ANÁLISE E PROJETO DE UM SUBSISTEMA DE CO
MUNICAÇÃO PARA UMA REDE DE COMPUTADORES

Jorge Luiz e Silva

Orientador:

Prof. Dr. Jan F. W. Slaets

Dissertação apresentada ao Instituto de Ciências Matemáticas de São Carlos, da Universidade de São Paulo, para obtenção do título de Mestre em Ciências de Computação.

que os frutos des-
te trabalho sejam
paz e felicidade
à Beth, Lia e Lyse.

AGRADECIMENTOS

Ao Prof. Dr. Jan Frans Willem Slaets pela orientação, amizade, disponibilidade e confiança a mim dedicada.

Ao Prof. e amigo Claudio Kirner de forma especial por todo incentivo e apoio na realização deste trabalho.

À Beth, que não medindo esforços, dedicou horas datilografando este trabalho.

À Sonia e Laura, pela amizade e atenção.

Aos meus pais e meus irmãos pelo apoio.

Finalmente a todos, que de forma direta ou indiretamente participaram deste trabalho.

RESUMO

Esta Dissertação relata o trabalho desenvolvido na implementação de um Subsistema de Comunicação dentro de um Projeto de Sistemas Distribuídos, abrangendo mais especificamente o Hardware implementado e o Software básico de suporte à Comunicação e Sincronização entre processos do sistema.

Inicialmente é feito uma análise considerando os aspectos de Arquitetura de Sistemas Distribuídos. Em seguida faz-se um levantamento dos vários Subsistemas de Comunicação enfocando basicamente; Topologia com exemplos tradicionais tipo "Ethernet", "Cmmp", e exemplos mais recentes tipo "Micronet", "HIVE", etc; e Software com referência a protocolos, também tradicionais tipo "BISYNC", "HDLC", X.25, etc.

Finalmente é descrito a implementação do Hardware e do Software desenvolvido até o nível de Primitivas de Comunicação e Sincronização entre Processos.

ABSTRACT

This work describes the development of the hardware and software used for communication and synchronization of processes in a Distributed System.

First a discussion of the classic topologies of Computer Network Architectures is presented, including some implementations such as Ethernet, Cmp, Micronet, HIVE; and software Communication protocols (Bisync, HDLC, X.25 ...).

Finally the developed hardware and software implementation, including Communication Primitives and Synchronization between processes, is presented.

INDICE

1. INTRODUÇÃO	01
2. ARQUITETURA DE REDES DE COMPUTADORES	03
3. CARACTERIZAÇÃO DOS SUBSISTEMAS DE COMUNICAÇÃO	09
3.1. Aspectos Básicos do Subsistema	09
3.2. Topologia	12
3.3. Software	31
4. PROJETO DE SISTEMAS DISTRIBUÍDOS	41
4.1. Considerações Preliminares Sobre o Sistema Desenvolvido	42
4.1.1. Aspectos do Hardware	44
4.1.2. Aspectos do Software	44
5. PROJETO DE UM SUBSISTEMA DE COMUNICAÇÃO	47
5.1. O Hardware do Processador de Comunicação	47
5.2. O Software do Processador de Comunicação	50
5.2.1. O Programa Supervisor	50
5.2.1.1. Protocolo	53
5.2.2. Rotinas de Comunicação pelas Linhas Seriais do Subsistema de Comunicação e Rotinas de Comunicação pelas Linhas Seriais conectadas ao Processador Operador	57
5.3. O Software do Processador Operador	66
5.3.1. Primitivas de Comunicação e Sincronização	66
5.3.1.1. Tipos de Primitivas de Comunicação e Sincronização	67

5.3.1.2. Primitivas de Comunicação e Sincronização da Rede DCE's	70
5.3.1.2.1. Rotinas de Tratamento das Primitivas	73
6. CONCLUSÕES	75
6.1. Sobre a Análise de Arquiteturas de Sistemas Distribuídos	75
6.2. Sobre o Desenvolvimento do Projeto	75
6.3. Sobre os Pontos Relevantes e Contribuição do Trabalho	77
6.4. A Continuidade do Trabalho	78
7. BIBLIOGRAFIA	79
ANEXO 1	
Diagrama de Circuito do Processador de Comunicação ..	85
ANEXO 2	
Configuração dos Pacotes na Comunicação entre Processos	89
ANEXO 3	
Programa Supervisor	95
ANEXO 4	
Procedimento das Primitivas "SEND", "RECEIVE" e SUPERVISOR no P.O.	103
ABREVIACÕES UTILIZADAS	111

1. INTRODUÇÃO

Muito se fala atualmente sobre Processamento Distribuído e Redes de Computadores, assuntos esses que já vem sendo estudados desde a década de 60 quando a tendência de evolução de sistemas estava indo em direção ao desenvolvimento de sistemas de grande porte com vários terminais espalhados.

Nesta época várias técnicas foram implementadas na tentativa de redução do Custo de Comunicação, Modelagem do Comportamento dos Sistemas e tentativas em resolver problemas de software na Comunicação [PARDO,1979].

Com a diminuição do custo de processamento e do custo de comunicação o fim dos anos 60 marcou uma nova era, a saber, Redes de Computadores; que com a tecnologia de Comutação por PACOTES possibilitou a implementação de Redes como por exemplo a ARPANET [ROBERTS e WESSLER,1970], CYCLADES [POVZIN,1973]. Basicamente estas Redes interligaram muitos computadores espalhados por vasta área geográfica de forma tal a oferecer um serviço de comunicação pública de computadores através de serviços telefônicos [KELLY,1978].

Com o avanço tecnológico foram desenvolvidas as chamadas Redes Locais de Computadores que consistem de Computadores interligados em área geográfica pequena, permitindo assim encontrar novos métodos de interconexão de Sistemas com um ganho substancial no custo e na performance da Rede como um todo, a exemplo de [LIU,1978], [CLARK et alii,1978].

No Brasil vários projetos envolvendo Redes e Sistemas Distribuídos vem sendo desenvolvidos, como por exemplo Rede Local LSD-EPUSP [RUGGIERO et alii,1979], MULTIPUC [EICHORST,1979],

Máquina de Arquitetura Distribuída MAD [SHIMIZU,1980], Rede Local NCE-UFRJ [TAKANO et alii,1981], Rede Cobra de Computadores [GADELHA,et alii,1982], Rede local REDPUC [SOARES et alii,1982] Rede local ITAU-TEC [FERRO,1985], Rede local para o DCC-UFMG [LOUREIRO et alii,1985], Rede local SCOPUS [NAVARRO e RIBEIRO, 1985], Rede CEPINNE [CUNHA et alii,1985].

Esta dissertação aborda uma parcela do trabalho realizado dentro de um Projeto de Sistema Distribuído em desenvolvimento; sob responsabilidade de um grupo de pesquisa do DCEs/UFSCar [KIRNER,1984] no qual eu estava inserido. O trabalho refere-se ao desenvolvimento do Subsistema de Comunicação mais especificamente ao Hardware implementado e ao software básico de suporte à comunicação e sincronização entre processos do sistema.

2. ARQUITETURA DE REDES DE COMPUTADORES

Na tentativa de reduzir a complexidade do projeto, as Redes de Computadores foram propostas como uma série de níveis e protocolos de controle [TANENBAUM, 1981], conforme figura (2.1) ; de tal forma que:

(a) Cada nível oferecesse certos serviços ao nível imediatamente superior através das respectivas interfaces;

(b) O nível n de uma máquina se comunicasse apenas com o nível n de outra máquina através de um protocolo do nível n ;

(c) Nenhum dado fosse transmitido diretamente de um nível n de uma máquina para o nível n de outra máquina, exceto no nível mais inferior. O que ocorre é que a informação de um nível n de uma máquina A a ser transmitida para o nível n de uma máquina B deve percorrer os níveis inferiores da máquina A até alcançar o nível onde realmente ocorre a transmissão. Neste nível deverá existir uma conexão física com um subsistema de comunicação ou diretamente com a máquina B. A máquina B por sua vez recebendo a informação, faz com que ela alcance o nível n desejado.

Este requisito caracteriza uma transmissão de informação virtual entre máquinas exceto no nível mais inferior.

figura 2.1 - Níveis, Protocolos e Interfaces

Estes níveis e protocolos é que definem a Arquitetura de uma Rede de Computadores.

Para o projeto destes níveis, algumas características devem ser observadas como: mecanismo de conexão (ex. se as mensagens possuem endereços); regras para transferência de dados (ex. se comunicação "simplex", "half-duplex", "full-duplex") controle de erros na comunicação; ordem das mensagens; controle de "TIMEOUT" na recepção; capacidades nos vários níveis para receber ou não mensagens longas; na existência de múltiplos caminhos entre fonte e

destino deve ser definido um algoritmo de rota para escolha do caminho por onde enviar a mensagem, etc.

A "ISO" (International Standards Organization) [ZIMMERMANN, 1980] propôs um modelo de padronização dos vários protocolos, o que é mostrado na figura (2.2).

figura 2.2.- Arquitetura de Redes Baseado no Modelo "ISO"

NÍVEL FÍSICO é onde efetivamente ocorre a transmissão de bits através do canal, portanto todas as características mecânicas, elétricas e de interfaceamento da rede devem ser observados neste nível.

NÍVEL DE LIGAÇÃO transforma uma linha física em uma linha livre de erros de transmissão para ser usado no nível de rede. Isto é feito agrupando os bits de informação para constituírem o que chamamos "quadros" e esses quadros são transmitidos sequencialmente. O Receptor por sua vez devolve quadros de reconhecimento para garantir que esta linha seja livre de erros uma vez que o nível 1 apenas aceita e transmite cadeia de bits.

NÍVEL DE REDE também chamado de subsistema de comunicação basicamente controla a operação da sub-rede. Ele determina as características básicas das interfaces Hospedeiro e Subsistema de Comunicação e como são direcionadas as mensagens na sub-rede. Existe aqui uma divisão de tarefas entre Hospedeiro e Subsistema de Comunicação para assegurar que todos os Pacotes sejam corretamente recebidos em seus destinos e na própria ordem.

O software deste nível aceita mensagens de uma máquina origem, converte para pacotes e faz com que o pacote seja dirigido para uma máquina destino. A questão é como determinar a rota; o que será feito através de tabelas de distâncias, tabelas de conexão e algoritmos de roteamento.

NÍVEL DE TRANSPORTE, este nível aceita dados do nível SESSÃO, divide-os em unidades menores, passa-os para o nível da REDE e assegura que todas as partes cheguem ao outro lado, liberando o nível SESSÃO das mudanças inevitáveis que venham ocorrer na tecnologia de hardware. Isto porque neste nível a comunicação passa a ser efetuada no nível HOSPEDEIRO - HOSPEDEIRO e não mais Hospedeiro-Subsistema de Comunicação ou vice-versa, mostrando claramente a desvinculação com o hardware envolvido na comunicação nos níveis inferiores.

NÍVEL SESSÃO é a interface entre o usuário e a REDE propriamente dita. É com este nível que o usuário deve negociar para a conexão com um processo sobre outra máquina, e uma vez estabelecido, o nível SESSÃO pode gerenciar o diálogo de maneira ordenada. Para isso o usuário deve providenciar o endereço que ele deseja se conectar e o nível SESSÃO se encarrega de uma série de tarefas para efetuar a conexão.

NÍVEL DE APRESENTAÇÃO, esse nível se presta a oferecer um conjunto de recursos tipo biblioteca de programas aos usuários da REDE, que estará disponível através do Sistema Operacional do Sistema.

Esses programas especificam rotinas do tipo Conversores de códigos "ASCII" para "EBCDIC", conversores de cadeias de códigos "ASCII" para seus respectivos caracteres, etc.

NÍVEL DE APLICAÇÃO, quando dois programas de usuários, sobre diferentes máquinas se comunicam, o nível de APLICAÇÃO trabalha com um conjunto de mensagens permitidas para a comunicação e executa ações sobre a recepção de cada uma delas.

Ocorre aqui todo um processo de transparência para o usuário quanto a distribuição física dos recursos da REDE, quanto a partição das tarefas, afim de se conseguir o máximo de desempenho na utilização dos recursos existentes na REDE como um todo.

Por se tratar de um Modelo de padronização, a referência "ISO" poderá ser adotada nos projetos das várias REDES ao longo dos anos por ser uma proposta bastante clara e objetiva.

Existe muito pouca variação entre os vários projetos existentes de REDES, e a proposta de implementação do modelo "ISO" como mostrado em [TANENBAUM, 1981] para as redes "Systems Networks

Architecture - SNA", "Advanced Research Projects Agency - ARPANET"
"Digital Equipment Corporation's - DECNET", etc; e acredita-se que
as tendências deverão seguir rumo a implementações tipo X.25 pro_
posto pela "CCITT" (Comité Consultante Internationale de
Telegraphique et Telephonique) que definem protocolos para os ní_
veis 1, 2 e 3 do modelo "ISO" mostrando assim uma padronização ca-
da vez maior no que se refere aos Modelos de Implementação de
projetos de REDES e SISTEMAS DISTRIBUÍDOS.

3. CARACTERIZAÇÃO DOS SUBSISTEMAS DE COMUNICAÇÃO

Por volta de 1975, Processamento Distribuído e Redes de Computadores, foi e continua sendo, uma área de pesquisa de grande interesse no que se refere à Arquitetura de Computadores.

Estes sistemas são implementados com estruturas que envolvem desde Dois Processadores Compartilhando Memória até Computadores Independentes se comunicando através de uma Rede Geograficamente Distribuída.

O principal problema no início das pesquisas nesta área era o grande número de publicações descrevendo os vários projetos e a dificuldade em comparar resultados dessas pesquisas. Não existia um contexto comum na qual tomar posições, não se debatiam um determinado conjunto de projetos e não existia uma nomenclatura comum.

Procurar-se-á aqui, compatibilizar as várias caracterizações existentes e adotar um enfoque detalhado para deixar claro os aspectos de subsistemas de comunicação, conforme proposto por [ANDERSON e JENSEN, 1975].

3.1. Aspectos Básicos do Subsistema

Em uma Rede de Computadores podemos identificar três unidades básicas, ou seja, Unidade de Hardware, Caminho e Elemento de Comutação.

As unidades de Hardware são os HOSPEDEIRO também referenciados como Processadores Operadores, (POs) capazes de executar processos. Basicamente são os Computadores que fazem parte da Rede.

Definimos também um mecanismo de interconexão tal que: um PO pode se comunicar com qualquer outro PO e essa comunicação se dará através de troca de mensagem. Nesta estrutura de interconexão distinguimos as outras duas unidades básicas, ou seja, caminho e elemento de comutação. O caminho é o meio pelo qual a mensagem é transferida entre os elementos do sistema. Normalmente são fios trançados comuns, cabo coaxial, fibra ótica, microondas, etc. A transmissão de uma mensagem sobre o caminho não deve alterar o conteúdo da mensagem.

Um elemento de comutação, também chamado Processador de Comunicação (PC), é uma entidade inteligente entre o Emissor e o Receptor de uma mensagem. Esses elementos de comutação são "inteligentes" pois podem afetar o destino de uma mensagem de duas formas: mudando seu endereço destino ou rotulando a mensagem para um dos diferentes caminhos existentes ou por ambas as formas.

