EXECUTIVO MULTI-TAREFA PARA PROCESSADOR FRONTAL DE REDE

David Turnell e Joberto S.B. Martins
UFPb - GRC: Grupo de Redes de Computadores
CP 10032
58100 - Campina Grande - Pb.

RESUMO

Este artigo descreve um Executivo que fornece serviços básicos de comunicação, temporização, sincronização, alocação de memória, e escalonamento de tarefas para a implantação de pacotes de rede multi-tarefa num Processador Frontal de Comunicação. O desenvolvimento leva em conta os critérios específicos de implantação de pacotes de rede. Os serviços, as primitivas, o protocolo de comunicação entre tarefas e as estruturas de dados utilizados são descritas em detalhe.

1. INTRODUCAO

O desenvolvimento de software de redes apresenta atualmente algumas tendências bem definidas.

Uma destas tendências é a adoção do Modelo de Referência da ISO ("International Standards Organisation") [1] como referência para as implantações. Neste modelo, a comunicação é quebrada em tarefas específicas implantadas por 7 camadas independentes definidas numa estrutura hierárquica. Do ponto de vista de implantação, a adoção deste modelo implica num pacote multi-tarefa de relativa complexidade [2].

Uma outra tendência perceptível é a utilização de Processadores Frontais de Comunicação (PFC's) ("front-ends") na implantação de pacotes de redes. A estratégia PFC é impulsionada pelos seguintes fatores:

- disponibilidade de protocolos de camadas inferiores em circuitos integrados (CI's), normalmente sob a forma de coprocessadores;
- redução de custos e melhora de desempenho de microprocessadores (UCP's), viabilizando o projeto de módulos autonomos, inteligentes e relativamente complexos;
- necessidade de redução da carga de processamento das UCP's, principalmente com a implantação de um pacote de software complexo como um pacote de rede ISO/OSI e
- crescente tendência de paralelização das estruturas computacionais de forma geral, levando a um ganho no desempenho global.

Em resumo, a solução PFC para redes implica na alocação de algumas camadas inferiores de protocolos numa unidade inteligente e independente da UCP principal.

Esta solução apresenta, no entanto, dificuldades que devem ser solucionadas. O principal problema vem do fato que, normalmente, não se aloca nenhum sistema operacional (SO) no PFC. Assim, necessita-se de um conjunto de serviços como sincronização, mensagens, escalonamento, etc..., para permitir a implantação de pacotes multi-tarefa. Estes serviços não devem ser tão gerais quanto os seus homólogos dos SO's e, sobretudo, devem levar em conta as necessidades específicas das aplicações alvo.

Em seguida, é apresentado um executivo multi-tarefa fornecendo serviços básicos para a implantação de um pacote de rede num PFC.

2. AMBIENTE DE OPERACAO DO EXECUTIVO

O executivo descrito neste artigo gerencia um PFC num super-microcomputador com as seguintes características gerais (fig. 1):

- arquitetura modular;
- barramento suportando multi-microprocessamento (várias UCP's podem ter acesso ao PFC) e
- sistema multi-usuário.

