José Antônio Gomes de Lima

Um Controlador Microprogramável para Comutadores ATM

Tese submetida à Coordenação dos Cursos de Pós-Graduação em Engenharia Elétrica da Universidade Federal da Paraíba - Campus II como parte dos requisitos necessários para obtenção do grau de Doutor em Ciências em Engenharia Elétrica.

Área de Concentração: Processamento da Informação

Orientador: Prof. Dr. Elmar Uwe Kurt Melcher

Campina Grande, Paraíba, Brasil

Junho de 1999

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
COORDENAÇÃO DOS CURSOS DE PÓS-GRADUAÇÃO EM
ENGENHARIA ELÉTRICA

UM CONTROLADOR MICROPROGRAMÁVEL PARA COMUTADORES ATM

JOSÉ ANTÔNIO GOMES DE LIMA

Campina Grande - PB
Junho/1999.

L732c Lima, Jose Antonio Gomes de
Um controlador microprogramavel para comutadores ATM / Jose Antonio Gomes de Lima. - Campina Grande, 1999.
158 f.

Tese (Doutorado em Engenharia Eletrica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Comutadores ATM 2. Redes de Computadores 3. Tese I. Melcher, Elmar Uwe Kurt II. Universidade Federal da Paraiba - Campina Grande (PB)

CDU 621.316.3(043)

UM CONTROLADOR MICROPROGRAMÁVEL PARA COMUTADORES ATM

JOSÉ ANTONIO GOMES DE LIMA

Tese Aprovada em 14.06.1999

ELMAR UWE KURT MELCHER, Dr., UFPB
Orientador

EDNA NATIVIDADE DA SILVA BARROS, Dr.rer.nat., UFPE
Componente da Banca

LUIGI CARRO, Dr., UFRGS
Componente da Banca

ANTONIO CARLOS CAVALCANTI, Dr., DI-UFPB
Componente da Banca

RAIMUNDO CARLOS SILVÉRIO FREIRE, Dr., UFPB
Componente da Banca

MARIA IZABEL CAVALCANTI CABRAL, D.Sc., UFPB
Componente da Banca

CAMPINA GRANDE - PB
Junho - 1999

Dedico este trabalho à minha esposa Maria Emilia e a nossos filhos Guilherme e Paula pelo apoio e compreensão nas inúmeras horas em que estive ausente do convívio com eles para poder elaborá-lo.

Agradecimentos

- Ao Prof. Dr. Elmar Uwe Kurt Melcher, pela dedicação e constante paciência na busca das soluções dos problemas e pela orientação em toda a extensão deste trabalho;
- Ao Prof. Hamilton Soares da Silva pelo empenho e cooperação demonstrados; pela ajuda, incentivo e companheirismo, indispensáveis ao desenvolvimento deste trabalho;
- Ao Prof. Dr. Antônio Carlos Cavalcanti, pelas idéias, críticas e sugestões que muito ajudaram na elaboração deste trabalho e pela participação na banca;
- À Profª Dra. Edna Natividade da Silva Barros, pelas idéias e sugestões na organização dos tópicos do trabalho e pela participação na banca;
- Aos Professores Mirabeau Dias, Raimundo Gouveia da Nóbrega Filho e Gustavo Henrique Matos Bezerra Motta pelo incentivo e companheirismo demonstrados;
- Ao Prof. Dr. Antônio Marcos Nogueira, coordenador de pós graduação em Engenharia Elétrica (COPELE) da UFPB pelo apoio demonstrado;
- À Profª Dra. Maria Isabel Cavalcanti Cabral pela participação na banca;
- Ao Prof. Dr. Raimundo Carlos Silvério Freire pela participação na banca;
- Ao Prof. Dr. Luigi Carro pela participação na banca;
- À Profª Odésia Correia de Amorim pela revisão ortográfica;
- À Ângela da COPELE;
- À CAPES/PICD, pelo apoio financeiro, através de Bolsa de Estudos.

Porque Deus amou o mundo de tal maneira que deu o seu filho unigênito para que todo aquele que nele crê não pereça, mas que tenha a vida eterna.

João 3.16

Resumo

Grande parte dos trabalhos sobre o processamento das funções da Camada ATM existentes na literatura se caracterizam pela implementação de arquiteturas em ASICs (Circuito Integrado de Aplicação Específica) desenvolvidos a partir de um conjunto próprio de especificações. Estas arquiteturas não são flexíveis o suficiente para absorver mudanças devido a novas padronizações, otimização, ou surgimento de um novo tipo de serviço que podem exigir alterações nos procedimentos de execução das funções da Camada ATM, definidos nas especificações iniciais. A incorporação dessas mudanças só é possível através de um novo ciclo de projeto e fabricação, devido a pouca ou nenhuma capacidade de reprogramação de tais arquiteturas.

Este trabalho propõe um arquitetura denominada: Controlador Microprogramável para Comutadores ATM (CMCA), para executar as funções da Camada ATM tendo como base uma unidade de controle microprogramada. Esta unidade permite alterações nas especificações iniciais do projeto sem mudanças no *hardware*, sendo suficiente para tal, sua reprogramação.

Como forma de validar a arquitetura e provar sua viabilidade, foi usado o conjunto de ferramentas dos ambientes de projeto de circuitos integrados CADENCE e ALTERA, para chegar às características do circuito final através de implementação *standard-cells* e FPGA.

Abstract

Most of works that propose architectures to perform the ATM layer functions obtain an ASIC or ASICs implementation based in their own specifications. These architectures are not flexible to allow adjusts due to changes in standards, optimization needs or new services requirements, that improve changes in original set of specifications like: a new parameter header; changes in current input rate policing algorithm; a new routing table structural definition; changes in switching element control, etc. A new development and fabrication cycles is need to incorporate these news specifications.

This work proposes a microprogrammable controller designed to perform the ATM layer functions of an ATM switch based in a microprogrammable control unit. This unit gives flexibility to change the initial specifications by microinstructions programming, with the same hardware. The controller logic validation and viability was made using CADENCE and ALTERA frameworks, resulting in a *standard-cells* and FPGA implementations.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo

1

Introdução

Já no final da década de 50 se pensava na integração das tecnologias de comutação e de transmissão, sendo o conceito formalizado em junho de 1971, numa reunião do grupo de trabalho 2 do grupo de estudo XI do CCITT [1], que definiu o termo Rede Digital de Serviços Integrados (RDSI - Integrated Services Digital Network (ISDN)).

A idéia da RDSI era a de dotar o usuário de uma "tomada de informações" [2], contendo uma interface comum para a transferência de dados dos mais variados tipos e com possibilidade para acomodar novos serviços, sem a necessidade de criar-se uma rede dedicada para os mesmos.

A evolução da RDSI levou ao surgimento da chamada RDSI-FL (Rede Digital de Serviços Integrados Faixa Larga) [3], [4] tendo, como característica principal, o uso do conceito de comutação rápida de pacotes, denominado Modo de Transferência Assíncrono (ATM), garantindo um uso mais eficiente da banda passante e menor complexidade no processamento da comutação [5], [6].

A partir de 1990, os primeiros protótipos de comutadores ATM começaram a surgir, enfocando principalmente o uso de malhas de interco-

nexão como elemento básico de comutação, estratégias de interligação dos elementos de comutação para a construção de grandes malhas (escalabilidade) e gerenciamento simples de filas. As propostas de comutadores ATM, apresentadas na exibição da Telecom em 1991 em Geneva, Suíça e no ISS'92 (International Switching Symposium - 1992) em Yokohama, Japão, tinham essas características [7].

Nos últimos 4 anos, a crescente convergência entre as tecnologias de comunicação e computação influiu decisivamente para o surgimento de novos serviços, afetando substancialmente os requisitos da nova geração de comutadores ATM que, agora, além de manter altas vazões e serem escaláveis, necessitam agregar novas características como: controle de fluxo das conexões, controle de congestionamento, estratégias de agendamento de prioridade de células, etc.

Apesar dos esforços de órgãos como o ITU-T e Fórum ATM (seção 2.1), a padronização básica de ATM, ainda hoje, não é suficiente para garantir uniformidade mínima de implementação, deixando ao projetista uma boa margem de criatividade para atender funcionalidades internas de seu comutador. A indústria de Comutadores ATM enfoca soluções proprietárias voltadas para atender as necessidades de grandes empresas e provedores de serviços, notadamente no que diz respeito à escalabilidade, velocidade e interoperacionalidade. Informações técnicas mais detalhadas sobre esses produtos são bastantes escassas, em parte por ser um mercado desenvolvido mais recentemente, mas certamente também por refletir o valor estratégico face ao acesso de empresas de menor porte às facilidades de projeto de circuitos integrados ASIC.

A literatura apresenta trabalhos sobre arquiteturas de comutadores ATM que podem ser divididos em três temas principais: Arquiteturas do Elemento Comutador [8], [9], [10], [11], [12], [13] Gerenciamento e Controle das células nas portas de saídas[14], [15] [16], [17], [18], [19], [20] e Processamento das Funções relacionadas com a Camada ATM [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31].

Grande parte dos trabalhos sobre o Processamento das funções da Camada ATM, se caracterizam pela implementação em um ou mais ASICs, desenvolvidos a partir de um conjunto próprio de especificações. Estas especificações resultam da forma de como as arquiteturas propostas executam as funcionalidades da Camada ATM, como por exemplo: a quantidade e tipo dos parâmetros que serão anexados ao novo cabeçalho da célula (endereço da porta de saída, quantidade e tipo de qualidade de serviço da conexão (QOS); parâmetros necessários ao gerenciamento das filas de células nas portas de saída e que dependem do tipo de gerenciamento praticado pelo comutador, etc.); a escolha do algoritmo de controle do fluxo de células (algoritmo de policiamento); definição da estrutura e organização da tabela de rotas do comutador, indispensável à execução das funções da Camada ATM; definição de uma interface específica para o controle do envio das células para o elemento comutador, etc.

O ASIC resultante destas especificações executa as funções da Camada ATM sem flexibilidade para suportar mudanças devido à absorção de novas padronizações, otimização ou aparecimento de um novo tipo de serviço. Isto pode requerer, por exemplo, a inclusão de novos parâmetros no cabeçalho da célula e a reorganização (ordem) destes parâmetros dentro do cabeçalho; o aumento do número de portas de saída do comutador; alteração no tipo de gerenciamento das filas de células nas portas de saída, impli-

cando na redefinição dos parâmetros associados a este gerenciamento; a modificação do algoritmo de policiamento praticado, ou inclusão de um novo algoritmo; redefinição da estrutura e organização da tabela de rotas, alterando a forma de acesso e o manuseio dos dados da tabela; alteração no elemento comutador tendo como conseqüência uma redefinição da interface com este elemento. A incorporação dessas mudanças só é possível através de um novo ciclo de projeto e fabricação, devido à pouca ou nenhuma facilidade de reprogramação de tais arquiteturas.

Este trabalho propõe a arquitetura de um sistema, denominado Controlador Microprogramável para Comutadores ATM (CMCA) [32], [33], [34] para a execução das funcionalidades da Camada ATM tendo como base uma unidade de controle microprogramada, de forma a torná-lo flexível às modificações de suas funcionalidades. A flexibilidade permite alterações nas especificações iniciais do projeto como a mudança dos algoritmos de roteamento, policiamento e gerenciamento praticados pelo sistema, bem como alterações no formato dos parâmetros e organização da tabela de rotas do comutador. Todas estas alterações, que podem refletir incorporações de novas padronizações, otimizações e novas especificações decorrentes do surgimento de novos serviços ATM são aceitas sem mudança no hardware, sendo suficiente a reprogramação da unidade de controle. Da mesma forma, o sistema permite conexão com vários tipos de elementos comutadores existentes, através de uma interface paralela, possibilitando a construção de um sistema comutador com múltiplas portas. Associado à microprogramação, é utilizado também o paralelismo de operações, como forma de agilizar o processamento para atender as altas taxas de transmissão das células (de 155 Mbps a 622 Mbps).

A motivação principal que levou ao desenvolvimento deste trabalho, foi nossa participação no projeto COMATM (Comutador ATM) - *ProTem-CC 1994* [35], [36], [37], [38], [39], que tinha como objetivo principal a formação de pessoal qualificado e a criação de um ambiente (*hardware e software*) para estudo e desenvolvimento de arquiteturas e protocolos para redes de alta velocidade, em particular redes ATM. Esse projeto cooperativo contava com a participação do Departamento de Engenharia Elétrica, do Departamento de Sistemas e Computação e do Departamento de Informática da UFPB, do Departamento de Informática da UFPE e do Laboratório de Microeletrônica da USP. As responsabilidades eram definidas por dois grupos criados no âmbito do projeto: o "grupo de redes" e o "grupo de VLSI". Cabia ao "grupo de redes" a definição das especificações de funcionalidades dos comutadores ATM e protocolos de rede. Cabia ao "grupo VLSI" a proposta de arquiteturas que atendiam a essas especificações, sua modelagem, simulação e síntese usando os ambientes de projeto de sistemas definidos inicialmente. Nossa participação no "grupo VLSI", a experiência adquirida em projetos de arquiteturas voltadas às redes ATM e a prática no uso das ferramentas, permitiram a concepção e desenvolvimento da arquitetura proposta neste trabalho.

O trabalho está organizado em sete capítulos. O capítulo 2 apresenta a evolução das redes de telecomunicações e introduz a Rede Digital de Serviços Integrados de Faixa Estreita (RDSI-FE) e a Rede Digital de Serviços Integradas de Faixa Larga (RDSI-FL). Apresenta o Modo de Transferência Assíncrono (ATM) e descreve a célula ATM. O capítulo 3 descreve a arquitetura geral de um Comutador ATM, o Algoritmo Genérico de Controle de Taxa (GCRA – Generic Cell Rate Algorithm), enfoca algumas implementações da indústria e aborda o estado da arte das arquiteturas

de Comutadores ATM. O capítulo 4 introduz o conceito de unidade de controle microprogramada, descreve a arquitetura proposta para a execução das funções da Camada ATM e o fluxograma geral que serviu como base para a sua concepção. O capítulo 5 apresenta o modelo utilizado para descrever a arquitetura do CMCA e descreve o cenário de teste utilizado para a sua simulação e validação lógica. O capítulo 6 apresenta os resultados obtidos na síntese do circuito em tecnologia ECPD07 da ATMEL, em tecnologia FPGA e outras considerações de sistema. Finalmente, no capítulo 7, são abordadas as conclusões e sugestões de trabalhos futuros.

Capítulo

2

Redes ATM

Este capítulo apresenta a evolução das redes de telecomunicações e introduz a Rede Digital de Serviços Integrados de Faixa Estreita (RDSI-FE) e a Rede Digital de Serviços Integradas de faixa Larga (RDSI-FL). Apresenta ainda os serviços de Faixa Larga, o Modo de Transferência Assíncrono, a célula ATM e, finalmente, as Camadas Física e ATM do modelo de referência de protocolos da RDSI-FL, utilizadas no desenvolvimento do presente trabalho.

2.1 Rede Digital de Serviços Integrados – RDSI

Antes do surgimento do conceito de Rede Digital de Serviços Integrados (RDSI), cada tipo de serviço especializado necessitava de uma rede dedicada para atender aos requisitos específicos. Na rede telefônica, por exemplo, canais de voz eram alocados no percurso entre os terminais, utilizando a "comutação de circuitos". Na rede de comunicação de dados era utilizada a "comutação de pacotes", onde os dados são divididos e "empacotados" com informações adicionais de endereçamento e entregues ao destinatário específico. Na rede de telex utilizava-se a "comutação de circuitos".

A existência de diversos tipos de serviços implicava na necessidade de conexões específicas, bem como na necessidade de desenvolvimento de equipamentos distintos, diminuindo a escala de fabricação e aumentando o custo.

A digitalização da rede telefônica, associada ao desejo de unificação das diversas redes, levou ao surgimento das Redes Digitais de Serviços Integrados. O termo Rede Digital de Serviços Integrados (RDSI - *Integrated Services Digital Network - ISDN*) surgiu em 1971 e tinha como idéia principal fornecer uma integração de serviços através de uma interface comum para a transferência de dados dos mais variados tipos. Nessa fase inicial, a integração ainda dependia de redes dedicadas para o atendimento dos mesmos (RDSI-FE - Rede Digital de Serviços Integrados Faixa Estreita) [40] conforme ilustra a Figura 1 .

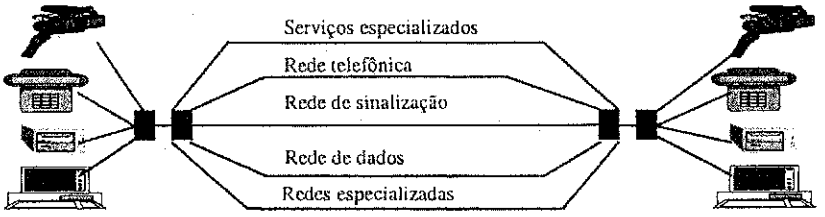


Figura 1 - Rede Digital de Serviços Integrados Faixa Estreita

A evolução da RDSI-FE levou ao surgimento da RDSI-FL (Rede Digital de Serviços Integrados - Faixa Larga), onde não apenas o acesso é integrado, mas, sobretudo, existe uma única rede de transporte comum, conforme ilustra a Figura 2.

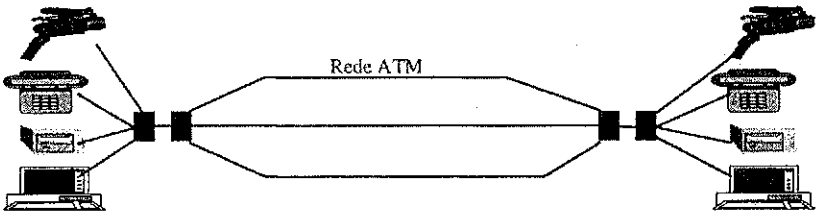


Figura 2 - Rede Digital de Serviços Integrados Faixa Larga

A existência de padrões universais, ou seja, padrões aceitos internacionalmente, constitui o ponto chave para a implementação de redes digitais de serviços integrados em todo o mundo, a exemplo do que é hoje a rede telefônica. Esses padrões são elementos fundamentais para a compatibilidade entre equipamentos de diferentes fabricantes. Neste sentido existem esforços de padronização da RDSI-FL em, basicamente, dois órgãos: o ITU-T e o Fórum ATM.

O ITU-T (*International Telecommunication Union Telecommunication Standardization Sector*) é um órgão permanente da *International Telecommunication Union* (ITU), com finalidade de emitir recomendações que favoreçam a padronização mundial das telecomunicações em questões técnicas, operacionais e tarifárias. Foi criado em março de 1993, substituindo o antigo CCITT (Comitê Consultivo Internacional de Telegrafia e Telefonia), onde as primeiras recomendações relativas a RDSI-FL foram aprovadas.

O Fórum ATM (*ATM Fórum*), na verdade, não é um órgão de padronização, mas trabalha em cooperação com órgãos de padronização como o ITU-T, tendo como objetivo principal a aceleração da instalação de produtos e serviços ATM, através da cooperação do setor industrial. O Fórum ATM conta com membros de vários setores tais como: fabricantes de equipamentos para redes de computadores, empresas de telecomunicações, fabricantes de semicondutores, instituições de pesquisas e universidades. Foi formado em outubro de 1991 e em junho de 1992, divulgou a sua primeira especificação da Interface Usuário - Rede (UNI), que contém informações sobre os serviços básicos ATM, opções de interface na camada física, gerenciamento de rede local e gerenciamento de tráfego.

A recomendação I.121 do ITU-T [41] define como características principais da RDSI-FL:

- Modo de transferência utilizado : Modo de Transferência Assíncrono (ATM - *Asynchronous Transfer Mode*)
- Suporte a conexões comutadas, permanentes e semi-permanentes
- Serviços fornecidos sob demanda, reservados e permanentes
- Suporte a serviços modo circuito ou modo pacote

- Suporte a serviços orientados ou não a conexão
- Configurações unidirecionais e bidirecionais

2.1.1 Serviços da RDSI-FL

Tendo sido concebida principalmente com o propósito de fornecer uma interface comum para dar suporte a tráfego de diversas naturezas [42], a RDSI-FL deve lidar com serviços já bem conhecidos, como é o caso da telefonia e TV a cabo, por exemplo, onde as características de tráfego e demanda já se encontram consolidadas, bem como lidar com novos serviços, onde não existe conhecimento desses parâmetros, fazendo com que nenhum serviço em particular esteja associado à RDSI-FL.

A recomendação I.211 do ITU-T classifica os serviços da RDSI-FL em quatro categorias [43]: serviços conversacionais, serviços de consulta, serviços de mensagens e serviços de distribuição.

Os serviços conversacionais são aqueles que asseguram uma comunicação bidirecional entre usuários, transferindo informações fim a fim em tempo real, sem armazenamento e retransmissão. Como aplicações deste tipo de serviço podem ser citados: videotelefonia, videoconferência, segurança, transmissão de dados em tempo real, etc.

Os serviços de consulta são aqueles que oferecem acesso às informações armazenadas em banco de dados remotos. Como aplicações deste tipo de serviços podem ser citados: videotexto, livrarias eletrônicas, vídeo sob demanda, etc.

Os serviços de mensagens são aqueles que oferecem comunicação entre usuários, através de unidades de armazenamento, com possibilidade

de retransmissão, mas não em tempo real, como por exemplo: correio eletrônico, correio de documentos multimídia, etc.

Os serviços de distribuição se caracterizam pelo fluxo de informação de um ponto para múltiplos pontos e dependendo ou não da intervenção do usuário podem ser divididos em dois tipos básicos: os serviços de distribuição sem controle do usuário e os serviços de distribuição com controle do usuário. Nos serviços sem controle do usuário, o início e a ordem das informações não podem ser controladas pelo mesmo, ao contrário dos serviços com controle do usuário, onde existe a possibilidade de seleção de itens, fornecidos na transmissão, onde o usuário pode, através de seleção desses itens, efetuar um controle das informações, tais como o início e a ordem da apresentação das mesmas. A Tabela 1 ilustra as categorias de serviços associadas a alguns exemplos de aplicações.

Diversos parâmetros tais como: taxa de chamada, taxa média de transmissão, taxa máxima de transmissão e duração da chamada, podem ser usados para caracterizar o tipo de serviço. Esses parâmetros variam de acordo com a natureza do tráfego gerado pela fonte do tráfego, que podem ser de três tipos: tráfego constante (CBR - *Constant Bit Rate*), tráfego de rajadas e tráfego variável (VBR - *Variable Bit Rate*).

O tráfego constante apresenta uma periodicidade na transmissão da informação, ou seja, sua taxa média é igual a sua taxa de pico, sendo esse parâmetro o único necessário para a caracterização desse tipo de tráfego.

SERVIÇO	APLICAÇÃO
Serviços conversacionais	Videotelefonia Videoconferência Supercomputação virtual Serviços de segurança
Serviços de consulta	Videotexto Livrarias eletrônicas Vídeo sob demanda
Serviços de mensagens	Correio eletrônico Manipulação de mensagens
Serviços de distribuição	1. *Sem controle do usuário* Distribuição de áudio Distribuição de vídeo Distribuição de documentos Difusão de TV 2. *Com controle do usuário* Substituição de documentos

Tabela 1 - Serviços da RDSI-FL

No tráfego de rajadas a informação é transmitida durante períodos, denominados períodos ativos, nos quais há uma grande geração de informações, trafegando na sua taxa de pico e períodos de silêncio, ou inativos, onde não existe tráfego. Parâmetros comumente utilizados para caracterizar esse tipo de tráfego incluem a duração média dos períodos de atividade e a explosividade (*burstiness*) da fonte, definida como a razão entre a taxa de pico e a taxa média de utilização do canal.

O tráfego variável apresenta taxas de transmissão variáveis ao longo do tempo. Os parâmetros de explosividade, a média e a variância da taxa de transmissão são utilizados para caracterizar esse tipo de tráfego.

2.1.2 Configuração de Referência da RDSI-FL

A recomendação I.413 do ITU-T [44] define os elementos presentes no ambiente do usuário, de forma a possibilitar o acesso integrado aos vários serviços oferecidos pela RDSI-FL. Essa definição está presente no esquema denominado: Configuração de Referência da RDSI-FL, composto pelos elementos: grupos funcionais e pontos de referência, conforme a Figura 3.

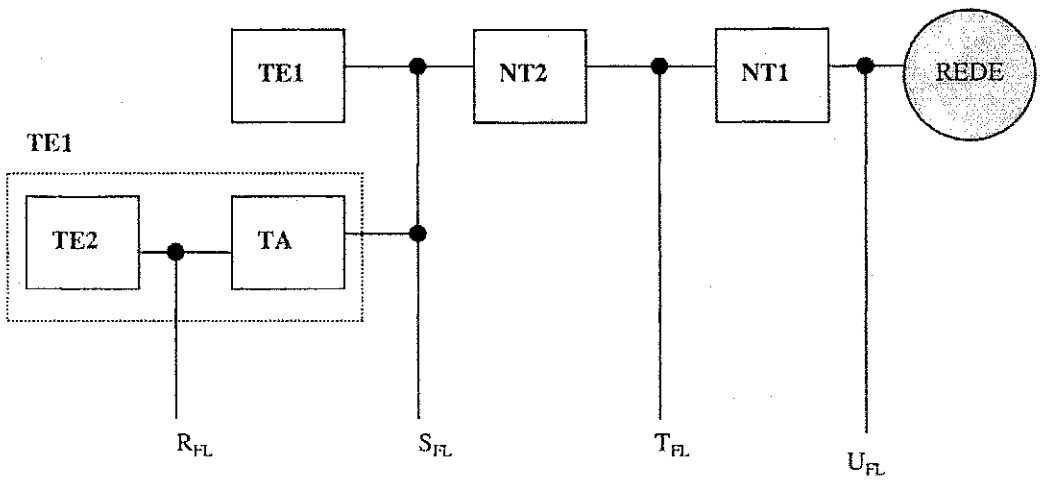


Figura 3 - Configuração de Referência da RDSI-FL

O grupo funcional TE1 (*Terminal Equipment 1*) compreende equipamentos compatíveis com a RDSI-FL, como por exemplo: telefones digitais, terminais de voz, terminais de dados, etc.

O grupo funcional TE2 (*Terminal Equipment 2*) compreende os equipamentos que não são diretamente compatíveis com a RDSI-FL e ne-

cessitam de um adaptador para se conectar à rede, como por exemplo: terminais seriais RS232, RS422, etc. Esses adaptadores (TA - *Terminal Adapters*) realizam a interface entre esse grupo funcional e a rede, tornando o conjunto {TE2, TA} funcionalmente equivalente ao grupo TE1.

O grupo funcional NT1 (*Network Termination 1*) realiza a interface entre o usuário e a rede, isolando o ambiente do usuário do meio de transmissão.

O grupo NT2 (*Network Termination 2*) é responsável pela concentração e comutação local.

As interfaces entre os grupos funcionais são definidas pelos pontos de referência de interface, denominados R_{FL}, S_{FL}, T_{FL} e U_{FL}. Diversas arquiteturas de ambiente do usuário podem ser construídas a partir da configuração de referência básica, combinando as funções definidas por um ou mais grupos funcionais.

Na especificação da interface usuário-rede (UNI), o Fórum ATM define duas formas de UNI : a UNI pública, que identifica a interface entre um usuário e o comutador ATM de uma rede pública e a UNI privada, que identifica a interface entre um usuário e o comutador ATM de uma rede privada (corporativa). Os pontos de referência T_{FL} e U_{FL} dizem respeito à UNI pública, enquanto o ponto de referência S_{FL} diz respeito à UNI privada. A Figura 4 ilustra formas de acessos possíveis para a conexão dos usuários com a rede, a NNI (*Network Node Interface*) define a interface entre nós de rede.

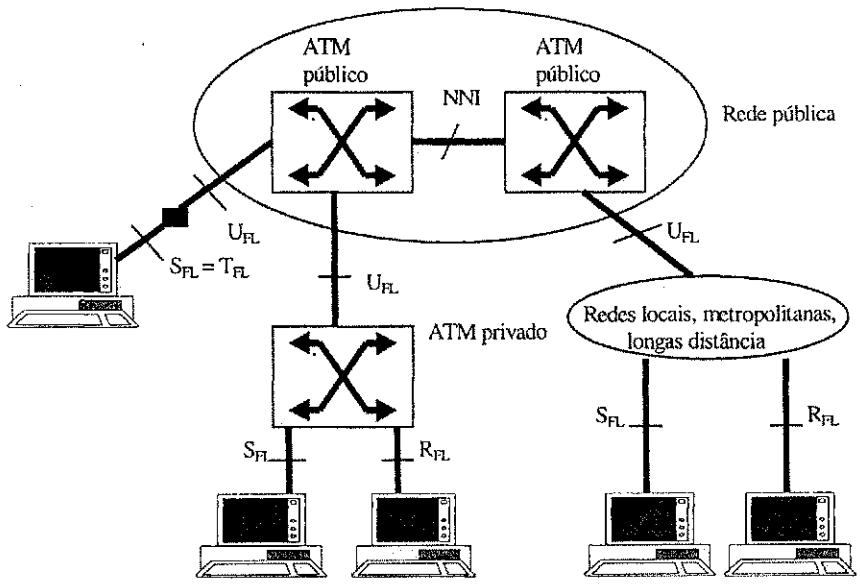


Figura 4 - Formas de acesso a Redes ATM

2.1.3 Modo de Transferência Assíncrono

O Modo de Transferência diz respeito aos aspectos que abrangem a transmissão, multiplexação e comutação da rede. Existem basicamente dois modos de transferência: O Modo de Transferência Síncrono (STM - *Synchronous Transfer Mode*) e o Modo de Transferência Assíncrono (ATM - *Asyncronous Tranfer Mode*)

O Modo de Transferência Síncrono é baseado no conceito de comutação por circuitos e na multiplexação por divisão de tempo síncrona, onde a capacidade total de um canal de transmissão é alocada periodicamente a cada um dos subcanais que o utilizam. A alocação de intervalos de tempo a subcanais é fixa e periódica.