A figura (3.1.1.) mostra estas unidades básicas dentro de uma Rede de Computadores.

figura 3.1.1 - Estrutura de uma Rede

Ainda com relação a troca de mensagem, ela pode se dar de duas formas diferentes, ou seja: comutação de mensagem ou comutação de pacotes.

Nos sistemas em que se utiliza comutação de mensagens, as mensagens são enviadas no seu todo ao longo de um caminho fixo de transmissão da fonte para o destino. O que ocorre é que no estabelecimento deste caminho, poderão existir PCs intermediários por onde passarão as mensagens, implicando no armazenamento e envio destas mensagens utilizando os discos dos POs correspondentes, devido ao tamanho das mensagens.

A comutação por pacotes, particionam uma mensagem em vários pacotes. A esses pacotes são acrescentados alguns bits como controle, endereços, etc; de forma tal a constituírem um "quadro", e esses quadros são enviados através do Subsistema de Comunicação.

A diferença básica entre este sistema e o anterior é que os quadros ao passar por PCs intermediários durante uma transmissão poderão ser armazenados em Memória Principal nos próprios PCs por serem de menor tamanho, reduzindo assim o tempo de atraso na transmissão de mensagens.

Uma entidade que não foi citada, mas que tem função fundamental nesta estrutura é a Interface entre os Processadores Operadores e o Subsistema de Comunicação. Esta interface pode ser projetada especificamente para cada equipamento com características específicas para a forma de comunicação ou simplesmente conectar os POs aos subsistemas através de portas de comunicação serial que a maioria dos equipamentos já dispõem.

3.2. Topologia

A partir de levantamentos feitos em uma série de projetos envolvendo interconexão de computadores, foi possível inserir, essas várias formas em uma estrutura do tipo árvore, que chamamos taxonomia, o que é mostrado na figura (3.2.1).

Também pode-se definir duas características básicas a serem consideradas dentre os vários projetos ou seja:

(a) MODULARIDADE onde a inclusão de um elemento ao sistema implica somente no custo do elemento, e se essa inclusão pode ser facilmente posicionada para produzir um aumento na Performance do sistema.

(b) TOLERÂNCIA A FALHA quando na ocorrência de falha de algum elemento do sistema, não implica em queda de todo o sistema mas apenas do elemento, e que através de uma reconfiguração o sistema continue funcionando mesmo sem todo seu potencial.

Na estrutura da figura (3.2.1) para se definir a arquitetura do subsistema de comunicação a primeira coisa a fazer é escolher a estratégia de transferência de mensagem, ou seja, transmissão DIRETA ou INDIRETA.

A diferença básica entre essas duas formas é que, no caso da transmissão INDIRETA toda mensagem enviada de um P0 origem, passa por um elemento de comutação PC o qual pode transformar endereços ou direcionar mensagens para uma série de caminhos alternativos, até alcançar o P0 destino. A transmissão INDIRETA caracteriza-se por uma distribuição inteligente de transmissão de mensagem porém requer uma complexidade maior na capacidade de comunicação e também um aumento na tolerância a falha de

figura 3.2.1 - Taxonomia

um sistema.

O método de controle de transferência, na estrutura de taxonomia, logicamente se refere ao caso em que a estratégia de transmissão de mensagem foi INDIRETA, podendo ser um controle CENTRALIZADO ou DESCENTRALIZADO.

No controle CENTRALIZADO, deverá ser usado um único elemento de comutação, e todas as decisões com relação ao direcionamento de mensagens serão indicado por ele. O que não ocorre no controle DESCENTRALIZADO uma vez que o controle de transferência das mensagens é efetuado por dois ou mais elementos de comutação.

O inconveniente na forma de controle CENTRALIZADO é a vulnerabilidade do sistema pela ocorrência de uma falha no elemento de comutação Central considerado uma catástrofe para o sistema. Diferentemente, o do controle descentralizado onde a falha em um dos elementos de comutação não compromete o Subsistema como um todo, necessita somente de uma reconfiguração do próprio subsistema.

Definindo os aspectos com relação a transferência de mensagem resta agora caracterizar a estrutura do caminho pelo qual fluirão as mensagens.

Na estrutura de taxonomia verifica-se a existência de caminhos com dois tipos de estruturas ou seja DEDICADO E COMPARTILHADO. A escolha de um dos dois métodos praticamente define toda a organização do Hardware do subsistema de comunicação, uma vez que o caminho é a própria interconexão física entre elementos de comutação, Processadores Operadores, etc. Existirá uma estrutura com caminho DEDICADO quando somente dois elementos estiverem conectados entre si através de um único caminho, (típico das comunicações seriais assíncronas como por exemplo o RS232,

ligações paralelas, etc). No caso COMPARTILHADO é quando se puder conectar mais que dois elementos entre si através de um caminho (ex. barramento paralelo, Memória compartilhada, etc).

Finalmente na estrutura da árvore de taxonomia verificamos os vários tipos de interconexão possíveis para o subsistema de comunicação, a qual passamos a considerar analisando alguns casos.

(a) "BUS" GLOBAL esta estrutura compartilha o caminho com controle de transferência DIRETO. A figura (3.2.2) caracteriza esta organização.

figura 3.2.2 - "Bus" Global

Na categoria "Bus" Global, P.O.s são interligados por um mesmo cabo e se comunicam através de um canal comum. Quando um P.O. origem deseja transmitir mensagem para um P.O. destino, ele o faz independente de haver outro P.O. transmitindo. Nota-se nesta estrutura o problema relacionado com o conflito das mensagens durante o processo de comunicação.

Alguns métodos foram implementados para resolver o pro-

blema citado acima, é o caso do Sistema "ALHOA" [ABRAMSON, 1973] em que as mensagens colididas são perdidas e como o PO destino não confirma o recebimento da mensagem ela é retransmitida pelo PO origem.

Um método alternativo é o "CSMA/CD" (Carrier Sensing Multiple Access/Colision Detect) [TOBAGI, 1980] onde a estação que deseja transmitir, verifica antes se há alguma mensagem fluindo pelo bus, em caso positivo aguarda até que fique disponível e então transmite. Ainda assim existe a possibilidade de haver colisão, o que pode ser resolvido com o mecanismo de detecção de colisão "CD" tal que no caso de uma colisão a transmissão é imediatamente abortada e é feita uma nova tentativa de transmissão de pois de intervalo de tempo. O intervalo de tempo é determinado segundo um algoritmo que procura evitar nova colisão gerando tempos diferentes para cada colisão.

Uma característica fundamental nestes métodos é que eles usam o Sistema "TDMA" (Time Division Multiple Access) proposto por [WILLARD, 1974].

Outra forma de solucionar este problema foi transmitir as mensagens pelo cabo através de RFs (Rádio Frequência) caracterizando o método de DFM (Divisão de Frequência Multiplexada). Neste método verifica-se a existência de vários canais com frequências diferentes podendo ser usados simultaneamente, o que implica em vários POs se comunicando ao mesmo tempo desde que cada canal em sua respectiva frequência.

Este sistema é Modular e Tolerante a falha com relação aos POs mas não com relação ao sistema de interconexão, pois uma falha na estrutura do caminho comprometerá todo o sistema.

Um exemplo bastante tradicional de subsistema de comuni

cação que utiliza "BUS"GLOBAL é a rede "ETHERNET" de computadores [METCALFE e BOGGS,1976].

(b) MEMÓRIA CENTRAL COMPARTILHADA como o próprio nome já diz, são sistemas de computadores com Memórias próprias e software próprios que partilham uma Memória Central apenas para troca de informações e não para execução de programas. A figura (3.2.3) mostra esta organização.

figura 3.2.3 - Memória Central Compartilhada

Da mesma forma que na estrutura "BUS"GLOBAL, este sistema é Modular e Tolerante a falha com relação aos P.Os mas não com relação a Memória Compartilhada.

O Sistema "Crossbar" de Comutação construído pela "Carnegie-Mellon University A Carnegie Mellon multi-mini-processor C.mmp" [WULF e BELL,1972], caracteriza exatamente esta estrutura de memo

ria central compartilhada.

(c) "LOOP" consiste de P0s interligados com dois P0s vizinhos através de um canal de comunicação dedicado unidirecional com controle de transferência DIRETO.

Quanto às características de Modularidade e Tolerância a falha é fácil ver que o sistema é Modular mas não Tolerante a falha, pois a falha em um dos P0s ocasionara a queda de todo o sistema.

Esta organização é mostrada na figura (3.2.4)

figura 3.2.4 - "LOOP"

(d) ANEL é outra estrutura topologicamente parecida com o "LOOP". Muitos autores adotam a notação "LOOP" para esta estrutura mas que referenciaremos por ANEL por se tratar de es-

estrutura diferente . É o caso de [WEST,1972], [HAFNER et alii,1974].

No ANEL podemos destacar três diferentes mecanismos de transmissão de mensagem, conforme figura (3.2.5).

figura 3.2.5 - Tipos de Anel

Circulação de Pacote Vazio

Alguns subsistemas de Comunicação possuem um mecanismo de transmissão de mensagem cuja característica é enviar blocos de dados chamados pacotes, um conjunto destes pacotes constitui uma mensagem.

Na metodologia PACOTE VAZIO, pacotes de tamanho fixo circulam no ANEL. Um bit no pacote indica se ele está cheio ou vazio. Ao detectar um pacote vazio, um PO que deseja transmitir, troca este bit para cheio, coloca a mensagem no pacote, seu endereço, o endereço destino e o pacote circula novamente no ANEL até alcançar o destino, quando então é novamente esvaziado.

A desvantagem neste método é que sendo os pacotes de tamanho fixo, uma mensagem muito grande terá de ser particionado em vários pacotes diferentes.

Passagem de "Token"

O "token" é uma sequência de bits pré-definidos que circulam no ANEL. Um PO que deseja transmitir uma mensagem, aguarda a passagem do "token" quando então ele o retira, coloca sua mensagem com os respectivos endereço fonte e destino e em seguida o "token". O PO destino identificando o "token" com seu endereço, retira a mensagem e devolve o "token" ao ANEL.

Nota-se aqui que não existe um tamanho fixo definido para a mensagem, porém o controle de transmissão e recepção desta mensagem é mais complexo que a metodologia anterior.

Inserção de atraso "Registro"

Nesse método a mensagem a ser transmitida é colocada em

um registro de deslocamento. A partir daí os bits da mensagem são transmitidos um a um sendo deslocados através dos registros. Vale observar que mensagens sô podem ser inseridos no ANEL quando não há mensagem passando, ou entre duas mensagens adjacentes.

Nas estruturas do tipo ANEL as características mudam com relação ao "LOOP" pois geralmente os P0s estão conectados ao ANEL através de uma interface, portanto o Sistema é Modular e Tolerante a falha com relação aos P0s mas não com relação ao Sistema de Interconexão pois uma falha em uma interface é fatal para todo o sistema. Geralmente é feito um esquema de reconfiguração por Hardware na própria interface tal que na ocorrência de uma falha a interface deixa de atuar no "bus" e libera o caminho para que as mensagens continuem circulando no ANEL. A única perda no sistema é do P0 que estava conectado àquela interface.

(e) ESTRELA conforme figura (3.2.6) existe um elemento central de comutação que controla toda a transmissão de mensagem entre P0s, caracterizando um mecanismo de transferência de mensagem INDIRETA.

figura 3.2.6 - Estrela

No mecanismo de transmissão deste sistema quando um $P0i$ deseja se comunicar com um $P0j$, o elemento central de comutação providencia a interconexão entre os dois $P0s$. Caso exista um $P0k$ que deseja se comunicar com um dos dois $P0i$ ou $P0j$, enquanto estes estão se comunicando, sua requisição é bloqueada até que desfeita a interconexão $P0i$ e $P0j$.

Quando um $P0i$ está se comunicando com um $P0j$, é possível que um $P0k$ se comunique com um $P0l$ sem que haja bloqueio nas requisições para a comunicação.

Este Sistema não é Modular e nem Tolerante a falha com relação ao Processador Central. Também não é Modular porém Tolerante a falha com relação aos $P0s$.

Por último vamos analisar a estrutura do tipo Rede IRREGULAR, uma vez que os diferentes mecanismos de interconexão na árvore de taxonomia são variações que não necessitam maiores detalhes.

Verificando na estrutura de taxonomia, Redes Irregulares são redes contendo um subsistema de comunicação com mecanismo de transmissão INDIRETO, através de caminhos DEDICADOS, como mostra dos na figura (3.2.7).

figura 3.2.7 - Rede Irregular

Cada caminho desse subsistema pode operar sobre meios diferentes de comunicação, com diferentes taxas e diferentes protocolos de controle. Além disso as mensagens são enviadas através do subsistema pelos Processadores de Comunicação. Pela irregularidade tanto dos caminhos, como dos protocolos e da própria estrutura de interconexão, o sistema é Modular e Tolerante a falha, o que pode ser considerado com um projeto altamente interessante pelas características que apresenta.

A taxonomia portanto se torna um recurso ao projetista de Redes para que ele possa definir com maior facilidade qual tipo de estrutura deve ser escolhida, em função da Modularidade, Tolerância a falha e dos serviços que a rede deve oferecer aos usuários do Sistema como um todo.

Na análise da estrutura de taxonomia, referenciamos vários projetos relacionados com os vários tipos de estruturas.

Passaremos então a analisar, com pouco mais detalhes, alguns projetos que utilizam Subsistema de Comunicação, mais especificamente Processadores de Comunicação, a fim de poder gerar um espectro maior para a definição do projeto de um Subsistema de Comunicação, objeto desta dissertação.

Existem poucos trabalhos disponíveis na literatura especializada que referenciam com detalhes a estrutura dos Processadores de Comunicação.

Dentre esses pode-se citar um sistema Multicomputador proposto por [AGRAWAL e ALEXANDER, 1984] denominado de "HIVE" (Heterogeneous, Interconnected, Versatile and Expandable), que é um sistema de computador multiprocessador com um total de 24 Processadores de Comunicação interligados conforme figura (3.2.8), acoplados a um Processador Operador para desenvolvimento e compilação de pro-

gramas.

figura 3.2.8 - Estrutura de Interconexão "HIVE"

O sistema opera em modo MIMD (Múltiplas Instruções e Múltiplos Dados) para aplicação com sinais digitais e processamento de imagem.

A estrutura interna de um Processador de Comunicação no Sistema de Multicomputador "HIVE" é mostrada na figura (3.2.9).

figura 3.2.9 - Arquitetura do Processador de Comunicação - HIVE

Na estrutura da figura (3.2.9), para qualquer requisição de Entrada/Saída, o controlador aloca um "buffer" disponível para a comunicação, faz a transferência dos dados e através da análise do cabeçalho do pacote, inicia as ações apropriadas, como ativar o canal "DMA" (Direct Memory Access), por exemplo.