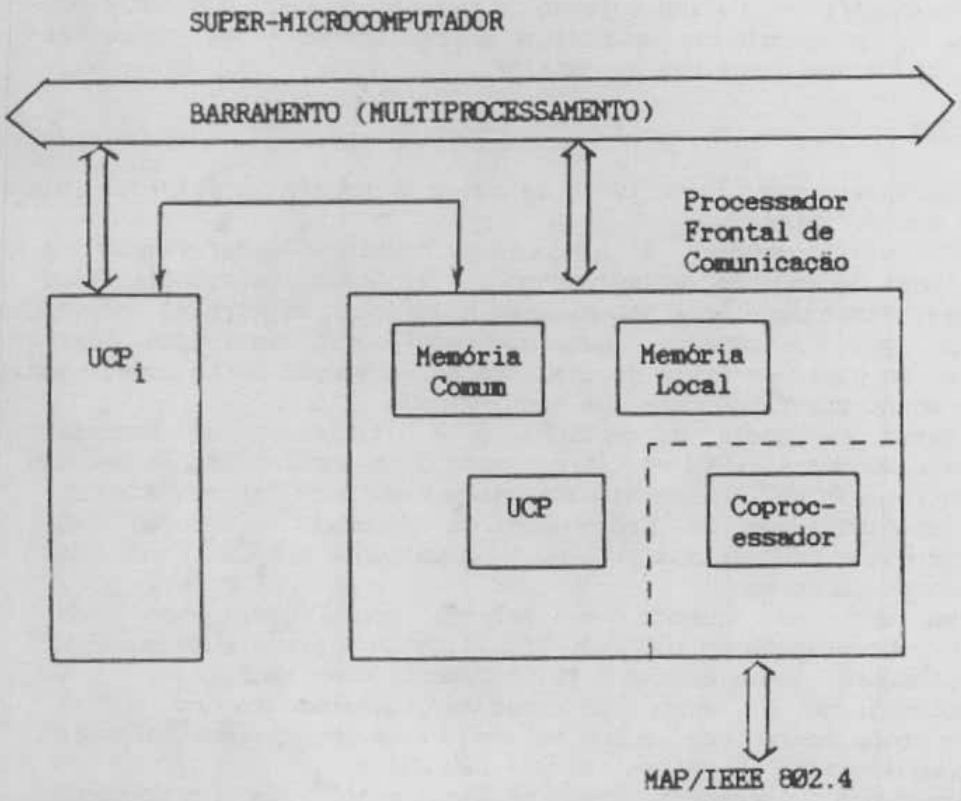

Fig. 1: Arquitetura Simplificada de um Computador com um Processador Frontal de Comunicação (PFC)

O PFC funciona sempre como módulo escravo da UCP principal e sua arquitetura é composta por:

- microprocessador de 16 bits (80186);
- coprocessador controlador de rede compatível com o padrão IEEE 802.4;
- memória comum de comunicação com a UCP;
- memória local de trabalho e
- barramento compatível com o PP-BAR do Processador Preferencial [3].

Do ponto de vista funcional, o PFC é um módulo inteligente que gerencia algumas camadas inferiores de protocolos. No caso, o PFC foi funcionalmente direcionado para implementar protocolos MAP ("Manufacturing Automation Protocols") [4] específicos para a automação industrial.

O executivo independe da funcionalidade específica das camadas inferiores implantadas no PFC, podendo igualmente ser utilizado, por exemplo, num PFC compatível com o Ethernet.

Para a implantação do executivo foi utilizada a linguagem C, exceto para as funções de baixo-nível, obrigatoriamente escritas em "assembler".

3. Especificação do Executivo

3.1 Mapa de Memória

O hardware do PFC é composto do microprocessador 80186 de 16 bits, coprocessador IEEE 802.4 MC68824, e até 1M de memoria RAM, com expansão em blocos de 256 Koctetos. O mapa da memória é o seguinte:

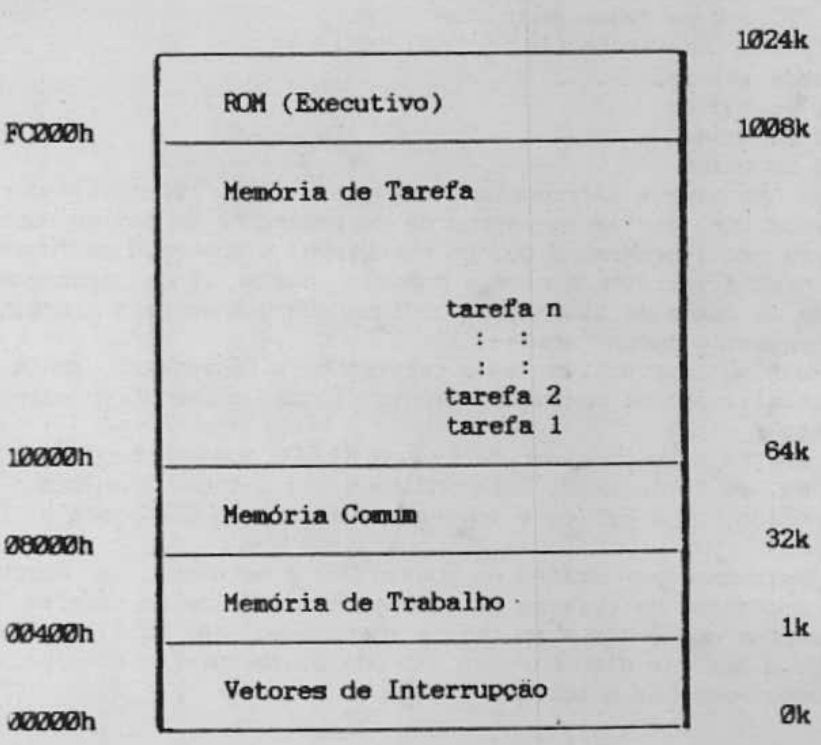

Fig. 2: Mapa da Memória

O primeiro kocteto de memória é usado para vetores de interrupção do 80186. Os vetores são numerados 0 - 255, e os vetores a partir de 128 são usados para chamar as primitivas do Executivo.

O Executivo tem 31k de memória RAM como área de trabalho dele, que é usada para buferização de mensagens, informação de estado das tarefas, lista de memória livre, etc.

A memória comum de 32k é usada para comunicação entre o PFC e o hospedeiro. Um protocolo de comunicação foi definido para permitir a

passagem de mensagens nas duas direções com o sincronismo sendo realizado através de linhas de interrupção.

O código das tarefas é instalado na memória de tarefa. O bloco de código de uma tarefa contem o código, os dados, e a pilha dela. As tarefas são carregadas consecutivamente na memória. A memória restante na área de tarefa é usada pelo Executivo para alocação dinâmica sob a demanda das tarefas.

O ultimo 16k de memória contem o Executivo em ROM.

3.2 Tarefas

Durante o carregamento ("boot") do sistema as tarefas são entregues ao PFC uma a uma pelo hospedeiro, usando o protocolo de mensagens Hospedeiro/PFC, e instaladas na memória de tarefa. O processo de carregar uma tarefa é executado da seguinte forma. Inicialmente, o hospedeiro manda uma mensagem ao PFC com os dados seguintes:

- identificador da tarefa (ID)
- prioridade inicial
- tamanho de código
- tamanho de dados
- tamanho de pilha

O PFC cria então uma tarefa diferenciada das demais pelo identificador de tarefa - ID. Em seguida, várias mensagens de carregamento de código contendo o codigo da tarefa são trocadas. O código executável é quebrado em funcão do limite de 32koctetos existente para a memória comum. Cada mensagem de carregamento tem um campo de tamanho de código, que quando zero termina o processo de carregamento desta tarefa.

Quando a última tarefa tiver sido carregada, o Hospedeiro manda uma mensagem de inicialização de operação, depois da qual o Executivo começa a comutar as tarefas.

A criação das tarefas pode ser feita sob MS-DOS com um compilador que produz código final da forma EXEC. Este código é 'load-time' localizado e um utilitário fornecido com o PFC pode converter um arquivo EXEC para a forma apropriada do PFC.

Como a organização das camadas no modelo OSI é estática, o Executivo não fornece um mecanismo de criação ou eliminação dinâmica de tarefas. Uma vez que as tarefas começam a executar, a instalação de mais tarefas é impossível porque a memória disponivel é alocada dinâmicamente como parte do processo de gerenciamento da memória.

3.3 Serviços do Executivo

O executivo desenvolvido fornece um conjunto básico de serviços para a implantação de um software multi-tarefa. Na definição destes serviços levou-se em conta igualmente alguns requisitos importantes para a implantação de um pacote de rede no PFC.

Os serviços do Executivo são os seguintes:

Prioridade:

Em sistemas multi-tarefa que operam em tempo real, como no caso de PFC's para redes, a mudança dinamica na prioridade das tarefas é um requisito básico. O Executivo fornece um serviço para uma tarefa alterar sua

prioridade em relação as outras. O algoritmo de roteamento de tarefa assegura que as tarefas com prioridades baixas conseguem tempo de UCP. A execução é tal que uma tarefa com prioridade 10 ganhará 10 vezes o número de fatias de tempo de uma tarefa de prioridade 100.

Auto-Bloqueio:

O Executivo fornece um servico para uma tarefa se desativar por um tempo especificado ("sleep"). Durante este tempo a tarefa não consome recursos da UCP.

Eventos:

O Executivo fornece um serviço de notificação de evento. Para uma camada do modelo OSI , um evento é, por exemplo, um estouro de temporização, ou uma chegada de mensagem. Este serviço permite uma tarefa se bloquear enquanto não há nada para ela fazer. Na ocorrência do evento, um parâmetro é devolvido à tarefa com um valor informando qual o tipo do evento.