O Modo de Transferência Assíncrono é baseado no conceito de comutação de pacotes de comprimento fixo e na multiplexação por divisão

de tempo assíncrona, onde não há alocação fixa de intervalos de tempo a subcanais (conexões). A ocupação é feita sob demanda de acordo com o tráfego de cada conexão, sendo cada canal identificado através de um rótulo no cabeçalho.

A maior vantagem do ATM em relação ao STM é a flexibilidade em acomodar serviços que requeiram taxas de transmissão variáveis durante a conexão, ou mesmo serviços com taxa constante mas que seja apenas uma fração da capacidade dos canais disponíveis na rede STM. Devido a esta característica o ATM foi escolhido para a RDSI-FL.

2.1.4 Célula ATM

Os pacotes de comprimento fixo do Modo de Transferência Assíncrono são denominados células. A recomendação I.361 do ITU-T [45], especifica o seu formato, que possui um comprimento de 53 bytes, sendo os 5 primeiros de cabeçalho e os 48 restantes de campo de informação, o *payload*, conforme ilustra a Figura 5.

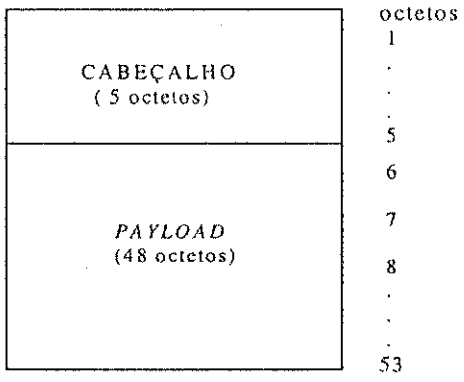


Figura 5 - Estrutura da Célula ATM

O cabeçalho da célula é formado por campos distintos e possui formatos ligeiramente diferentes para a interface usuário-rede (UNI) [46] e rede-rede (NNI), conforme ilustra a Figura 6.

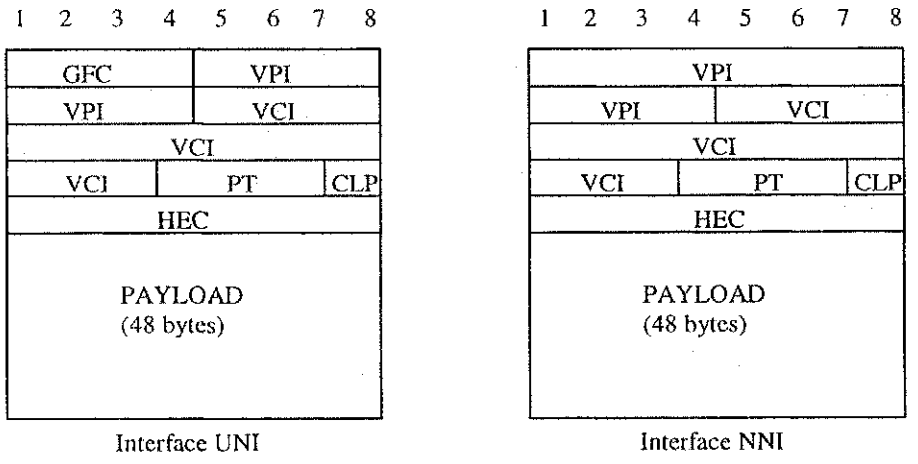


Figura 6 - Formato do cabeçalho da Célula ATM

O campo GFC (*Generic Flow Control*), de 4 bits, foi introduzido com a finalidade de regular o fluxo de tráfego numa rede ATM e até o momento não foi padronizada a sua utilização. O campo PT (*Payload Type*), de 3 bits, especifica o tipo de informação contido no *payload* da célula, ou seja, se o *payload* contém dados de usuário ou informações de gerenciamento do sistema. O campo CLP (*Cell Loss Priority*), de 1 bit, especifica o nível de prioridade da célula, identificando as de alta prioridade (CLP = 0) e as de baixa prioridade (CLP = 1). Essa identificação possibilita o descarte das células de baixa prioridade, quando a rede estiver em situação de congestionamento. O campo HEC (*Header Error Control*), de 8 bits, permite a delimitação da célula dentro do fluxo de bytes recebidos e a verificação da integridade do cabeçalho da mesma.

Sendo o Modo de Transferência Assíncrono orientado a conexão, é necessário se estabelecer uma ligação entre os parceiros da comunicação antes da transmissão das informações propriamente ditas. Neste estabelecimento prévio são especificados os endereços dos parceiros e associados identificadores para a conexão estabelecida. Os campos VPI (*Virtual Path Identifier*), de 8 ou 12 bits, e VCI (*Virtual Channel Identifier*), de 16 bits, são usados como identificadores da conexão. Na interface NNI não existe o campo GFC, como conseqüência o campo VPI possui, para essa interface, um comprimento de 12 bits, conforme ilustra a Figura 6.

2.1.5 Modelo de Referência dos Protocolos das RDSI-FL

O modelo de referência dos protocolos da RDSI-FL (definido pela recomendação I.321 do ITU-T) [47] é composto por três planos: plano do usuário, plano de controle e plano de gerenciamento, conforme ilustra a Figura 7.

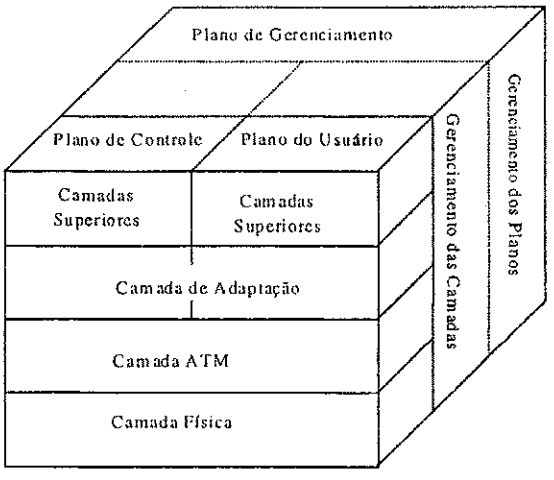


Figura 7 - Modelo de Referência dos Protocolos da RDSI-FL

O plano do usuário é responsável pela transferência de informações dos usuários. O plano de controle trata da sinalização necessária para ativar, manter e desativar conexões. O plano de gerenciamento é responsável por dois tipos de funções: gerência dos planos e gerência das camadas. A gerência dos planos trata do gerenciamento entre os três planos. A gerência das camadas é estruturada em camadas de protocolos que tratam dos fluxos de informações de operação, administração e manutenção (OAM) relativos a cada camada, incluindo o gerenciamento de recursos e parâmetros associados às entidades de protocolo.

O Plano de Usuário é dividido em três camadas: Camada Física, Camada ATM e a Camada de Adaptação ao ATM (AAL). As camadas física e ATM são comuns ao plano de controle e ao plano de usuário.

A Camada Física tem como função básica, fornecer à Camada ATM uma interface independente do meio físico de transmissão, tratando dos aspectos básicos da transmissão dos bits, como por exemplo, a codificação, alinhamento, recepção e conversão eletro-ótica dos bits que trafegam no meio físico. Também cabe à Camada Física a execução de funções para a identificação do início e do final de cada célula, verificação de erros no cabeçalho da célula, geração do campo HEC, entre outras.

A Camada ATM lida com o transporte das células para todos os tipos de serviços, sejam eles orientados ou não a conexão, com taxa de transmissão constante ou variável. Ela também trata do estabelecimento e da liberação de circuitos virtuais e do controle de congestionamento. A camada ATM será tratada com mais detalhes na seção 2.2.

A Camada de Adaptação ao ATM (AAL) [48] utiliza os serviços de transporte da camada ATM para oferecer ao usuário da rede ATM serviços

com requisitos específicos, isto é, converte o formato de serviço do usuário em células ATM, na transmissão e células ATM no formato de serviço desejado pelo usuário, na recepção. A camada AAL é implementada nas estações dos usuários que estão conectados à rede ATM.

2.2 Camada ATM

Um comutador ATM pode ser entendido como um conjunto de portas de entrada e saída, associadas cada uma às linhas físicas da rede. A comutação da célula ATM é uma função aplicada sobre cada célula que chega em uma determinada porta de entrada do comutador e corresponde à retransmissão desta célula através de uma outra porta de saída do comutador [49].

Uma conexão ATM, denominada "conexão de canal virtual" (*Virtual Channel Connection - VCC*) é formada pela concatenação de conexões virtuais estabelecidas nos vários enlaces da rede denominadas "enlace de canal virtual" (*Virtual Channel Link - VCL*), formando um caminho único através do qual as células serão encaminhadas. A VCL é identificada em cada comutador através de dois campos presentes no cabeçalho da célula, o campo VPI (*Virtual Path Identifier*) e do campo VCI (*Virtual Channel Identifier*). O campo VPI especifica o "caminho virtual" no qual está localizado a conexão virtual, enquanto o campo VCI identifica o "canal virtual" do "caminho virtual" especificado. A Figura 8 ilustra este conceito.

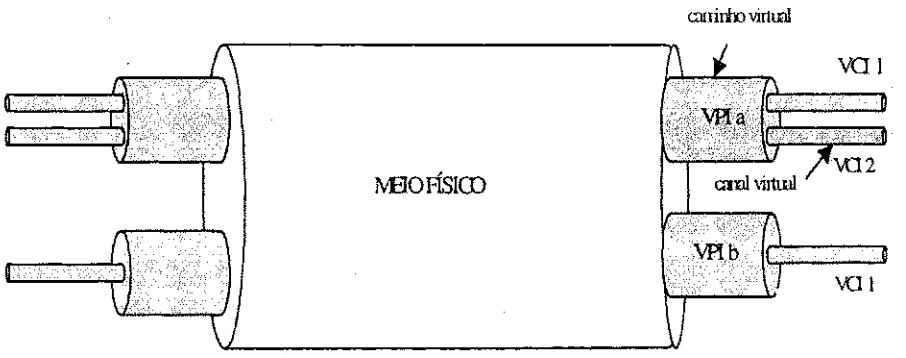


Figura 8 - Caminho e Canal Virtual

A Figura 9 ilustra uma conexão ATM estabelecida entre duas estações X e Y, formada por quatro VCLs (VCL1, VCL2, VCL3 e VCL4).

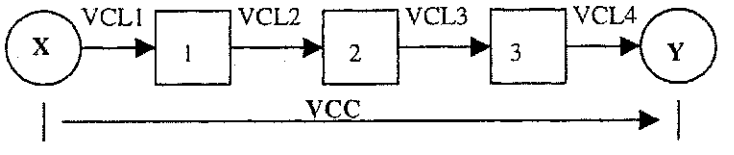


Figura 9 - Conexão ATM

Quando a estação X solicita uma conexão para ligá-la à estação Y, a rede devolve o rótulo VCL1 a ser utilizado para transmitir as informações e devolve à estação Y o rótulo VCL4 que identifica, para Y, o VCL desta conexão.

Cada conexão está associada a um conjunto de parâmetros que caracterizam o tipo de tráfego contratado pelo usuário, como por exemplo: tráfego *unicast* (células que possuem um único destino), tráfego *multicast* (células que possuem vários destinos), taxa máxima de transmissão, taxa média de transmissão, tipo de prioridade da conexão, etc. Estes parâmetros são usados na comutação e pelos mecanismos de policiamento implemen-

tados na rede, com o fim de identificar um usuário que desrespeita o que foi contratado, visando o controle de congestionamentos.

No encaminhamento das células da origem para o destino, cada comutador existente ao longo do caminho deve possuir informações sobre cada canal virtual que chega às suas entradas; essas informações estão contidas na chamada "Tabela de Rotas". As células que chegam através de um determinado VCL devem ser encaminhadas ao próximo comutador com um novo VCL, e assim por diante até o destino. Cada comutador possui a sua própria "Tabela de Rotas" que relaciona cada VCL de entrada ao próximo VCL e porta de saída a ser utilizada. O cabeçalho da célula (os campos VPI e VCI) identifica a posição dos dados referentes ao VCL em questão, dentro da tabela. O cabeçalho é atualizado para identificar o novo VCL e a célula é retransmitida pela porta de saída especificada na tabela. O processo é ilustrado na Figura 10.

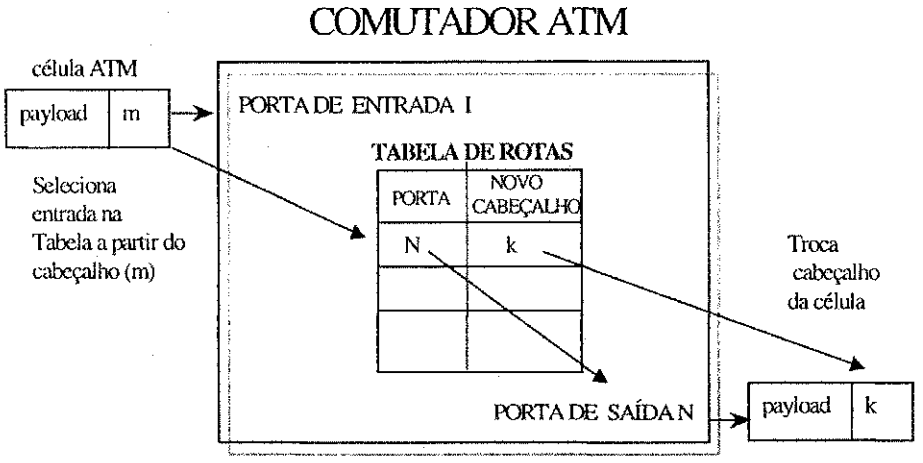


Figura 10 - Comutação da célula ATM

Como o VPI é apenas uma parte do rótulo da célula utilizado para o seu encaminhamento, existem dois modos de comutação: a comutação de

caminho virtual ou comutação de VP e a comutação de canal virtual ou comutação de VC, conforme ilustra a Figura 11.

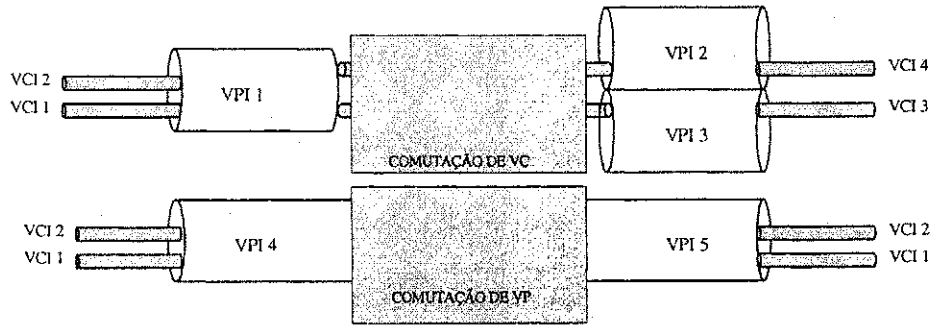


Figura 11 - Comutação de VP e de VC

Na comutação de VP apenas o VPI de entrada é trocado, todos os VCIs daquela VPI são comutados para um outro VPI, é o que ocorre com o VPI 4 da Figura 11, comutado no novo caminho VPI 5. A comutação de VC é baseada no rótulo completo (VPI + VCI), ou seja, são alterados tanto o VPI como o VCI. O canal de entrada VCI 2, do caminho VPI 1 é comutado como VCI 4 no novo caminho VPI 2, enquanto o canal VCI 1 é comutado como VCI 3 no novo caminho VPI 3.

2.2.1 Funções da Camada ATM

A Camada ATM lida com a movimentação de células da origem para o destino, executando processamentos sobre os campos do cabeçalho da célula para o controle dessa movimentação. A recomendação I.150 do ITU-T [50] especifica as funções executadas pela Camada ATM e entre as quais podem ser citadas:

- Remoção do cabeçalho da célula

- Uso das informações do cabeçalho para a identificação da conexão à qual a célula pertence.
- Consulta à tabela que contém informações sobre a conexão.
- Processamento de algoritmos de policiamento da conexão.
- Remontagem do cabeçalho da célula com novos valores dos campos VCI e VPI e endereço da porta de saída para onde se destina a célula.
- Encaminhamento da célula baseada nas novas informações do cabeçalho.

2.2.2 Interface com Camada Física- Padrão Utopia

A fim de padronizar a transferência de células entre a Camada ATM e a Camada Física, foi proposto pelo Fórum ATM uma interface denominada UTOPIA (*Universal Test & Operations PHY Interface for Atm*), que estabelece um protocolo padrão de sinais, visando controlar a forma de transferência das células entre essas duas camadas[51].

O padrão especifica dois modos de transferência de dados entre as Camadas Física e ATM : modo 8 bits e modo 16 bits. O modo 8 bits define a transferência de dados em palavras de 8 bits, ou seja, a célula é transferida por byte, a partir do *Header 1* e no formato indicado na Tabela 2. Este modo de transferência assume que a camada física processa o campo HEC, de acordo com as especificações da recomendação I.432 do ITU-T [52], que inclui como função da camada, entre outras, a geração do campo HEC e delimitação das células. Dessa forma, o campo UDF (*User Defined Field*), presente no formato de transferência modo 8 bits, transporta o campo

HEC. O padrão recomenda o uso do modo de transferência em 8 bits, para taxas de transmissão de células entre 100 Mbps a 155 Mbps, usando o relógio das interfaces (transmissor/receptor) com freqüência máxima de 25 MHz.

Header 1
Header 2
Header 3
Header 4
UDF
Payload 1
:
Payload 48

Tabela 2 - Formato de transferência da célula ATM no modo 8 bits

O modo 16 bits define a transferência de dados em palavras de 16 bits, ou seja, a célula é transferida em um barramento de 16 bits, dois bytes simultaneamente, com o formato indicado na Tabela 3.

Header 1	**Header 2**
Header 3	**Header 4**
UDF1	**UDF2**
Payload 1	**Payload 2**
:	:
Payload 47	**Payload 48**

Tabela 3 - Formato de transferência da célula ATM no modo 16 bits

A especificação recomenda o uso deste modo de transferência para taxas de transmissão de células de 155 Mbps a 622 Mbps. A especificação

define, ainda, duas interfaces independentes entre as camadas ATM e Física: interface de transmissão e interface de recepção, conforme a Figura 12.

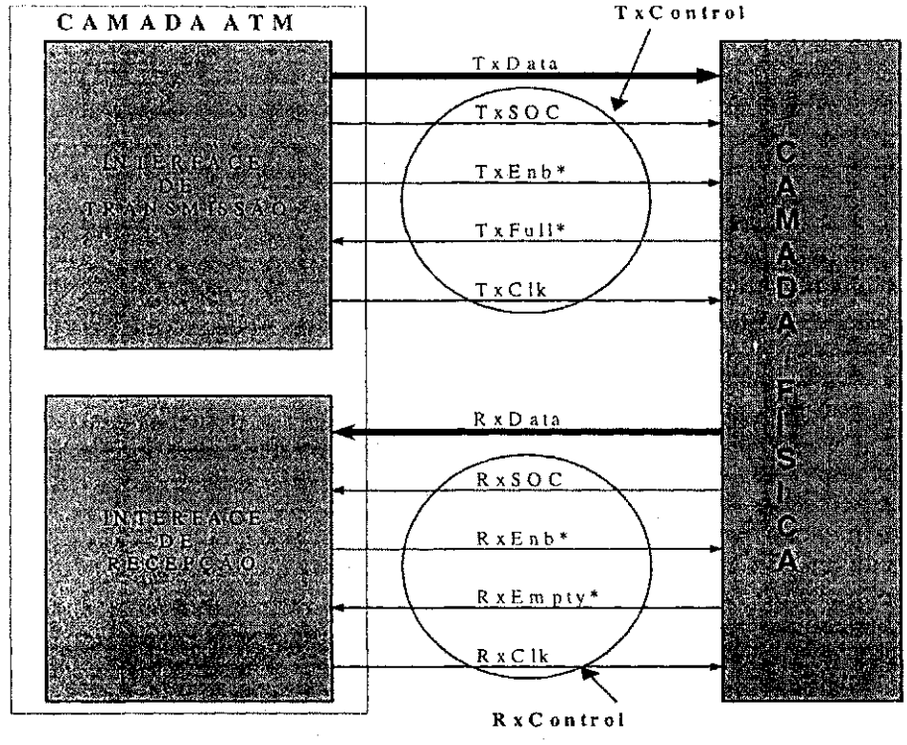


Figura 12 - Interfaces de Transmissão e Recepção

A interface de transmissão é composta por um barramento unidirecional de 8 ou 16 bits (TxData) onde trafegam os dados oriundos da camada ATM, sob controle dos sinais do conjunto TxControl.

A interface de recepção é composta por um barramento unidirecional de 8 ou 16 bits(RxData) onde trafegam os dados oriundos da camada física, sob controle dos sinais do conjunto RxControl.

A transferência de dados entre as camadas é efetuada pelas interfaces, independentes entre si e controladas pelos respectivos sinais, conforme indicam as Tabela 4 e Tabela 5.

SINAL	SIGNIFICADO	OBSERVAÇÃO
TxData[7..0]	Barramento unidirecional da camada ATM para a camada física, que contém os dados a serem transferidos. TxData[7] é o MSB	Para o modo 16 bits, o barramento é definido como: TxData[15..0]
TxSOC	*Start of Cell.* Sinal ativo alto, enviado pela camada ATM, que indica a presença do primeiro byte válido da célula a ser transportada	Para o modo 16 bits, indica a primeira palavra de 16 bits
TxEnb*	*Enable.* Sinal ativo baixo, enviado pela camada ATM, indicando a presença de dados válidos no barramento TxData	
TxFull*	*Full* Sinal ativo baixo, enviado pela camada física, indicando que um máximo de 4 bytes serão aceitos pela camada física	Para o modo 16 bits, indica que uma célula completa será aceita pela camada física(TxClav - Cell flow available)
TxClk	*Clock* de transferência. Sinal de clock gerado pela camada ATM para o sincronismo de transferência dos dados presentes no barramento TxData[7..0]	Transferência efetuada na borda de subida do clock
TxPrty[0]	*Parity.* Indica a paridade (ímpar) do barramento TxData[7..0]. Enviado pela camada ATM	Sinal opcional. Para o modo 16 bits, é definido um segundo sinal TxPrty[1], que indica a paridade do barramento TxData[15..8]
TxRef*	*Transmit Reference.* Sinal enviado pela camada ATM para propósitos de sincronismo (Ex: 8 KHz marker, indicador de frame, etc.)	Sinal opcional

Tabela 4 - Sinais da Interface de Transmissão

SINAL	SIGNIFICADO	OBSERVAÇÃO
RxData[7..0]	Barramento unidirecional da camada física para a camada ATM, que contém os dados a serem transferidos. RxData[7] é o MSB	Para o modo 16 bits, o barramento é definido como: RxData[15..0]
RxSOC	*Start of Cell.* Sinal ativo alto, enviado pela camada física, que indica a presença do primeiro byte válido da célula a ser transportada	Para o modo 16 bits, indica a primeira palavra de 16 bits
RxEnb*	*Enable.* Sinal ativo baixo, enviado pela camada ATM, indicando que os sinais RxData e RxSOC serão amostrados no próximo ciclo	
RxEmpty*	*Empty.* Sinal ativo baixo, enviado pela camada física, indicando que no ciclo atual, não existe byte a ser enviado à camada ATM	Para o modo 16 bits, substitui RxEmpty*, o sinal RxClav (Cell Available), ativo alto, indicando a existência de uma célula completa disponível para transferência
RxClk	*Clock* de transferência. Sinal de clock gerado pela camada ATM para o sincronismo de transferência dos dados presentes no barramento RxData[7..0]	Transferência efetuada na borda de subida do clock
RxPrty[0]	*Parity.* Indica a paridade (ímpar) do barramento RxData[7..0]. Enviado pela camada física	Sinal opcional. Para o modo 16 bits, é definido um segundo sinal RxPrty[1], que indica a paridade do barramento RxData[15..8]
RxRef*	*Receive Reference.* Sinal enviado pela camada física para propósitos de sincronismo	Sinal opcional

Tabela 5 - Sinais da Interface de Recepção

2.3 Resumo

Este capítulo apresentou a evolução das redes de telecomunicações que levou ao surgimento de uma nova tecnologia denominada ATM, baseada fundamentalmente na transmissão de pequenos pacotes de informação, denominados células. As principais características dessa tecnologia foram apresentadas, com ênfase na abordagem da Camada ATM do modelo de referência de protocolos da Rede Digital de Serviços Integrados de Faixa Larga (RDSI-FL), utilizada no desenvolvimento deste trabalho. Os conceitos básicos dessa tecnologia permitirão um entendimento maior das funções de um Comutador ATM, assunto apresentado no próximo capítulo.

Capítulo

3

Comutadores ATM

Este capítulo apresenta a arquitetura geral de um Comutador ATM, descrevendo as funcionalidades de cada bloco específico. Apresenta ainda alguns mecanismos de policiamento utilizados em Comutadores ATM, dando ênfase ao Algoritmo Genérico de Controle de Taxa (GCRA – Generic Cell Rate Algorithm); relata os problemas inerentes aos Comutadores ATM e enfoca algumas implementações existentes. Finalmente é abordado o estado da arte dividido em três problemas centrais: Arquiteturas do Elemento Comutador, Gerenciamento e controle das células nas portas de saída e Processamento das funções da Camada ATM, sendo este último o tema focalizado no presente trabalho.

3.1 Função básica de um Comutador ATM

A função básica de um Comutador ATM é a transferência de blocos de informação (contendo dados do usuário, sinalização, controle ou de manutenção) das portas de entrada para as portas de saída (comutação propriamente dita) [53], [54], [55]. Para a execução desta função básica, o comutador deve possuir elementos específicos, conforme ilustra a Figura 13.

As células trafegam no meio físico como uma seqüência de bits e são agrupadas em bytes pela Interface de Entrada (IE) da porta de entrada correspondente, que executa funcionalidades inerentes à Camada Física como a delimitação das células dentro da seqüência de bits e verificação de erros no cabeçalho da célula, por exemplo. Analogamente, as células nas portas de saída são transformadas em uma seqüência de bits e entregues ao meio físico.

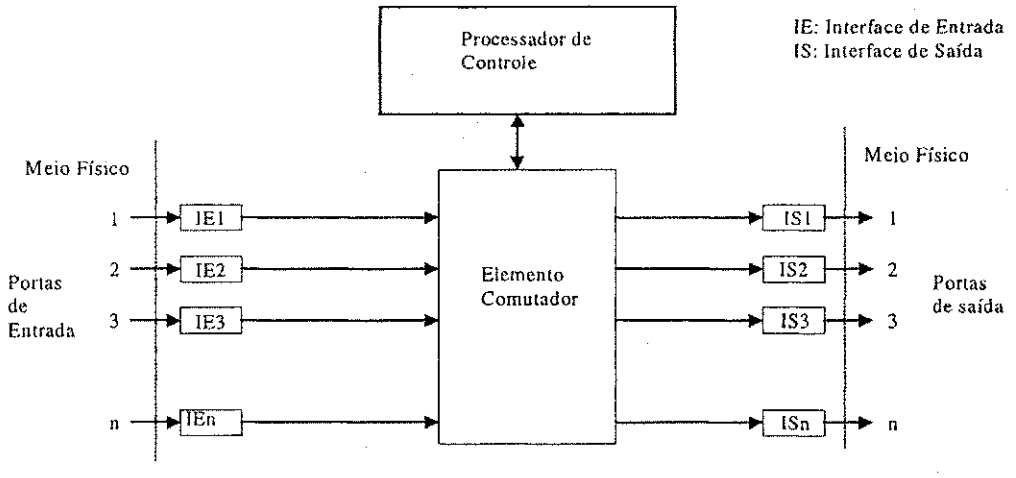


Figura 13 - Arquitetura Geral de um Comutador ATM

3.2 O Elemento Comutador

O elemento comutador é responsável pela implementação da comutação física, transferindo informações das portas de entrada para as portas de saída e a sua arquitetura pode ser classificada em dois tipos básicos: arquitetura baseada em divisão de tempo e arquitetura baseada em divisão de espaço [56] como mostra a Figura 14.