Um outro trabalho que pode ser citado refere-se ao projeto de uma Rede de Computadores Reconfigurável "MICRONET" proposto por [WITTIE, e TILBORG, 1980], com o objetivo de testes para equipamentos na área de Arquitetura de Redes com Sistema Operacional Distribuído.

A "MICRONET" é uma rede reconfigurável de Módulos de Computadores autônomos, cada um compartilhando dois "buses" de comuni-

cação, de um conjunto de "BUSES". A figura (3.2.10) mostra a organização da Rede "MICRONET".

figura 3.2.10 - Sistema de Interconexão da Rede "MICRONET"

Cada nó consiste de um Computador Hospedeiro e um Processador de Comunicação, conforme figura (3.2.11).

figura 3.2.11 - Esquema de um Nó da Rede
"MICRONET"

Existe um "DMA" através do qual a memória do Processador Hospedeiro e do Processador de Comunicação estão interligados.

O Processador de Comunicação possui um microprocessador Z80 com 48k "bytes" de "RAM" (Random Access Memory), 8k "bytes" de "PROM" (Programable Read Only Memory) que contém um conjunto de rotinas pré-definidas e uma interface serial.

A este Processador de Comunicação está acoplado um microcontrolador 8X300 da "Signetics" controlado por rotinas em uma

"ROM" (4k palavras de 16 bits) que providencia 4 canais "DMA" ligados: um ao Hospedeiro, dois para outros Nós através do "bus" de comunicação, e um para o Processador de Comunicação. O 8X300 implementa o "CSMA" no "bus".

Um trabalho bastante recente é o projeto de um Sistema Suporte para desenvolvimento de Sistemas Distribuídos . baseado num Barramento Paralelo Centralizado proposto por [KIRNER e MARQUES,1985]. Este projeto é composto por vários Processadores de Comunicação interligados por um barramento paralelo centralizado (backplane) com uma taxa de transmissão da ordem de 1 a 10 Megapacotes/s, correspondendo a um barramento serial a taxa de 40 a 400 Mbit/s. O objetivo do projeto é gerar a infraestutura necessária ao estudo de primitivas de comunicação e sincronização para sistemas de aplicação em tempo real e ao desenvolvimento de Sistemas Operacionais Distribuídos.

A arquitetura do Sistema é mostrado na figura (3.2.12).

O Processador de Comunicação é formado por três módulos: Módulo de Interface com o barramento; Módulo de Interface com o PO e Módulo de Controle das interfaces conforme figura (3.2.13).

O módulo de Interface com o barramento se encarrega em determinar os acessos ao barramento através de circuitos de determinação de prioridades, além de sincronizar as disputas e liberações do barramento; contando para essa tarefa de circuitos sincronizadores e "buffers" de recepção e transmissão.

O módulo de interface com o Processador Operador consiste de uma interface serial padrão de Entrada/Saída do tipo (8251, "SI0", etc). Quando não se utiliza Processadores Operadores, e mediante a expansão do Processador de Comunicação, essas

figura 3.2.12 - Arquitetura do Sistema Barramento Paralelo Centralizado

figura 3.2.13 - Configuração do Processador de Comunicação

linhas poderão estar ligadas a periféricos, sensores, atuadores e outros elementos.

O módulo de controle das interfaces consiste de circuitos com microprocessadores ou de Hardware equivalente. Sua função se resume na coordenação dos trabalhos das interfaces, e na transferência de informação entre elas.

No primeiro exemplo "HIVE" podemos verificar a existência de seis "buffers" no Processador de Comunicação que possibilita a comunicação independente envolvendo até 12 Processadores de Comunicação limitados a uma taxa máxima permitida para comunicação serial, mas que no conjunto demonstra uma alta taxa de informações circulando no Sistema com um todo. Além disso o projeto de Hardware do Processador de Comunicação é mais simples por se tratar de um microprocessador e interface seriais para a comunicação.

Para o segundo exemplo "MICRONET" a utilização do microprocessador Z80 consiste em mais um projeto utilizando microprocessador.

No terceiro exemplo, Sistema com Barramento Paralelo Centralizado, pode-se notar que apenas dois processadores de Comunicação poderão se comunicar em um determinado tempo, devido a existência de apenas dois "buffers" em cada PC, ou seja, "buffer" de RECEPÇÃO e "buffer" de EMISSÃO. Como o tempo de preenchimento destes "buffers" é bem maior que o tempo de transferência de informação pelo barramento é fácil verificar que vários PCs poderão se comunicar sem que haja monopólio no barramento de algum PC em particular. Isso resulta em alta taxa de transmissão mas também em grande complexidade na implementação do Hardware do PC.

No projeto desta dissertação optou-se pela implementa-

ção de um subsistema parecido com o proposto por [AGRAWAL e ALEXANDER,1984], de tal forma que o Subsistema de Comunicação possuisse apenas alguns Processadores de Comunicação baseado em Microprocessador com "buffers" para a manipulação de informações, características estas mais próximas da proposta inicial por [KIRNER,1984].

3.3. Software

Na análise das topologias, suas características e exemplos, verificamos os aspectos relacionados com o Hardware de um Subsistema de Comunicação que estão diretamente ligados aos níveis 1, 2 e 3 do Modelo "ISO", faltando apenas caracterizar os protocolos de Comunicação para que se tenha definido também o Software destes níveis e em seguida discutiremos o Software dos níveis superiores.

Algumas firmas vêm definindo protocolos padrões para as comunicações, é o caso dos protocolos "BISYNC" (Binary Synchronous Communication), "HDLC" (High Level Data Link Control), X.25 da "CCITT" já referenciado, etc; cada um com características diferentes [TANENBAUM,1981].

Os primeiros protocolos a serem desenvolvidos foram os protocolos orientados por caracter "palavras". É o caso do protocolo "BISYNC" desenvolvidos pela "IBM" (International Business Machine) que utiliza os tradicionais códigos "ASCII", "EBCDIC", etc.

O formato da mensagem "BISYNC" é mostrado a seguir

SYN	SYN	SOH	CABEÇALHO	STX	DADO	ETB ou ETX	CHECKSUM
-----	-----	-----	-----------	-----	------	------------------	----------

SYN - "SYNchronize"

SOH - "Start of Header"

STX - "Start of Text"

ETB - "End of Transmission Block"

ETX - "End of Text"

ETB usado para terminar um bloco quando existe mais que um bloco a transmitir.

ETX usado para terminar o último bloco.

SYN são bits de sincronismo.

Esses protocolos são bastantes utilizados em 'Pooling' de terminais remotos, linhas operando em "half-duplex", linhas multipontos ou ponto-a-ponto.

A partir dos anos 70, com o avanço das Redes de Computadores, começaram a surgir os protocolos orientados por bit, isto é, não se baseavam em caracteres, pelo inconveniente de se ter que trabalhar com códigos pré-definidos como o "ASCII", "EBCDIC", que possuem um número limitado de bits para a representação de informações (6 a 8 bits), e que em se tratando de comunicação entre Processadores implicaria em uma complexidade maior no tratamento da informação e consequentemente atraso na comunicação. O protocolo 'HDLC' da 'SNA' da 'IBM', é um exemplo desse protocolo.

As características básicas deste protocolo e em geral dos protocolos orientados por bits são a existência de QUADROS de dados que contêm um número arbitrário de bits. O tamanho dos

QUADROS não serão um múltiplo integral de qualquer tamanho específico de caracter. Cada QUADRO inicia e termina com um padrão de bits, ou seja, 01111110.

Esta última característica implica em um cuidado especial que é como transmitir um dado exatamente igual ao padrão de bits. Para isso foi criado o esquema de inserção de bits 0's tal que a cada 5 bits 1's é inserido um 0 para a transmissão. Consequentemente na recepção os bits 0's são retirados para se manter a confiabilidade dos dados transmitidos.

Ainda com relação a quantidade de bits, a existência de sete a quatorze 1's consecutivos na linha implica em uma sequência de aborto que é utilizado para o cancelamento da transmissão de um quadro. Uma linha ociosa implica em quinze ou mais 1's consecutivos que é utilizado para assinalar que a linha está desocupada.

Todo protocolo orientado por bit utiliza a estrutura de QUADRO abaixo:

01111110	END	CONTROLE	DADOS ≥ 0	CHECKSUM	01111110
----------	-----	----------	-----------	----------	----------

01111110 - "Flag" de início e fim de 1 QUADRO

END - Usado para identificar o destino do QUADRO

CONTROLE - Usado para o controle do fluxo de informação no sistema

DADO - Informações gerais

"CHECKSUM" - Variação pequena do Código de Redundância Cíclica para verificação de possíveis erros de transmissão.

Esse protocolo fornece ao nível 3 do Modelo "ISO" um pacote de informação livre de erros de comunicação.

O software do nível de Rede, como visto anteriormente, determina as características básicas das interfaces P0s e PCs e como são direcionadas as mensagens no Subsistema de Comunicação.

O direcionamento das mensagens será feito através de algoritmos de determinação de rota utilizando tabelas de conexão, tabelas de distâncias, etc; com um protocolo de comunicação que pode ser uma versão simplificada do protocolo "HDLC", ou seja:

CONTROLE	END	DADO $\geq$ 0	CHECKSUM
----------	-----	---------------	----------

e também com características parecidas, isto é, existência de QUADROS de informação do tipo: Informação REJEITADA, Informação ACEITA, quadro de INFORMAÇÃO, etc.

Esses protocolos podem ser facilmente implementados no subsistema de comunicação, mas quando se trata de um processador Operador a ser conectado ao subsistema de comunicação, torna-se difícil definir protocolos para esta interconexão em função das várias máquinas existentes com diferentes características e também pela diversificação das várias Redes desenvolvidas em países diferentes. Desta forma a "CCITT", como já referenciada, propôs os protocolos de acesso padrão X.25, que englobam os níveis físicos, de ligação e da rede para a conexão P0 e PC.

São definidos nesta padronização alguns nomes como: "DTE" (Data Terminal Equipment) equivalente aos Processadores Operadores; "DCE" (Data Circuit-Terminating Equipment) equivalentes às

través da interface PO-PC(DCE).

Basicamente no protocolo X.21 existem 8 linhas seriais como descrito na figura (3.3.2).

figura 3.3.2 - Linhas de sinais usadas no Protocolo X.21

O "DTE" usa os sinais T e C para transmitir dados e controle. O "DCE" usa da mesma forma os sinais R e I, ou seja, para dado e controle. A linha S contém uma cadeia de sinais emitidos pelo "DCE" para informar o "DTE" quando inicia e termina cada intervalo de informações. A linha opcional B é utilizada para agrupar os bits em quadros de 8 bits. Se isto ocorrer o "DTE" deve posicionar os caracteres nos limites de um Quadro. Caso contrário, ambos "DTE" e "DCE" devem iniciar toda sequência de controle com ao menos dois caracteres "SYN".

No Processador de Comunicação, supor que "DTE" deseje posicionar uma chamada. Ele o faz tornando $T=0$ e $C=0$. Se "DCE" está pronto para aceitar uma chamada, ele inicia transmitindo o "ASCII" "+" em R, indicando que pode começar o diálogo. O "DTE" sentindo que pode começar a enviar o endereço do "DTE" que deseja se comunicar, através de uma série de caracteres "ASCII" usando para isso a linha T e transmitindo 1 bit por vez.

Neste ponto o "DCE" envia o que são chamados sinais de progresso de chamada, para informar ao "DTE" sobre o resultado da chamada. Estes sinais consistem de um número com dois dígitos, tal que o primeiro deles, dá a classe geral dos resultados, e o segundo os detalhes. A classe geral basicamente é composta pelas informações:

- colocar chamada
- tentar novamente
- falha na chamada e provavelmente falhará novamente a próxima vez
- pequeno período de congestão na rede.

Se a chamada pode ser colocada, o "DCE" fixa a linha 1 para o nível 0 para indicar que a transferência pode iniciar. Neste ponto foi estabelecido uma conexão digital "full-duplex" e um ou outro "DTE" pode enviar informações.

Para finalizar a comunicação um dos dois "DTE's" pode fixar C em 1 e encerrar o envio de dados, apesar que o receptor deve continuar a receber dados até que o "DTE" transmissor tenha finalizado.

O nível de quadro e o nível de pacote são os mesmos níveis de ligação e nível de rede respectivamente, referenciado no Modelo "ISO" tal que no nível de quadro seja utilizado o protocolo

lo "HDLC" e no nível de pacote seja garantido a existência de quadros transmitidos ou recebidos, livre de erros, a serem utilizados no Processador Operador.

O software dos níveis superiores não definidos pelo protocolo X.25 consistem na implementação de Primitivas, nome que se dá as tarefas básicas a serem oferecidas por um S.O.D. (Sistema Operacional Distribuído) que completaria o que denominamos núcleo do Sistema conforme figura (3.3.3), restando finalmente a implementação de programas aplicativos que utilizam as interligações existentes como, por exemplo, um sistema servidor de periféricos, caixa postal eletrônica, etc.

figura 3.3.3 - Núcleo do Sistema

Procuramos dar uma visão básica dos aspectos de Software e Hardware relacionados com uma Rede de Computado-

res, mais especificamente com o Subsistema de Comunicação.

Passaremos então a descrição do projeto, tema desta dis
sertação.

4. PROJETO DE SISTEMAS DISTRIBUÍDOS

A evolução natural do uso de recursos computacionais tem sofrido a influência de uma série de fatores. De um lado constata-se a abundância de microprocessadores, microcomputadores, e máquinas de pequeno porte a um custo bem acessível, e de outro percebe-se que cada vez mais se torna viável a utilização de recursos compartilhados e maior imunidade a queda total do sistema. Nesse sentido, a interligação de micros formando redes ou multiprocessadores tem aumentado consideravelmente, exigindo o conhecimento das técnicas utilizadas na comunicação entre máquinas em todos os seus níveis.

Visando dominar a técnica de desenvolvimento de sistemas distribuídos, tanto em hardware quanto em software, um grupo de pesquisa do Departamento de Computação e Estatística da UFSCar resolveu investir na elaboração de Suporte para Desenvolvimento de Sistemas Distribuídos, correspondendo a um laboratório de estudo em sistemas distribuídos. Esse laboratório, cuja implantação por si só já tem função de gerar conhecimento e aumentar a motivação do grupo para a pesquisa de sistemas distribuídos, tem como objetivo a exploração de algumas classes de multiprocessadores e redes. Dentre as atividades previstas para o uso do laboratório pode-se citar: teste de funcionamento do hardware de sistemas distribuídos, desenvolvimento de software básico (Núcleo de sistemas distribuídos, sistemas operacionais, linguagens para programação concorrente, estruturas de banco de dados distribuídos, etc) e de software de aplicação, verificação de aspectos de desempenho, estudo de tolerância a falha, etc.

Tendo em vista os objetivos delineados e a disponibilidade de componentes a um custo não proibitivo, optou-se pela escolha de um suporte para desenvolvimento de sistemas distribuídos com uma estrutura que, do ponto de vista acadêmico, possibilitasse um alto grau de especulação científica. Assim usando a proposta de [BRAYER e LAFLEUR, 1982] como referência, passou-se a desenvolver um sistema baseado no uso de microprocessadores, que daria maior flexibilidade de projeto, com uma arquitetura de rede irregular com conexão ponto-a-ponto.