Gerenciamento de Memoria:

As tarefas executando as camadas de modelo OSI usam extensivamente "buffers" para manipulação de dados. Uma tarefa tem normalmante alguns "buffers" alocados estaticamente na sua área de dados e, eventualmente, solicita "buffers" adicionais segundo necessidade determinada pelos usuários da camada.

O executivo fornece um serviço de gerenciamento dinâmico de memória mantendo um conjunto de "buffers" de onde os processos podem solicitar e devolver memória.

A memória alocada pelo executivo é o restante da memória não utilizada como memória de tarefa. A lista de blocos livres é mantida com um algoritmo "first-fit" e, para evitar uma fragmentação excessiva, o executivo mantem uma granularidade de 256 octetos.

Temporização:

Temporização é um requisito fundamental de operação do modelo OSI. Uma tarefa temporiza da seguinte maneira. Ela usa o serviço de temporização para lançar um temporizador depois de, por exemplo, mandar uma mensagem a outra tarefa. No lançamento, a tarefa define um identificador ao temporizador. Assim usando identificadores, uma tarefa pode lançar várias temporizações em paralelo. Posteriormente, a tarefa usa o serviço de evento, para ser notificada do estouro de uma temporização.

Durante a temporização, a tarefa pode obter o valor atual da contagem.

Comunicação entre Tarefas:

As camadas do modelo OSI se comunicam através de mensagens de controle e dados. As mensagens variam em tamanho e formato. O Executivo fornece um serviço de roteamento de mensagens baseado em filas. Como não há necessidade para sincronismo entre tarefas além da passagem de mensagens, um serviço complementar de semáforo não é implementado.

O roteamento de mensagens é feito usando os identificadores (ID's) das tarefas envolvidas. Mensagens podem ser enviadas entre tarefas sem necessidade de abrir/fechar conexões. Na recepção, a tarefa emissora é identificada.

A comunicação entre um processo no Hospedeiro e uma tarefa no PFC é feita da mesma forma que entre tarefas, só que o identificador (ID) dos processos no Hospdeiro é na faixa de 128 a 255. O Executivo, quando recebe um ID nesta faixa, manda uma mensagem de dados ao Hospedeiro usando a memória comum. O gerenciador ("handler") no Hospedeiro é responsável pelo direcionamento do tráfego na interface para seus processos.

Depuração de Tarefas:

A questão de depuração de uma tarefa é muito importante considerando que o software residente no PFC pode ser razoavalmente complexo e que ele vai rodar num ambiente isolado das ferramentas normais de depuração existentes no computador.

A primeira fase de depuração seria feita na UCP com o teste dos módulos isolados de uma tarefa. A segunda fase de testes também deve ser realizada na UCP, com a tarefa sendo testada num "test-bed" como uma unidade. O "test-bed" gera as condições de entrada e monitora a resposta feita pela tarefa. Estas duas fases devem detetar mais de 90% dos erros existentes.

Finalmente vem a terceira e mais difícil fase da depuração em que a tarefa parece funcionar e é instalada na placa. Com certeza, ainda vão haver problemas, que serao de dois tipos. O primeiro é do tipo em que a tarefa faz uma operação legal mas na hora errada. O segundo tipo é aquele em que a tarefa faz um erro grosseiro, por exemplo, escrever na área de código dela ou talvez de uma outra tarefa. Os dois tipos de erro tem uma maneira diferente de detecção.

O primeiro tipo pode ser detectado através do uso de uma tarefa adicional de monitoração, para a qual a tarefa sob teste continuamente manda mensagens de estado.

O segundo tipo de erro pode ser detectado através de um modo de operação especial do Executivo. Na mensagem M_Inicia_Tarefa da UCP para a placa, a UCP pode especificar se a tarefa é para ser rodada normalmante, ou vai ser interpretada. Se for interpretada, o Executivo não permite que a UCP rode o código da tarefa diretamente, mas através de um interpretador. Cada acesso à memória pode ser testado para detectar acessos fora da área de tarefa e também fora de um bloco de memória alocado à tarefa, e tambem detectar escrita na área de código.