Nas arquiteturas baseadas em divisão de tempo, todas as células fluem através de uma única via de comunicação compartilhada por todas as portas de entrada e saída. Essa via compartilhada pode ser um meio

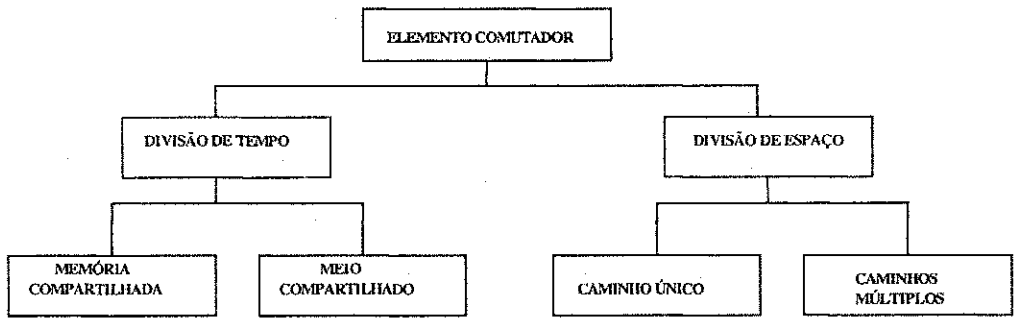


Figura 14 - Classificação de Arquiteturas de Elemento Comutador ATM

compartilhado, como um barramento, ou uma memória compartilhada, como mostra a Figura 15. A vazão da via de comunicação compartilhada define a capacidade máxima de comutação desse elemento comutador.

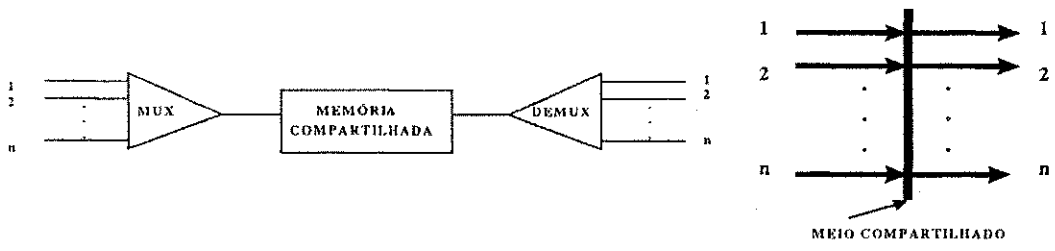


Figura 15 - Elemento comutador baseado em divisão de tempo

Elementos comutadores baseadas em divisão espacial podem ser de dois tipos: os que possuem um único caminho entre qualquer par de portas de entrada e saída, denominados "elementos com caminho único" e os que possuem mais de um caminho entre pares de entrada e saída, denominados "elementos com caminhos múltiplos" [57], [58], [59], [60]. A Figura 16 mostra um elemento comutador com caminho único. A complexidade deste tipo de elemento comutador cresce com o número de portas, devido ao aumento do número de caminhos entre as portas.

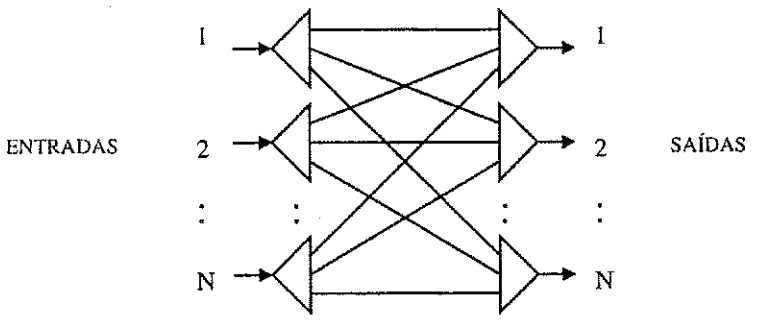


Figura 16 - Elemento comutador com caminho único

A Figura 17 mostra um elemento comutador com caminhos múltiplos formado pela interligação de elementos básicos de 2 entradas e 2 saídas (2x2). Entre uma porta de entrada e uma de saída existe mais de um caminho possível, como ilustra a figura para dois possíveis caminhos entre a porta de entrada 1 e a porta de saída 2. O primeiro caminho é formado através dos elementos básicos A, F, G H, E, sendo o segundo através dos elementos básicos A, B, C, D, E.

Os elementos comutadores com caminhos múltiplos são utilizados para facilitar a construção de comutadores com maior número de portas, já

que possuem um menor número de ligações entre as portas de entradas e saídas se comparados aos de caminho único. Entretanto, uma vez que vários caminhos existem, introduz-se a possibilidade de conflitos entre células solicitando os mesmos caminhos tornando-se necessário um controle adicional para a resolução dos conflitos, aumentando a sua complexidade. Conflitos em elementos internos ao comutador são denominados "bloqueios".

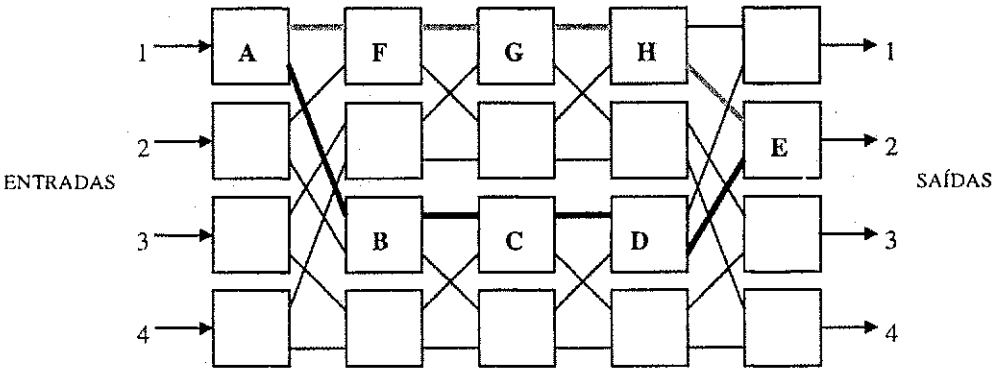


Figura 17 - Elemento comutador com caminhos múltiplos

A escolha entre os projetos de comutadores ATM, baseados em elementos comutadores com divisão espacial ou divisão de tempo, considera requisitos de velocidade de memória, vazão de barramento e bloqueios. Por exemplo, a velocidade de acesso à memória e vazão de barramento são problemas típicos dos Comutadores ATM baseados em divisão de tempo, limitando o número máximo de portas que o comutador pode ter, enquanto bloqueios internos são problemas típicos de comutadores baseados em divisão espacial, limitando a velocidade do comutador.

3.3 O Processador de Controle

O processador de controle é responsável pelos procedimentos de controle da comutação, como, por exemplo: leitura dos parâmetros da tabela de rotas referentes à célula a ser roteada, manutenção dos dados da tabela, monitoração de utilização do comutador (policiamento da conexão) e geração de estatísticas da comutação (número de células roteadas, número de células descartadas, etc.).

Para a execução das suas funções, o processador de controle deve interagir com uma tabela, denominada tabela de rotas, que corresponde a uma memória local, interna ou não ao processador, cuja finalidade é armazenar informações referentes a cada conexão ativa. Entre estas informações estão os valores **novo VPI**, **novo VCI** e **endereço da porta de saída** (usados na remontagem do novo cabeçalho da célula); estão também os parâmetros referentes às especificações de tráfego da conexão, usados nos algoritmo de policiamento das conexões e dados resultantes do monitoramento do tráfego, como número de células roteadas, numero de células descartadas, etc. O tamanho máximo da tabela depende do número de conexões possíveis no comutador.

O processador de controle acessa os dados referentes a uma determinada célula de entrada (que pertence a uma determinada conexão), usando os campos **VPI** e **VCI** do cabeçalho da célula, como endereço. Após a verificação da obediência às especificações de tráfego contratadas (ver seção 2.2), a célula terá o seu cabeçalho remontado com novos valores de VPI e VCI e anexada a informação da porta de saída à qual se destina (endereço da porta de saída), para que possa ser finalmente comutada pelo elemento comutador. Cabe também ao processador de controle gerar esta-

tísticas da comutação, atualizando a tabela com informações referentes ao número de células roteadas e número de células descartadas pelo policiamento. A Figura 18 representa a interação entre o Processador de Controle e a Tabela de Rotas.

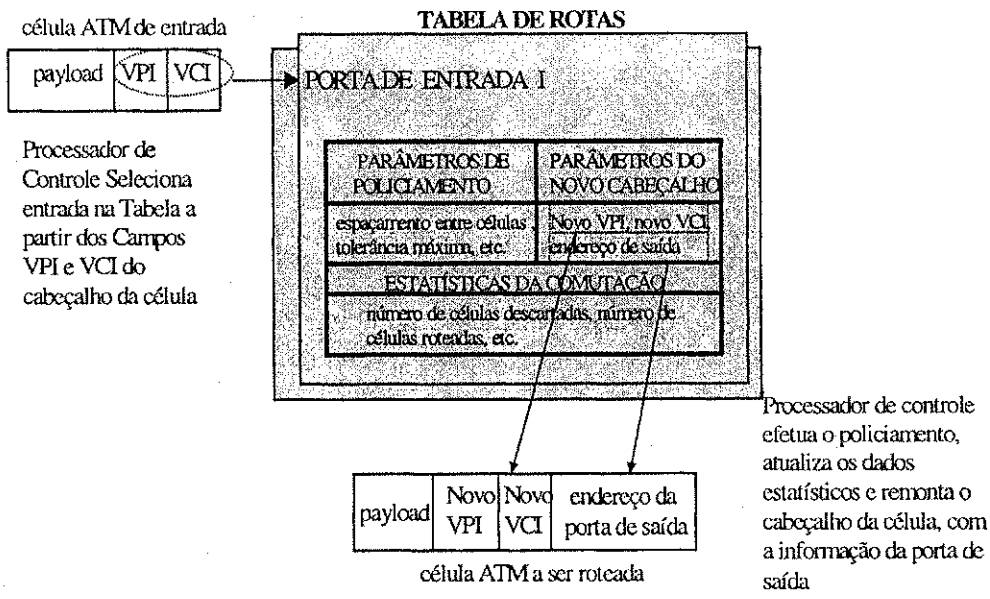


Figura 18 - Interação Processador de Controle e Tabela de Rotas

3.4 Mecanismos de Policiamento em Comutadores ATM

O congestionamento, de uma maneira geral, diz respeito à degradação sofrida pelo fluxo de tráfego de um sistema causada por uma excessiva solicitação de seus recursos. No caso de redes ATM, o congestionamento ocorre quando o número de células em trânsito é superior ao máximo acomodável.

A fim de garantir um desempenho adequado da rede para todos os usuários e prevenir que um usuário específico degrade a qualidade dos serviços prestados aos demais, são necessários mecanismos de controle de

congestionamento [61], [62], [63], [64], [65], [66], classificados como preventivos e reativos pelo ITU-T. Conforme o próprio nome indica, os controles, classificados como preventivos tentam prevenir a ocorrência de congestionamento, enquanto os reativos, reagem à sua existência.

O controle de congestionamento, denominado policiamento, é classificado como sendo, ao mesmo tempo, preventivo e reativo [54] e tem como função básica assegurar que sejam respeitadas pelos usuários as especificações de tráfego negociadas no estabelecimento da conexão, de forma a evitar e corrigir eventuais congestionamentos na rede.

Entre os critérios utilizados para comparar a eficácia dos mecanismos de policiamento, podem ser citados: Transparência, Tempo de Reação e Complexidade de implementação[54].

O critério da Transparência considera o efeito que o mecanismo exerce sobre as conexões que respeitam as especificações negociadas, causado pela identificação errada de células que obedecem as especificações como sendo células que não obedecem as especificações e vice-versa.

A Conformidade com o mecanismo ideal considera o comportamento de um mecanismo real em relação à probabilidade de rejeição devido à variação da taxa média de células emitidas pela fonte monitorada [67].

O Tempo de Reação considera o tempo que o mecanismo leva para detectar as conexões que violam os parâmetros negociados.

A Complexidade de implementação diz respeito ao número de elementos de *hardware* (comparadores, contadores, registradores, etc.) requisitados para a implementação do mecanismo.

A Tabela 6 resume os resultados das comparações feitas com quatro tipos de mecanismos de policiamento encontrados na literatura: "Janelas Saltitantes", "Janelas Deslizantes", "Contadores de Pico" e "Balde Furado" (*Leaky Bucket)* [67], [68], [69].

Mecanismo	Conformidade	Transparência	Reação	Complexidade
Janela Saltitante	Boa	Ruim	Bom	Excelente
Janela Deslizante	Boa	Ruim	Bom	Excelente
Contador de Pico	Ruim	Excelente	Ruim	Ruim
"Balde Furado"	Excelente	Ruim	Bom	Excelente

Tabela 6 - Comparação entre mecanismos de policiamento

Na maioria dos critérios o mecanismo "Balde Furado" é superior aos demais, como indica a Tabela 6.

Este algoritmo pode ser melhor compreendido usando-se a seguinte analogia: imagine um "balde" com um pequeno furo na parte de baixo, conforme ilustra a Figura 19.

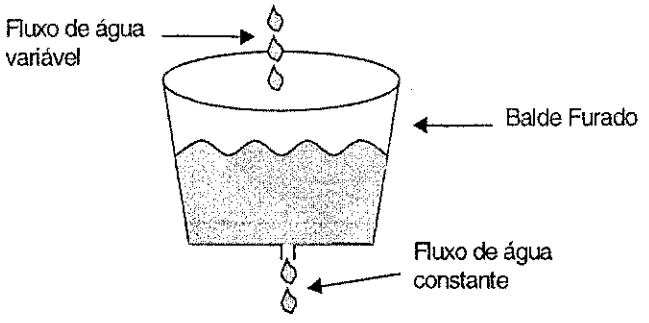


Figura 19 - O "Balde Furado"

O fluxo de saída está sempre a uma taxa constante, independente da velocidade com que a água entra no balde. Quando o balde está cheio, a água

que entra nele transborda e é perdida, ou seja, não aparece no fluxo de saída sob o furo.

A mesma idéia pode ser aplicada às células de uma conexão que está sendo policiada. Um contador ("balde") possui um valor máximo de contagem (N) sendo decrementado a uma taxa constante T (se o valor é maior que zero). Toda célula pertencente a mesma conexão policiada que chega no comutador incrementa o contador e, se encontrar o contador com o valor máximo (N), será considerada uma célula excessiva (que desrespeita as especificações contratadas).

A recomendação I.371 do ITU-T propõe um algoritmo de referência para o controle de policiamento de conexões em Redes ATM, denominado Algoritmo Genérico de Controle de Taxa (GCRA – *Generic Cell Rate Algorithm*) [69], [70], usando o algoritmo "Balde Furado", (Figura 20). O algoritmo utiliza as seguintes variáveis:

X – valor do contador no instante de chegada da última célula bem comportada, ou seja, a célula de uma dada conexão que não viola as especificações de tráfego contratadas no estabelecimento da conexão.

LCT – (*Last Conformance Time*), instante de chegada da última célula bem comportada.

I – (Incremento), corresponde ao espaçamento ideal entre células bem comportadas, ou seja, o intervalo de tempo entre duas células da mesma conexão, que determina a taxa de transmissão contratada no estabelecimento da conexão.

L – (Limite de antecipação), tolerância admitida na violação do espaçamento ideal entre as células.

$t_{a(k)}$ – instante de chegada da célula a ser policiada.

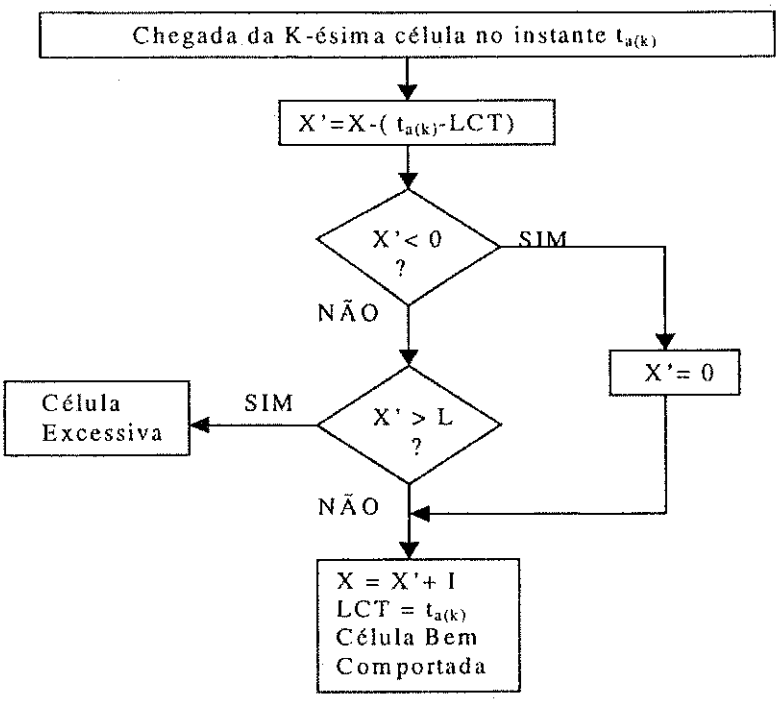


Figura 20 - Algoritmo Genérico de Controle de Taxa com o "Balde Furado"

O valor máximo do contador corresponde a I+L e o "balde" é esvaziado na razão de uma unidade de conteúdo por unidade de tempo. Sempre que uma célula é aceita, o "balde" é enchido com uma quantidade I; quando o "balde" transborda (o que equivale a X>L), a célula viola a especificação de tráfego e é rejeitada; caso contrário, a célula é aceita e os valores LCT e X devem ser atualizados para verificação da célula seguinte. Os valores de I e L são definidos na negociação com o usuário.

3.5 Outras características dos Comutadores ATM

A arquitetura de um Comutador ATM deve ser **flexível** em diversos aspectos para acomodar a diversidade de requisitos dos serviços atuais e de serviços futuros, de forma a suportar qualquer tipo de tráfego e substituir qualquer outra rede dedicada, segundo a proposta da RDSI-FL.

Uma outra característica importante é a capacidade de expansão ou **escalabilidade**, entendida como a habilidade de manusear uma grande quantidade de portas a partir da interligação de elementos básicos com poucas portas de entrada e saída.

O comutador ATM deve possuir, também, capacidade para suportar tráfegos multidestinos (*multicast*), implementada na própria arquitetura ou através de módulos adicionais que multiplicam as mensagens de acordo com o número de portas distintas pelas quais elas devem ser enviadas [71], [72], [73].

Finalmente, devido às altas taxas de transmissão de células envolvidas (centenas de megabits por segundo), as arquiteturas que se propõem a implementar comutadores ATM, devem conter blocos que operem em alta velocidade, para a execução de suas funcionalidades. Por exemplo, o processador de controle deve executar todos os acessos à tabela de rotas, algoritmos de policiamento da célula, cálculos das estatísticas de comutação e remontagem do novo cabeçalho em um tempo inferior a 2,75 μs (tempo mínimo entre células sucessivas que chegam ao comutador na taxa de 155 Mbps) ou 680 ns para taxas de transmissão de células de 622 Mbps.

3.6 Problemas inerentes aos Comutadores ATM

A manipulação de conexões *multicast* é fundamental para as aplicações de multimídia tais como; videoconferência, áudio, vídeo comercial, etc. Os comutadores que usam um elemento comutador baseado em divisão espacial, possuem grande dificuldade para implementar conexões *multicast*, muitos deles geram várias cópias da célula e roteiam cada uma das cópias para a sua porta de destino, o que ocasiona aumento de tráfego interno e possível perda de desempenho. Os comutadores baseados em divisão de tempo fazem o *multicast* sem nenhuma dificuldade, apenas utilizando o meio compartilhado sem gerar cópia de célula e sem perda de desempenho [16], [19], [69], [74], [75], [76], [77].

Por outro lado, devido às limitações tecnológicas das velocidades das memórias e vazão de barramentos, os comutadores baseados em divisão de tempo possuem limites de tecnologia de *hardware* para atingir grandes tamanhos (escalabilidade), ao contrário daqueles baseados em divisão de espaço, que teoricamente não teriam tais restrições, a menos das restrições físicas tais como: número de pinos dos dispositivos, conectores e considerações de sincronismo, que limitam o tamanho do elemento comutador.

Finalmente, a contenção nas portas de saída é um problema inerente a qualquer comutador. Ocorre quando várias células chegam de diferentes portas de entrada, cada uma delas requisitando a mesma porta de saída. A porta de saída só pode transmitir uma célula de cada vez, assim as outras ou são descartadas, ou armazenadas. A solução técnica mais utilizada é o armazenamento na porta de saída, sendo cada porta responsável pelo monitoramento da sua própria fila [78], [79].

3.7 Comutadores ATM comerciais

A indústria de comutadores ATM enfoca soluções voltadas para atender as necessidades de grandes empresas e provedores de serviços, notadamente no que diz respeito a **escalabilidade , velocidade** e **suporte a *multicast***

As arquiteturas apresentam, em sua maioria, malhas de interconexão de solução proprietária com elemento comutador trabalhando na ordem de gigabits por segundo, a fim de suportar a escalabilidade e evitar a contenção interna. As velocidades das portas de saída se situam entre 10 a 622 Mbps e se faz sentir uma maior obediência a padrões e recomendações de órgãos como o ITU-T e Fórum ATM, através da enumeração desses padrões nas especificações técnicas dos produtos [41], [43], [44], [51], [70], [80], [81].

A *Fore Systems* apresenta um modelo de comutador ATM, denominado *ForeRunner ASX-4000* [82], cuja principal característica é possuir flexibilidade para atender as necessidades crescentes do cliente em termos de vazão e número de portas, garantindo a escalabilidade. O sistema é composto por portas de entrada e saída que trabalham entre 10 a 622 Mbps, suportando até 40 Gbps de tráfego de entrada com o uso de até 4 elementos comutadores de 10 Gbps, conforme ilustra o diagrama da Figura 21.

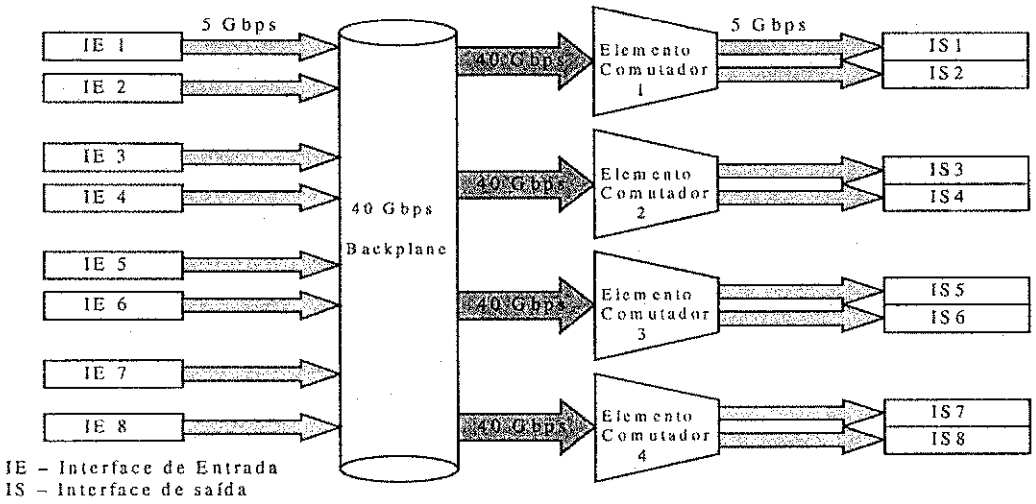


Figura 21 - Arquitetura do *ForeRunner ASX-4000*

O sistema pode ser montado com a quantidade de portas de entrada e saída e elementos de comutação, de forma a atender as necessidades reais do cliente. A Tabela 7 apresenta algumas especificações técnicas deste comutador.

Elemento comutador	Malha de interconexão não bloqueante com 10, 20, 30 ou 40 Gbps
Número de portas	1 a 64
Velocidade das portas	155 Mbps e 622Mbps
Mecanismo de policiamento	GCRA com "Balde Furado"
Multicast	Sim
Buffer de saída	Até 128 K células por porta
Obediência a padrões e normas	AF-PHY 0046000 do Fórum ATM Recomendação I.432 do ITU-T Recomendação I.371 do ITU-T UNI 3.0/3.1/4.0 do Fórum ATM

Tabela 7 - Especificações técnicas do *ForeRunner ASX-4000* da *Fore System*

A *Bay Networks* [83], empresa especializada em sistemas para redes locais, apresenta um modelo denominado *Centillion 50*, conforme as especificações da Tabela 8.

Elemento comutador	Malha de interconexão não bloqueante com 10 Gbps
Número de portas	3 para 622 Mbps 12 para 155 Mbps
Velocidade das portas	155 Mbps e 622Mbps
Mecanismo de policiamento	Não informado
Multicast	Não implementado
Buffer de saída	Não informado
Obediência a padrões e normas	*LAN Emulation V.1* –Fórum ATM *Private NNI* – Fórum ATM

Tabela 8 - Especificações técnicas do *Centillion 50* da *Bay Networks*

A *Alcatel Data Networks* apresenta o modelo *1100 HSS* para redes a longas distância, de acordo com a Tabela 9.

Elemento comutador	Malha de interconexão 10 Gbps
Número de portas	Máximo de 128 portas
Velocidade das portas	622 Mbps
Mecanismo de policiamento	Não informado
Multicast	Não implementado
Buffer de saída	Não informado
Obediência a padrões e normas	*ITU-T*, Fórum ATM

Tabela 9 - Especificações técnicas do *1100 HSS* da *Alcatel Data Networks*

3.8 Estado da arte das arquiteturas de Comutadores ATM

Muitos trabalhos sobre Comutadores ATM, existentes na literatura, apresentam soluções para três problemas centrais. As Arquiteturas do Elemento

Comutador, o Gerenciamento e Controle das células nas portas de saída e o Processamento das Funções relacionadas com a Camada ATM.

3.8.1 Arquiteturas do Elemento Comutador

Chao et al. [9] descreve um elemento comutador, denominado *Abacus Switch*, com capacidade para *multicast* e expansão (escalabilidade), formado basicamente por uma malha central de interconexão (elemento comutador baseado em divisão espacial), chamado MGN (*multicast grouping network*), seguido de pequenas malhas interligadas a conjuntos de portas de saída chamadas SSMs (*small switch modules*), conforme ilustra a Figura 22.

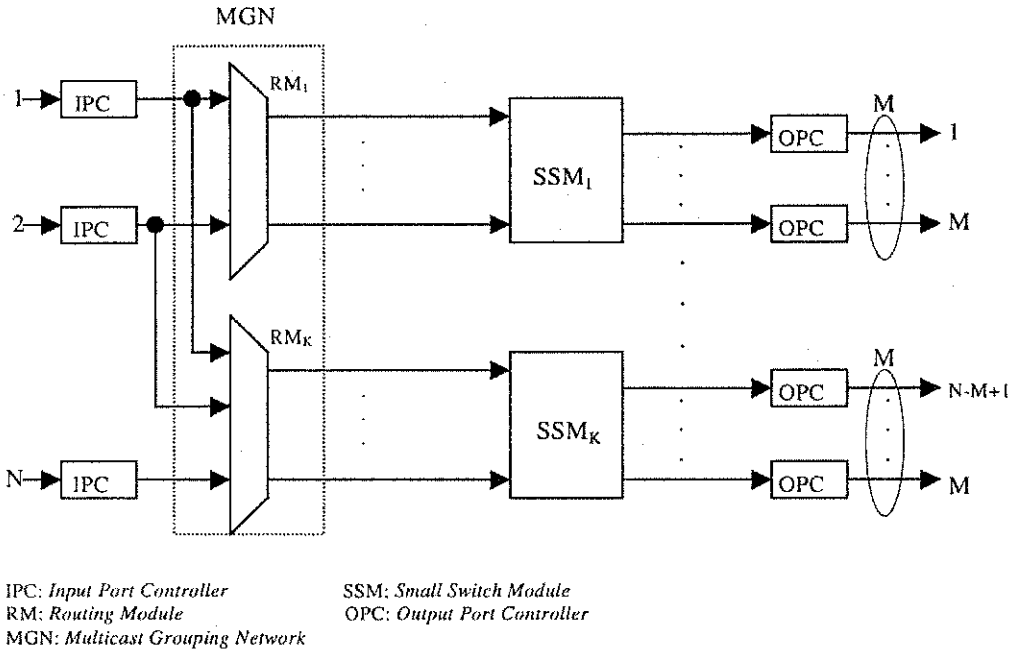


Figura 22 – Arquitetura do *Abacus Switch*

O bloco MGN é uma malha de interconexão formada pela interligação de pequenos blocos básicos agrupados, de forma a atingir um ele-

mento comutador do porte desejado. As saídas desta malha principal são interligadas às entradas dos módulos SSM, que funcionam como pequenas malhas de interconexão ligando suas entradas a um grupo de portas de saída onde estão conectados. O uso desses módulos facilita a implementação da função de *multicast*.