4.1. Considerações Preliminares sobre o Sistema Desenvolvido

A arquitetura de rede irregular com conexão ponto-a-ponto foi escolhida porque, além de permitir a exploração de muitos aspectos de hardware e software, poderia utilizar microprocessadores em sua implementação, apresentaria funcionamento com visualização sugestiva, e possibilitaria o estudo do roteamento de informações e dos aspectos de reconfiguração de sistemas. Essa arquitetura, utilizando processadores operadores e processadores de comunicação pode ser vista, em configuração genérica, na figura (4.1.1).

figura 4.4.1 - Configuração Genérica do Sistema Distribuído

As características principais do sistema são:

- subsistema de comunicação implementado por conexão ponto-a-ponto em topologia irregular;
- processador de comunicação implementado com microprocessador, conectável a um processador operador através de linha serial (RS-232), e três processadores de comunicação também através de linha serial (RS-232);
- transferência de informação por comutação de pacotes com a técnica armazena e envia;

- processador operador podendo ser qualquer máquina monousuário ou multiusuário.

4.1.1. Aspectos do Hardware

O hardware consiste na implementação do subsistema de comunicação e em sua conexão física com os processadores operadores. O subsistema de comunicação resume-se na interligação de vários processadores de comunicação através de linhas seriais que estarão conectados aos processadores operadores também através de linha seriais.

A descrição detalhada do hardware do subsistema de comunicação será vista no capítulo 5.

4.1.2. Aspectos do Software

O software de base do sistema distribuído consiste do software de cada Processador de Comunicação, devidamente complementado com a parcela de software do Processador Operador, necessária para a comunicação PO-PC.

O software do Processador de Comunicação envolve o programa Supervisor, rotinas de Comunicação pelas linhas seriais do subsistema de comunicação, rotinas de comunicação pela linha serial conectada ao processador Operador, e rotinas de suporte para a implementação das primitivas do sistema e do protocolo de comunicação.

O software do Processador Operador consiste do software próprio da máquina, rotinas de comunicação pelas linhas seriais conectadas ao Processador de Comunicação e rotinas referentes

as primitivas de comunicação e sincronização do sistema distribuído.

O detalhamento do software do projeto também será visto no capítulo 5.

5. PROJETO DE UM SUBSISTEMA DE COMUNICAÇÃO

A topologia proposta no Sistema Distribuído do DCEs, em Rede Irregular com conexão ponto-a-ponto torna necessário que o Processador de Comunicação seja capaz de receber e transmitir mensagens aos POs e aos PCs interligados a ele, e atuar sobre essas mensagens de tal forma a viabilizar a comunicação de maneira confiável e com um potencial de processamento significativo.

Isto implica em um PC com capacidade de armazenamento e Processamento de informações, além de ser capaz de manipular várias interfaces para formarem a estrutura de interconexão.

Passaremos a detalhar o projeto do Hardware e Software desenvolvidos neste subsistema.

5.1. O Hardware do Processador de Comunicação

Por se tratar da implementação de um processador, o primeiro passo foi definir a Unidade Central de Processamento e em seguida os elementos de interface para a comunicação conforme especificado na figura (5.1.1).

A UCP escolhida foi a "Z80-ZILOG", pelo fato de existir no mercado, ser de baixo custo e ter documentação disponível de todo seu software [ZILOG,1981a].

Entre os elementos periféricos disponíveis no mercado foi escolhido o "SIO" (Serial Input Output) da "ZILOG", que permite transmissões a uma taxa máxima de 800k bits/s, trabalhando em ambiente de Interrupção Vetorizada [ZILOG,1981b].

figura 5.1.1 - Arquitetura do Processador de Comunicação

Os dois circuitos "CTC-Z80" (Counter Timer Circuit) foram utilizados com o objetivo de alimentar os "clocks", geradores de tempo para os "SIOs", e como elementos geradores de "TIMEOUT" no sistema, característica essa fundamental para funcionamento de um Subsistema de Comunicação [ZILOG,1981c].

Foram utilizados 8k de 'EPROM' com os programas que gerenciam o Processador de Comunicação e 8k de "RAM" para manipulação de "buffers" e filas na comunicação.

Antes de definir as características gerais do sistema é interessante analisar uma característica importante que é a taxa de transmissão possível para o atendimento de quatro canais distintos no Processador de Comunicação.

A recepção de um "byte" em um canal é executada através de um processo de interrupção vetorizada. Nesse processo o dispositivo "SIO", liberado pela "UCP", posiciona um vetor de interrupção na via principal do Processador de Comunicação que será utilizado em concatenação ao registro I do microprocessador Z80 para se definir o endereço da rotina de tratamento da interrupção em questão.

Em se tratando de um PC com 4MHz de "clock" serão necessários 4,5 μ s para o posicionamento do vetor e salto para a rotina de tratamento de interrupção. Considerando o conjunto das instruções existentes na rotina de tratamento da interrupção, serão necessários 94.5 μ s para seu atendimento. Isto significa um total de aproximadamente 100 μ s para o recebimento de 1 "byte" de dados, ou seja, 80kb/s em um canal. Para um máximo de quatro canais no Processador de Comunicação, essa taxa será dividida, o que significa uma taxa de 20kb/s por canal de comunicação para a recepção.

Como as rotinas de transmissões possuem um total de instruções aproximadamente igual à rotina de recepção, os valores se aproximam, e a taxa por canal também será aproximadamente 20kb/s na transmissão.

As características gerais do sistema portanto são: máximo de três canais com conexão serial do Processador de Comunicação com seus vizinhos contidos no subsistema de Comunicação através do padrão RS232, funcionando em regime de comutação síncro

na a uma taxa de transmissão de 20kbits e um canal serial com conexão do Processador de Comunicação com o Processador Operador utilizando também o padrão RS232, funcionando em regime de comunicação síncrono igual ao anterior ou assíncrono a uma taxa de transmissão selecionável de até 9600bit/s, viabilizando a conexão de qualquer máquina comercial, através de suas interfaces seriais, ao subsistema de comunicação.

No anexo 1 está o diagrama de circuitos do Processador de Comunicação.

5.2. Software do Processador de Comunicação

Como citado anteriormente o software do PC consiste do Programa Supervisor, rotinas de comunicação pelas linhas seriais do subsistema de comunicação, rotinas de comunicação pelas linhas seriais conectadas ao P0, e rotinas de suporte para a implementação das primitivas do sistema e do protocolo de comunicação.

5.2.1. O Programa Supervisor

O programa supervisor basicamente controla o fluxo da informação no subsistema de comunicação, cuidando do estabelecimento de rotas para a mensagem, bem como da manipulação de filas de mensagens, O programa supervisor portanto é quem elabora o nível 3 da proposta do Modelo "ISO".

Isto foi proposto através de três filas, cada uma associada a uma linha serial, e uma fila de anúncios como suporte pa

ra comunicação a nível de Primitivas de Comunicação e Sincroniza_
ção que será visto na secção 5.3..

A estrutura das filas no PC é mostrada na figura (5.2.1.1).

figura 5.2.1.1 - Estrutura das Filas

As filas serão manipuladas da seguinte forma: Uma mensa_
gem é recebida pelo Supervisor e encaminhada para uma das três fi

las, sendo duas filas de espera de transmissão para outro PC e uma de transmissão para o Processador Operador. Em se tratando de um anúncio, este é encaminhado para a fila de anúncios.

A determinação da fila de transmissão para onde a mensagem deve ser enviada para ser transmitida é feita simplesmente verificando o endereço destino da mensagem.

Para a manipulação das mensagens foram implementados três "buffers" de recepção e três "buffers" de transmissão associado as três filas de espera para transmissão, conforme figura (5.2.1.2). Esta estrutura é estática de tal forma que os ponteiros de "buffers" e de filas são pré-determinados com área também pré-estabelecida, ou seja, "buffer" com um total de 52 "bytes", fila de transmissão com disponibilidade de no máximo 4 quadros num total de 208 "bytes" e fila de anúncios com no máximo 20 anúncios com 9 "bytes" cada. As filas serão manipuladas na forma "LIFO" (Last In First Out). Na recepção das mensagens, à cada "byte" recebido em qualquer das linhas, ocorrerá interrupção e a rotina de recepção irá preenchendo o "buffer" de mensagem correspondente até acabar, quando então haverá uma anotação de "buffer" com recepção terminada.

A transmissão de mensagens será feito, "byte" a "byte" numa única linha por vez até esgotar-se a mensagem.

O programa Supervisor passa alternadamente por etapas de recepção e de transmissão. Na etapa de recepção, o programa Supervisor fará a varredura de todos os "buffers" que estejam com recepção terminada, de forma que, para cada mensagem, seja feita a coleta, determinação da rota, e colocação do conteúdo do "buffer" na fila apropriada. Na etapa de transmissão, o programa supervisor, a partir da última fila atendida na etapa de transmissão anterior, pegará um quadro da próxima fila e comandará sua trans -

missão pela linha serial correspondente.

figura 5.2.1.2 - Estrutura das Filas e Buffers

5.2.1.1. Protocolo

O protocolo utilizado entre os processadores de comunicação é baseado na troca de pacotes de controle visando indicar reconhecimento positivo com autorização de nova transmissão, reconhecimento negativo, etc. O aspecto fundamental do protocolo é que deve existir uma autorização do receptor para que a transferência ocorra. A condição inicial do sistema prevê a existência prévia de autorização em todas as linhas.

O protocolo utilizado é uma simplificação do protocolo "HDLC" sendo basicamente constituído de dois tipos de Quadros ou seja:

Quadros com informação no formato abaixo,

e Quadros de controle no formato abaixo,

sendo que:

CONTROLE - Especifica o tipo de quadro que pode ser:

1. Informação - indica que este quadro carrega uma informação.
2. Transmissão Rejeitada - indica que a última transmissão foi recebida com erro, e que deve ser transmitido novamente o último quadro.
3. Recebido e Liberado - indica que a última transmissão foi recebida corretamente e uma nova transmissão pode ser feita.
4. Recebido e não Liberado - indica que a última transmissão foi recebida corretamente, mas não existe espaço na fila e não pode ser feita outra transmissão.
5. Inicialização - indica que o subsistema de comunicação vai ser inicializado.

ENDEREÇO DESTINO - Endereço do PO destino pa-

ra onde o quadro deve ser transmitido.

INFORMAÇÃO - Este campo contém o texto da mensagem.

Uma vez definido o protocolo, passamos a descrever como se dá a sincronização na comunicação entre PC's pelo Supervisor, que será efetuado em cada canal de comunicação através da consulta de uma variável de Estado, que informa o estado atual do processo de comunicação naquele canal.

Esta variável pode especificar três estados possíveis, ou seja: "Recebido e Liberado", "Recebido e não Liberado" e "Esperando Confirmação". Mais especificamente um PC transmissor envia um quadro e fica aguardando reconhecimento. Esta situação é caracterizada configurando a variável de Estado para "Esperando Confirmação". O PC receptor recebendo o quadro irá alocar este quadro em uma de suas filas. Com a alocação deste quadro na fila apropriada, pode ter ou não sobrado espaço para mais um quadro. O reconhecimento portanto deverá acontecer de tal forma que o PC receptor transmita ao PC transmissor a informação de "Recebido e Liberado" caso tenha espaço na fila e "Recebido e não Liberado" caso não tenha espaço na fila.

Ainda com relação ao PC receptor, pode acontecer do quadro não ser recebido por algum problema. O PC receptor portanto transmite ao PC transmissor a informação de "Transmissão Rejeitada" o que fará o PC transmissor transmitir novamente o quadro. A figura (5.2.1.1.1) descreve os diagramas de estado na sincronização no Processo de Comunicação entre PCs.

a) UM CANAL PC TRANSMISSOR

b) UM CANAL PC RECEPTOR

figura 5.2.1.1.1 - Diagrama de Estados na Sincronização do Processo de Comunicação - Supervisor

O Supervisor faz três recepções e somente uma transmissão a cada vez. Esta decisão baseou-se no fato que cada transmissão gera no subsistema de comunicação outra transmissão de reconhecimento, e o subsistema de comunicação funciona como amortecedor de mensagens fazendo com que as mensagens fluam com maior ou

Desta forma o número de transmissões é menor que o número de recepções e o subsistema de comunicação tem que ter capacidade para receber, dos PO, mensagens que estes desejem transmitir, armazenando-as até que o processador Operador destino possa consumi-la. Por isso temos que dar uma maior prioridade para a recepção e optou-se intuitivamente para a proporção três para um.

Os Procedimentos que compõem o Programa Supervisor estão no anexo 3.

5.2.2. Rotinas de Comunicação pelas Linhas Seriais do Subsistema de Comunicação e Rotinas de Comunicação pelas Linhas Seriais Conectadas ao PO

As rotinas de comunicação [MARQUES, 1984], foram implementadas para a comunicação nas linhas seriais do subsistema de comunicação e para a comunicação nas linhas seriais conectadas ao Processador Operador.

A comunicação no subsistema de comunicação foi implementada utilizando o protocolo 'HDLC'. Isto porque a troca de mensagens no subsistema se dá através do dispositivo 'SIO', que facilita bastante a implementação deste protocolo, uma vez que ele incorpora características básicas do 'HDLC'.

No caso de conexão PO-PC foi preciso elaborar um protocolo assíncrono, com o objetivo de manter a compatibilidade com a maioria dos microcomputadores disponíveis no mercado, uma vez que eles geralmente possuem uma interface assíncrona tipo RS-232.

Uma vez que esses protocolos fazem parte do nível 2, houve a necessidade de se elaborar uma interface para a interligação

com o nível 3. A função desta interface é a de passar as informações necessárias ao programa Supervisor.

A comunicação no subsistema de comunicação como já citado, é totalmente implementada na própria interface "SIO". O Software que controla a interface está basicamente distribuído na forma de rotinas de atendimento de interrupção.

São quatro as rotinas que atendem as interrupções em um canal, sendo dispostas na seguinte forma:

SRC - Condição especial de Recepção

RCA - Caracter de Recepção disponível

TBE - "Buffer" de transmissão vazio

ESC - Mudança do Estado Externo

sendo que SRC tem maior prioridade de atendimento sobre RCA, TBE, ESC e assim sucessivamente.

Essas rotinas são divididas em dois grupos transmissão e recepção.

A transmissão é composta pelas rotinas TBE e ESC, que são responsáveis pela transmissão de um quadro. Para iniciar uma transmissão, é preciso carregar o primeiro "byte" do quadro na interface, e emitir alguns comandos necessários à sua reinicialização (por ex: "reset" no gerador de CRC, etc..). Feito isto, a interface começa a gerar interrupções TBE (uma para cada "byte"), pedindo "bytes" para a UCP, na medida em que forem sendo transmitidos. Este processo para quando ocorrer um "TX-underrun" (ou seja, ocorre uma interrupção TBE, mas não é carregado dado na interface) avisado por uma interrupção ESC. O fluxograma simplificado da rotina de atendimento de interrupção TBE, é mostrado na figura (5.2.2.1).

figura 5.2.2.1 - Fluxograma da Rotina TBE

Enquanto não acabam-se dados, a rotina faz a carga normal para a interface, a menos que seja enviado uma sequência de cancelamento pela rotina ESC. Se a rotina TBE detecta esses sinais, uma variável é assinalada, indicando o ocorrido para o programa Supervisor, um comando "reset" na interrupção TBE pendente é emitido.

do, e nenhum dado a mais é carregado na interface.