4. Primitivas do Executivo

4.1 Grupos de Primitivas

As primitivas do Executivo fornecem os serviços descritos na seção anterior. Elas são chamadas através de um "reset" de software (RST), e são agrupadas em quatro tipos - básicas, comunicação, gerenciamento de memória e temporização. As primitivas são:

grupo 1 - básicas
- X_Prioridade
- X_Dormir

X_Espera_Evento

grupo 2 - comunicação
X_Manda_Mensagem
X_Recebe_Mensagem

grupo 3 - gerenciamento de memória
X_Aloca_Bloco
X_Devolve_Bloco

grupo 4 - temporização
X_Lança_Temporizador
X_Ler_Temporizador

Na descrição que segue, a sintaxe das primitivas é apresentada juntamente com uma descrição de sua parametrização.

4.2 Primitivas Básicas

Primitiva X_Prioridade (Prioridade):

A primitiva X_Prioridade é usada para alterar a prioridade da tarefa chamadora, com valor do parâmetro Prioridade variando de 1 a 255, e 1 sendo a mais alta prioridade.

Primitiva X_Dormir (Período):

A primitiva X_Dormir coloca a tarefa num estado bloqueado por um tempo dado por Período x 10ms. Enquanto dormir, a tarefa não consome recursos da UCP.

Primitiva X_Espera_Evento (Evento,Dado):

A primitiva X_Espera_Evento coloca a tarefa num estado bloqueado até o acontecimento de um evento. Isso pode ser o estouro de um temporizador ou a chegada de uma mensagem. O Executivo devolve o parâmetro "Evento" com o código do evento. No caso de estouro de temporização "Dado" seria o identificador do temporizador, e, no caso de chegada de mensagen "Dado" seria o identificador da tarefa emissora.

4.3 Primitivas de Comunicação

Primitiva X_Recebe_Mensagem (Tempo,Posição,Tamanho,Emissor,
RecTamanho,Estado):

A tarefa espera por uma mensagem. O tempo de espera é dado pelo parâmetro "Tempo", cada unidade representando 10ms. Um valor de FFFFh significa espera para sempre. As variáveis "Posição" e "Tamanho" descrevem o "buffer" disponível para receber a mensagem.

Na chegada de uma mensagen, o Executivo devolve o parâmetro "Emissor" com o identificador da tarefa emissora. A variável "RecTamanho" é carregada

com o tamanho da mensagem recebida. A variável "Estado" é carregada com o valor zero para recepcão correta ou não-zero em condição de erro.

Primitiva X_Manda_Mensagem (Receptor,Posição,Tamanho,Estado):

Esta primitiva manda uma mensagem a outra tarefa. O parâmetro "Receptor" indica a tarefa a receber a mensagem, e "Posição" e "Tamanho" descrevem o buffer que vai ser transmitido. Qualquer que seja o estado da tarefa receptora, a tarefa emissora não fica bloqueada, as mensagens sendo buferizadas pelo Executivo se for necessário. Depois da chamada, o Executivo devolve um valor em "Estado" zero se a mensagem foi aceita.

6.4 Primitivas de Alocacão de Memória

Primitiva X_Aloca_Bloco (Tamanho,Posição,Estado):

Um bloco igual a "Tamanho" é alocado à tarefa. O Executivo devolve a posição do bloco na variavel "Posição". O valor de "Tamanho" deve ser um múltiplo de 256 octetos. "Estado" contem um valor dependente do sucesso da chamada.

Primitiva X_Devolve_Bloco (Posição,Tamanho):

Com esta primitiva, uma tarefa devolve um bloco de memória ao Executivo. O bloco é colocado na lista de blocos livres mantida pelo Executivo. "Posição" e "Tamanho" definem o bloco, com "Tamanho" sendo um múltiplo de 256 octetos;

6.5 Primitivas de Temporizacão

Primitiva X_Lança_Temporizador (ID,Tempo):

Esta primitiva cria um temporizador com identificador ID. Este temporizador é local a esta tarefa. O temporizador começa a rodar logo depois da criação. O estouro do temporizador é um evento detectado pela primitiva X_Espera_Evento. O parâmetro "Tempo" define o período em múltiplos de 10ms.

Primitiva X_Ler_Temporizador (ID,Tempo):

O Executivo devolve o período restante do temporizador identificado pelo ID na variável "Tempo". A leitura não afeta a contagem.