As células, que trafegam no elemento comutador proposto, necessitam de informações adicionais que devem ser enviadas antes do cabeçalho da célula ATM que será roteada, tais como: tipo de operação a ser executada (*multicast* ou não), endereço da porta de saída (ou portas de saída caso seja operação de *multicast*), prioridade da célula a ser usada em caso de bloqueio interno, etc.

No caso de operação *multicast*, a célula será enviada para o módulo SSM interligado às portas de destino, que executa o roteamento desta célula para as portas desejadas. A arquitetura proposta foi implementada em um ASIC com 32x32 elementos comutadores apresentando as seguintes características:

Tecnologia CMOS 0,8 μm - 3 níveis de metal	
Área do *core*	49,56 mm^2
No. de transistores	81000
No. de pinos funcionais	145
Freqüência máxima	240 MHz

Tabela 10 - Resultados da implementação do *Abacus Switch*

Yamanaka et al. [8] apresenta uma arquitetura de elemento comutador baseado em memória compartilhada (com memória de único acesso), tendo como principal característica a estrutura em funil (*funnel structure*) onde o

número de portas de entrada do elemento comutador é maior que o número de portas de saída.

A memória compartilhada é dividida em estruturas menores denominadas *shared memory buffer* (SBM), conectados às portas de entrada e portas de saída através de duas malhas de interconexão específicas, conforme ilustra a Figura 23. As células de entrada são armazenadas nos SBMs e transferidas para as portas de saída correspondentes; como o número de entradas (**m**) é maior que o número de saídas (**n**), o número de acessos máximo que cada bloco SBM está sujeito é (**m+n**) por ciclo de célula, ou seja, cada SBM poderá ser acessado **n** vezes no período de uma célula, totalizando **kn** acessos a todo comutador em funil, sendo **k** o número de SBMs. A relação $\mathbf{m+n \leq kn}$ é a condição que torna a arquitetura "não bloqueante". Com $\mathbf{k = 1}$, o comutador torna-se um comutador com memória compartilhada convencional, mas com $\mathbf{k > 1}$ a sua escalabilidade torna-se melhor, entretanto a complexidade do controle dos módulos de memória cresce com o aumento dos SBMs.

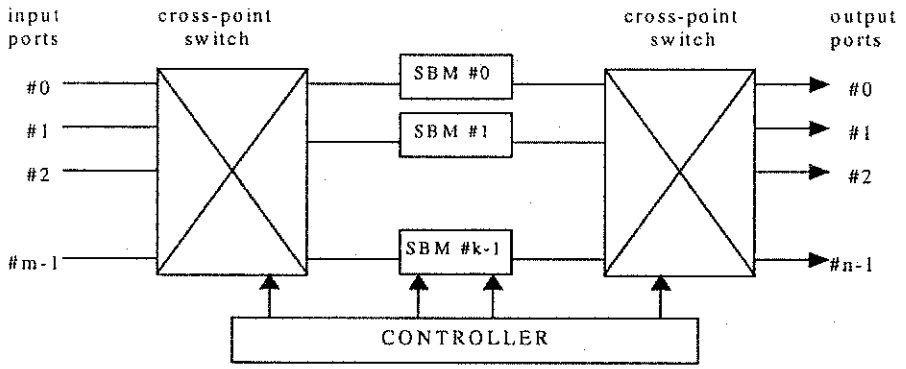


Figura 23 - Arquitetura multi-buffer compartilhado

A implementação da arquitetura para 32 entradas e 8 saídas, resultou em dois ASICS, denominados **BX-LSI** e **CX-LSI**, com as seguintes características:

BX-LSI	**Contém os módulos SBMs e realiza as funções de acesso aos SBMs**
Tecnologia CMOS 0,5 µm	
Área do *chip*	222,75 mm^2
No. de células	174000
No. de pinos funcionais	473
Freqüência máxima	155,52 MHz
Total de RAM (SBMs)	344 k-bits

Tabela 11 - Resultados da implementação do BX-LSI

CX-LSI	**Controla o BX-LSI, gerenciando os acessos aos SBMs**
Tecnologia CMOS 0,5 µm	
Área do *chip*	222,75 mm^2
No. de células	115000
No. de pinos funcionais	473
Freqüência máxima	155,52 MHz
Total de RAM implementada	132 k-bits

Tabela 12 - Resultados da implementação do CX-LSI

3.8.2 Gerenciamento e controle das células nas portas de saída

Chao et al.[15] apresenta uma arquitetura para gerenciamento das células nas portas de saída, denominada *Generalized Priority Queue Manager* (GPQM).

Essa arquitetura se fundamenta basicamente no cálculo e armazenamento de dois parâmetros que caracterizam a prioridade da célula. O primeiro parâmetro, denominado *due time* (T), representa o instante de chegada da célula na porta de saída correspondente, acrescido do tempo máximo tolerável de permanência no *buffer* antes de ser lida. O segundo parâmetro, denominado *virtual finishing time* (F), está associado a cada classe de serviço oferecida (classes I, II, III, e IV) e representa uma indicação da prioridade da célula dentro de cada classe.

A arquitetura do GPQM é constituída, basicamente, por uma memória de células (*cell mem*), dois microprocessadores (um para cálculo do parâmetro T e outro para cálculo do parâmetro F), uma área de *buffer* para armazenamento dos ponteiros associados às células em *cell mem* e quatro blocos denominados *Priority Content Addressable Memory* (PCAM), responsáveis pela determinação da prioridade das células em cada classe de serviço.

Quando uma célula chega ao GPQM, é armazenada na memória de células; paralelamente são calculados os parâmetros T e F da célula e guardados na área de *buffer,* juntamente com o ponteiro que identifica sua localização na memória.

Na transmissão para a porta de saída, cada PCAM, associado à sua classe de serviço, determina a célula que possui menor valor F, dessa forma quatro células disputarão a preferência de transmissão, sendo escolhida aquela que tiver o menor valor de T.

Hashemi et al. [14] introduz uma arquitetura que se propõe a gerenciar as células nas portas de saída de um Comutador ATM, podendo ser usada em conjunto com elementos comutadores baseados em memória compartilhada.

A arquitetura se interliga ao elemento comutador controlando a leitura das células para as respectivas portas de saída de acordo com o algoritmo de prioridade executado e se baseia em duas estruturas denominadas: *tagging unit* e *sequencer*. A *tagging unit* tem como função associar a cada célula informações que serão usadas como parâmetros pelo algoritmo de prioridade executado na unidade *sequencer*. Esta unidade armazena os ponteiros referentes às células armazenadas na memória compartilhada (elemento comutador) juntamente com as informações de prioridade oriundas da *tagging unit*.

A seqüência dos ponteiros armazenados é dividida em grupos, de tal forma que o primeiro grupo armazena os ponteiros que correspondem às células de maior prioridade que chegaram para as portas de saída mas ainda não foram lidas; o segundo grupo armazena todos os ponteiros que correspondem às células com prioridade inferior às do primeiro grupo que ainda não foram lidas e assim sucessivamente em ordem decrescente de prioridade. Cada célula que chega, destinada a uma determinada porta, tem seus parâmetros de prioridade comparados com os da célula do primeiro grupo. A célula vencedora será lida da memória e transmitida à porta. A célula perdedora terá seus parâmetros comparados com a do segundo grupo e assim sucessivamente, de forma que células vencedoras se deslocam para os grupos de maior prioridade e células perdedoras se deslocam para grupos de menor prioridade.

3.8.3 Processamento das funções da Camada ATM

Merayo et al. [26] apresenta um Comutador ATM cuja arquitetura se baseia essencialmente em dois blocos básicos denominados ICM (*switch core*) que

implementa as funções do elemento comutador e o bloco CMC (Cell to Microcell Converter) que executa as funções de controle do elemento comutador e funções relacionadas com a Camada ATM.

O bloco CMC recebe as células ATM de entrada (53 bytes), através de uma interface dedicada, realizando uma expansão da célula (de 53 bytes para 64 bytes, os bytes adicionais delimitam as fronteiras das células). A célula expandida é segmentada em palavras de 64 bits, o que constitui a chamada microcélula. Paralelamente os campos VPI e VCI são extraídos do cabeçalho e utilizados para acesso a uma memória externa que contém o endereço de roteamento da célula a ser passada para o ICM. O bloco não executa funções de policiamento, nem operação *multicast*. Possui também uma unidade de interface dedicada, que permite conexão com um microprocessador externo que executa funções auxiliares ao bloco CMC, como contagem do número de células roteadas, armazenamento de dados processados, etc. O CMC foi implementado em um ASIC, uma parte em CMOS e uma parte em ECL, como mostra a Tabela 13.

Tecnologia BiCMOS 0,7 μm – 2 níveis de metal	
Parte CMOS	**Parte ECL**
No. de transistores: 300.000	No. de transistores: 8300
Freqüência máxima: 74 MHz	Freqüência máxima: 311 MHz
No. de pinos funcionais: 319	

Tabela 13 - Resultados da implementação do bloco CMC

Katevenis et al. [21] apresenta um circuito integrado único, denominado ATLAS I (*single chip ATM switch*), que se propõe a executar as funções inerentes à camada ATM e ao Elemento Comutador para 16 portas de entrada/saída.

As células de entrada têm o *payload* armazenado em um *buffer* compartilhado e seu cabeçalho (campos VPI e VCI) utilizado para acesso à tabela de rotas interna (4K entradas), contendo o endereço da porta de saída e o parâmetro denominado classe de serviço, que classifica o tipo de células. Existem três tipos de classes de serviço associadas às células que trafegam no comutador, Alta, Média e Baixa. A Alta é relativa aos serviços em tempo real, a Média relativo a tráfego de dados tipo VBR (*Variable Bit Rate*) e finalmente a Baixa, relativo a tráfego de dados tipo CBR (*Constant Bit Rate*).O mecanismo de policiamento utilizado se baseia no conceito da existência de "crédito" para a emissão da célula, explicado mais adiante.

Associado a cada célula armazenada no *buffer* compartilhado, existe um ponteiro que indica o seu posicionamento no *buffer,* este ponteiro se localiza em uma das três estruturas denominadas *Ready Cell Queues* (RQ). Existe um RQ para serviços da classe Alta, um outro para a classe Média e um terceiro para a classe Baixa. As células que pertencem a classe Alta de serviço não são policiadas, as das outras classes só terão o ponteiro armazenados nos seus respectivos RQs se houver um "crédito" associado ao RQ correspondente. Uma outra estrutura denominada CT (*Credit Table*) armazena os "créditos" disponíveis por classe de serviço; cada célula roteada, incrementa o total de créditos da sua classe, da mesma forma que cada célula que tem o seu ponteiro armazenado no RQ correspondente decrementa o total de créditos deste RQ. A célula que ao chegar não encontrar crédito disponível no respectivo RQ, terá seu ponteiro armazenado em uma outra estrutura denominada *Creditless Cell List* (CLL) à espera de um futuro "crédito" disponível. O circuito não executa operação de *multicast*, não possui flexibilidade para reprogramação das suas funções e não executa polici-

amento da conexão conforme o conceito de serviço contratado. As características do ASIC apresentadas a seguir são baseadas em estimativas.

Tecnologia CMOS 0,5 μm	
Área do core	120 mm^2
Área do chip	225 mm^2
Freqüência máxima	50 MHz

Tabela 14 – Estimativa de implementação do *Atlas I*

Nakamura et al. [24] mostra a arquitetura de uma unidade de policiamento de tráfego de um comutador ATM (modelada em linguagem VHDL) que utiliza o algoritmo Genérico de Controle de Taxa (GCRA) versão "Balde Furado" (*Leaky Bucket*).

As células que devem ser policiadas não passam pela unidade, para que não haja alteração no fluxo de células nem introdução de atrasos. A unidade interage apenas com um processador externo, de onde recebe os parâmetros correspondentes à célula que será policiada (e que dependem do VPI/VCI da célula) e uma indicação para início do policiamento. Após a execução do algoritmo, a unidade retorna ao processador os valores atualizados dos parâmetros e a indicação do resultado do policiamento (se a célula violou ou não as especificações contratadas). A unidade trabalha com uma freqüência de 20 MHz e se faz necessário o uso de um processador externo para os devidos acessos à tabela de rotas onde estão contidos os parâmetros referentes à célula. A unidade proposta executa apenas o algoritmo de policiamento e repassa ao processador a informação de violação ou não das especificações contratadas, não possuindo capacidade para o descarte da célula; também não é dotada de flexibilidade para executar ou-

tro tipo de controle de policiamento nem atualizar a tabela de rotas com dados resultantes do policiamento efetuado.

Rathgeb et al. [22] propôs uma arquitetura de Comutador ATM com o objetivo principal de fornecer versatilidade, modularidade e escalabilidade no suporte a serviços.

O sistema é montado a partir de dois blocos principais: *Line Interface Circuit* (LIC) e o *ATM Switching Network* (ASN), combinados na quantidade e de forma a atender os requisitos dos serviços desejados. O Bloco ASN desempenha a função do elemento comutador e o bloco LIC é formado por um conjunto de circuitos que executam, entre outras, as funcionalidades da Camada Física, algumas funcionalidades da Camada ATM e funcionalidades da Camada de Adaptação ao ATM (AAL), conforme ilustra a Figura 24.

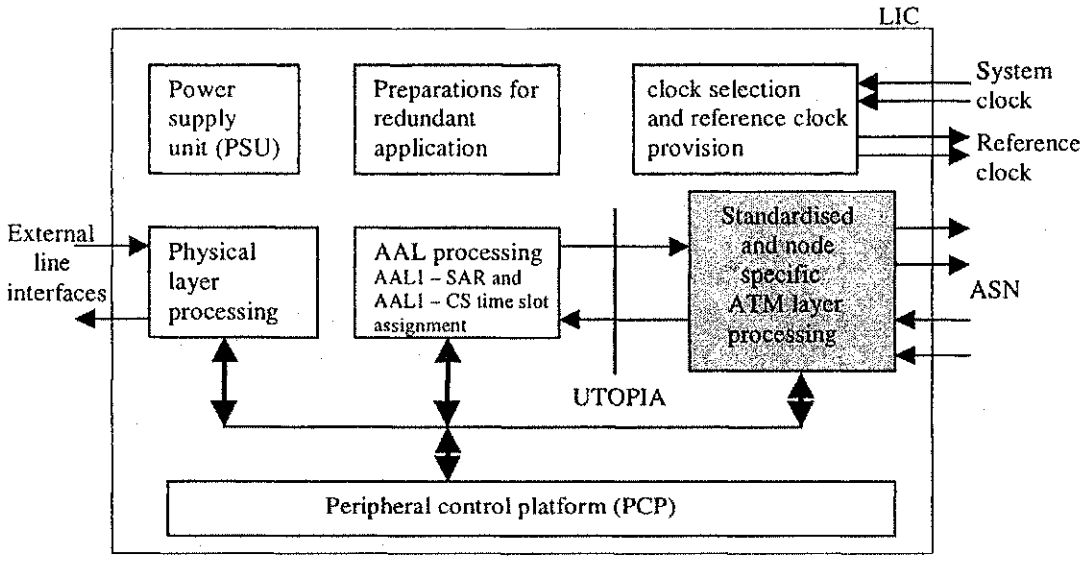


Figura 24 - Arquitetura do LIC *(Line Interface Circuit)*

Das funções da Camada ATM, são executadas as de translação de VPI/VCI, ou seja, a remontagem do cabeçalho da célula ATM com novos valores dos campos VPI/VCI, lidos de uma memória SRAM (tabela de rotas). Não é executado policiamento do fluxo de células. A leitura da memória é feita usando os campos VPI/VCI da célula como entrada numa memória associativa (*Content Addressable Memory* - CAM) resultando em um endereço de 13 bits para o acesso à memória, SRAM de 8 K, determinando o total de conexões simultâneas que podem ser suportadas (máximo de 8192 conexões). Como interface entre as Camadas Física e ATM, bem como entre as Camada AAL e ATM é utilizado o padrão UTOPIA.

Silva et al. [23] discute aspectos da implementação de um sistema comutador ATM e propõe uma arquitetura composta por um conjunto de ASICs, implementando as funções específicas de comutação ATM em torno de um circuito roteador, inicialmente projetado para interligação de processadores e que desempenha a função de elemento comutador, conforme a Figura 25.

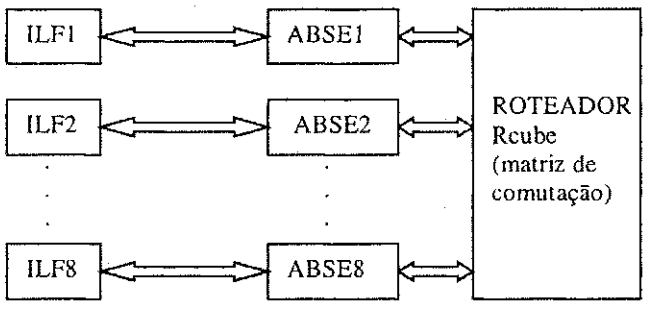


Figura 25 - Arquitetura do módulo ATM usando o roteador RCube

As funções de comutação (entre elas as funções relacionadas com a Camada ATM) estão concentradas em uma estrutura denominada ABSE (*ATM Basic Switching Element*), que realiza operações sobre uma célula ATM

de entrada; esta estrutura é composta por uma FIFO, um processador de controle, uma tabela de rotas e uma unidade de serialização. A FIFO armazena o cabeçalho da célula ATM enquanto o *payload* é armazenado em registradores pertencentes ao processador de controle.

Cabe a esse processador efetuar operações sobre o cabeçalho da célula, para separação dos campos usados como endereço no acesso à tabela de rotas que contém informações da porta de saída do comutador, do tipo de serviço a ser associado à célula, das novas informações dos campos de cabeçalho, etc. O processador também é responsável por uma seqüência de procedimentos necessários para compatibilização dos formatos de dados entre as células ATM e o formato aceito pelo elemento comutador. Estes procedimentos são necessários pois o elemento comutador foi concebido para interligação entre processadores e aceita formato de dados específico para tal. Pela mesma razão, uma unidade de serialização é requerida na arquitetura, tendo como causa a característica de trabalho do elemento comutador (entradas seriais). Não há execução de policiamento sobre as células.

Silva enfatiza que o número exato de ASICs que comporá o sistema só será definido após validação lógica do módulo completo e rigoroso estudo sobre o mapeamento dos blocos para as diferentes tecnologias.

Banniza et al. [25] relata as dificuldades existentes na implementação de um Comutador ATM e propõe um sistema, através da divisão das funcionalidades do comutador em placas e circuitos integrados dedicados.

O sistema é constituído por duas placas denominadas *Switch Module Board* (SM) e *Termination Link Board* (TLK). O SM executa as funções do elemento comutador e é composto por uma matriz de comutação de

16x16 associada a outros sub-blocos que executam multiplexação, demultiplexação e geração dos sinais de relógio necessários.

A placa TLK por sua vez desempenha a conversão serial-paralela das células ATM de entrada e paralela-serial das células ATM de saída, assim como as funções da Camada ATM.

As funções da Camada ATM são executadas por três diferentes circuitos integrados, como ilustra a Figura 26.

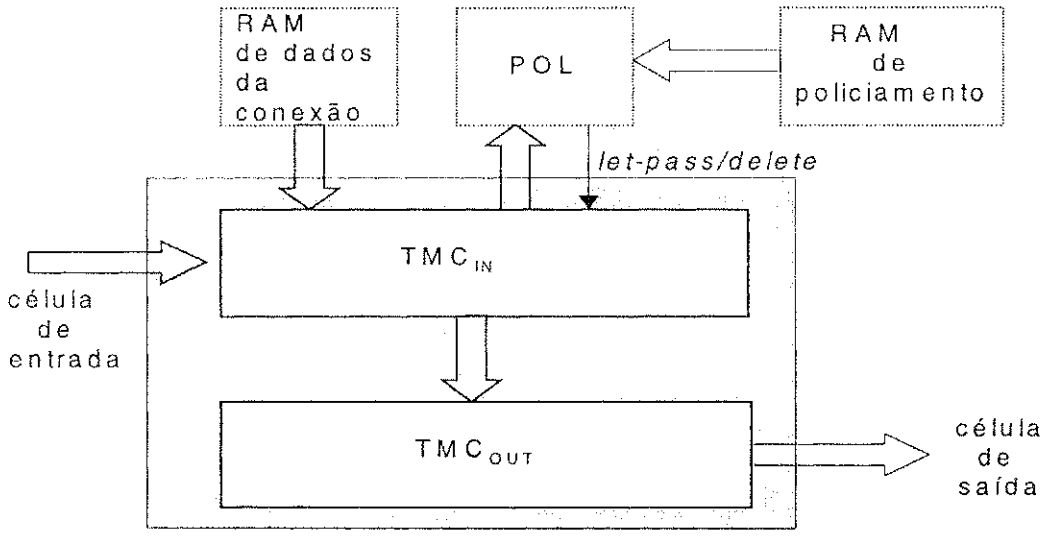


Figura 26 - Arquitetura da TLK (*Termination Link Board*)

O primeiro deles o TMC_{IN} recebe a célula ATM através de um barramento unidirecional de 8 bits, processa a separação dos campos VPI/VCI os quais são traduzidos em um índice que identifica a conexão virtual a qual a célula pertence. Este índice é utilizado para acessar os parâmetros referentes à conexão, contidos numa memória RAM externa.

O segundo circuito, POL (*Police*), executa as funções de policiamento da célula e opera como um coprocessador para o TMC_{IN}, recebendo

deste um parâmetro identificador da conexão a ser policiada e retornando a informação de sucesso ou não do policiamento, através de um sinal (*let-pass/delete*). Uma RAM externa (*POL RAM)* acessada pelo circuito contém as varáveis e parâmetros usados. Banizza enfatiza que a escolha de um circuito separado para esta função se deve à inexistência de padrões para o policiamento.

O terceiro circuito, TMC_{OUT}, executa a remontagem do cabeçalho da célula com os novos valores de VPI/VCI e outras informações referentes ao controle do elemento comutador. O circuito não executa operações *multicast* e não possui flexibilidade para reprogramação de suas funções a menos do policiamento.

As características principais estimadas dos ASICs propostos são enumeradas abaixo:

TMC_{IN}	**Tecnologia CMOS 0,8 µm**
No. de células	28000
Freqüência máxima	38,88MHz
TMC_{OUT}	**Tecnologia CMOS 0,8 µm**
No. de células	27000
Freqüência máxima	38,88MHz

Figura 27 - Resultados estimados dos ASICs propostos

3.9 Resumo

Este capítulo apresentou a arquitetura básica de um Comutador ATM, descrevendo as funcionalidades dos seus blocos principais, os problemas relativos à sua implementação e algumas soluções adotadas. O estado da arte

apresentado indica que as novas arquiteturas propostas se concentram em três pontos principais: o Elemento Comutador, o Gerenciamento das células nas portas de saída e o Processamento das funções da Camada ATM.

Muitos trabalhos sobre o Processamento das funções da Camada ATM apresentados na literatura, se caracterizam pela implementação dessas funções em um ou mais ASICs, desenvolvidos a partir de um conjunto próprio de especificações, não sendo flexíveis o suficiente para suportar mudanças ditadas por incorporações de novas padronizações, criação de novos serviços ou necessidade de otimização.

O próximo capítulo apresenta uma arquitetura denominada: Controlador Microprogramável para Comutadores ATM (CMCA), objeto deste trabalho, que executa o processamento das funções da Camada ATM, tendo como base uma unidade de controle microprogramada.

Capítulo

4

O Controlador Microprogramável para Comutadores ATM (CMCA)

Este capítulo apresenta os requisitos necessários para um sistema executar as funções da Camada ATM e introduz o conceito de unidade de controle microprogramada utilizada na arquitetura proposta. Apresenta ainda a definição da tabela de rotas, aqui denominada: Tabela de Roteamento, Gerenciamento e Sinalização (TRGS), que é a interface da arquitetura proposta como um sistema hospedeiro. Apresenta o fluxograma geral da Camada ATM que serviu como base para a concepção da arquitetura do CMCA, seu escalonamento temporal e a montagem e execução de um programa C^{++} a partir do fluxograma geral para avaliação em computadores com velocidade de 75 a 333 MHz. Mostra ainda a arquitetura geral de um Comutador ATM utilizando o controlador proposto. Finalmente, os blocos componentes do controlador são descritos, com detalhamento dos Módulos Microprocessadores seu conjunto de operações, os sinais de entrada e saída, o jogo e a estrutura das microinstruções.

4.1 Requisitos de um Sistema para execução das funções da Camada ATM

O projeto de um sistema que se propõe a executar as funcionalidades da Camada ATM envolve algumas definições preliminares:

1. Definição dos parâmetros de roteamento – deve haver uma prévia definição dos parâmetros que serão anexados ao novo cabeçalho da célula, como, por exemplo: o endereço da porta de saída da célula (que define o número de portas do comutador), a identificação do tipo de conexão (unicast, multicast), qualidade de serviço da conexão (QOS) [84], parâmetros para gerenciamento dos *buffers* de células nas portas de saída, parâmetros específicos a um novo tipo de serviço oferecido pelo comutador, definição de *flags* usados no algoritmo de roteamento, etc.
2. Definição do(s) algoritmo(s) de controle do fluxo das células – deve haver uma prévia escolha do tipo de policiamento que será executado, para que sejam definidos os parâmetros específicos (tamanho da palavra, quantidade, etc.).
3. Definição da tabela de rotas – deve haver uma definição da estrutura da tabela, no que diz respeito à localização e forma de acesso dos parâmetros descritos.

Além do mais, tal sistema deve ser rápido o suficiente para executar todas as funcionalidades referentes a uma determinada célula, antes da chegada da próxima (isto representa um tempo útil de 2,75 μs para taxas de transmissão de 155 Mbps e 680 ns para taxas de 622 Mbps).

A literatura relata diversas implementações de circuitos que executam as funções da Camada ATM [21], [74], [78], [85], [86], [87], [88], [89] a maioria delas resultam em um ou mais circuitos dedicados (ASICs), desenvolvidos a partir de um conjunto próprio de especificações, não sendo flexíveis o suficiente para suportar mudanças ditadas por incorporações de novas padronizações, criação de novos serviços ou necessidades de otimização. Para contemplar estas mudanças, tais arquiteturas, por incorporarem pouca ou nenhuma reprogramabilidade, devem ser modificadas, causando um novo ciclo de projeto e fabricação de novos ASICs, com os custos inerentes ao processo.

4.2 Unidade de controle convencional

A unidade de controle, dentro de uma UCP (Unidade Central de Processamento), dirige todas as atividades de hardware, controlando a busca, decodificação e execução da instrução, determinando desta forma o movimento dos dados, a fonte e o destino dos mesmos.

Os projetos convencionais de uma unidade de controle são feitos de maneira tal que uma malha lógica gera sinais de tempo e controle em uma seqüência pré-determinada para cada instrução em nível de máquina. Para cada instrução da máquina, existe uma determinada malha lógica que gera os sinais associados aos eventos que devem ocorrer em determinado tempo para executar a instrução específica e, uma vez definido o conjunto de instruções, toda mudança ou inclusão de nova instrução gera a necessidade de modificação na estrutura da malha ou inclusão de nova malha. A

lógica de controle é fixa, definida no projeto, para o conjunto de instruções que o sistema executa.

Unidades de controle convencionais são implementadas como Máquinas de Estados Finitos, as quais incluem estados para a busca, decodificação e execução das instruções. Em cada estado sinais são ativados numa seqüência preestabelecida de forma que a unidade de processamento manipule os dados de forma correta. Esta filosofia de projeto é denominada controle convencional ou *hardwired control.*.

4.3 Unidade de controle microprogramada

O conceito de microprogramação foi introduzido por **Wilkes** [90], [91] (professor do Laboratório de Matemática em Cambridge) em 1951, que propôs uma maneira sistemática e ordenada para o projeto de uma unidade de controle, observando que a execução de uma instrução de máquina requer movimentos de dados entre registradores que podem ser seqüenciais ou paralelos (simultâneos). Propôs, então, uma máquina que executasse simples instruções de movimentos de dados entre registradores, denominadas microinstruções. Cada microinstrução consistiria de um ou mais comandos, ou eventos, associados às linhas de controle que comandavam os movimentos de dados entre registradores; estes comandos são chamados microcomandos e uma operação controlada ou afetada por um microcomando é chamada microoperação. Toda execução de uma instrução de máquina seria uma seqüência de microinstruções, denominada microprograma.

A Figura 28 mostra uma máquina hipotética com oito pontos de controle (1, 2, 3, 4, 5, 6, 7 e 8). As microinstruções requeridas, para executar

uma instrução de máquina, podem ser construídas ativando-se ou inibindo-se linhas de controle, sendo cada uma representada por um bit de memória.