Quando acabam-se os dados, a interface deve avisar que a transmissão do quadro encerrou-se, para que o CRC seja enviado. O CRC é enviado quando ocorre um "TX-underrun". A maneira de forçar a ocorrência deste "TX-underrun", é não carregarmos mais dados na interface, e assinalar uma variável indicando que o procedimento para o envio automático do CRC já foi efetuado. Após isto nova interrupção TBE ocorre, assinalando que o "flag" de encerramento já foi enviado. Esta interrupção é a última, e em seu atendimento, deve-se assinalar uma variável de interface para o programa Supervisor, indicando que o quadro foi totalmente transmitido, e emitir o comando de resete na interrupção TBE pendente.

A função da rotina ESC na transmissão, é gerenciar a ocorrência de condição "TX-underrun". A ocorrência de "TX-underrun" pode ser esperada ou não. Quando acabam-se os dados, ela é esperada, bastando então emitir um comando "reset" na interrupção ESC. A ocorrência não esperada é quando, por algum motivo, não foi carregado dado na interface. Neste caso deve-se emitir um comando de envio da sequência de cancelamento e assinalar uma variável para que a rotina TBE possa detectar o ocorrido. O fluxograma simplificado da rotina de atendimento de interrupção ESC para a transmissão é descrito na figura (5.2.2.2).

A recepção é composta pelas rotinas RCA, SRC e ESC que são responsáveis pela recepção de um quadro. A interrupção RCA ocorre toda vez que um dado chega ao topo da fila de recepção da interface. O procedimento da rotina RCA, implica na leitura de dado na interface, e no seu armazenamento em um "buffer" na memória.

figura 5.2.2.2 - Fluxograma da Rotina ESC

A figura (5.2.2.3) mostra o fluxograma simplificado da rotina de atendimento da interrupção RCA.

figura 5.2.2.3 - Fluxograma da Rotina RCA

A interrupção tipo SRC ocorre no final da recepção de um quadro (detecção do "flag" de encerramento) ou quando ocorre um erro de "overrun" (a fila de recepção da interface está com três posições ocupadas, e um novo dado é recebido, sobrepondo um dado na fila).

O erro de CRC não gera uma interrupção SRC, mas é analisado quando ocorre a interrupção que acusa fim de um quadro. Para detectar-se qual é a causa da interrupção SRC, faz-se a leitura do registrador apropriado da interface.

Na ocorrência de um erro de "overrun" ou erro de CRC o quadro é descartado. Os "flags" de abertura e encerramento não são carregados na fila, no entanto, um dos "bytes" de CRC (são dois) é carregado indevidamente na fila, devendo ser lido e descartado para que não ocorra novas interrupções RCA.

Na figura (5.2.2.4) é apresentado o fluxograma simplificado da rotina de atendimento de interrupção SRC.

A interrupção tipo ESC ocorre na recepção quando o receptor detecta uma sequência de cancelamento ou quando o receptor entra em sincronismo por ter recebido "flag". Para se determinar a causa da interrupção ESC, deve-se também ler o registrador apropriado na interface. Ao se detectar uma sequência de cancelamento deve-se assinalar uma variável para que o quadro seja posteriormente descartado. Ao entrar em sincronismo, emite-se apenas o comando de "reset" da interrupção ESC pendente.

figura 5.2.2.4 - Fluxograma da Rotina SRC

O fluxograma simplificado da rotina de interrupção ESC para a recepção é mostrado na figura (5.2.2.5).

figura 5.2.2.5 - Fluxograma da Rotina ESC
(Recepção)

Existem muitos outros detalhes sobre as rotinas de interrupção, que foram omitidos, mas que podem ser encontrados em [CROMEMCO,1981].

Finalmente na comunicação entre Processador Operador e Processador de Comunicação foi escolhido uma comunicação serial assíncrona RS-232, devido ao fato de que a maioria dos P0s disponíveis no mercado, possuem uma interface nesse padrão.

O protocolo empregado possui o formato abaixo:

2B	00	DADOS-TAMANHO PEQUENO ≥ 0	D4	D4
----	----	--------------------------------	----	----

2B	FF	DADOS-TAMANHO GRANDE. ≥ 0	D4	D4
----	----	--------------------------------	----	----

O protocolo de tamanho pequeno é utilizado para as mensagens de controle, e o grande para mensagens do usuário.

Com essa comunicação foi elaborado para P0s que utilizam um dispositivo periférico com interface TMS 5501 (Texas), que às vezes duplica o primeiro "byte" de dados recebidos, escolheu-se os dígitos "2B" do protocolo como caracteres que servem para assinalar se o dado foi duplicado ou não, comparando-o com o segundo caracter recebido pela interface. Esta comparação é válida, uma vez que o segundo caracter do quadro será obrigatoriamente "00" ou "FF" caso não haja duplicações. O valor "2B" foi escolhido aleatoriamente.

Os dígitos "00" e "FF" indicam quantos "bytes" a interface tem que receber, ou seja, "00" indica que o tamanho do quadro a receber é pequeno e "FF" indica tamanho grande.

Para indicar o fim de um quadro são utilizados dois dígitos "D4", também escolhido aleatoriamente.

O software que controla essa comunicação é disposto em duas rotinas (RECEBE e TRANSMIT) utilizadas pelo programa Super

visor.

Também, como no programa Supervisor, existe uma variável de estado que indica o estado de um canal. Essa variável serve de interface entre o programa Supervisor e este programa que somente transmite e recebe quadros.

5.3. O Software do Processador Operador

A nível de Supervisor, a sincronização no processo de comunicação entre Processador Operador e Subsistema de Comunicação deve se manter conforme utilizado neste nível para a comunicação entre PC's, exceto que no Processador Operador pode-se dispensar toda àquela manipulação de filas. Portanto o programa Supervisor a nível de Processador Operador basicamente opera no protocolo de Comunicação neste nível, e faz a interface entre este nível e o nível superior para a troca de mensagens. O programa que descreve o Procedimento Supervisor a nível de Operador está no anexo 4.

Os níveis inferiores ao nível Supervisor no Processador Operador foram implementados conforme descrito no item 5.2.2. mais especificamente na Comunicação entre PO e PC.

5.3.1. Primitivas de Comunicação e Sincronização

Todo software desenvolvido até aqui, está relacionado com os níveis 1, 2 e 3 da proposta pelo Modelo "ISO", ou seja: nível físico, nível de ligação e nível de rede.

O nível 4, ou nível de Transporte, é onde efetivamente ocorre a troca de mensagens entre processos situados em processa-

dores operadores diferentes, uma vez que os níveis inferiores tratam da manipulação de mensagens internas ao subsistema de comunicação e asseguram canais livres de erros na comunicação entre os Processadores Operadores.

Primitivas de Comunicação e Sincronização são operações através do qual se faz a comunicação entre processos, operações estas a serem implementadas no nível de Transporte utilizando basicamente os canais livres de erros citados acima e um protocolo de comunicação específico para este nível.

5.3.1.1. Tipos de Primitivas de Comunicação e Sincronização

As ações dessas Primitivas consistem basicamente numa emissão de mensagem, realizada por um processo denominado emissor, ou numa recepção de mensagem, realizada por um processo denominado receptor, acompanhado de algumas ações de controle [SEGRE e KIRNER, 1982].

A característica mais importante das primitivas de comunicação e sincronização, está no fato delas bloquearem, ou não, os processos que executam as operações de emissão e recepção de mensagem.

Desta forma é possível ter quatro modos de sincronização e comunicação entre processos, dependendo das combinações possíveis de primitivas bloqueadas e não bloqueadas [MANNING, et alii, 1980], que são as seguintes:

(Eb, Rb) (En, Rn) (En, Rb) (Eb, Rn)

onde Eb significa emissão bloqueada, Rn significa Recepção não bloqueada e assim sucessivamente.

Basicamente uma Emissão é bloqueada quando o processo que envia a mensagem fica bloqueado, esperando a sincronização com o processo receptor, até que o mesmo esteja pronto para recebê-la, quando então, depois da mensagem ser transferida, o emissor será desbloqueado. Da mesma maneira, se não houver nenhuma emissão pendente, esperando sincronização, o processo receptor se bloqueia na operação de recepção até que uma mensagem seja enviada. A figura (5.3.1.1.1) ilustra esta descrição.

a) Situação com emissão pendente

b) Situação com recepção pendente

figura 5.3.1.1.1 - Emissão e Recepção bloqueados

Na Emissão não bloqueada o que ocorre é que, o processo que executar a primitiva, não ficará bloqueado esperando a sincronização com o outro processo, ele continuará a sua execução, logo depois que sua mensagem for encaminhada para transmissão, sem precisar esperar por nenhuma sincronização. Com a Recepção não bloqueada, o processo deverá continuar a sua execução, mesmo quando não existir nenhuma mensagem pronta para ser recebida.

A figura (5.3.1.1.2) ilustra o processo não bloqueado.

~~~~~  
(En)  
~~~~~  
↓

*transfere a mensagem
em qualquer situação
e continua sua execu-
ção.*

~~~~~

· (Rn)

~~~~~
↓

*se existir emissão ou mensagem
pendente, ela será recebida e
continua a execução senão na-
da ocorre e continua a execu-
ção.*

figura 5.3.1.1.2 - Emissão e Recepção não
bloqueados

A principal vantagem das primitivas não bloqueadas é que elas permitem explorar ao máximo o paralelismo de um conjunto de processos concorrentes, em contraposição ao que ocorre com as primitivas síncronas ou bloqueadas.

Visando atender as necessidades fundamentais na comunicação entre processos da rede DCE's, optou-se em uma primeira instância pela implementação de primitivas consistindo de "SEND" bloqueado, "RECEIVE" bloqueado e "RECEIVE" não bloqueado que passamos a discutir.

5.3.1.2. Primitivas de Comunicação e Sincronização da Rede DCE's

A comunicação entre processos na REDE DCE's será feita de uma forma síncrona ou seja emissão e recepção bloqueados.

Dessa forma se um Emissor quiser enviar uma mensagem, o Receptor só a aceitará quando ele estiver pronto para recebê-la. Quando isso ocorrer o sistema onde se encontra o receptor deve enviar uma autorização ao sistema onde se encontra o emissor, permitindo que a transmissão da mensagem seja iniciada.

O Emissor que quer enviar a mensagem não irá transmiti-la enquanto o receptor não estiver pronto para recebê-la. Uma das maneiras de fazer com que o receptor saiba que existe uma emissão pendente é remeter-lhe um anúncio avisando-o que há uma mensagem para ele.

Se não fosse enviado esse anúncio, a própria mensagem é quem teria que ser transmitida. Em se tratando de uma Rede de Computadores, várias mensagens poderiam alcançar um único receptor, o que acarretaria numa sobrecarga no Processador de Comunicação, que é onde essas mensagens deveriam estar para que o receptor quando pronto, pudesse recebê-la. Justifica-se portanto esse anúncio, que é uma mensagem simplesmente anunciando a existência de uma mensagem a ser transmitida de um Emissor qualquer.

Verifica-se portanto a existência de uma fila de anúncios em cada Processador de Comunicação existente no Subsistema de Comunicação, de tal forma que a chegada de uma mensagem em um determinado PC é analisada e se for anúncio é colocado na fila de anúncios, se for mensagem é enviada ao P0 ligado a ele.

A iteração entre os processos acontecem conforme a figura (5.3.1.2.1) onde $P0_i$ é o operador transmissor e PC_i o seu processador de comunicação, $P0_j$ é operador receptor e PC_j o seu processador de comunicação correspondente.

figura 5.3.1.2.1 - Protocolo de Comunicação entre dois Processadores Operadores

A iteração se dá na seguinte forma:

$P0_i$ querendo enviar uma mensagem ao $P0_j$, chama a ro-

tina "SEND" que cria um anúncio e o envia para o PC*i* correspondente. O PC*i* transmite esse anúncio ao PC*j* que coloca este anúncio na fila de anúncios. Enquanto isso o PO*i* fica bloqueado esperando a autorização para enviar mensagem.

Quando o PO*j* chama a rotina "RECEIVE", ocorre a autorização de anúncio ao PC*j*. O PC*j* por sua vez verifica na fila de anúncios se a fila está vazia ou não. Se a fila não estiver vazia o primeiro anúncio que estiver na fila é transformado em autorização de anúncio e transmitido para o PC*i*. No caso da fila não estar vazia é devolvido ao PO*j* um "liberou anúncio" caso contrário é devolvido "fila vazia". O PC*i* recebendo a autorização de anúncio transmite essa autorização ao PO*i*.

Quando a autorização alcançar o PO*i*, a mensagem é dividida em pacotes e transmitidas ao PO*j*, passando obviamente pelos PC*i* e PC*j*.

Enquanto a mensagem é transmitida, seus "bytes" são somados. O mesmo ocorrendo na hora da recepção de mensagem. Terminando a transmissão da mensagem, é transmitido logo em seguida o resultado da soma dos "bytes" "CHECKSUM" para o PO*j*.

O PO*j* recebendo este "CHECKSUM" compara com o seu, faz o reconhecimento positivo se forem iguais, ou negativo se forem diferentes.

No caso do reconhecimento ser positivo, o PO*j* envia este reconhecimento ao PO*i* e encerra seu processo. Caso contrário, o reconhecimento também é enviado ao PO*i*, e o PO*j* fica na espera da retransmissão da mensagem pelo PO*i*.

O PO*i* recebendo o reconhecimento, o analisa e se for positivo também encerra seu processo, caso contrário retransmite a mensagem.

Tanto a emissão quanto a recepção de mensagens terão as sociados esquemas específicos de reconhecimento, "timeout" e re-transmissão para evitar a perda de mensagem. Isto vai no sentido de complementar tratamento parecido já realizado na comunicação entre processadores de comunicação.

Todas as informações que circulam pela rede neste nível estão em forma de pacotes e estão descritos no anexo 2.

5.3.1.2.1. Rotinas de Tratamento das Primitivas

As primitivas de comunicação e sincronização consistem de duas rotinas básicas e estrutura de dados divididas entre o Processador Operador e Processador de Comunicação.

As rotinas são: "SEND" bloqueado e "RECEIVE" bloqueado, com os seguintes parâmetros:

SEND (OP, Pdest, Porig, ORIGmsg, TAMmsg) onde

OP-Tipo de operação a ser executado conforme anexo 2;

Pdest-Processador para onde vai ser enviado as Mensagens;

Porig-Processador de onde veio a Mensagem (no caso o próprio PO que chamou a rotina);

ORIGmsg-Endereço de memória onde está posicionado a mensagem a ser transmitida;

TAMmsg-Tamanho da mensagem, em "bytes", a ser transmitida;

RECEIVE (OP, Pdest, Porig, ORIGmsg, TAMmsg) onde

OP-Tipo de operação que vai ser executado;

Pdest-Processador que vai receber a mensagem (no caso

o que chamou a rotina);

Porig-Processador de onde vem uma mensagem;

ORIGmsg-Endereço da memória a partir do qual vai ser depositada a mensagem recebida;

TAMmsg-Tamanho da mensagem recebida em 'bytes';

Os procedimentos que compõem o Programa das Primitivas de Comunicação e Sincronização entre Processos "SEND" bloqueado e "RECEIVE" bloqueado estão no anexo 4.