5. Comunicacão Executivo-Hospedeiro

5.1 Protocolo

O PPC e o hospedeiro se comunicam através de 32k de memória comum. Só um lado pode estar no processo de transmissão por vez. O acesso é controlado por um mecanismo de semáforo.

O semáforo usa o primeiro octeto da memória comum. Se o valor for não-

zero significa que a memória não está sendo usada. Então, através das instrucões LOCK e XCHG do 80186, aquele lado que consegue instalar zero neste primeiro octeto consegue acesso exclusivo à memória comum para transmissão. As instruções LOCK e XCHG evitam conflito.

Depois de processar a mensagem recebida, o receptor coloca um valor não-zero no semaforo, e a memória fica disponível mais uma vez. O processo de mandar uma mensagem é o seguinte:

1: emissor instala zero no semáforo
2: emissor coloca mensagem na memória
3: emissor inicia interrupção ao outro lado
4. receptor processa mensagem
5. receptor instala não-zero no semáforo

Há cinco tipos de mensagem usadas no protocolo PFC/Hospedeiro, que são M_Inicializa, M_Inicia_Tarefa, M_Carrega_Código, M_Inicia_Operação e M_Dados.

5.2 Mensagens

Mensagem M_Inicializa:

Esta mensagem força uma inicialização da PFC, sendo unidirecional do Hospedeiro para o PFC.

Mensagem M_Inicia_Tarefa:

Esta mensagem começa o processo de carregamento de uma tarefa. Dados como parâmetros são o identificador da nova tarefa, a prioridade inicial dela, e os tamanhos de dados, código e pilha. A mensagem é unidirecional do Hospedeiro para o PFC.

Mensagem M_Carrega_Código:

Esta mensagem entrega uma parte do código de uma tarefa ao PFC. Uma bandeira ("flag") indica se esta é a última parte do código. A mensagem é unidirecional do Hospeiro para o PFC.

Mensagem M_Começa_Operação:

Esta mensagem inicia a roteamento das tarefas. Depois desta mensagem, nenhuma tarefa mais pode ser criada no PFC porque a memória não utilizada pelas tarefas criadas será usada para alocação dinâmica. Esta mensagem é tambem unidirecional do Hospedeiro para o PFC.

Mensagem M_Dados:

Esta mensagem bidirecional permite a passagem de dados entre tarefas na placa e processos no hospedeiro. Dados como parâmetros são os identificadores das tarefas/processos emissor e receptor. O Executivo não impõe nenhum formato no campo de dados.

6. CONSIDERAÇÕES FINAIS

O executivo descrito é uma solução alternativa para o gerenciamento de um PFC. Ele é uma solução melhor adaptada que os monitores de carregamento de SO's ou adaptações e sub-conjuntos de SO's. Os monitores são pobres de recursos e as adaptações de SO's são, normalmente, "pesadas" e complexas.

O conjunto mínimo de facilidades oferecidas pelo Executivo endereça as necessidades específicas de implantação dos softwares de rede.

O mecanismo de comunicação entre tarefas oferece o suficiente para uma implantação ISO/OSI, onde as camadas/tarefas dialogam via primitivas de serviço. Neste contexto, troca de mensagens com sinalização do evento de recepção é o necessário.

A temporização é feita com um degrau e uma precisão razoável para protocolos. Isto é conseguido graças à existência de poucas tarefas sem a sobrecarga ("overhead") do SO, o que torna o chaveamento de tarefas eficiente e a recepção de eventos precisa.

O mecanismo de gerência de tarefas usa prioridades. Isto flexibiliza a implantação do pacote rede, permitindo um ajuste fino no desempenho de operação das camadas de protocolo instaladas no PFC.

7. REFERÊNCIAS BIBLIOGRÁFICAS

[1] ISO/DIS 7498: Information Systems - Open System Interconnection - Basic Reference Model, april, 1982.

[2] Joberto S.B. Martins e William Giozza; "Uma Proposição de Arquitetura de Rede Local RM OSI/ISO para supermicros"; 5º Simpósio Brasileiro de Telecomunicações; Campinas; 1987.

[3] TELEBRÁS - CPqD; Documentação Técnica do Projeto PP (Processador Preferêncial); 1987.

[4] General Motors, "Manufacturing Automation Protocols," 1984.