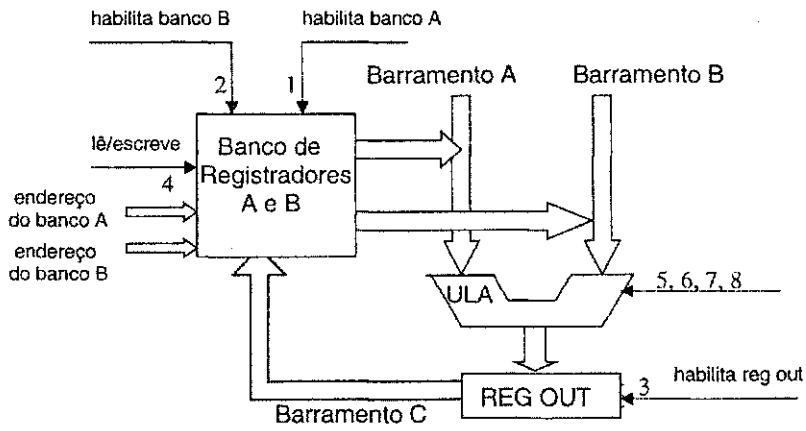


Figura 28 - Máquina hipotética com controle microprogramado

Uma palavra de memória com os bits representativos das linhas de controle corresponde a uma microinstrução, armazenada na memória de controle, ver Figura 29.

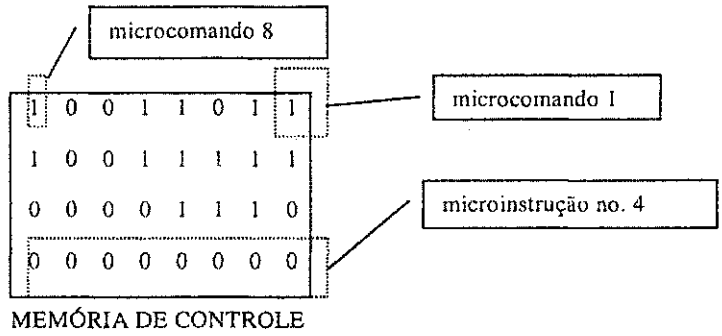


Figura 29 - Memória de controle

4.3.1 A microinstrução

A microinstrução é o conjunto de comandos, ou eventos, associados às linhas de controle que comandam os movimentos de dados entre registradores numa máquina com controle microprogramado e pode ser classificada em dois tipos básicos: microinstrução horizontal e microinstrução vertical.

A microinstrução horizontal é definida como aquela que contém todas as microoperações paralelas (simultâneas) possíveis que podem ser executadas pela máquina. Tem como vantagens a flexibilidade na utilização do hardware e o aproveitamento do paralelismo do fluxo de dados e como desvantagem a dificuldade para programação.

A microinstrução vertical é definida como sendo aquela que possui uma única microoperação a ser executada. Apresenta, como uma das vantagens, a simplicidade no formato, possibilitando um melhor entendimento e programação da mesma. Possui como desvantagem o excessivo número de microinstruções necessárias para se construir um microprograma.

O ciclo da microinstrução é dividido em dois passos: a busca da microinstrução na memória de controle e a execução. A execução consiste na análise dos vários campos da microinstrução e movimentação dos dados entre registradores especificados pelos microcomandos e pode ser de dois tipos: a execução monofásica e a execução polifásica.

A execução monofásica se caracteriza pela aplicação simultânea (na mesma fase) de todos os sinais de controle das microoperações especificadas na microinstrução, segundo ilustra a Figura 30.

Na execução polifásica, as microoperações especificadas na microinstrução não são executadas simultaneamente, mas em tempos diferentes como indica a Figura 31.

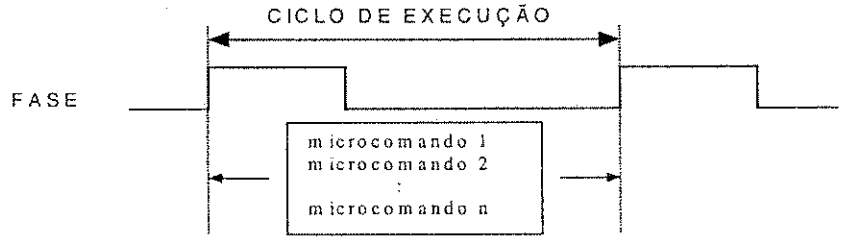


Figura 30 - Microinstrução monofásica

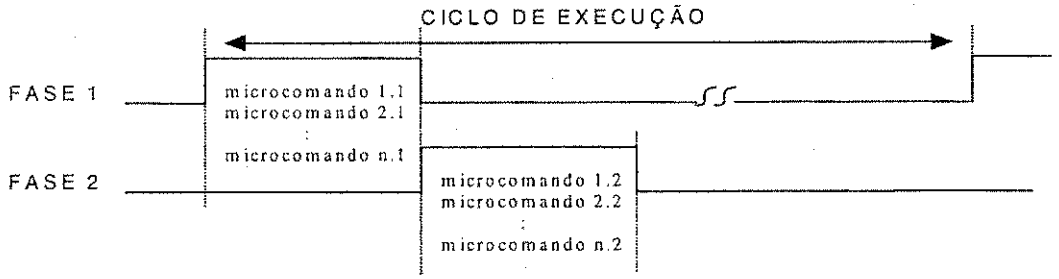


Figura 31 - Microinstrução polifásica

4.3.2 Estrutura do hardware de uma máquina microprogramada

A Figura 32 abaixo mostra a estrutura do hardware de uma máquina microprogramada, composta basicamente pelos blocos: Memória de Controle, Registrador Contador de Endereço da memória de controle (RCE), Regis-

trador de Microinstrução (RMI), Decodificador de Microinstrução (DM), Unidade Lógica e Aritmética (ULA) e Registradores de uso Geral (RG).

A memória de controle substitui a malha lógica da unidade de controle convencional. Contém as microinstruções para a execução das instruções da máquina. O RCE contém o endereço da microinstrução a ser executada, que funciona como um contador que é incrementado para a execução da próxima microinstrução ou carregado com um valor como resultado da microinstrução anterior. O Registrador de Microinstrução (RMI) contém a microinstrução que está sendo executada. Através da decodificação dos seus microcomandos pelo bloco DM, são controladas as operações da ULA, o fluxo de dados dos Registradores de uso geral (RG) e outros blocos da máquina.

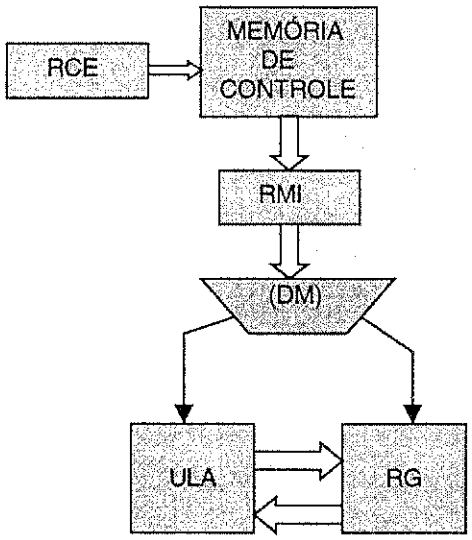


Figura 32 - Estrutura do hardware de uma máquina microprogramada

A arquitetura do sistema, denominado Controlador Microprogramável para Comutadores ATM (CMCA) [32], [33], [34] proposto neste trabalho, executa as funcionalidades da Camada ATM tendo como base uma unidade de controle microprogramada, de forma a torná-la flexível às modificações de suas funcionalidades. Esta característica, a exemplo do utilizado na evolução dos protocolos da indústria de modems, permite a utilização de uma mesma estrutura física, configurável através dos diversos microprogramas que poderão residir em memória interna, carregados através de *download*, por um sistema hospedeiro, ou de uma memória externa, por exemplo as do tipo *flash*. Associado à microprogramação, esta arquitetura buscou o uso do paralelismo de operações, como forma de agilizar o processamento para atender às altas taxas de transmissão das células (de 155 Mbps a 622 Mbps), através da replicação de uma estrutura de processamento capaz de executar algoritmos dirigidos a qualquer das três tarefas ligadas à Camada ATM: roteamento, policiamento e gerenciamento.

4.4 CMCA: Controlador Microprogramável para Comutadores ATM

A Figura 33 apresenta a arquitetura de um Comutador ATM usando o CMCA. Células são transferidas de uma porta de entrada para uma porta de saída em alta velocidade.

No meio físico, as células ATM são transmitidas como uma seqüência de bits, a IE (Interface de Entrada) referente a cada porta de entrada, transforma a seqüência de bits em bytes, de acordo com o padrão

UTOPIA (ver seção 2.2.2), submetendo-os ao CMCA correspondente, que realiza as funções relacionadas com a Camada ATM na seqüência:

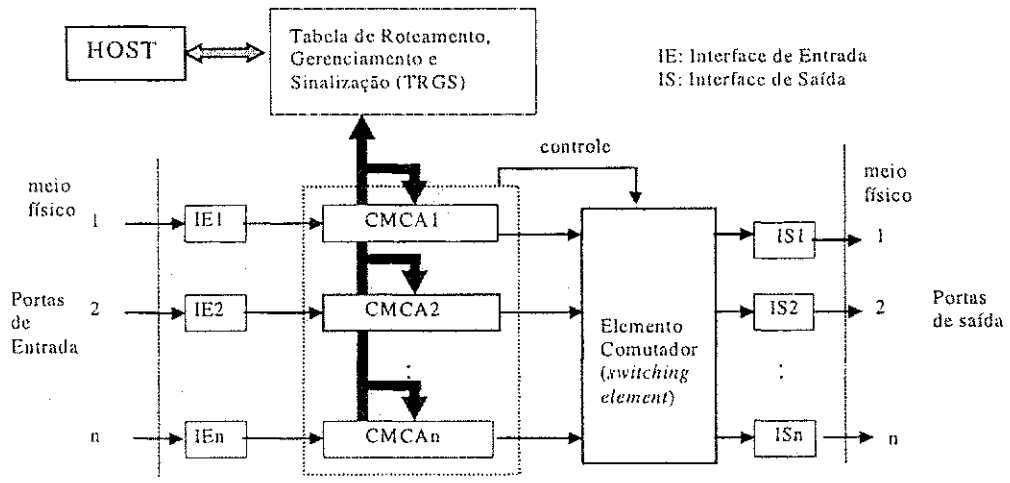


Figura 33 - Arquitetura de um Comutador ATM usando o CMCA

- Separação dos campos do cabeçalho da célula ATM (VPI, VCI, CLP, PT, GFC, HEC);
- Separação do payload da célula ATM;
- Identificação das células de usuário, de sinalização e de gerenciamento;
- Acesso à tabela de rotas para identificação da conexão e leitura dos parâmetros associados à mesma (novos campos VPI e VCI, endereço da porta de saída, parâmetros de policiamento, etc.);
- Execução do controle de fluxo (policiamento da conexão);

- Geração do novo cabeçalho da célula (troca dos campos VPI e VCI originais pelos novos valores lidos da tabela de rotas);
- Remontagem da célula com informações adicionais (porta de saída da célula, tipo da conexão, qualidade de serviço da conexão, etc.);
- Geração de estatística de gerenciamento (quantidade de células descartadas pelo policiamento, quantidade de células roteadas e outros);
- Atualização da tabela de rotas;
- Envio da célula e controle do elemento comutador (*switching element*).

O elemento comutador realiza as funções de roteamento físico e a IS (Interface de Saída) transforma os bytes da célula ATM em uma seqüência de bits.

As funções relativas ao plano de controle (entre elas a ativação, manutenção e desativação de conexões) e ao plano de gerenciamento, são executadas pelo *host*, que usa a TRGS como elo de ligação com os CMCAs.

4.5 Tabela de Roteamento, Gerenciamento e Sinalização (TRGS)

A TRGS contém todos os parâmetros e identificadores de tráfego das conexões para que as células possam ser roteadas até seu destino final. É um bloco de memória compartilhada, acessada tanto pelo sistema hospedeiro

(*host*) para a carga dos parâmetros das conexões contratadas, como pelo CMCA. A tabela é constituída por cinco tipos de informações: parâmetros de roteamento, parâmetros de policiamento, parâmetros de gerenciamento, células de gerenciamento e células de sinalização, conforme a Figura 34.

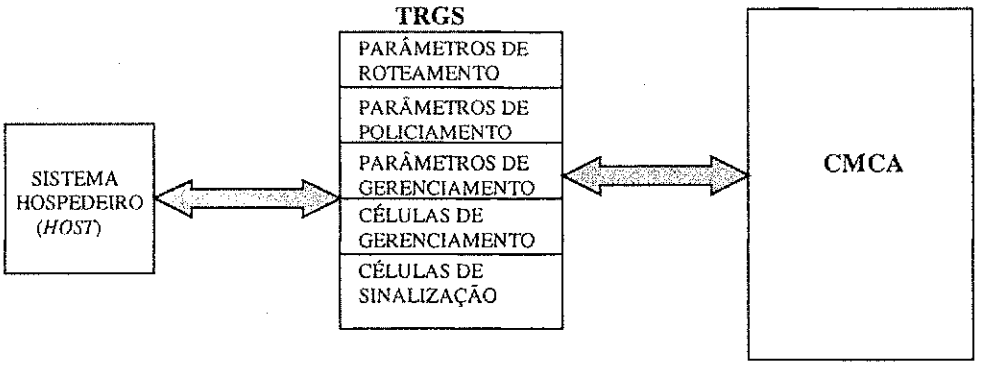


Figura 34 - Tabela de Roteamento, Gerenciamento e Sinalização

Os dados de roteamento compreendem: os novos campos VPI e VCI usados na remontagem do novo cabeçalho da célula, a porta de saída da célula, o tipo da conexão (*unicast* ou *multicast*), o número de portas em caso de conexão *multicast*, *flags* e ponteiros utilizados no algoritmo de roteamento e a qualidade de serviço da conexão, conforme especificado na Tabela 17 da seção 5.2. Os dados de policiamento compreendem as variáveis e constantes associadas ao tipo de controle de fluxo implementado no comutador (Tabela 18, seção 5.2.). Os dados estatísticos traduzem o número de células descartadas pelo policiamento e o número de células roteadas (Tabela 19, seção 5.2). As duas últimas áreas armazenam células de gerenciamento e sinalização que trafegam na rede. Estas células podem ser geradas pelo sistema hospedeiro ou destinadas a ele.

4.6 Fluxograma Geral da Camada ATM

A execução das funções da Camada ATM está baseada no processamento de parâmetros localizados na tabela de rotas (ver seção 4.5) que podem ser de três tipos básicos: processamento de roteamento, processamento de policiamento e processamento de gerenciamento.

O processamento de roteamento envolve a leitura dos novos campos VPI e VCI usados na remontagem do novo cabeçalho e outras informações que serão anexadas à célula (porta de saída, qualidade de serviço da conexão, etc.) para que seja roteada até seu destino final.

O processamento de policiamento executa a monitoração da conexão, ou seja, a verificação da respeitabilidade às condições de tráfego contratadas. Esse processamento envolve a leitura e atualização de parâmetros que são específicos ao tipo de policiamento praticado.

O processamento de gerenciamento compreende a contagem do número de células roteadas e descartadas por conexão e o armazenamento destes valores na tabela de rotas. A Figura 35 mostra o Fluxograma Geral da Camada ATM que foi utilizado como especificação e que levou à concepção e simulação da arquitetura do CMCA.

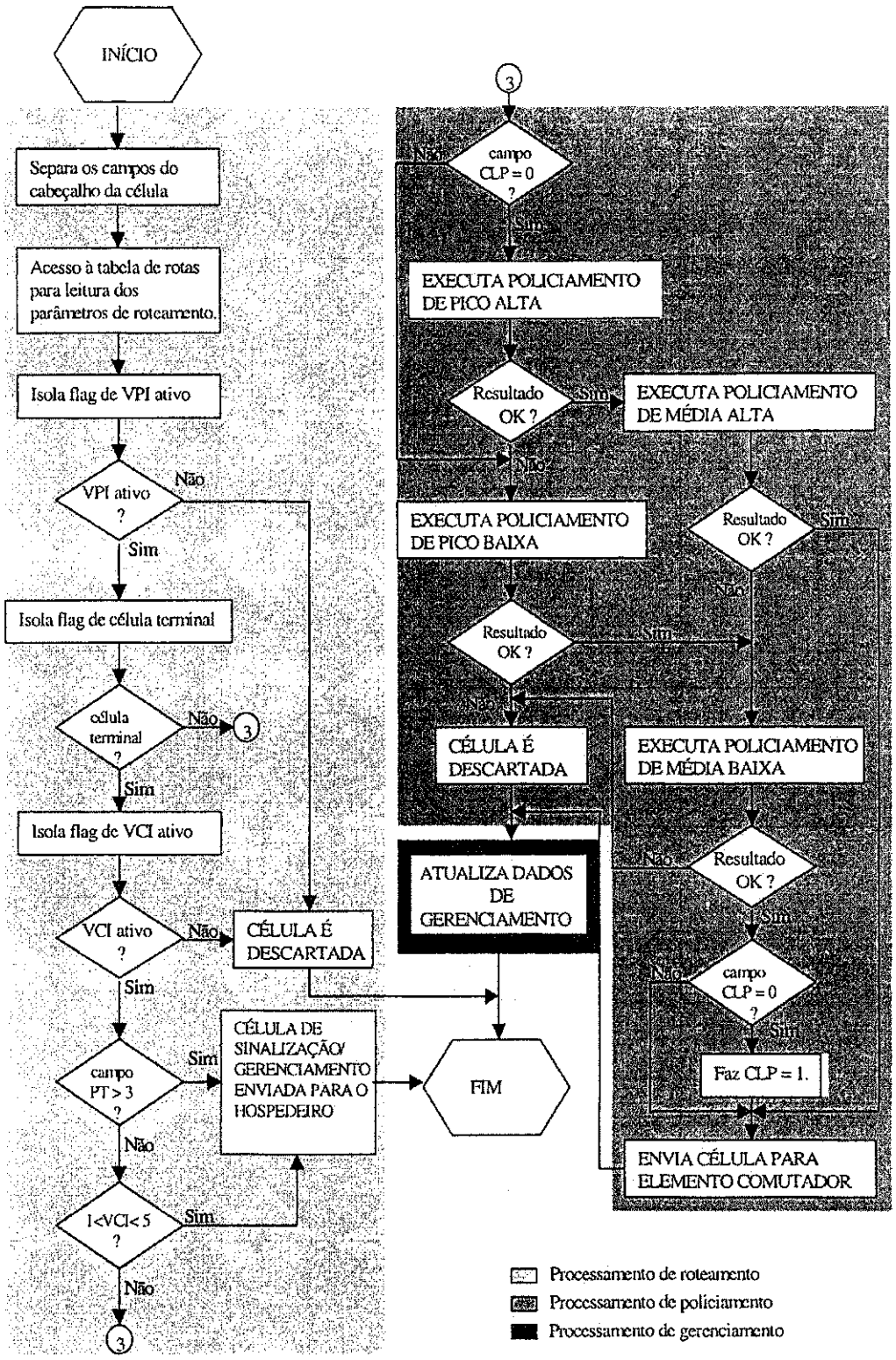


Figura 35 - Fluxograma Geral da Camada ATM

O processamento de policiamento é composto por chamadas a 4 tipos de procedimentos denominados: policiamento de pico alta, policiamento de pico baixa, policiamento de média alta e policiamento de média baixa. Os policiamentos de pico alta e pico baixa, representam o conjunto de procedimentos cujo propósito é monitorar a conexão com relação a taxa máxima de geração de células contratada e efetuar ações como o aceite ou descarte da célula, por exemplo. Os termos pico de alta e pico de baixa se referem à prioridade da célula que está sendo policiada (indicada pelo campo CLP do cabeçalho) sendo célula de alta prioridade se CLP = 0 e baixa prioridade se CLP = 1.

Os policiamentos de média alta e média baixa se referem ao conjunto de procedimentos cujo propósito é monitorar a conexão com relação a taxa média de geração de células contratada, sendo os termos média de alta e média de baixa referentes à prioridade da célula.

Esta estratégia de policiamento foi obtida a partir dos trabalhos desenvolvidos no âmbito do projeto COMATM [24], [36], [37], [38], [39].

Com a finalidade de avaliar o desempenho do processamento seqüencial do fluxograma geral da Camada ATM, utilizando um processador comercial existente, foi elaborado um programa em C^{++} a partir do fluxograma usando o compilador Borland C^{++} 5.0. O tempo de execução do programa para três processadores *Pentium* de diferentes freqüências e usando os sistemas operacionais: Windows95, Windows 98 e NT foi mensurado, conforme indica a Figura 36. O tempo mínimo de execução obtido foi de 3 microssegundos para o *Pentium* 333 MHz, ainda insuficiente para atender os requisitos de processamento de células sucessivas com taxa de

transmissão de 155 Mbps a 622 Mbps (2,735 μs e 0,680 μs, respectivamente).

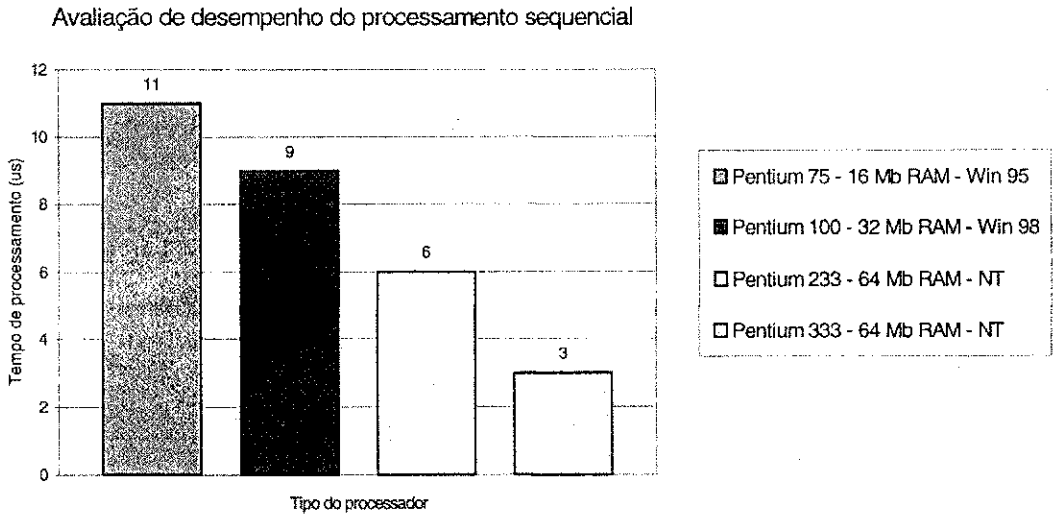


Figura 36 - Processamento seqüencial do fluxograma geral da Camada ATM

4.7 Concepção da arquitetura do CMCA

A análise do Fluxograma Geral da Camada ATM Figura 35 leva às seguintes considerações:

1) Os processamentos só são iniciados quando da chegada de uma célula e separação dos campos de seu cabeçalho.

2) De acordo com a estratégia de policiamento, uma mesma célula pode ser policiada várias vezes.

3) O processamento de gerenciamento é dependente do policiamento, ou seja, só pode ser executado após o término do mesmo.

4) Os processamentos apresentam predominância de operações de comparação e acesso à tabela de rotas para leitura, escrita e manipulação dos diversos parâmetros contidos nas palavras de roteamento, policiamento e gerenciamento.

5) Os processamentos são altamente dependentes do formato dos parâmetros e organização da tabela.

Estas considerações indicam a necessidade de um circuito que execute a interface com o meio físico, a separação dos campos do cabeçalho da célula e geração de sinais de sincronismo para início dos processamentos. A interface com o meio físico é efetuada usando o protocolo de sinais UTOPIA [37], [51], [80].

Uma análise mais cuidadosa (Figura 37),mostra que células de baixa prioridade (CLP = 1) da mesma conexão são submetidas a até dois tipos de policiamento (policiamento de pico baixa e policiamento de média baixa)

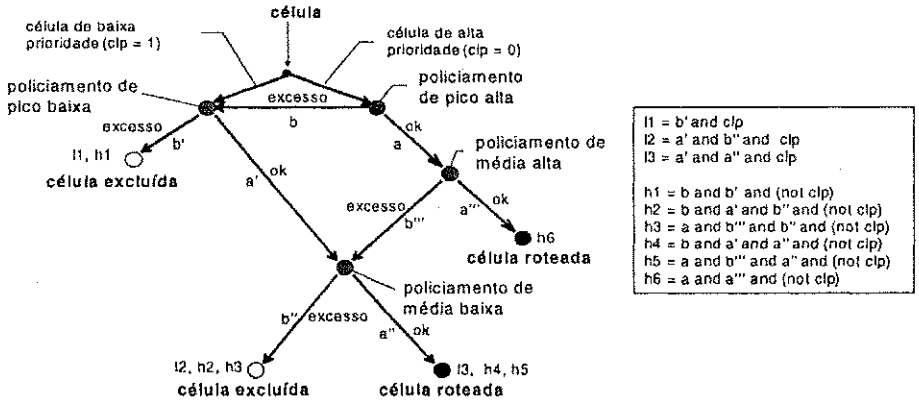


Figura 37 - Fluxo de policiamento do Fluxograma Geral da Camada ATM

e são roteadas se ambos os resultados dos policiamentos derem positivos (condição **l3**). Células de alta prioridade (CLP=0) de uma mesma conexão

são submetidas a até quatro tipos de policiamento (pico alta, média alta, pico baixa e média baixa) sendo apenas três deles possíveis de cada vez.

O escalonamento no tempo do Fluxograma Geral da Camada ATM, considerando o uso de circuitos independentes, denominados Módulo Microprocessador (MM), é mostrado na Figura 38. Em cada nível, as operações concorrentes executadas por cada MM são especificadas, bem como a dependência seqüencial entre os níveis.

No nível 1, os parâmetros de roteamento e policiamento são lidos da TRGS. Os parâmetros de roteamento são processados pelo MM1, os policiamentos de pico alta, média alta e média baixa são executados pelos MM2, MM3 e MM4 respectivamente. Por motivo de otimização, todos os tipos de policiamento são executados independente de seus resultados (*speculative execution*) mesmo que eventualmente nem todos os resultados sejam necessários posteriormente.

No nível 2, os parâmetros de policiamento de pico alta e média alta são atualizados pelos MM2 e MM3; o policiamento de pico baixa é executado pelo MM1. Dependendo do fluxo de policiamento (ver Figura 37) é possível se ter uma decisão do policiamento para células de alta prioridade, ainda neste nível, ou seja, células que satisfazem as condições: **h3**, **h5** e **h6** têm o seu policiamento finalizado neste nível.

No nível 3, os parâmetros de policiamento de pico baixa e média baixa são atualizados pelos MM4 e MM1. MM2 executa a decisão que satisfaz as condições: **l1**, **l2**, **l3**, **h1**, **h2** e **h4** e MM3 envia a célula para o Elemento Comutador.

No nível 4 o MM1 submete as células que satisfazem as condições: l1, l2, l3, h1, h2 e h4 ao Elemento Comutador.

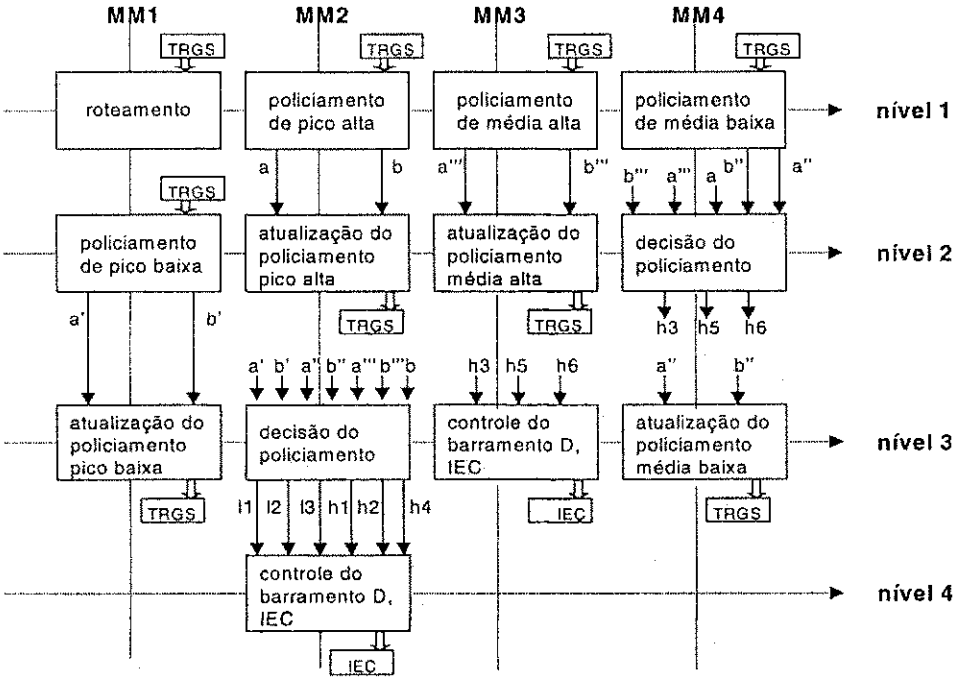


Figura 38 - Escalonamento do Fluxograma Geral da Camada ATM

Todas estas considerações conduziram à arquitetura apresentada a seguir.