Essas rotinas basicamente se utilizam de dois "buffers" de mensagem de 7 "bytes" cada, um para transmissão de pacotes e outro para recepção de pacotes, além de duas áreas de memória uma contendo a mensagem a ser transmitida e outra contendo a mensagem recebida.

Portanto uma mensagem a ser transmitida de um P0i para um P0j deverá ser colocada na memória no P0i a partir do qual serão definidas as variáveis ORIGmsg, TAMmsg, Pdest, Porig, OP e em seguida é chamado o Procedimento "SEND" que se encarrega de enviar a mensagem.

Da mesma forma o P0j receptor, através do Procedimento "RECEIVE", recebe a mensagem com os parâmetros Porig, Pdest, TAMmsg e OP, que será posicionada em área também determinada pela variável ORIGmsg na memória, podendo então ser utilizada pelo P0j configurando assim a troca de mensagem entre processos localizados em Processadores distintos.

6. CONCLUSÕES

6.1. Sobre a Análise de Arquiteturas de Sistemas Distribuídos

A análise desenvolvida nos capítulos iniciais permitiu uma visão geral sobre Sistemas Distribuídos destacando principalmente os aspectos de topologia e protocolos de Comunicação, tomando como base as várias implementações existentes. Com essa análise foi possível familiarizar-se com os vários conceitos ligados à Arquitetura de Sistemas Distribuídos e com isso definir adequadamente o subsistema de Comunicação, os protocolos de comunicação e o Software até o nível de Primitivas de Comunicação e Sincronização entre processos.

6.2. Sobre o Desenvolvimento do Projeto

A nível de Hardware foram desenvolvidos, testados e estão em funcionamento:

- Um Processador Operador denominado "BIJU" [SILVA e SAITO, 1986], com uma Unidade Central de Processamento Z80, 64k "bytes" de Memória "RAM" estática, 4k "bytes" de Memória "PROM", uma interface paralela para impressora, uma interface serial RS-232 e quatro interfaces para Unidades de Disco flexível de 8 ou 5 1/4 polegadas, distribuídos em três cartões de circuito impresso interligados por um barramento paralelo no padrão S100 [GARETS e LIBES, 1981]. Estão acoplados a este sistema um terminal de vídeo, uma Unidade de Disco flexível para Discos de 8 polegadas, e uma impressora. O sistema o-

pera através do "CDOS" (Cromemco Disk Operating System) que é compatível com os programas desenvolvidos no sistema CP/M com pequenas alterações.

- Um Módulo de Interfaceamento serial para o Processador Operador "BIJU", utilizando o componente da "INTEL-8251", com acesso ao barramento no padrão S100 providenciando a conexão do "BIJU" ao Subsistema de Comunicação, [SILVA,1985].

- Três Processadores de Comunicação com até quatro canais de comunicação através da interface "S10" e um canal de comunicação serial através do porto serial 8251.

A nível de Software foram desenvolvidos e estão em funcionamento:

- O Software de Comunicação entre Processadores de Comunicação utilizando o protocolo "HDLC", elaborando assim os níveis 1 e 2 no Subsistema de Comunicação [MARQUES,1984], software este instalado em cada processador de comunicação através de memórias "EPROM".

- O Software de Comunicação entre Processador Operador e Processador de Comunicação, com protocolo próprio de comunicação conforme citado no item 5.2.1.1 [MARQUES,1984].

- Um Programa Monitor [SILVA e ROSSIGALLI,1985], contendo instrução como amostragem de trechos de memória, substituição de conteúdo de memória e executa programa a partir de determinada posição de memória, com objetivo de facilitar os testes de comunicação no subsistema de comunicação a nível de linguagem de máquina para microprocessador Z80.

- O Programa Supervisor elaborando o nível 3 no Subsistema de Comunicação e no Processador Operador, cujos procedimentos estão no anexo 3 e 4 respectivamente.

- O Programa elaborando as Primitivas de Comunicação e Sincronização "SEND" e "RECEIVE" descritos no anexo 4.

6.3. Sobre os Pontos Relevantes e Contribuição do Trabalho

Como pontos relevantes e contribuições do trabalho, destacam-se:

- o projeto servir primeiramente como um laboratório de pesquisa na área de Redes de Computadores e Sistemas Distribuídos, tanto nos aspectos relacionados com o hardware quanto aos aspectos relacionados com o software.

- a proposta de um Processador de Comunicação utilizando interfaces "SIO" (Serial Input Output), por se tratar de um componente altamente poderoso no processo de comunicação.

- a existência dos PC's, com protocolo que permite aviso prévio de envio de mensagem e com a manipulação de filas de mensagens em cada PC, através do qual é possível o envio de pacotes de dados sem a conexão física imediata do Processador Operador fonte com o Processador Operador destino, evitando assim a perda destes pacotes, uma vez que estes preencherão as filas de mensagens nos PC's, e que na medida do possível serão enviados ao Processador Operador destino.

6.4. A Continuidade do Trabalho

Como proposta de continuidade do trabalho, pode-se:

- incluir no programa Monitor desenvolvido, instruções do tipo acesso a 'buffers' de mensagens, acesso à variáveis de estado, inserção de mensagens em 'buffers' de transmissão, etc, tornando-se um recurso de grande valia nos testes de comunicação no subsistema, à nível de linguagem de máquina.
- ampliar o Programa Supervisor, inserindo o tratamento de mais um canal de comunicação, em um total de quatro canais por Processador de Comunicação.
- implementação de um Sistema Operacional Distribuído e programas aplicativos em geral.

7. BIBLIOGRAFIA

- ABRAMSON, N. - The ALHQA system - In: Computer Communication Network. New Jersey, Prentice Hall. 1973.
- AGRAWAL, D.P. & ALEXANDER, W.E. - B-HIVE: a heterogeneous, interconnected, versatile and expandable multicomputer system. Comp. Archit. News, 12(2): 7-13, 1984.
- ANDERSON, G.A. & JENSEN, E.D. - Computer interconnection structures: taxonomy, characteristics, and examples. Comp. Surv., 7(4): 197-213, dec., 1975.
- ARAUJO, J.F.M.de - Redes locais de computadores. Cad. Tec. INFO, 1983. 16p. (Digibras,6).
- BRAYER, K. & LAFLEUR, V. - A testbed approach to the design of a computer communication network. Computer IEEE, 15(10): 14-23, oct., 1982.
- CLARK, D.D.; POGRAN, K.T.; REED, D.P. - An introduction to local area networks. Proceedings IEEE, 66(11): 1497-1517, nov., 1978.
- COTTON, I.W. - Technologies for local area computer networks. Proceedings of the LACN Symposium. p.25-44, 1979.
- CROMENCO, I.N.C. - Quadart-instruction manual. California, may, 1981. 214p. (Part nº 023-2005).
- CUNHA, P.R.F.; MONTEIRO, J.A.S.; LUCENA, E.B. - Implementação de um protocolo de transporte para a Rede Cepinne. Simpósio Brasileiro de Redes de Computadores, 3. Rio de Janeiro, 1985. (9.1-9.26).
- DEMARINES, V.A. & HILL, L.W. - The cable bus in data communication. Datamation, aug., 1976.

- EICHORST, H.P. - Multipuc: um sistema distribuído de microcomputadores. Seminário Integrado de Software e Hardware, 6. Campinas-SP. Anais. p.26-27, 1979.
- ENSLOW JR., P.H. - Multiprocessor organization. Comp. Surv., 9(1): 103-129, mar., 1977.
- FERRO, V.N. - Rede local ITAUTEC. Simpósio Brasileiro de Redes de Computadores, 3. Rio de Janeiro-RJ. 1985. (2.1-2.19).
- FLYNN, M. - Some computer organization and their effectiveness. IEEE Trans. Comp., 21(9), sept.; 1972.
- GADELHA, A.E.R.; VAZ, F.A.; NOVOA, M.A.A. - Rede Cobra de computadores: estratégia de implementação. Congresso da Soc. Brasileira de Computação, 2. Anais e Seminário Integrado de Software e Hardware, 9. Ouro Preto-MG. Anais. p.173-190, 1982.
- GARETZ, M. & LIBES, S. - Interfaces to S100/IEEE 696 Microcomputers. Califórnia, McGraw-Hill, 1981. 322p.
- HAFNER, E.R.; NENADAL, Z.; TSCHANZ, M. - A digital Loop communication system. IEEE Trans. Commun. C-22(6): 877-881, jun., 1974.
- HOPKINS, G.T. - Multimod communications on the MITRINET. Comp. Networks, 4(5): 229-233, 1980.
- KELLY, P.T.F. - Public packet switched data networks: international plans and standards. Proceedings IEEE, 66(11): 1539-1548, nov., 1978.
- KIRNER, C. - Suporte para desenvolvimento de sistemas distribuídos: uma implementação. Congresso da Sociedade Brasileira de Computação, 4. Anais e Seminário Integrado de Software e Hardware, 9. Viçosa-MG. Anais. p.109-121, 1984.

- KIRNER, C. & MARQUES, E. - Suporte para desenvolvimento de sistemas distribuídos baseado num barramento paralelo centralizado. Congresso da Sociedade Brasileira de Computação, 5. Anais e Seminário Integrado de Software e Hardware, 11. Porto Alegre-RS. Anais. p.423-435, 1985.
- LIU, M.T. - Distributed Loop computer networks. In:Advances on Computers, New Yorks, Academic Press, v.17. p.163-221. 1978.
- LOUREIRO, A.A.F.; et alli - Proposta de uma rede local para o DCC. Simpósio Brasileiro de Redes de Computadores, 3. Rio de Janeiro-RJ, 1985 (3.1-3.13).
- MANNING, E.; LIVESEY, N.J.; TODUKA, H. - Interprocess communication in distributed systems: one view. IFIP, 1980.
- MARQUES, E. - Suporte para desenvolvimento de sistemas distribuídos: implementação do software básico de comunicação. Simpósio de Software Básico, 4. S.J.dos Campos-SP. Anais, p.184-196, 1984.
- MENASCE, D.A. & SCWABE, D. - Redes de computadores: aspectos técnicos e operacionais. Escola de Computação, 3. jul., 1982 184p.
- METCALFE, R.M. & BOGGS, D.R. - ETHERNET: distributed packet switching for local computer networks. Comm. ACM, 19(7): 395-404, jul., 1976.
- NAVARRO, L.C. & RIBEIRO Fº, H. - Rede local SCOPUS. Simpósio Brasileiro de Redes de Computadores, 3. Rio de Janeiro-RJ, 1985 (5.1-5.41).
- PARDO, R. - Interprocess communication and synchronization for distributed system. Dissertação apresentada a Ohio State University, 1979, 229p.

- POVZIN, L. - Presentation and major design aspects of the CYCLADES networks. Data Comm. Symp., 3. Proc. p.80-85, 1973.
- ROBERTS, L.G. & WESSLER, B.D. - Computer networks development to achieve resource sharing. Proc. AFIPS 1970 SJCC p. 543-549.
- RODRIGUES, G.C. - Protocolos da rede-local NCE-UFRJ. Seminário Integrado de Software e Hardware, 8. Florianópolis-SC, A nais. p.423-252, 1981.
- RUGGIERO, W.V.; et alii - Especificação da rede local do LSD-EP-USP. Seminário Integrado de Software e Hardware, 6. São Paulo-SP, Anais. p.95-99 1979.
- SEGRE, L. & KIRNER, C. - Primitivas de comunicação e sincronização de processos por troca de mensagens: uma análise. São Carlos-SP, UFSCar-DCEs 41p. 1982. (Relatório Técnico).
- SHIMIZU, E.Y. - Componentes básicos de uma máquina de arquitetura distribuída: identificação e implementação. Seminário Integrado de Software e Hardware, 7. Campinas-SP. Anais. p.77-86, 1980.
- SILVA, J.L. - Suporte para desenvolvimento de sistemas distribuídos: interface serial de um processador operador. São Carlos-SP, UFSCar-DCEs 16p. 1985. (Relatório Técnico).
- SILVA, J.L. & ROSSIGALLI, M.S. - Um monitor para um processador de comunicação da rede DCEs. Simpósio sobre Desenvolvimento de Software Básico, 5. Belo Horizonte-MG, Anais. p.65-78, 1985.
- SILVA, J.L. & SAITO, J.H. - Implementação de um microcomputador Z80: BIJU. São Carlos-SP, UFSCar-DCEs. (Relatório Técnico a ser publicado).

SOARES, L.F.G; et alii - Descrição do hardware e software da rede local, REDPUC. Congresso Nacional de Informática, 15, Anais. p.447-451. 1982.

STALLING, W. - Local networks. Comp.Surv. 16(1): 3-41, mar., 1984.

TAKANO, D.F.; AUDE, E.P.L.; AZEVEDO, M.H.C. - Arquitetura e implementação da rede local NCE/UFRJ. Seminário Integrado de software e hardware, 8. Florianópolis-SC. Anais. p.253-266. 1981.

TANENBAUM, A.S. - Computer networks. New Jersey, Prentice-Hall, 1981, 517p.

TOBAGI, F.A. - Analysis of a two hop centralized packet radio networks. Part II: Carrier sense multiple access. IEEE Trans. Commun. C-28(2): 208-216, fev., 1980.

WEITZMAN, C. - Distributed micro/mini-computer systems. New Jersey, Prentice Hall, 1980 403p.

WEST, L.P. - Loop - transmission control structures. IEEE Trans. Commun. 20(3): 531-539, jun., 1972.

WILLARD, D.G. - A time division multiple access for digital communication. Comp. Des. 13(8): 79-83, jun., 1974.

WITTIE, L.D. & TILBORG, A.M.V. - Micros, a distributed operating system for MICRONET: a reconfigurable network computer. IEEE Trans. Comp. 29(12): 1133-1144, dec., 1980.

WULF, W.A. & BELL, C.G. - C.mmp-a multi-mini-processor. AFIPS, 1972, p.765-777.

ZILOG, 1981a - Z80: CPU Central Processing Unit-Product

ZILOG, 1981b - Z80: SIO Serial Input/Output Controller-Product Specification. mar., 1981 17p.

ZILOG, 1981c - Z80: CTC Counter/Timer Circuit-Product Specifications. mar., 1981 13p.

ZIMMERMANN, H. - OSI reference model - the ISO Model of architecture of open systems interconnection. IEEE Trans. Commun. C-28(4): 425-432, apr., 1980.

