na CPU se o processador estava executando código do núcleo. Desta forma, as estruturas de dados do núcleo não precisam ser protegidas, a não ser aquelas referenciadas por rotinas de interrupção, e assim, os dados podem ser protegidos inibindo-se as interrupções durante a sua manipulação.

Para os sistemas multiprocessadores, a solução acima não é suficiente, pois as estruturas podem ser acessadas por diversos processadores ao mesmo tempo. Neste caso, é necessário utilizar-se um mecanismo que assegure a exclusão mútua dos processos que tentam acessar concorrentemente as estruturas compartilhadas do núcleo. O mecanismo adotado é o de semáforos binários, implementados através de instruções "test-and-set", que garantem operações indivisíveis nos semáforos. Enquanto um processo está de "posse" do semáforo, os outros ficam bloqueados até que este seja liberado e, desta forma, o processo poderá obter um acesso exclusivo (temporário) à estrutura.

Existem 2 alternativas para os processos bloqueados no semáforo ([RUSS87]) ([BACH84]):

a) esperar em "busy wait" até o semáforo ser liberado; é a melhor na maioria dos casos, pois geralmente as estruturas são utilizadas por pouco tempo e a sobrecarga ("overhead") de travamento e destravamento do semáforo torna-se bastante grande.

b) suspender-se, sendo ativado pelo processo que o liberou; só é utilizado quando houver a possibilidade de um processo ser suspenso com o semáforo travado, o que leva a uma grande contenção do recurso compartilhado pela indeterminação do tempo

de suspensão do processo.

Para evitar que a espera em "busy wait" leve a um congestionamento no barramento, duas providências foram tomadas: a análise do código, para evitar que os semáforos possam ficar travados por muito tempo e a execução do "busy wait" com menor frequência de acesso ao semáforo.

AGRADECIMENTOS:

Gostariamos de agradecer ao Prof. Dr. João Antonio Zuffo pelo incentivo dado para a elaboração deste trabalho e a FINEP pela colaboração financeira ao projeto do Minissuper MS8701 e do Sistema Operacional LSI-SO.01.

REFERENCIAS BIBLIOGRAFICAS:

[BACH84] BACH, M.J. e BUROFF, S.J. "Multiprocessor UNIX Operating Systems", AT&T Bell Labs. Tech. Journal, vol.63, no.8, Outubro de 1984, pp.1733-1749.

[BACH86] BACH, M.J. "The Design of the UNIX Operating System" Prentice-Hall, 1986, xiv + 471 pags.

[DENN68] DENNING, P.J. "The Working Set Model for program behavior" Comm.ACM, vol.11, no. 03, maio de 1968, pp. 323-333.

[GOUL86] GOULD E. "The Network File System Implemented on 4.3BSD" USENIX Summer Conference Proceedings", Atlanta, Georgia, Junho de 1986, pp.294-298.

[McKU84] MCKUSICK, M.K. e outros, "A Fast File System for UNIX" ACM Trans.on Computer Systems, vol.2, no.3, Agosto de 1984, pp.181-197.

[RUSS87] RUSSEL C.H. e WATERMAN,P.J "Variations on UNIX for Parallel-Processing Computers", Comm.ACM, vol.30, no. 12, Dezembro de 1987, pp.1048-1055.

[RIFK86] RIFKIN, A.P. e outros,"RFS Architectural Overview" USENIX Summer Conference Proceedings", Atlanta, Georgia, Junho de 1986, pp. 248-259.

[RITC74] RITCHIE, D.M. e THOMPSOM, K. "The UNIX Time-Sharing System" Comm.ACM, vol.17, no.7, julho de 1974, pp.365-375.

[SVID86] AT&T, "System V Interface Definition" - 3 volumes, 1986.