4.8 Arquitetura do CMCA

A Figura 39 mostra a arquitetura do CMCA, composta pelo seguintes blocos:

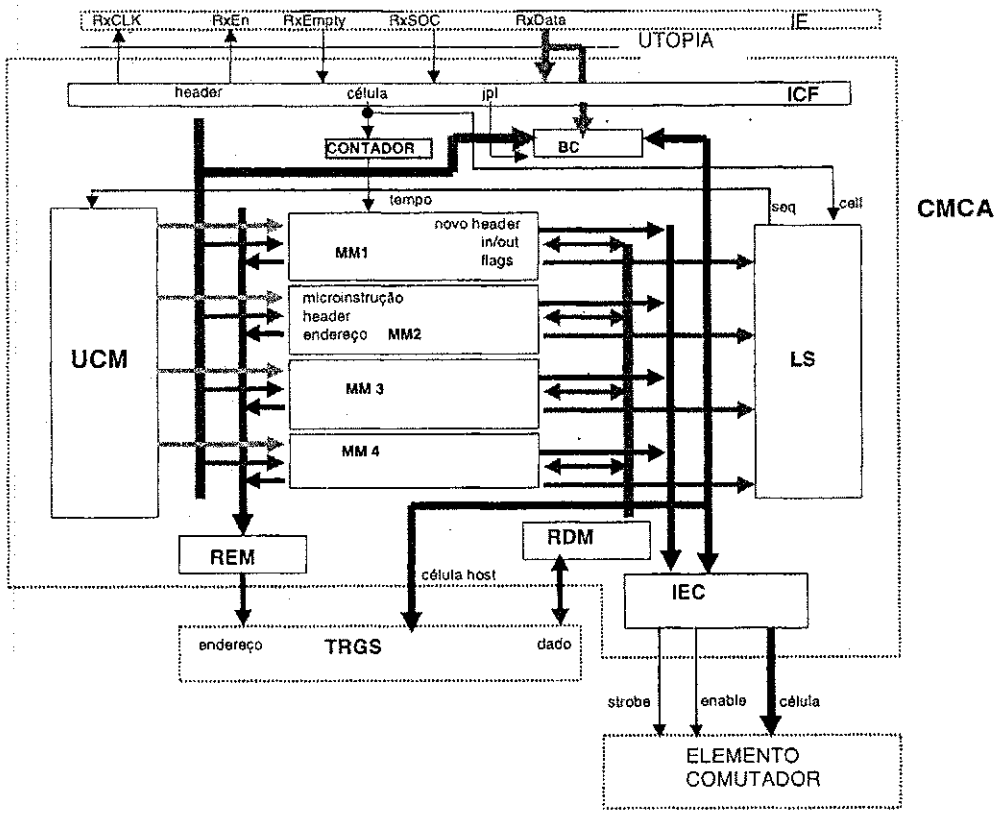


Figura 39 - Arquitetura do CMCA

- Interface com Camada Física (ICF).
- 4 Módulos Microprocessadores (MM).
- Lógica de Sequenciamento (LS).
- Unidade de Controle Microprogramável (UCM).
- Buffer de Célula (BC).
- Contador.

- Interface com Elemento Comutador (IEC).
- Registrador de Endereço da Memória (REM).
- Registrador de Dados da Memória (RDM).

A célula proveniente da Camada Física é recebida pela Interface Com Camada Física (ICF) que separa seus campos de cabeçalho e *payload* e sincroniza a operação dos blocos restantes do CMCA. Os Módulos Microprocessadores (MMs) executam os algoritmos de roteamento e policiamento de acordo com o microprograma da Unidade de Controle Microprogramada (UCM). Os parâmetros referentes a cada conexão e usados nos algoritmos são lidos de uma memória externa, a Tabela de Roteamento, Gerenciamento e Sinalização (TRGS) através do Registrador de Endereço de Memória (REM) e Registrador de Dados da Memória (RDM). Todos os MMs podem acessar a TRGS (em operações seqüenciais) carregando o endereço de acesso no REM e lendo ou escrevendo os dados via o RDM. A Lógica de Sequenciamento (LS) controla a escolha da próxima microinstrução a ser executada, permitindo desvios condicionais ou incondicionais no fluxo do microprograma. Esses desvios são acionados pelos *flags* de cada MM, como resultado da microinstrução anterior. A célula é armazenada no Buffer de Células (BC) e dependendo do seu tipo (célula de gerenciamento de sinalização ou de usuário) e do resultado do policiamento poderá ser enviada para o sistema hospedeiro ou elemento comutador. A Interface com o Elemento Comutador (IEC) controla o Elemento Comutador através de uma interface paralela, enviando a célula em pacotes de 32 bits. Os intervalos de tempo entre as células de uma mesma conexão são medidos pelo

CONTADOR de 32 bits e usados no processamento dos algoritmos de policiamento praticados.

Em seguida, cada bloco será descrito.

4.8.1 Interface com Camada Física (ICF)

O bloco ICF tem como função receber da Camada Física a célula ATM em palavras de 8 bits (para taxas de 155 Mbps) ou 16 bits (para taxas de 622 Mbps), controlando a recepção através da implementação do protocolo de sinais UTOPIA [51] (ver seção 2.2.2), padronizado pelo Fórum ATM. O bloco também executa a separação dos campos da célula ATM: cabeçalho e seus campos internos (VPI, VCI, CLP, PT, GFC e HEC) e o payload.

A ICF é constituída pelas unidades: "recepção", "payload," "habilitação" e "extração", conforme ilustra a Figura 40.

A unidade "recepção" gera sinais internos de controle para as demais unidades, a partir do protocolo UTOPIA. A unidade "payload" indica o início do payload da célula, gerando o sinal "jpl" para o bloco BC. A unidade "habilitação" gera os sinais de habilitação para as unidades de extração dos campos internos do cabeçalho da célula ATM e o sinal "cell" para os blocos CONTADOR e LS, indicando o instante de chegada da célula.

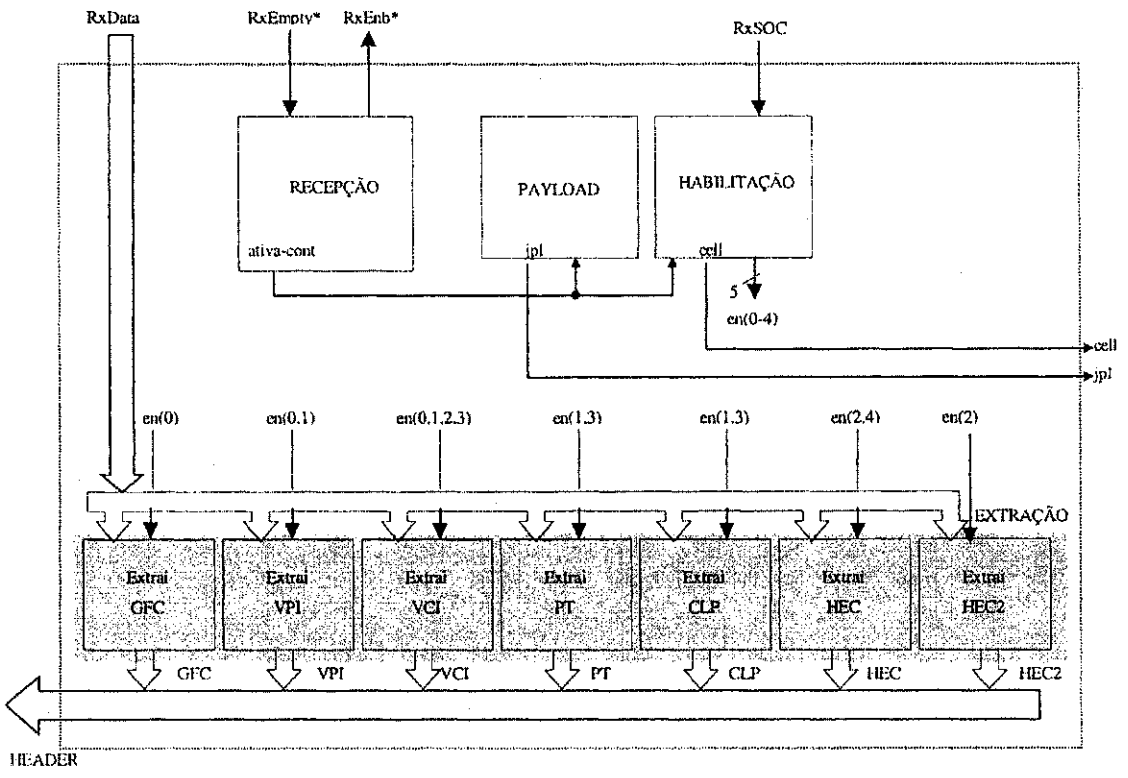


Figura 40 - Arquitetura da ICF

4.8.2 Módulo Microprocessador

A arquitetura geral do Módulo Microprocessador é mostrada na Figura 41. Ela contém seis registradores de 32 bits , denominados: RA, RB, RC, RD, RE e RF que formam a "memória rascunho". Cada registrador pode ser lido via os barramentos A e B e cada um pode ser carregado a partir de um terceiro barramento, presente na arquitetura, o barramento C.

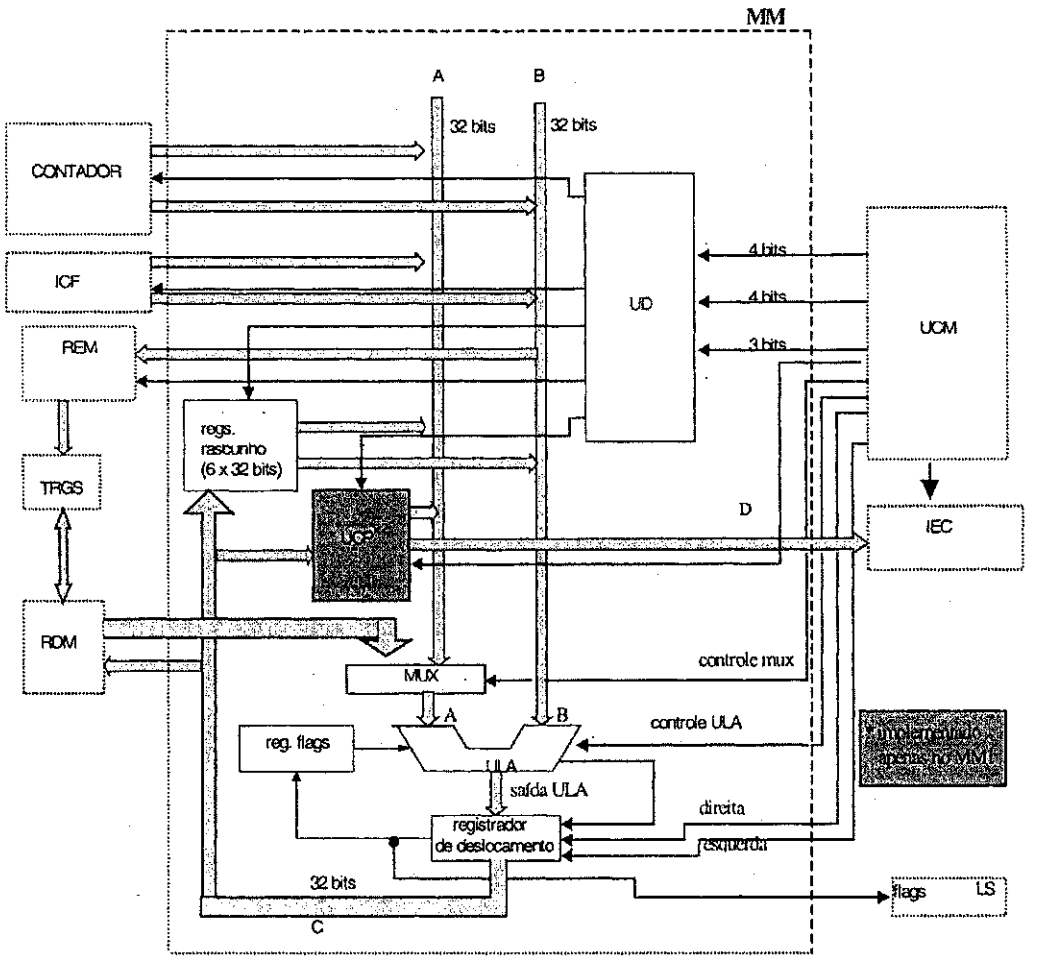


Figura 41 - Módulo Microprocessador

A palavra de 32 bits oriunda da TRGS contém parâmetros usados nos algoritmos de processamento da célula. Dependendo dos algoritmos, da definição e da organização da TRGS, esses parâmetros podem possuir tamanhos diversos (4, 8 16 ou 32 bits, por exemplo). Como forma de agilizar o manuseio desses parâmetros dentro da palavra de 32 bits, o MM contém uma unidade denominada Unidade de Controle de Parâmetros

(UCP), formada por um registrador denominado Registrador de Campo (RC), usado para armazenar a palavra de 32 bits lida da memória externa (TRGS). Este registrador pode ser lido via os barramentos A e D, em campos de 4, 8, 16 ou 32 bits em apenas um ciclo de instrução, evitando operações de deslocamento via o registrador de deslocamento para a separação de campos da palavra, e que gastam vários ciclos de instrução. Na implementação do CMCA, a unidade UCP foi inserida apenas no Módulo Microprocessador 1 (MM1), tendo em vista que, de acordo com o escalonamento do fluxograma de simulação (seção 4.7), esse módulo era responsável pelas operações de roteamento e controle do elemento comutador tornando-se redundante a presença do citado bloco nos outros módulos. A Figura 42 mostra a unidade UCP, levando-se em conta os parâmetros de roteamento usados.

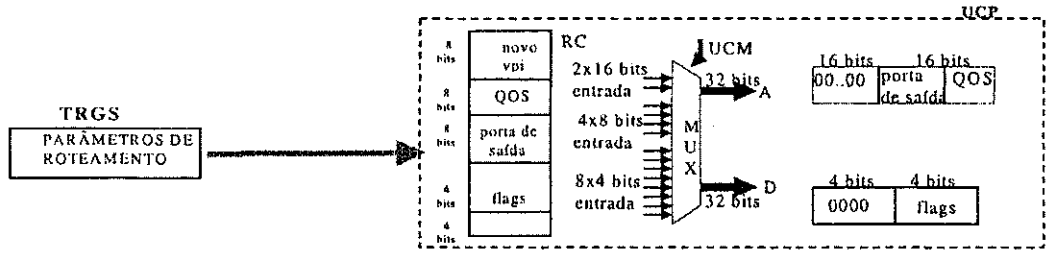


Figura 42 - Unidade de Controle de Parâmetros (UCP)

Os barramentos A e B são conectados às entradas da ULA de 32 bits, que pode executar 13 operações lógicas e aritméticas, especificadas pelas quatro linhas de controle, como ilustra a Tabela 15. A Unidade de Decodificação (UD) contém a lógica combinacional necessária para controlar o fluxo de dados do módulo processador.

OPERAÇÃO DA ULA	CONTROLE	FLAGS AFETADOS
A + B	0000	n,z, c
A-B	0001	n,z c
A comp B	0010	eq , gt, lt
A and B	0011	n, z
A or B	0100	n, z
saída ULA: = A	0101	n, z
saída ULA: = B	0110	n, z
saída ULA: = not A	0111	n, z
saída ULA: = not B	1000	n, z
flag c := not flag c	1001	c
flag c : = 0	1011	c
flag c := 1	1100	c
nenhuma operação	1100	--

FLAGS

eq	le	gt	c	n	z

z : zero flag
n: negative flag
c : carry flag
gt: greater than flag (gt = 1 se A >B)

lt: lesser than flag (lt = 1 se A < B)
eq: equal flag (eq =1 se A= B)

Tabela 15 - Operações da ULA e flags

4.8.3 Lógica de Sequenciamento (LS)

A escolha da próxima microinstrução a ser executada é determinada pela Lógica de Sequenciamento (LS), conforme ilustra a Figura 43. Os flags de cada Modulo Microprocessador (MM), o sinal de saída *cell* do bloco ICF e o campo COND da microinstrução corrente indicam a próxima microinstrução a ser executada, conforme ilustra a Tabela 16. O sinal de saída *seq* da Lógica de Sequenciamento (LS) indica à UCM o endereço da próxima microinstrução (ADR se *seq* = 1 ou "endereço corrente + 1" se *seq* = 0). ADR é um campo da microinstrução.

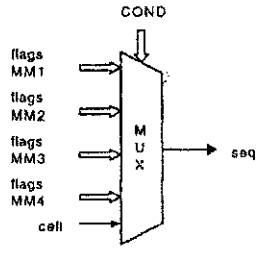


Figura 43 - Lógica de sequenciamento (LS)

COND	Escolha
0	Não desvie
1	Desvie para ADR se c = 1
2	Desvie para ADR se z = 1
3	Desvie para ADR se eq = 1
4	Desvie para ADR se gt = 1
5	Desvie para ADR se lt = 1
6	Desvie para ADR se n = 1
7 até 12	Mesmo acima para MM2
13 até 18	Mesmo acima para MM3
19 até 24	Mesmo acima para MM4
25	Desvie para ADR se cell = 1
26	Desvio incondicional

Tabela 16 - Possibilidades de escolha da próxima microinstrução

4.8.4 Unidade de Controle Microprogramada (UCM)

A UCM controla os blocos do CMCA, através de microinstruções horizontais, conforme ilustra a Figura 44. A unidade é composta por um multiplexador, um registrador incrementador de endereços (RIE), uma memória de controle e um registrador de microinstrução (RMI), conforme ilustra a Figura 45.

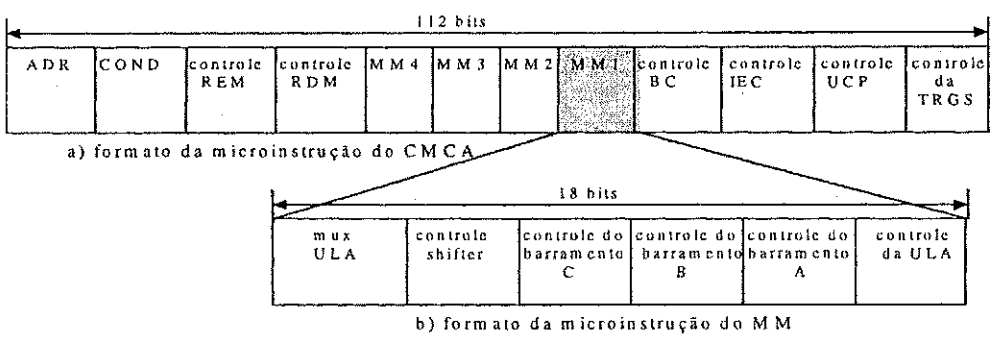


Figura 44 - Formato das microinstruções

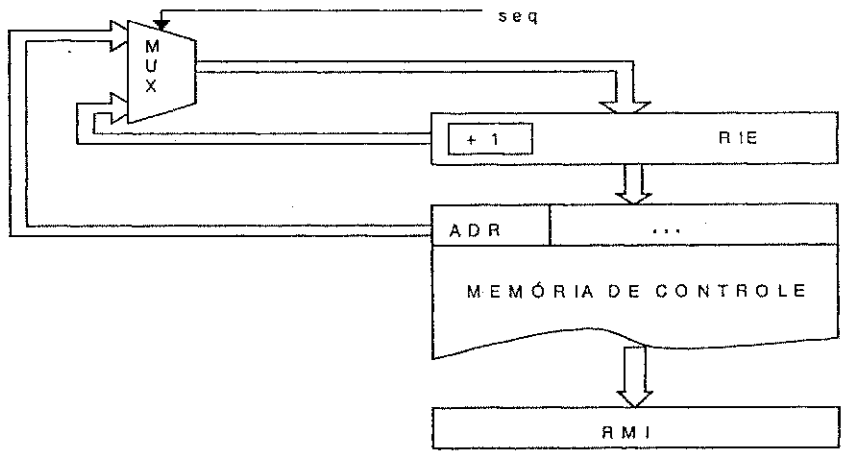


Figura 45 - Unidade de Controle Microprogramada (UCM)

A memória de controle (do tipo RAM) armazena as microinstruções que compõem os algoritmos de controle. O uso de memória RAM torna o controle flexível; como os microcomandos da microinstrução controlam todo o fluxo de dados do CMCA, diversos microprogramas podem ser elaborados, cada um executando um algoritmo em particular, por exemplo: um determinado microprograma executa o algoritmo de policiamento baseado no "Balde Furado", um outro pode ser elaborado para executar o algoritmo "Janelas Deslizantes", um outro para executar o algoritmo "Janelas Saltitantes" [67], [68], [69], etc. O microprograma desejado pode então ser carregado na RAM de microprograma, a partir da carga pelo hospedeiro e o CMCA passa a executar o algoritmo em particular sem nenhuma alteração no seu *hardware.*

A microinstrução a ser executada é lida da memória no endereço contido no RIE e armazenada no RMI. Durante a execução, o RIE incrementa seu conteúdo para apontar para a próxima microinstrução, a lógica

de sequenciamento (LS) controla a busca da próxima microinstrução através do sinal *seq*, para a escolha da próxima a ser executada; a de endereço ADR (contido no campo da microinstrução corrente) ou a de "endereço corrente + 1".

4.8.5 Buffer de Células (BC)

O Buffer de Células (BC) armazena a célula oriunda da ICF ou da Tabela de Roteamento, Gerenciamento e Sinalização (TRGS). É constituída por duas RAMs e uma unidade que executa o controle do fluxo de células, conforme ilustra a Figura 46.

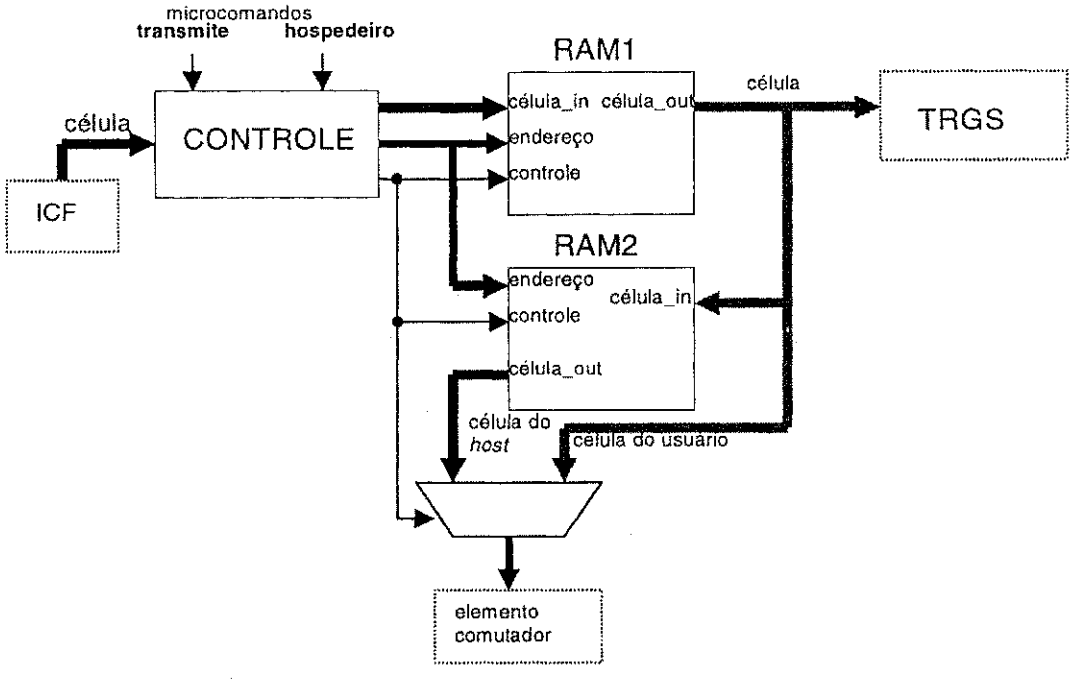


Figura 46 - Buffer de Células (BC)

O algoritmo de processamento determina quando e para onde a célula armazenada deve ser transferida. Existem três possibilidades:

1. A célula oriunda da ICF é célula de usuário e deve ser roteada. Neste caso a RAM1 armazena a célula que é submetida ao elemento comutador, após o policiamento. A operação de transmissão é acionada através do microcomando **transmite**.
2. A célula oriunda da ICF é célula de sinalização/gerenciamento terminal, ou seja, é uma célula que deve ser enviada para o hospedeiro (*host*). Neste caso a RAM1 armazena a célula que será transferida para a TRGS. A operação de transferência é acionada através do microcomando **hospedeiro**.
3. A célula oriunda da TRGS é célula de sinalização/gerenciamento enviada pelo *host* e deve ser roteada. Neste caso a RAM2 armazena a célula que será submetida ao elemento comutador.

4.8.6 Interface com o Elemento Comutador (IEC)

A interface com o Elemento Comutador (IEC) controla o elemento comutador externo, através de uma interface paralela (sinais *enable*, *strobe* e saída). A célula flui do CMCA para o elemento comutador através de um barramento de 32 bits (saída), controlado pelos sinais *enable* e *strobe* , conforme ilustra a Figura 47. O sinal *enable* ativo, indica que o dado presente no barramento é válido, nesse caso o elemento comutador sincroniza a leitura do dado através do sinal *strobe*. Esta abordagem dá ao CMCA flexibilidade para se conectar aos vários tipos de elementos comutadores existentes, mediante

um adaptador de interface simples que pode ser implementado em FPGA. Esse adaptador traduz os sinais da interface paralela nos sinais específicos do elemento comutador utilizado.

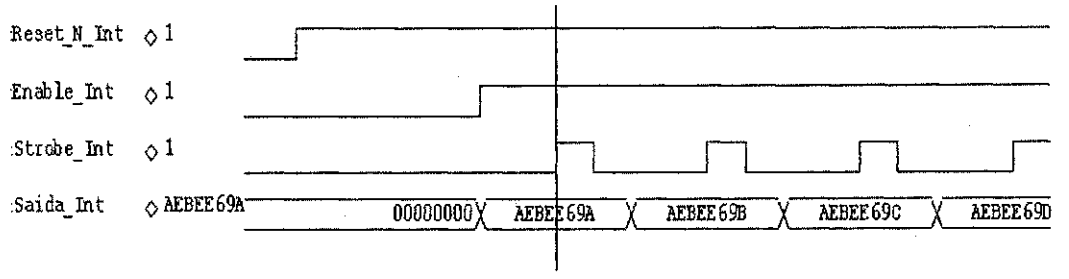


Figura 47 - Sinais da Interface com Elemento Comutador

4.8.7 Contador

O contador de 32 bits é responsável pela medida dos intervalos de tempo entre as células oriundas da Interface de Entrada (IE). Esta informação está disponível para os Módulos Microprocessadores (MM) e é usada nos algoritmos de policiamento.

4.8.8 REM e RDM

Toda troca de informação entre os Módulos Microprocessadores (MM) e a Tabela de Roteamento, Gerenciamento e Sinalização (TRGS) é feita através do Registrador de Endereço da Memória (REM) e Registrador de Dados da Memória (RDM). O REM armazena o endereço da TRGS que será lido ou escrito pelo Módulo Microprocessador (MM), enquanto o RDM armazena o dado lido ou a ser escrito. Cada Módulo Microprocessador (MM) pode escrever no REM, ler e escrever no RDM, para trocar dados com a Tabela.

As operações de leitura e escrita são controladas por campos específicos das microinstruções.

4.8.9 Ciclos da microinstrução

Embora os Módulos Microprocessadores (MMs) que compõem o CMCA possam executar operações concorrentes, o ciclo de operação básica de cada módulo produz eventos que ocorrem em seqüência. A fim de se determinar uma seqüência correta dos eventos foi introduzida uma base de tempo com 4 fases de relógio (isto é, um relógio com 4 subciclos). Essa base de tempo é fornecida através de um circuito, interno ao CMCA, que gera as fases phy1, phy2, phy3 e phy4, a partir do sinal de relógio externo, *master*, conforme ilustra a Figura 48.