ANEXO 1

"DIAGRAMA DE CIRCUITO DO PROCESSADOR DE COMUNICAÇÃO"

ANEXO 2

"CONFIGURAÇÃO DOS PACOTES NA COMUNICAÇÃO ENTRE PROCESSOS"

FORMATO DOS PACOTES

TAM, PACOTE	OPERAÇÃO	TIPO	DESTINO	ORIGEM	*	*
----------------	----------	------	---------	--------	---	---

TAM.PACOTE - será 7 se não for mensagem, caso contrário será o tamanho de um pacote de mensagem mais 7.

OPERAÇÃO - leitura em disco, gravar em disco, usar a impressora, etc.

TIPO - anúncio, autorização de anúncio, fila vazia, liberou anúncio, autorização, "checksum", reconhecimento e mensagem.

DESTINO - para onde vai o pacote.

ORIGEM - de onde vem o pacote.

, - se o tipo for:

- (a) - Mensagem → número do pacote e tamanho da mensagem respectivamente.
- (b) - Reconhecimento → 0 e Positivo/Negativo respectivamente.
- (c) - "Checksum" → resultado da soma.
- (d) - 0 Restante → tamanho da mensagem

PACOTE DO TIPO ANÚNCIO

Pacote capaz de caracterizar uma mensagem a ser transmitida.

7	*	ANUNCIO	*	*	TAM _{msg}
---	---	---------	---	---	--------------------

* - parâmetros fornecidos na chamada da rotina "SEND".

PACOTE DO TIPO AUTORIZAÇÃO DE ANÚNCIO

Este pacote autoriza o Processador de Comunicação a retirar um anúncio da fila e transformá-lo em Autorização.

7	*	AUTO, ANUN.	*	*	TAM _{msg}
---	---	----------------	---	---	--------------------

* - parâmetros fornecidos na chamada da rotina "RECEIVE".

PACOTE DO TIPO FILA VAZIA

Pacote enviado do Processador de Comunicação ao Processador Operador ligado a ele indicando que a fila de anúncios está vazia.

7	*	FILA VAZIA	ORIG.	*	TAM _{msg}
---	---	---------------	-------	---	--------------------

* - parâmetros fornecidos na chamada da rotina "RECEIVE".

PACOTE DO TIPO LIBEROU ANÚNCIO

Pacote enviado do Processador de Comunicação ao Processador Operador ligado a ele indicando que liberou um anúncio.

7	*	LIBE, ANUN.	ORIG.	*	TAM _{msg}
---	---	----------------	-------	---	--------------------

* - parâmetros fornecidos na chamada da rotina "RECEIVE".

PACOTE DO TIPO AUTORIZAÇÃO

Pacote especificando a liberação para envio de mensagem.

7	*	AUTORI.	*	*	TAM _{msg}
---	---	---------	---	---	--------------------

* - parâmetros fornecidos na chamada da rotina "RECEIVE".

PACOTE DO TIPO "CHECKSUM"

Pacote transmitido depois do envio de uma mensagem, usado no reconhecimento da mensagem.

7	*	CHECKSUM	*	*	RESULTADO CHECKSUM
---	---	----------	---	---	-----------------------

* - parâmetros fornecidos na chamada da rotina "SEND".

PACOTE DO TIPO RECONHECIMENTO

Pacote cujo conteúdo especifica se uma mensagem deve ou não ser retransmitida.

7	*	RECONH.	*	*	Ø	Ø / 1
---	---	---------	---	---	---	-------

* - parâmetros fornecidos na chamada da rotina "RECEIVE".

PACOTE DO TIPO MENSAGEM

Pacote com 7 parâmetros e acompanhada de um quadro de informação.

TAM, PACO.+7	*	MENSA.	*	*	Ø	Nº DO PACOTE	INFORM.
-----------------	---	--------	---	---	---	-----------------	--------------

* - parâmetros fornecidos na chamada da rotina "SEND".

ANEXO 3

"PROGRAMA SUPERVISOR"

```

A.TYPE PRO2.PAS
PROGRAM SUPERVISOR(INPUT,OUTPUT);
(*****)
(*)
(*)
(*)   PROGRAMA SUPERVISOR IMPLEMENTADO PARA
(*)   CADA PROCESSADOR DE COMUNICACAO NO
(*)   SUBSISTEMA DE COMUNICACAO.
(*)
(*)
(*)
(*****)
LABEL 10,20,30;
TYPE   BUFFER=ARRAY [1..52] OF INTEGER;
        FILA=ARRAY [1..208] OF INTEGER;
VAR     STAT:INTEGER;
        FILA1,FILA2,FILA3:FILA;
        PIB1,PIB2,PIB3,PIBT1,PIBT2,PIBT3:BUFFER;
        PFF1,PFF2,PFF3,ESPLA,ESPL1,ESPL2,ESPL3;
        PCA,PCF1,PCF2,PCF3,PFFLA,PFFL1,PFFL2,PFFL3,I,STAT1,
        STAT2,STAT3,NBT,TLIMIT,TOUT1,TOUT2,TOUT3,P,SINAL:INTEGER;
        INICIO,FAROL,SUCESSO1,SUCESSO2,SUCESSO3:BOOLEAN;
PROCEDURE TRANSMIT(VAR BURT:BUFFER;ENDFI:INTEGER);
BEGIN
  (* TRANSMISSAO A NIVEL DE HDLC *)
  (* NO SUBSISTEMA DE COMUNICACAO *)
END;
PROCEDURE TRATAEXCECAO;
BEGIN
  (* PROCEDIMENTO QUE TRATA OCORRENCIA *)
  (* DE FALHA, QUE SERA ANALISADO EM *)
  (* OUTRA ETAPA DO PROJETO. *)
END;
(*)
(*)
(*)   PROCEDIMENTO PARA TRANSMISSAO DE BUFFER E
(*)   ATUALIZACAO DE PONTEIROS DE FILA
(*)
(*)
(*)
PROCEDURE TREFT (VAR TPIB:BUFFER;PCFX,ESPLX,PFILA:INTEGER);
VAR I:INTEGER;
BEGIN
  FOR I:=1 TO 52
  DO BEGIN
    CASE PFILA OF
      1:TPIB[I]:=FILA1[PCFX];
      2:TPIB[I]:=FILA2[PCFX];
      3:TPIB[I]:=FILA3[PCFX]
    END;
    PCFX:=PCFX+1
  END;
  ESPLX:=ESPLX+52;
  TRANSMIT(TPIB,PFILA);
  (*ESPERANDO CONFIRMACAO*)
  CASE NBT OF
    1:STAT1:=3;
    2:STAT2:=3;
    3:STAT3:=3
  END
END;
(*)
(*)   PROCEDIMENTO QUE INTERPRETA UMA MENSAGEM
(*)   SE FOR CONTROLE ENTAO ATUALIZA STATUS
(*)   SE MENSAGEM ENTAO MOVE PARA RESPECTIVA FILA
(*)

```

```

(*)      E ATUALIZA PONTEIROS DE FILA.      *)
(*)
PROCEDURE INTERPRETAR(VAR TCTRL,TPIB:BUFFER;INICIO:BOOLEAN);
VAR TIRA,I,FNC,PFFLX,ESPLX,PCFX:INTEGER;PIBT:BUFFER;
PROCEDURE ATFIL;
BEGIN
    CASE TCTRL[2] OF
        1:FILA1[PFFLX]:=TCTRL[1];
        2:FILA2[PFFLX]:=TCTRL[1];
        3:FILA3[PFFLX]:=TCTRL[1]
    END
END;
PROCEDURE FIVALB(COD,FLA:INTEGER);
VAR I:INTEGER;
BEGIN
    FOR I:=1 TO 9
    DO PIBT3[1]:=TCTRL[1];
    PIBT3[3]:=COD;
    PIBT3[1]:=0;
    PIBT3[8]:=FILA3[PCA+8];
    TRANSMIT(PIBT3,3);
    TIRA:=FLA
END;
BEGIN
    INICIO:=FALSE;FNC:=0;
    IF TCTRL[1]=0 (*MENSAGEM*)
    THEN IF ((TCTRL[5]<>1) OR (P=3)) AND (TCTRL[5]<>2)
        (*SE NAO E ANUNCIO NEM AUTORIZA ANUNCIO*)
        THEN BEGIN CASE TCTRL[2] OF
            1:BEGIN PFFLX:=PFFL1;ESPLX:=ESPL1;PCFX:=PCF1 END;
            2:BEGIN PFFLX:=PFFL2;ESPLX:=ESPL2;PCFX:=PCF2 END;
            3:BEGIN PFFLX:=PFFL3;ESPLX:=ESPL3;PCFX:=PCF3 END
        END;
        IF ESPLX=0
        THEN BEGIN
            TPIB[1]:=3; (*REJEITADO*)
            TRANSMIT(TPIB,P)
        END
        ELSE BEGIN
            (*MOVE PARA FILA RESPECTIVA*)
            FAROL:=TRUE;
            WHILE FAROL
            DO BEGIN
                IF PFFLX<>208
                THEN BEGIN
                    FOR I:=1 TO 52
                    DO BEGIN
                        ATFIL;
                        PFFLX:=PFFLX+1
                    END;
                    ESPLX:=ESPLX-52;
                    IF ESPLX=0 THEN FNC:=1;
                    FAROL:=FALSE
                END
            ELSE IF PCFX=PFFLX
            THEN BEGIN
                PCFX:=1;
                PFFLX:=1
            END
        ELSE BEGIN
            FOR I:=1 TO (208-ESPLX)
            DO BEGIN

```

```

END;
PCFX:=PCFX+1
END;
PCFX:=1;
PFFLX:=I+1
END
END;
IF FNC=0
THEN BEGIN TPIBC1J:=1;(*RECEBIDO E LIBERADO*)
TRANSMIT(TPIB,P)
END
ELSE BEGIN TPIBC1J:=2;(*RECEBIDO E NAO LIBERADO*)
TRANSMIT(TPIB,P)
END
END
END
END
ELSE IF TCTRL5J=1 (*SE ANUNCIO*)
THEN BEGIN
FAROL:=TRUE;
WHILE FAROL
DO BEGIN IF PFFLA<>180
THEN BEGIN (*MOVE PARA FILA DE ANUNCIO*)
FOR I:=1 TO 9
DO BEGIN
FILAA[PFFLAJ]:=TCTRL1J;
PFFLA:=PFFLA+1
END;
ESPLA:=ESPLA-9;
FAROL:=FALSE;
END
ELSE IF PCA=PFFLA
THEN BEGIN PCA:=1;PFFLA:=1 END
ELSE BEGIN
FOR I:=1 TO (180-ESPLA)
DO BEGIN FILAACIJ:=FILAA[PCA];
PCA:=PCA+1
END;
PCA:=1;
PFFLA:=I+1
END
END;
END
END
ELSE BEGIN (*SE AUTORIZA ANUNCIO*)
IF ESPLA=180
THEN FIVALE(3,0) (*FILA VAZIA*)
ELSE FIVALE(8,1); (*LIBEROU ANUNCIO*)
IF TIRA=1
THEN BEGIN
FOR I:=1 TO 9
DO BEGIN
CASE FILAA[PCA+7J] OF
1:BEGIN
PIBT1C1J:=FILAA[PCA];
PIBTC1J:=PIBT1C1J
END;
2:BEGIN
PIBT2C1J:=FILAA[PCA];
PIBTC1J:=PIBT2C1J
END;
3:BEGIN
PIBT3C1J:=FILAA[PCA];
PIBTC1J:=PIBT3C1J
END
END
END
END

```

```

        ESPLA:=ESPLA+9;
        PIETC1:=0;
        PIETC2:=PIETC7;
        PIETC3:=4;(*AUTORI*)
        TRANSMIT(PIBT,FILAALPCA+7)
    END
END
ELSE IF TCTRLC1=1
    THEN BEGIN (*RECEBIDO E LIBERADO*)
        CASE P OF
            1:STAT1:=1;
            2:STAT2:=1;
            3:STAT3:=1
        END
    END
ELSE IF TCTRLC1=2
    THEN BEGIN (*RECEBIDO E NAO LIBERADO*)
        CASE P OF
            1:STAT1:=2;
            2:STAT2:=2;
            3:STAT3:=2
        END
    END
ELSE IF TCTRLC1=3 (*TRANSMISSAO REJEITADA*)
    THEN TRANSMIT(TPIB,P)
    ELSE INICIO:=TRUE
END;
BEGIN
10:  ESPLA:=208;ESPL1:=208;ESPL2:=208;ESPL3:=208;
    PCA:=1;PCF1:=1;PCF2:=1;PCF3:=1;
    PFFLA:=1;SINAL:=0;PFFL1:=1;PFFL2:=1;PFFL3:=1;
    NBT:=0;STAT1:=1;STAT2:=1;STAT3:=1;
    TOUT1:=0;TOUT2:=0;TOUT3:=0;TLIMIT:=255;P:=0;
    SUCESSO1:=FALSE;SUCESSO2:=FALSE;SUCESSO3:=FALSE;
20:  P:=P+1;
    CASE P OF
        (*
        (* OS CASOS 1,2 E 3 ANALISAM A RECEPCAO
        (* O CASO 4 PROVIDENCIA UMA TRANSMISSAO
        (*
        1:IF NOT SUCESSO1
            (* SUCESSO GERADO ATRAVES DE
            (* INTERRUPCAO NO TRATAMENTO
            (* DO PROTOCOLO HDLC.
        THEN BEGIN
            IF STAT1<>3 (*ESPERANDO CONFIRMACAO*)
            THEN GOTO 20
            ELSE IF TOUT1=TLIMIT
                THEN BEGIN
                    TOUT1:=0;
                    TRATAEXCECAO
                END
            ELSE BEGIN TOUT1:=TOUT1+1;GOTO 20 END
        END
    ELSE BEGIN
        TOUT1:=0;
        (*INTERPRETAR A MENSAGEM*)
        INTERPRETAR(FIB1,PIBT1,INICIO);
        IF INICIO THEN GOTO 10
        ELSE GOTO 20
    END;
2:IF NOT SUCESSO2

```

THEN BEGIN

```
TOUT2:=0;
TRATAEXCECAO
```

END

```
ELSE BEGIN TOUT2:=TOUT2+1;GOTO 20 END
```

END

ELSE BEGIN

ROUT2:=0;

INTERPRETAR(FIE2,PIET2,INICIO):

```
IF INICIO THEN GOTO 10
```

ELSE GOTO 20

END;

3. IF NOT SUCCESSFUL

THEN BEGIN

IF STAT3<3

THEN GOTO 20

```
ELSE IF TOUT3=TLIMIT
  THEN BEGIN
```

```
TOUT3:=0;
TRATAEXCECAO
```

END

```
ELSE BEGIN TOUT3:=TOUT3+1;GOTO 20 END
```

END

```
ELSE BEGIN
```

```
TOUT3:=0;
```

```
INTERPRETAR(FIE3,PIBT3,INICIO);
```

IF INICIO THEN GOTO 10

```
ELSE GOTO 20
```

END:

4:BEGIN

(*
(*
(*
(*

DEFINICAO DO PROXIMO BUFFER A SER TRANSMITIDO.