Os eventos chave durante cada um dos quatro subciclos são:

- phy1 - Leitura da próxima microinstrução a ser executada
- phy2 - Transferência de dados dos registradores para os barramentos A e B
- phy3 - Carga do Registrador de Endereços da Memória (REM)
- phy4 - Transferência de dados do barramento C para os registradores rascunho, UCP e RDM

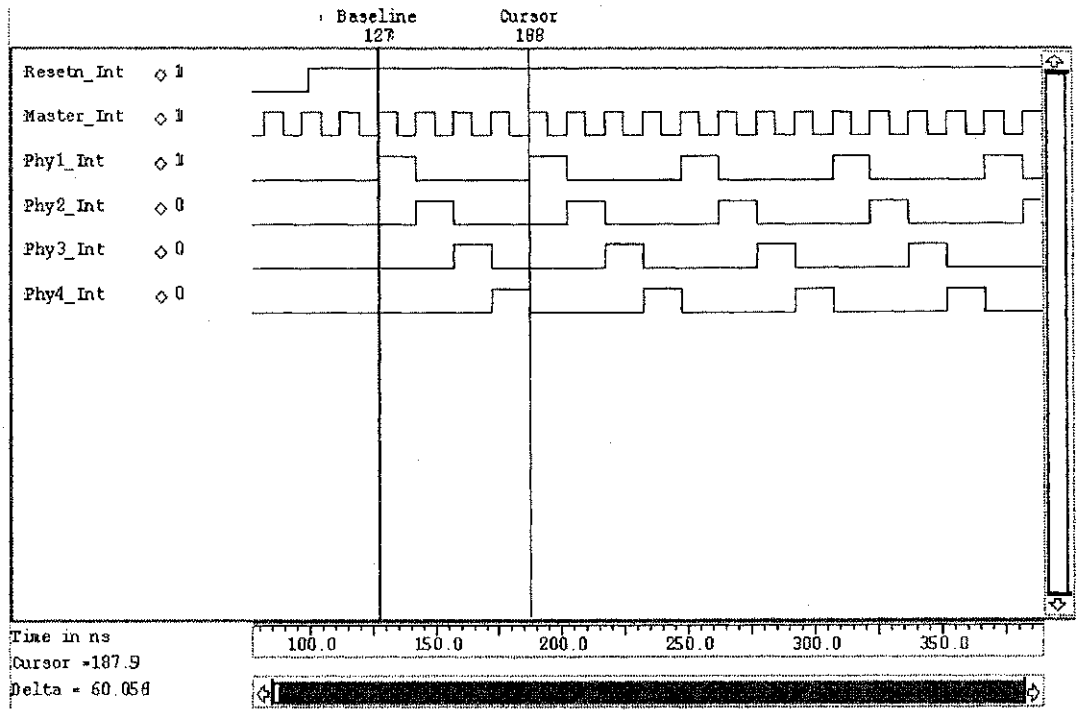


Figura 48 - Gerador das fases do relógio

4.9 Resumo

Este capítulo descreveu a arquitetura do Controlador Microprogramável para Comutadores ATM (CMCA), que se propõe a executar as funções da Camada ATM de um Comutador ATM, tendo como principal característica a flexibilidade para a execução de diversos algoritmos, devido a sua unidade de controle microprogramada. O próximo capítulo apresenta a metodologia de implementação e simulação da referida arquitetura.

Capítulo

5

Metodologia de Implementação e Simulação do CMCA

Este capítulo apresenta a definição e organização dos dados da Tabela de Roteamento, Gerenciamento e Sinalização (TRGS), onde estão armazenadas as informações referentes às conexões ativas do comutador. Apresenta ainda o modelo utilizado para descrever a arquitetura do CMCA e o cenário de teste utilizado para a simulação e validação lógica.

5.1 A linguagem VHDL

A linguagem VHDL (*VHSIC* (*Very High Speed Integrated Circuits*) *Hardware Description Language*) [92], [93] é uma linguagem padrão para a descrição e especificação de hardware, aceita pela maioria das ferramentas de síntese comerciais [94], [95] que permite descrições de hardware em diferentes níveis de abstração, oferecendo as seguintes facilidades no projeto de circuitos:

- Documentação: a própria descrição VHDL do hardware que levará ao circuito sintetizável ajuda na sua documentação.
- Simulação: simulação dos modelos comportamentais e estruturais do hardware.
- Tecnologia: A síntese do circuito pode ser feita em princípio em qualquer tecnologia a partir da descrição VHDL do mesmo.
- Reutilização: módulos de um projeto desenvolvidos em VHDL podem ser incorporados às bibliotecas existentes, permitindo seu uso em outros projetos.

A descrição de um componente em VHDL consiste de uma especificação de interface e de uma especificação de arquitetura. Como ilustrado na Figura 49, a descrição de interface inicia com a palavra reservada ENTITY e contém a descrição das portas de entrada e saída do componente. O nome do componente vem após a palavra ENTITY e é seguida pela palavra reservada IS. A especificação de arquitetura inicia com a palavra reservada ARCHITECTURE, e descreve as funcionalidades do com-

ponente, que dependem dos sinais de entrada e saída e outros parâmetros especificados na descrição de interface.

```
ENTITY nome_do_componente IS

(
Portas de entrada e saída e
outros parâmetros
);

END nome_do_componente;

ARCHITECTURE comportamental OF nome_do_componente IS

Declarações

BEGIN

Descrição das funcionalidades do componente

END comportamental;
```

Figura 49 - Descrição da interface e arquitetura do componente

A linguagem VHDL foi usada na descrição de todos os blocos do CMCA.

5.2 Definição da Tabela de Roteamento, Gerenciamento e Sinalização

A tabela de rotas, aqui denominada Tabela de Roteamento, Gerenciamento e Sinalização (TRGS), possui informações necessárias ao processamento das funções da Camada ATM. A Tabela 17 mostra os parâmetros de roteamento definidos e usados na simulação do CMCA. A Tabela 18 e Tabela 19 mostram os parâmetros de policiamento e gerenciamento respectivamente.

PARÂMETRO	SIGNIFICADO	TAMANHO (bits)
Novo VPI	Novo valor do campo vpi da célula	8
Novo VCI	Novo valor do campo vci da célula	16
QOS	Qualidade de serviço da conexão, inserida na célula	4
Flag VPI válido	Indica dados da célula existente na tabela (=1 se válido)	1
Flag VCI válido	Indica dados da célula existente na tabela (=1 se válido)	1
Flag VP terminal	Indica célula de conexão terminal (=1 se célula terminal)	1
Flag multicast	Indica operação de multicast (=1 se multicast)	1
Quantidade de portas multicast	Indica o numero de cópias da célula a ser transmitida	8
Porta de saída da célula	Endereço da porta de saída da célula	8
Ponteiro de segmento de caminho	Índice para consulta aos dados de roteamento	16
Ponteiro de segmento de canal	Índice para consulta aos dados de roteamento	16

Tabela 17 - Parâmetros de roteamento

PARÂMETRO	SIGNIFICADO	TAMANHO (bits)
LCT - Pico de alta	Valor do tempo de chegada da última célula válida	32
X - Pico de alta	Instante de chegada da última célula válida	32
L – Pico de alta	Tolerância admitida na violação do incremento	32
I – Pico de alta	Espaçamento nominal entre duas células válidas	32
LCT - Pico de baixa	Valor do tempo de chegada da última célula válida	32
X - Pico de baixa	Instante de chegada da última célula válida	32
L – Pico de baixa	Tolerância admitida na violação do incremento	32
I – Pico de baixa	Espaçamento nominal entre duas células válidas	32
LCT - Média de alta	Valor do tempo de chegada da última célula válida	32
X - Média de alta	Instante de chegada da última célula válida	32
L – Média de alta	Tolerância admitida na violação do incremento	32
I – Média de alta	Espaçamento nominal entre duas células válidas	32
LCT - Média de baixa	Valor do tempo de chegada da última célula válida	32
X - Média de baixa	Instante de chegada da última célula válida	32
L – Média de baixa	Tolerância admitida na violação do incremento	32
I – Média de baixa	Espaçamento nominal entre duas células válidas	32

Tabela 18 - Parâmetros de policiamento

PARÂMETRO	SIGNIFICADO	TAMANHO (bits)
CDPP	Contador de células descartadas por policiamento de pico	32
CDPM	Contador de células descartadas por policiamento de média	32
CDPC	Contador de células descartadas por congestionamento	32
CDR	Contador de células roteadas	32

Tabela 19 - Parâmetros de gerenciamento

Os parâmetros de roteamento definidos na Tabela 17 possibilitam dois modos de comutação de células : comutação de caminho virtual e co-

mutação de canal virtual [54]. Na comutação de caminho virtual (seção 2.2), apenas o VPI da célula pertencente ao caminho virtual é trocado pelo novo valor existente na tabela, mantendo-se o VCI original; neste tipo de comutação o processamento é reduzido, uma vez que não há necessidade de examinar nem mapear VCIs. Na comutação de canal virtual, é necessário o mapeamento não apenas dos VPIs, mas também dos VCIs.

A célula é dita ser uma célula não terminal, quando é realizada sobre ela uma comutação de caminho virtual, por outro lado a célula é dita ser uma célula terminal quando sofre uma comutação de canal virtual. Levando-se em conta a existência de comunicação multicast (ponto a multiponto), os parâmetros de roteamento definidos na TRGS possibilitam quatro tipos de comutação: célula não terminal unicast, célula não terminal multicast, célula terminal unicast e célula terminal multicast.

Os parâmetros de policiamento definidos possibilitam a execução do Algoritmo Genérico de Controle de Taxa (GCRA – Generic Cell Rate Algorithm) versão "Balde Furado", de forma a monitorar o tráfego da conexão com relação à taxa de pico e taxa média contratadas e de acordo com a prioridade da célula dentro da conexão.

Finalmente, os parâmetros de gerenciamento correspondem aos dados da tabela que contêm informações sobre o desempenho do nó de comutação, atualizados a cada processamento de célula.

5.3 Organização e acesso dos dados da tabela

Os parâmetros da tabela são organizados em três áreas distintas, definidas como módulo de roteamento, módulo de policiamento e módulo de gerenciamento. Os três módulos são divididos, cada um, em dois segmen-

tos, o segmento de caminho e o segmento de canal que armazenam os dados para a comutação de caminho virtual e canal virtual, respectivamente.

5.3.1 Comutação de caminho virtual

Na comutação de caminho virtual, o campo VPI da célula é utilizado como indexador para acessar a palavra que contém os novos dados da célula. O endereço físico da memória (tabela) é gerado a partir do campo VPI somado a um valor programável de *offset* de endereço que proporciona flexibilidade para escolha do endereço inicial do módulo dentro da memória. Este *offset* é carregado pelo sistema hospedeiro, na própria tabela (os primeiros endereços da tabela foram usados para armazenar os *offsets* de roteamento, policiamento e gerenciamento carregados pelo hospedeiro) e usado pelo CMCA no cálculo do endereço físico. A Figura 50 ilustra o acesso ao módulo de roteamento e o formato da palavra de roteamento para a comutação de caminho virtual *unicast*.

O acesso ao módulo de policiamento é feito também usando o campo VPI da célula como indexador e um outro *offset* programado pelo hospedeiro de forma a flexibilizar o uso da tabela. Cada procedimento de policiamento tem associado quatro parâmetros, representados por quatro palavras sucessivas (LCT, X, L, I) de 32 bits, conforme ilustra a Figura 51.

O acesso ao módulo de gerenciamento é análogo aos anteriores, duas palavras de 32 bits são suficientes para acomodar os parâmetros de gerenciamento, conforme ilustra a Figura 52.

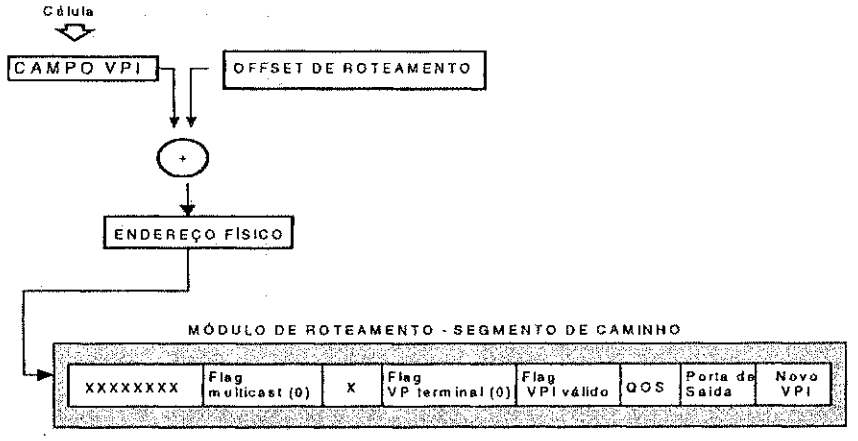


Figura 50 – Módulo de roteamento comutação de caminho virtual *unicast*

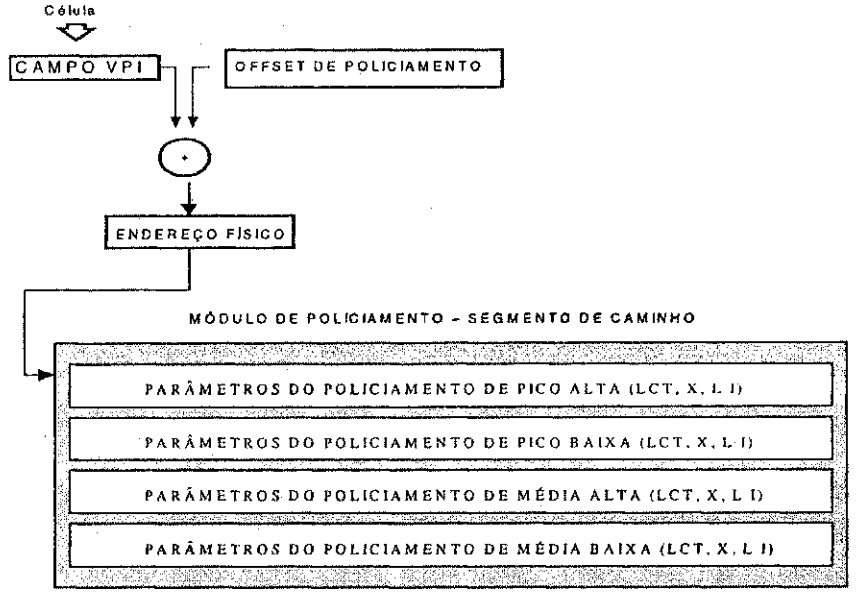


Figura 51 - Módulo de policiamento comutação de caminho virtual *unicast*

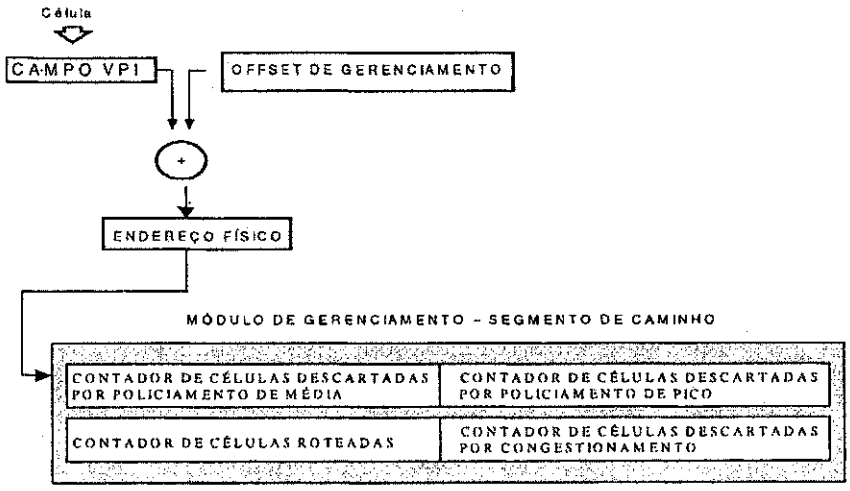


Figura 52 - Módulo de gerenciamento comutação de caminho virtual *unicast*

Na comutação de caminho virtual *multicast,* a célula deve ser enviada para mais de uma porta física com um novo valor de VPI relativo a cada porta. A tabela deve conter os novos valores de campos VPIs que correspondem à quantidade de portas físicas necessárias para a operação, sendo preciso mais de um acesso ao segmento de caminho.

O campo VPI da célula, a exemplo da operação *unicast,* continua sendo usado como indexador para acessar o primeiro dado no segmento de caminho. O parâmetro "Ponteiro de segmento de caminho" gera, juntamente com o valor do campo VPI da célula, o endereço físico para acessar os valores dos vários novos campos VPIs e portas de saída correspondentes para completar a operação *multicast.* A Figura 53 ilustra o processo.

Como se trata de uma comutação virtual de caminho (é policiado o caminho), a forma de acesso aos módulos de policiamento e de gerenciamento é idêntica à operação *unicast.*

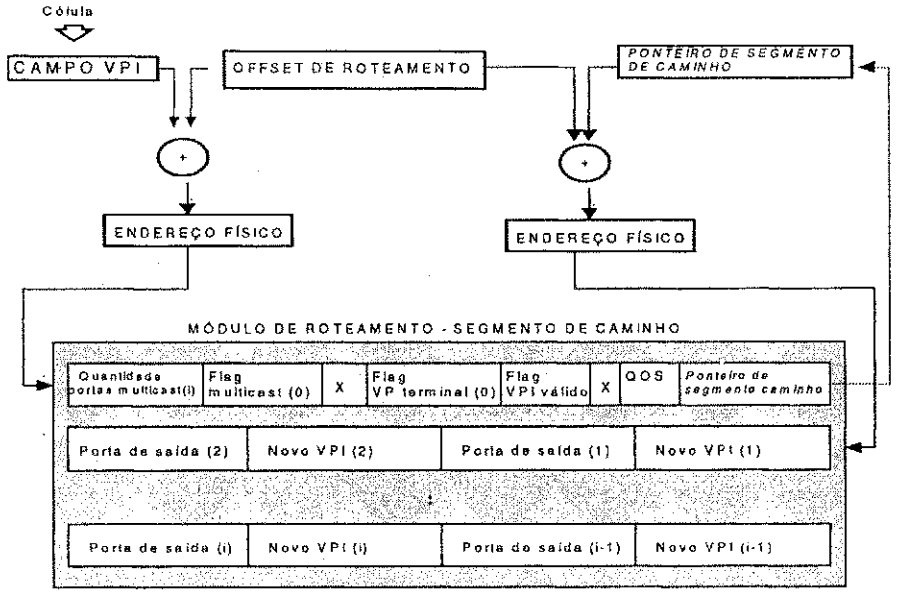


Figura 53 - Módulo de roteamento comutação de caminho virtual *multicast*

5.3.2 Comutação de canal virtual

Na comutação de canal virtual, tanto os VPIs como os VCIs são trocados, e desta forma ambos os segmentos dos módulos são acessados. O campo VPI da célula é utilizado como indexador, para acessar a palavra correspondente no segmento de caminho da mesma forma que na comutação de caminho virtual. A palavra lida contém o parâmetro denominado "Ponteiro de segmento de canal" que aponta para a área de segmento onde se iniciam os dados de canal, referente à conexão em questão. Para este segmento, o endereço físico é gerado utilizando-se o campo VCI da célula. Desta forma, para cada célula do mesmo caminho é gerado um endereço único no segmento de canal. A Figura 54 ilustra o acesso aos segmentos de caminho e canal do módulo de roteamento, para a comutação de canal virtual *unicast*.

O acesso aos segmentos do módulo de policiamento e gerenciamento é feito de maneira análoga, conforme ilustra a Figura 55 e Figura 56.

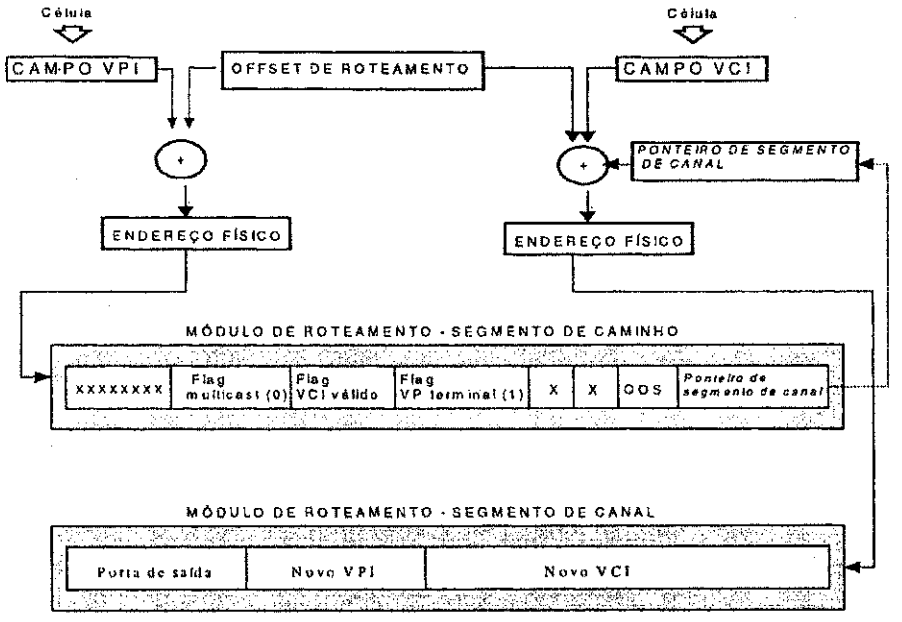


Figura 54 - Módulo de roteamento comutação de canal virtual *unicast*

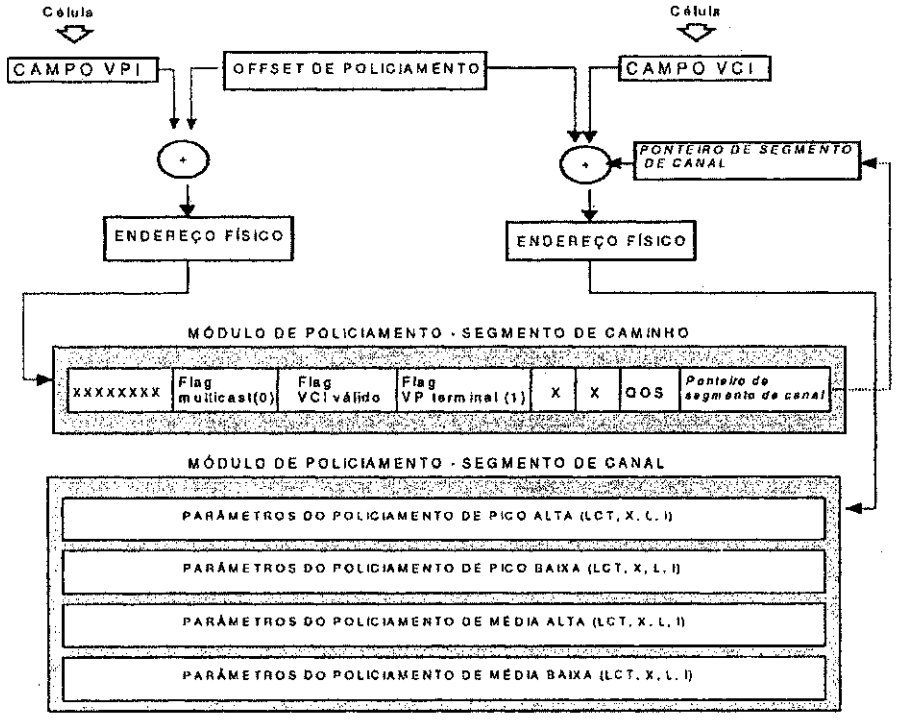


Figura 55 - Módulo de policiamento comutação de canal virtual *unicast*

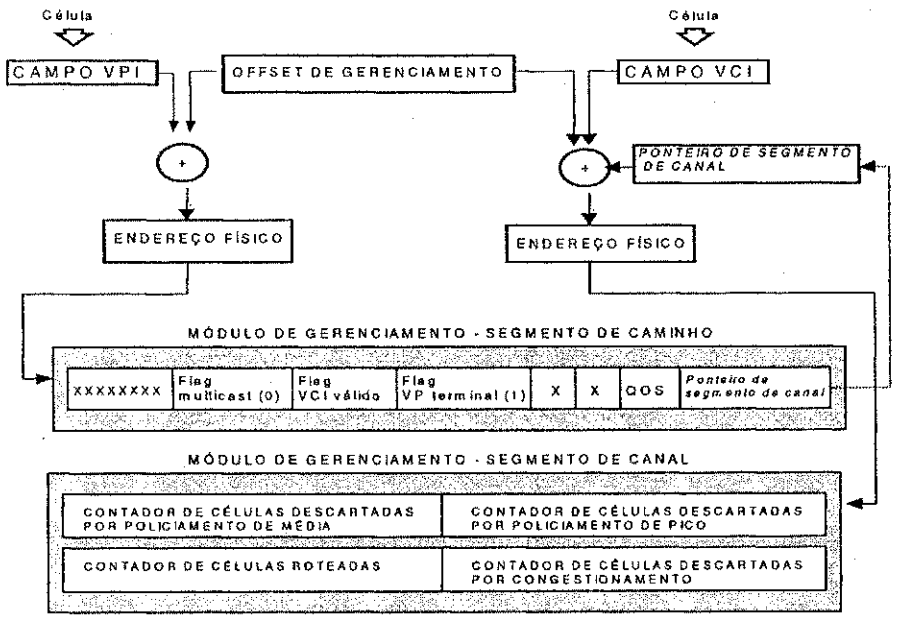


Figura 56 - Módulo de gerenciamento comutação de canal virtual *unicast*

Na comutação de canal virtual *multicast*, a célula deve ser enviada para mais de uma porta física com novos valores de VPI e VCI, a tabela deve conter os novos valores dos campos VPI e VCI que correspondem à quantidade de portas físicas necessárias para a operação, sendo preciso mais de um acesso ao segmento de canal.

O campo VPI da célula, a exemplo da operação *unicast*, continua sendo usado como indexador para acessar o primeiro dado no segmento de caminho. O parâmetro "Ponteiro de segmento de canal" gera, juntamente com o valor do campo VCI da célula, o endereço físico para acessar os valores dos vários novos campos VPIs e VCIs e portas de saída correspondentes para completar a operação *multicast*. A Figura 57 ilustra o processo.

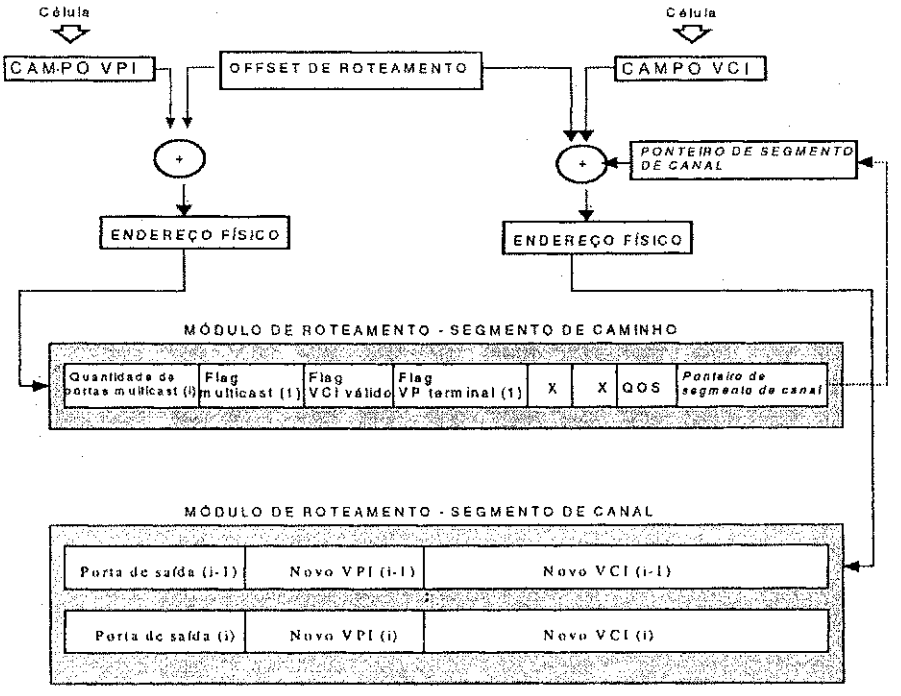


Figura 57 – Módulo de roteamento comutação de canal virtual *multicast*

Como se trata de uma comutação virtual de canal (é policiado o canal), a forma de acesso aos módulos de policiamento e de gerenciamento são idênticos à operação *unicast*.