✕ }
 ✕ }
 ✕ }
 ✕ }

30:

$$F_z = 0$$

NET := NET + 1;

```
SINAL:=SINAL+1;
```

```
IF NET=4 THEN NET:=1;
```

```
IF SINAL=4 THEN BEGIN SINAL:=0;GOTO 20 END;
```

CASE NET OF

1: BEGIN

```
IF ESFL1=208
```

THEN GOTO 30

```
ELSE IF STAT1=1
```

THEN BEGIN

```
TREFT(PIET1,PCF1,ESFL1,1);
GOTO 20
```

END

```
ELSE BEGIN STAT1:=1;GOTO 30 END
```

GOTO 20

END

ELSE BEGIN STAT3:=1;GOTO 30 END

END

END

END

END

ANEXO 4

**"PROCEDIMENTO DAS PRIMITIVAS SEND,
RECEIVE E SUPERVISOR NO P.O."**


```

A.TYPE PRO1.PAS
PROGRAM TESE(INPUT,OUTPUT);
(*****)
(*)
(*)
(*)    CONJUNTO DE PROCEDIMENTOS QUE IMPLEMENTAM    (*)
(*)    PRIMITIVAS DE COMUNICACAO E SINCRONIZACAO    (*)
(*)    ENTRE PROCESSOS E O PROGRAMA SUPERVISOR A    (*)
(*)    NIVEL DE PROCESSADOR OPERADOR.              (*)
(*)
(*)
(*****)
TYPE    BUFFER=ARRAY [1..50] OF INTEGER;
VAR      BUFT,BUFR:ARRAY [1..50] OF INTEGER;
          FILA1,FILA2:ARRAY [1..200] OF INTEGER;
          STAT:INTEGER;

(*)
(*)
(*)    PROCEDIMENTO DA PRIMITIVA SEND              (*)
(*)
(*)
PROCEDURE SEND(VAR OP,PDEST,PORIG,ORIGMSG,TAMMSG:INTEGER);
LABEL    10,20,30;
CONST    ANUN=1;(*ANUNCIO*)
          AUTOANU=2;(*AUTORIZA ANUNCIO*)
          FILAVA=3;(*FILA VAZIA*)
          AUTORI=4;(*AUTORIZA MENSAGEM*)
          CHSUM=5;(*CHECKSUM*)
          RECON=6;(*RECONHECIMENTO*)
          MSG=7;(*MENSAGEM*)
          TLIMIT=255;(*TEMPO DE TIMEOUT*)
          LIBANU=8;(*LIBEROU ANUNCIO*)
          TRANS=2;(*TRANSMITIR*)
          RECEBE=1;(*RECEBER*)
VAR      PFT,(*PONTEIRO DE FILA DE TRANSMISSAO*)
          PFR,(*PONTEIRO DE FILA DE RECEPCAO*)
          RECMTD,(*RECONHECIMENTO*)
          TAMPA,(*TAMANHO DO PACOTE*)
          NOPACO,(*NUMERO DO PACOTE*)
          CHECKSUM,(*SOMA DOS BYTES*)
          ORIG,(*POSICAO DE ORIGEM DA MENSAGEM*)
          TOUT,(*TIMEOUT*)
          TAMM,(*TAMANHO DA MENSAGEM*)
          I:INTEGER;
          FAROL : BOOLEAN;

(*)
(*)
(*)    PROCEDIMENTO DE MONTAGEM DE BUFFER          (*)
(*)    PARA TRANSMISSAO.                          (*)
(*)
(*)
PROCEDURE MBUFM (VAR BUFF:INTEGER);
BEGIN
          BUFT[1]:=7;
          BUFT[2]:=OP;
          BUFT[4]:=PDEST;
          BUFT[5]:=PORIG;
          CASE BUFF OF
            ANUN:BUFT[6]:=TAMMSG;
            MSG:BEGIN BUFT[1]:=BUFT[1]+TAMPA;
                      BUFT[6]:=0;
                      BUFT[7]:=NOPACO;
            END;
          END;

```

```

RECON:BUFTC6J:=0;
CHSUM:BUFTC6J:=CHECKSUM;
AUTORI:BUFTC6J:=0;
AUTOANU:BUFTC6J:=0
END
END;
BEGIN
10:   PFT:=1;PFR:=1;STAT:=1;
      (* *)
      (*ENVIA ANUNCIO DE MENSAGEM*)
      (* *)
      BUFTC3J:=ANUN;
      MBUFM(BUFTC3J);
      SUPERVIOP(TRANS);
      (* *)
      (*ESPERA AUTORIZACAO*)
      (* *)
      FAROL:=TRUE;TOUT:=0;
      WHILE FAROL AND (TOUT<=TLIMIT)
      DO BEGIN
          SUPERVIOP(RECEBE);
          IF BUFTC3J=AUTORI
          THEN FAROL:=FALSE
          ELSE TOUT:=TOUT+1
      END;
20:   IF FAROL
      THEN TRATAEXCECAO (*NAO RECEBEU AUTORIZACAO DE ANUNCIO*)
      ELSE BEGIN
          (* *)
          (*INICIALIZA ALGUMAS VARIAVEIS PARA TRANSMISSAO*)
          (* *)
          TAMM:=BUFTC6J;(*TAMANHO DA MENSAGEM*)
          ORIG:=ORIGMSG;
          NOPACO:=0;
          CHECKSUM:=0;
          TAMPA:=43
      END;
      (* *)
      (*ENVIO DOS PACOTES*)
      (* *)
30:   BUFTC3J:=MSG;
      TAMM:=TAMM-TAMPA;
      IF TAMM<0 THEN BEGIN TAMPA:=(TAMM+43);TAMM:=0 END;
      MBUFM(BUFTC3J);
      FOR I:=8 TO (TAMPA+7)
      DO BEGIN
          BUFTCIJ:=FILA1[PFT];
          PFT:=PFT+1;
          CHECKSUM:=CHECKSUM+BUFTCIJ
      END;
      SUPERVIOP(TRANS);
      (* *)
      (*SE TERMINOU ENVIA O CHECKSUM*)
      (* *)
      IF TAMM<>0
      THEN BEGIN NOPACO:=NOPACO+1;GOTO 30 END
      ELSE BEGIN
          BUFTC3J:=CHSUM;
          MBUFM(BUFTC3J);
          SUPERVIOP(TRANS)
      END;
      (* *)
      (*FICA AGUARDANDO O RECONHECIMENTO*)

```

```

DU BEGIN
    SUPERVIOF(RECEBE);
    IF BUFRC[3]=RECON
    THEN BEGIN
        IF BUFRC[7]=0
        THEN RECMTO:=0
        ELSE RECMTO:=1;
        FAROL:=FALSE;
        END
    ELSE BEGIN TOUT:=TOUT+1;RECMTO:=0 END
    END;
    IF FAROL THEN TRATAEXCECAO
    ELSE IF RECMTO=0
    THEN BEGIN
        TOUT:=0;FAROL:=TRUE;
        BUFTC[6]:=TAMMSG;
        GOTO 20
    END
END;
(* *)
(* *)
(* PROCEDIMENTO DA PRIMITIVA RECEIVE *)
(* *)
(* *)
PROCEDURE RECEIVE(VAR OP,PDEST,PORIG,ORIGMSG,TAMMSG;INTEGER);
LABEL 10,20,30;
CONST ANUN=1;
      AUTOANU=2;
      FILAVA=3;
      AUTORI=4;
      CHSUM=5;
      RECON=6;
      MSG=7;
      TLIMIT=255;
      LIBANU=8;
      TRANS=2;
      RECEBE=1;
VAR PFT,PFR,RECMTO,TAMPA,CS,
    NOPACO,CHECKSUM,TOUT,TAMM,I;INTEGER;
    CONT,FAROL;BOOLEAN;
PROCEDURE MEUFM (VAR BUFF;INTEGER);
BEGIN
    BUFTC[1]:=7;
    BUFTC[2]:=OP;
    BUFTC[4]:=PDEST;
    BUFTC[5]:=PORIG;
    CASE BUFF OF
        ANUN:BUFTC[6]:=TAMMSG;
        MSG:BEGIN BUFTC[1]:=BUFTC[1]+TAMPA;
                  BUFTC[6]:=0;
                  BUFTC[7]:=NOPACO;
                END;
        RECON:BUFTC[6]:=0;
        CHSUM:BUFTC[6]:=CHECKSUM;
        AUTORI:BUFTC[6]:=0;
        AUTOANU:BUFTC[6]:=0
    END
END;
END;
BEGIN
    STAT:=1;
10:   PFR:=ORIGMSG;
      TAMPA:=43;
      (* *)
      (*AUTORIZAR ANUNCIO*)

```

```

      MBUF(M(BUFTC3));
      SUPERVIOP(TRANS);
20:   CHECKSUM:=0;CONT:=FALSE;
      (*                                     *)
      (*ESPERA ALGUMA INFORMACAO*)
      (*                                     *)
30:   FAROL:=TRUE;
      TOUT:=0;
      WHILE FAROL OR (TOUT<=TLIMIT)
      DO BEGIN
          SUPERVIOP(RECEBE);
          IF ((BUFR13=CHSUM) AND CONT ) OR
             ((BUFR13=FILAVA) AND NOT CONT ) OR
             ((BUFR13=MSG) AND CONT ) OR
             ((BUFR13=LIBANU) AND NOT CONT )
          THEN FAROL:=FALSE
          ELSE TOUT:=TOUT+1
      END;
      IF FAROL
      THEN TRATAEXCECAO (*ESTOUROU TIMEOUT NA RECEPCAO*)
      ELSE (*ANALISE DA INFORMACAO*)
          CASE BUFR13 OF
          5:BEGIN (*CHECKSUM*)
              CS:=BUFR16;
              BUFTC3:=RECON;
              CS:=CS-CHECKSUM;
              IF CS=0
              THEN BEGIN BUFTC7:=1;RECMTD:=1 END
              ELSE BEGIN BUFTC7:=0;RECMTD:=0 END;
              MBUF(BUFTC3);
              SUPERVIOP(TRANS);
              IF RECMTD=0
              THEN GOTO 20
          END;
          3:GOTO 10;(*FILA VAZIA*)
          7:BEGIN (*MENSAGEM*)
              TAMPA:=BUFR11-7;
              TAMB:=TAMB-TAMPA;
              FOR I:=8 TO (TAMPA+7)
              DO BEGIN
                  FILA2(PFR):=BUFR11;
                  PFR:=PFR+1;
                  CHECKSUM:=CHECKSUM+BUFR11
              END;
              GOTO 30
          END;
          8:BEGIN (*LIBEROU ANUNCIO*)
              TAMB:=BUFR16;CONT:=TRUE;GOTO 30 END
          END
      END;
END;
(*                                     *)
(*                                     *)
(*   PROCEDIMENTO SUPERVISOR NO PROCESSADOR   *)
(*   OPERADOR.   *)
(*                                     *)
(*                                     *)
PROCEDURE SUPERVIOP( TRANS:INTEGER);
LABEL 10;
TYPE   BUFFER=ARRAY [1..50] OF INTEGER;
VAR    PIB,PIBT:BUFFER;
        SUCESSO:BOOLEAN;
        TOUT,TLIMIT,I:INTEGER;

```

```

THEN BEGIN
  IF STAT=3 (*ESPERANDO CONFIRMACAO*)
  THEN IF TOUT=TLIMIT
    THEN TRATAEXCECAO
    ELSE BEGIN TOUT:=TOUT+1;GOTO 10 END
END

```

```

ELSE BEGIN

```

```

  IF PIBCI=0 (*PACOTE COM MENSAGEM*)
  THEN BEGIN

```

```

    FOR I:=1 TO 50

```

```

      DO BUFCI:=PIBCI+2;

```

```

      PIBCI:=1;

```

```

      TRANSMIT(PIBT) (*RECEBIDO E LIBERADO*)

```

```

    END

```

```

    ELSE IF PIBCI=1 (*RECEBIDO E LIBERADO*)

```

```

      THEN STAT:=1

```

```

    ELSE IF PIBCI=2 (*RECEBIDO E NAO LIBERADO*)

```

```

      THEN STAT:=2

```

```

    ELSE IF PIBCI=3 (*REJEITADO*)

```

```

      THEN BEGIN

```

```

        TRANSMIT(PIBT);

```

```

        STAT:=3;

```

```

        GOTO 10

```

```

      END

```

```

    END;

```

```

  2: IF STAT=1 (*TRANSMISSAO LIBERADA*)

```

```

  THEN BEGIN

```

```

    PIBCI:=0; (*MENSAGEM*)

```

```

    FOR I:=1 TO 50

```

```

      DO PIBCI+2:=BUFCI;

```

```

      PIBCI:=BUFCI+4;

```

```

      TRANSMIT(PIBT);

```

```

      STAT:=3;

```

```

      TRANS:=1;GOTO 10

```

```

    END

```

```

  ELSE BEGIN

```

```

    FOR I:=1 TO 255 DO STAT:=1; (*AGUARDA OPORTUNIDADE

```

```

    GOTO 10

```

```

  END

```

```

END

```

```

END;

```

```

PROCEDURE TRANSMIT(VAR BUFT:BUFFER);

```

```

BEGIN

```

```

  (* TRANSMISSAO NO PROTOCOLO NIVEL 2

```

```

  *)

```

```

END;

```

ABREVIACES UTILIZADAS

ARPANET - Advanced Research Projects Agency Networks.

BISYNC - Binary Synchronous Communication.

CCITT - Comit Consultante Internationale de Telegraphique et Telephonique.

Cmmp - Carnegie Mellon multi-mini-processor.

CRC - Cdigo de Redundncia Cclica.

CSMA - Carrier Sensing Multiple Acess.

CSMA/CD - Carrier Sensing Multiple Acess/Colision Detect.

CTC - Counter Timer Circuit.

DCE - Data Circuit - Terminating Equipment.

DCE's - Departamento de Computao e Estatstica.

DECNET - Digital Equipment Corporation's.

DFM - Diviso de Frequncia Multiplexada.

DMA - Direct Memory Acess.

DSE - Data Switching Exchange.

DTE - Data Terminal Equipment.

Eb - Emisso bloqueada.

En - Emisso no bloqueada.

ESC - Mudana do Estado Externo.

ETB - End of Transmission Bloch.

ETX - End of Text.

HDLC - High Level Data Link Control.

HIVE - Heterogeneous, Interconnected, Versatile and Expandable.

IBM - International Busyness Machine.

ISO - International Standard Organization.

LIFO - Last in First Out.

PC - Processador de Comunicação.

PO " Processador Operador.

PROM - Programable Read Only Memory.

RAM - Random Access Memory.

Rb - Recepção bloqueada.

RCA - Caracter de Recepção Disponível.

RF - Rádio Frequência.

Rn - Recepção não bloqueada.

ROM - Read Only Memory.

SIO - Serial Input Output.

SNA - Systems Network Architecture.

SOD - Sistema Operacional Distribuído.

SOH - Start of Header.

SRC - Condição Especial de Recepção.

STX - Start of Text.

SYN - Synchronize.

TBE - Buffer de Transmissão Vazio.

TDMA - Time Division Multiple Access.

UCP - Unidade Central de Processamento.