5.3.3 Implementação da tabela

A tabela foi descrita através de um modelo VHDL, sem propósito de síntese, sendo usada apenas como um dos componentes externos do cenário de simulação do CMCA. O modelo comportamental é composto por seis processos que descrevem cada segmento (caminho e canal) dos módulos da tabela (roteamento, policiamento e gerenciamento), como mostra a Figura 58 para o segmento de caminho do módulo de roteamento. As cadeias de bits, associadas a cada processo, correspondem às palavras que contêm os parâmetros dos módulos usados na simulação.

```
-- Modelo da tabela de rotas do CMCA (TRGS)
-- O modelo simula os módulos de roteamento, policiamento e
-- gerenciamento

library STD;
library IEEE;
use STD.STANDARD.ALL;
use ieee.std_logic_1164.all;
use IEEE.std_logic_arith.all;


ENTITY TRGS IS
   PORT( CE_ROTn, CE_POLn, CE_GERn: IN std_logic;
         Rd_Wrn : in std_logic;
         addr : in std_logic_vector(16 downto 0);
         datain: IN dword;
         phy1 : in std_logic;
         dataout: OUT dword);
END TRGS;

ARCHITECTURE comportamental OF memstat IS

  CONSTANT MEM_DELAY: TIME := 20 ns;
  TYPE dword_memory is array(integer range <>) ofdword;
  signal datarot_cam, datarot_can, datapol_can, datager_can,
  datapol_cam, datager_cam : std_logic_vector(dataout'range);
  signal aux_signal : dword;
  signal aux_signal1 : dword;
  signal aux_signal3 : dword;


BEGIN

rot_caminho : process

 variable mem_caminho : dword_memory(0 to 281) :=

("00000000000000000000000000000100", -- status de inicializacao
 "00000000000000000000000000000100", -- indice de roteamento
 "00000000000000000000000000000000", -- indice de policiamento
 "00000000000000000000000000001000", -- indice de gerenciamento
 "00000000001000000000000000000000", -- 1a. celula VPI invalido = 1
 "00000000011000000000011000000000", -- VPI = 1 celula roteada por PMA
 "00000000010000000000000000000000", -- VPI = 2 celula roteada por PMB
 "00000000010000000000000000000000", -- VPI = 3 descartada por PPB
 "00000000010000000000000000000000", -- VPI = 4 descartada por PMB
 "00000000010000000000000000000000", -- VPI = 5 descartada por PMB
 "00000000010000000000000000000000", -- VPI = 6 celula roteada
 "00000000010000000000000000000000", -- VPI = 7 celula roteada por PMB
 "00000000010000000000000000000000", -- VPI = 8 celula roteada por PMB
 "00000000001000000000000000000000", -- VPI = 9 descartada por PMB
 "00000000000000000000000000001010", -- VPI = 10
 "00000000000000000000000000000000",

 variable ia : integer;

 begin

  wait on phy1;
   wait for 1 ns;
   if CE_ROTn = '0' then
    if Rd_Wrn = '1' then
     ia := to_integer(addr);
     datarot_cam <= mem_caminho(ia);
    end if;
   end if;
 end process rot_caminho;

 dt : for i in dataout'range
    generate
     dataout(i) <=((datarot_cam(i) and not (ce_rotn or not Rd_wrn)) or (datapol_cam(i) and not (ce_poln or
     not Rd_wrn)));
   end generate dt;

end comportamental;
```

Figura 58 - Descrição VHDL de parte da TRGS

5.4 Implementação da arquitetura do CMCA

O CMCA foi descrito em VHDL usando a seguinte metodologia: a arquitetura geral foi dividida em blocos, os blocos divididos em sub-blocos, conforme ilustra a Figura 59. No nível de sub-blocos foram feitas descrições comportamentais. No nível dos blocos foram feitas descrições estruturais conectando os sub-blocos.

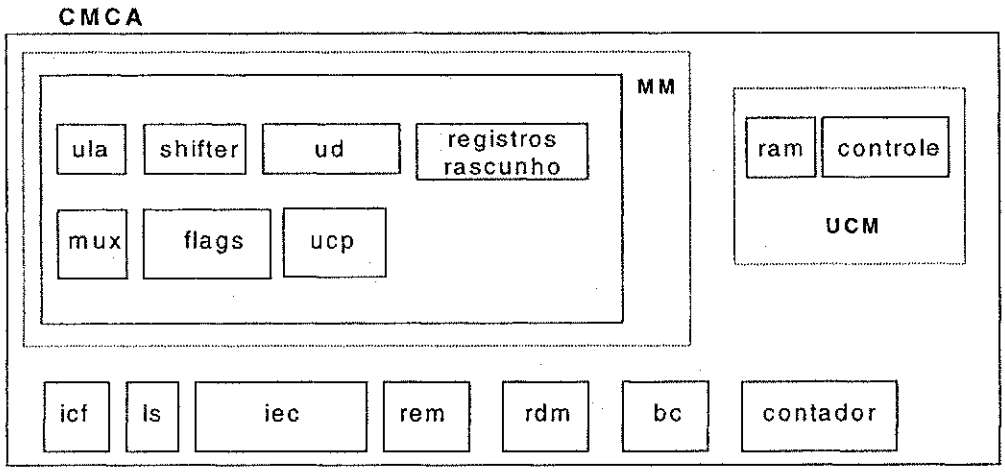


Figura 59 - Hierarquia da descrição da arquitetura

As descrições comportamentais VHDL foram feitas, tendo-se o cuidado de gerar um código sintetizável para a ferramenta de síntese *Synergy* do ambiente CADENCE [96], [97]. Grande parte das descrições usa máquinas de estados explícitas, que são muito bem aceitas por este ambiente. A Figura 60 ilustra, como exemplo, a descrição comportamental VHDL do registrador padrão de 32 bits usado no conjunto de registradores rascunho dos Módulos Microprocessadores.

```
library STD;
library IEEE;
use STD.STANDARD.ALL;
use ieee.std_logic_1164.all;
use IEEE.std_logic_arith.all;

ENTITY reg32x2 IS
  PORT (
       c        : in std_logic_vector (32 downto 1);
       a        : out std_logic_vector (32 downto 1);
       b        : out std_logic_vector (32 downto 1);
       scanin     : in std_logic;
       scanout    : in std_logic;
       scanmode   : in std_logic;
       enable1    : in std_logic;
       enable2    : in std_logic;
       write      : in std_logic;
       phy1       : in std_logic;
       phy2       : in std_logic;
       reset_n    : in std_logic
      );
END reg32x2;

ARCHITECTURE behavioral of reg32x2 IS
  signal reg_int : std_logic_vector (32 downto 1);

begin

  process(phy2, reset_n)
    begin
      if reset_n = '0'
       then
         reg_int <= "00000000000000000000000000000000";
       elsif rising_edge (phy2)
          then
            if write = '1'
             then
               reg_int  <= c;
            end if;
       end if;
end process;

  process (phy1, reset_n)
    begin
      if reset_n = '0'
       then
          a <=  "00000000000000000000000000000000";
          b <=  "00000000000000000000000000000000";
       elsif rising_edge (phy1)
         then
          if enable1 = '1'
           then
             a <= reg_int;
          end if;
          if enable2 = '1'
           then
             b <= reg_int;
          end if;
     end if;
    end process;

end behavioral;
```

Figura 60 - Descrição VHDL, registrador de 32bits

5.5 Simulação Lógica do CMCA

Para a simulação do CMCA foi elaborado um cenário de teste (*test bench*), composto por dois componentes externos, a TRGS e a ILF (Interface de Linha Física), conforme ilustra a Figura 61.

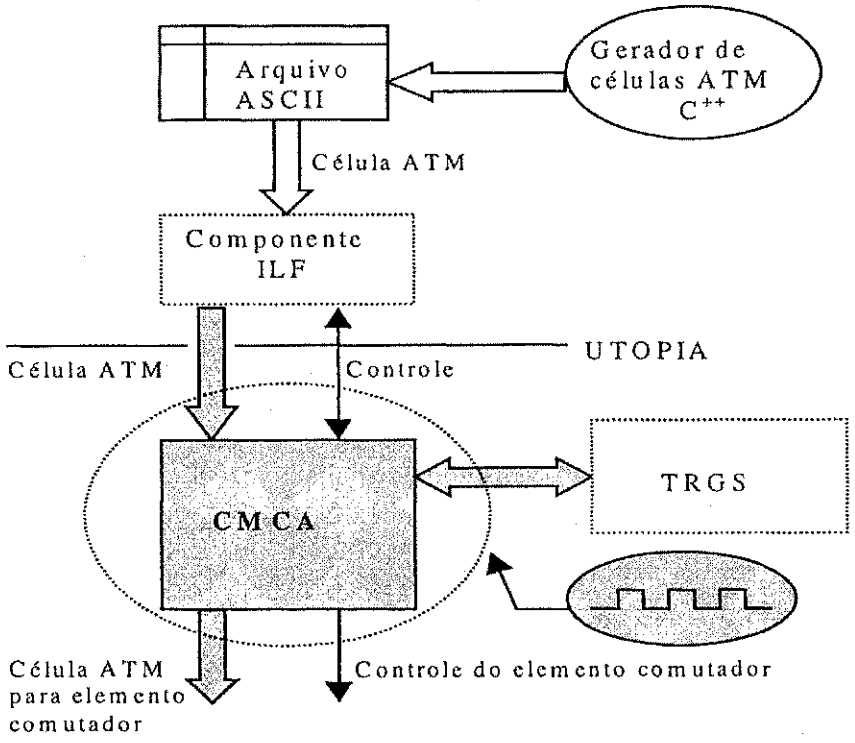


Figura 61 - Cenário de teste do CMCA

O componente ILF, descrito através de um modelo comportamental VHDL, tinha como propósito ler as células ATM armazenadas em um arquivo ASCII (geradas através de um programa desenvolvido em C^{++}) e transmiti-las para o CMCA, usando o protocolo padrão UTOPIA. O propósito do programa gerador de células era facilitar a criação de conjuntos de células ATM a partir de valores de campos de cabeçalho previamente

definidos, para ser usados na simulação. Cada célula do arquivo tinha os seus correspondentes parâmetros de roteamento, policiamento e gerenciamento armazenados no segundo componente externo TRGS. Dessa forma, cada célula do arquivo era processada pelo CMCA de acordo com os dados da TRGS. A arquitetura foi validada através da captura e análise do diagrama de tempo (*cwaves*), produzido pela ferramenta de simulação nos diferentes pontos do circuito.

A Figura 62 mostra um diagrama de tempo de simulação obtido para o processamento de uma célula válida (célula que deve ser roteada). Os acessos à TRGS são feitos através dos sinais: *Csrotn_Int*, *Cspoln_Int* e *Csgern_Int*, que selecionam os módulos de roteamento, policiamento e gerenciamento, respectivamente, e do sinal *Rd_Wrn_Int* (escrita/leitura). O início do processamento é sincronizado pelo sinal *chega*, que indica a presença da célula com os campos do cabeçalho disponíveis. A partir daí são feitos os acessos aos módulos da tabela, para os processamentos específicos e as atualizações dos parâmetros de policiamento e gerenciamento, identificadas pelas variações do sinal *Rd_Wrn_Int*. O sinal *Transmite_Int*, enviado para o Elemento Comutador, indica que é uma célula válida e deve ser roteada. Conforme indica ainda a Figura, o tempo total de processamento foi de 1,98 μs, computados entre os sinais *chega e Transmite_Int*, que corresponde a 32 ciclos do relógio de 16 MHz usado na simulação.

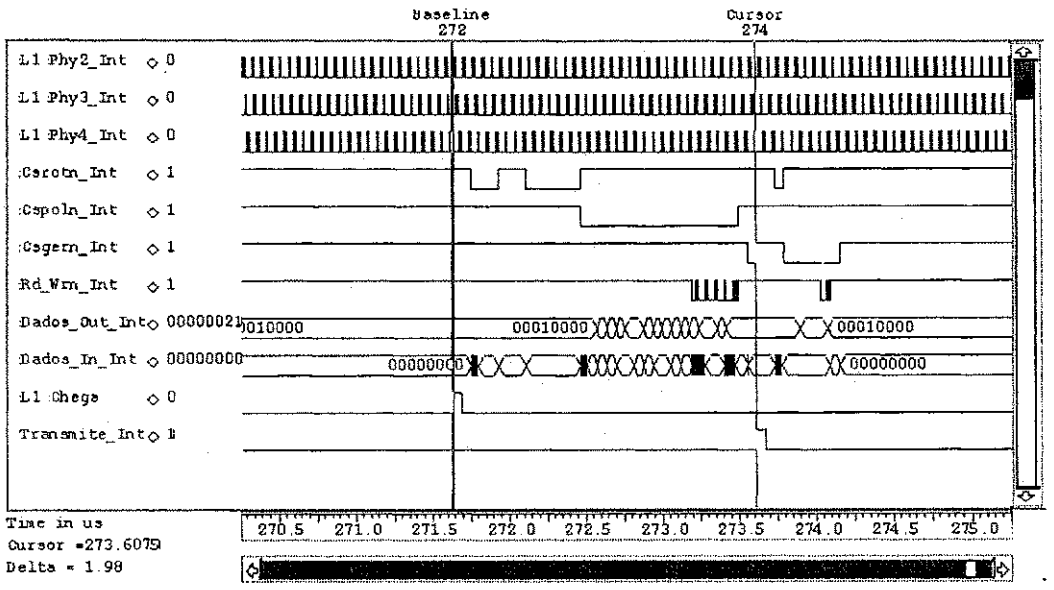


Figura 62 - Processamento de uma célula ATM

5.6 Resumo

Este capítulo descreveu a metodologia de implementação e simulação do CMCA, apresentando o modelo VHDL utilizado para descrever a arquitetura, bem como o cenário de teste utilizado para a simulação lógica. O próximo capítulo apresenta os resultados obtidos após a síntese do circuito em tecnologia ECPD07 da ATMEL e em tecnologia FPGA.

Capítulo

6

Resultados

Os resultados das simulações do CMCA (Figura 63) indicam que o processamento das funções da Camada ATM para uma célula é realizado em 32 ciclos de instrução no pior caso, conforme indica a Figura 62 para uma célula válida submetida a três policiamentos.

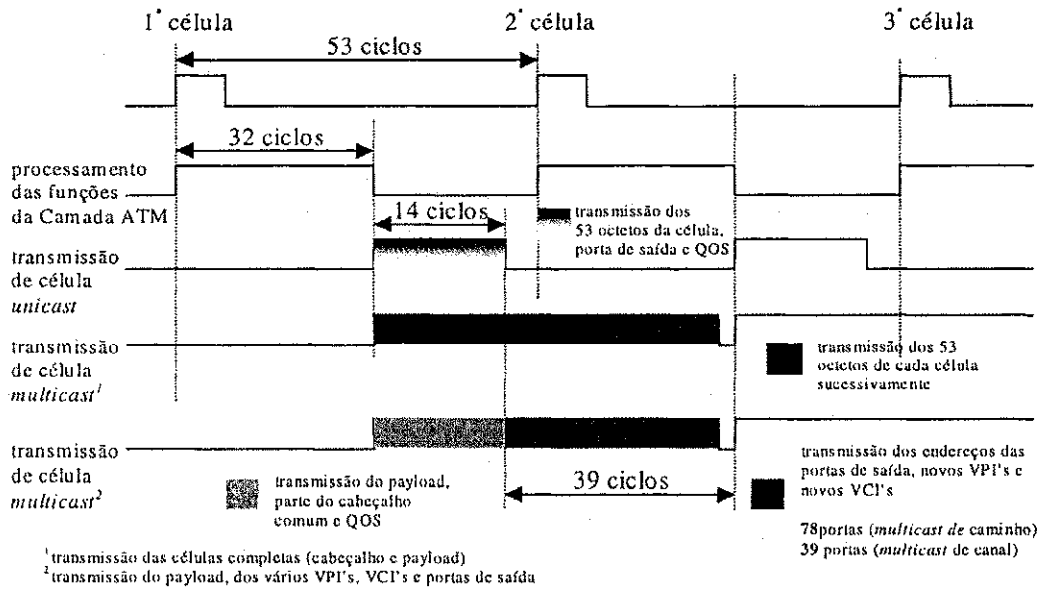


Figura 63 – Avaliação de desempenho do CMCA

A transmissão da célula *unicast* para o Elemento Comutador (barramento de 32 bits) gasta 14 ciclos, devido ao envio de 56 bytes (53 bytes da célula mais os bytes correspondentes aos parâmetros: "QOS", "endereço da porta de saída" e "quantidade de portas multicast"). Dessa forma a operação completa é executada em 46 ciclos, não havendo perda de células.

Na operação *multicast*, duas formas de transmissão (que dependem basicamente do microprograma) foram abordadas:

1) transmissão sucessiva das células *multicast* completas

2) transmissão do *payload* comum, dos novos campos de cabeçalho e dos endereços das portas de saída.

No caso 1) células completas são enviadas sucessivas vezes (o número máximo depende da quantidade de portas *multicast*) com os respectivos cabeçalhos e o mesmo *payload*. Como a transmissão de cada célula requer 14 ciclos, o número máximo de portas *multicast* (n) possível sem haver perda de célula é dado pela relação: $14n \leq 53$, ou seja, $n_{max} \leq 3$

No caso 2) o *payload* e parte do cabeçalho, comuns a todas as células, são enviados apenas uma vez, requerendo 14 ciclos. Na operação *multicast* de caminho virtual, o campo VPI (1 byte) e o "endereço da porta de saída"(1 byte) devem ser enviados para cada porta, desta forma o número máximo de portas que podem ser atendidas é dado pela relação: $14+0.5n \leq 53$, sendo $n_{max} \leq 78$ portas. Analogamente, na operação *multicast* de canal virtual, o número máximo de portas é $n_{max} \leq 39$, obtido da relação: $14+n \leq 53$; agora, o campo VCI (2 bytes) também é transmitido. Os elementos comutadores mais adequados para compor um sistema baseado no CMCA, são aqueles com capacidade para montar as células das diversas

portas de saída, contendo os campos VCI e VPI específicos, recebidos do CMCA. Os elementos comutadores baseados em divisão temporal possuem maior facilidade para esta operação se comparados com os baseados em divisão espacial.

Na implementação *standard-cells*, a área total ocupada pelo circuito resultante, após a síntese e roteamento usando a tecnologia ECPD07 da ATMEL (CMOS - 0,7 µm - 2 níveis de metal) e incluindo os *pads*, foi de 115,02 mm^2 (10,86 mm x 10,67 mm) com 29438 células, conforme ilustra a Figura 64.

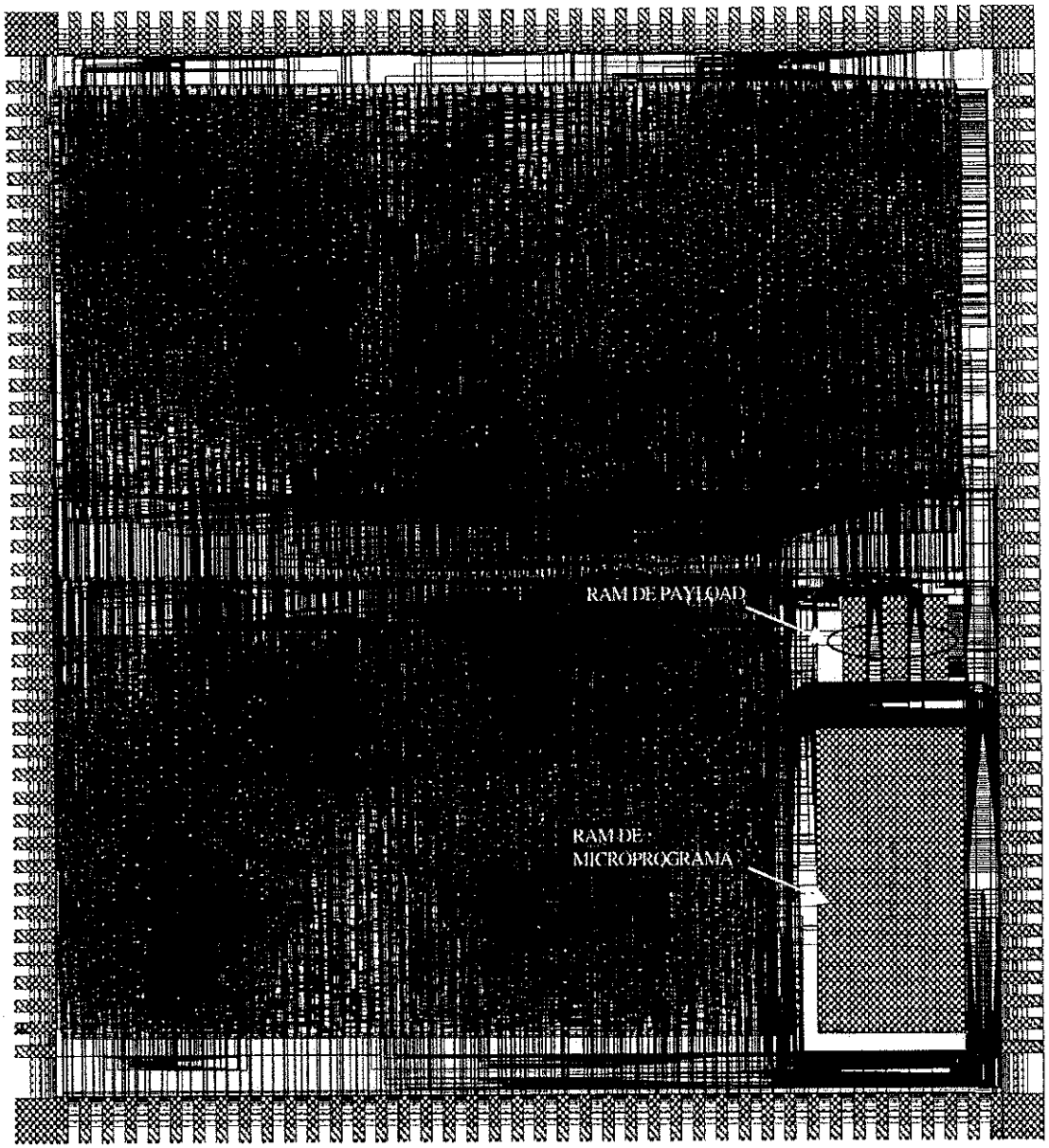


Figura 64 - Layout do CMCA

A área de roteamento (canais) ocupa 70%, devido a grande quantidade de barramentos internos, e o circuito possui 166 pinos funcionais (Apêndice C), segundo ilustra a Tabela 20. Os resultados das sínteses por bloco apresentados na Tabela 21 e Tabela 22, indicam que os módulos microprocessadores (MM), contêm 90,87% das células do CMCA, sendo os blocos ULA e UCM, os que limitam a freqüência máxima do circuito em 70 MHz. Esta freqüência foi obtida dos relatórios gerados pela ferramenta de síntese e para um resultado mais preciso deve ser feita a extração das capacitâncias do circuito.

Tecnologia CMOS 0,7 µm - 2 níveis de metal	
Área do *core*	98,34 mm^2
Área do *chip*	115,02 mm^2
Área dos *pads*	16,68 mm^2
Área de silício sem roteamento	27,43 mm^2
Área de roteamento	70,91 mm^2
No. de células	29438
No. de pinos funcionais	166
Freqüência máxima[1]	70 MHz

[1] obtida dos relatórios gerados na síntese

Tabela 20 - Resultado da síntese do CMCA usando o *Design Kit da ATMEL*

BLOCO	**FREQÜÊNCIA (MHz)**	**ÁREA (mm^2)**	**CÉLULAS**
ULA	70	0,47	504
SHIFTER	317	0,09	242
UD	125	0,96	3387
REGS. RASC	581	1,40	1704
MUX	1985	0,03	97
REG. FLAGS	383	0,03	31
UCP	202	0,33	723
MM	**70**	**3,31**	**6688**

Tabela 21 - Resultados da síntese por bloco - MM

BLOCO	**FREQÜÊNCIA (MHz)**	**ÁREA (mm^2)**	**CÉLULAS**
4 MMs	70	13,24	26752
ICF	297	0,25	349
BC	113	0,76	937
CONTADOR	112	0,22	287
LS	417	0,03	72
RDM	644	0,21	435
REM	105	0,15	429
UCM	75	9,14	117
CMCA	**70**	**24,00**	**29438**

Tabela 22 - Resultados da síntese por bloco – CMCA

A implementação FPGA [98] efetuada, tinha como propósito, verificar a viabilidade para fins de validação do projeto a baixa freqüência. O dispositivo resultante, após a síntese usando o MAX + plus II, versão 9.01 da ALTERA, foi o EPF10K200EG599-1 da família 10KE. Dos 599 pinos do dispositivos, apenas 154 são necessários às funções do circuito, 71% das células lógicas disponíveis foram utilizadas, e a freqüência máxima de operação alcançada foi de 8 MHz , conforme ilustra a Tabela 23.

Componente	EPF10K200EGC599-1
Família	10KE
Total de *LCs* usadas/disponíveis	7156/9984 (71%)
Total de flip-flops usados	2506
Total de EABs usados/disponíveis	7/24 (29%)
Total de *embedded cells* usadas/disponíveis	96/384 (25%)
Total de pinos dedicados usados/disponíveis	6/6 (100%)
Total de pinos	154
Total de pinos de entrada	22
Total de pinos de saída	68
Total de pinos bidirecional	64
Freqüência de operação	8 MHz

Tabela 23 - Resultado da síntese do CMCA em FPGA da ALTERA

Nesta implementação foi usado o código VHDL inicialmente descrito para a implementação *standard-cells* usando a ferramenta CADENCE [96] sem nenhuma preocupação de otimização para o mapeamento em FPGA. Como o conjunto de multiplexadores usados no controle do fluxo de dados dos Módulos Microprocessadores (MM) necessitavam de um número excessivo de célula lógicas, foram trocados por gates *tri-state*s. Devido à característica de reconfigurabilidade das FPGAs, a implementação da memória de microprograma (memória de controle) foi feita em ROM, eliminando dessa forma a necessidade do bloco de carga da RAM, presente na implementação *standard-cells.*

Visando comparar os resultados deste trabalho com os da literatura, escolheram-se os trabalhos de Merayo [26], Katevenis [20], Banniza [25] e Rathgeb [22].

Merayo propõe um ASIC que realiza as funções da Camada ATM de um Sistema Comutador ATM específico, possuindo as seguintes características: o ASIC contém uma parte CMOS e uma parte ECL, não possui funções de policiamento, nem executa operação de *multicast.* Os parâmetros de roteamento das células são armazenados em uma tabela de rotas externa ao *chip* e se faz necessário o uso de um microprocessador externo para a execução de operações que auxiliam nas funções de roteamento. Não possui flexibilidade para controle de elementos comutadores diversos ou alterações nas funções definidas nas especificações iniciais.

Katavenis propõe um ASIC que executa as funções da Camada ATM com elemento comutador interno baseado em divisão temporal (*buffer* compartilhado), com capacidade para 16 portas de entrada e saída. O *chip* executa um policiamento baseado no conceito de "crédito", conforme des-

crito na seção 3.8.3, possui também uma tabela de rotas interna, que armazena os parâmetros de no máximo 4096 conexões e não executa função de *multicast.* Devido a existência de um elemento comutador interno, não há flexibilidade para controle de outros elementos comutadores. O ASIC também não possui flexibilidade para se adaptar a alterações de funcionalidades.

Banniza propõe um sistema que executa as funções de um Comutador ATM, através da divisão das funcionalidades em placas e circuitos integrados dedicados. As funções da Camada ATM são executadas por três diferentes circuitos denominados: TMC_{IN}, TMC_{OUT} e POL (conforme a seção 3.8.3). A tabela de rotas é implementada externamente, através do uso de memória RAM, sendo as funções de policiamento executadas pelo circuito POL, que atua como um coprocessador. Devido a esta estrutura, existe a flexibilidade para alterações nos algoritmos de policiamento. Não há execução de *multicast.*

Rathgeb propõe um sistema que executa as funções de um Comutador ATM, através da divisão das funcionalidades em dois blocos básicos: *Line Interface Circuits* (LIC) e *ATM Switching Network* (ASN), conforme a seção 3.8.3. As funções da Camada ATM são executadas pelo bloco LIC. A tabela de rotas é implementada externamente, através do uso de memória SRAM, que armazena parâmetros de no máximo 8192 conexões. Devido a existência de um elemento comutador próprio, localizado no bloco ASN, não há flexibilidade para uso de outros elementos comutadores , bem como para se adaptar a modificações nas especificações iniciais do projeto. A função de *multicast* é executada pelo bloco ASN.

O ASIC proposto neste trabalho, executa as funções da Camada ATM, considerando o uso de um elemento comutador externo. Possui uma interface paralela que permite a conexão com diferentes tipos de elementos comutadores, porém o desempenho na operação *multicast* é maior, utilizando elementos comutadores baseados em divisão temporal (memória compartilhada, *buffer* compartilhado). Utiliza uma memória SRAM externa como tabela de rotas e possui capacidade de endereçamento de 128K posições (17 linhas de endereço). Devido a existência de uma unidade de controle microprogramada o circuito possui como principal característica, a flexibilidade para permitir a absorção de mudanças causadas por novas padronizações das funções da Camada ATM , novos tipos de serviços ou otimização, sem mudança no hardware. Como todas as funções, inclusive as de policiamento, dependem apenas do microprograma que está sendo executado, a modificação deste permite, por exemplo, alterar a quantidade de parâmetros anexados ao cabeçalho da célula, retirando ou incluindo um novo parâmetro; alterar o(s) algoritmo(s) de policiamento(s) praticado(s); controlar diversos elementos comutadores através da interface paralela existente; se adaptar às mudanças na estrutura e organização da tabela de rotas do comutador, etc. A adaptação do circuito às novas condições de funcionamento é feita sem alteração no *hardware*. A Tabela 24 apresenta um resumo do estudo comparativo efetuado.

CIRCUITO	CARACTERÍSTICAS										
	POLICIAMENTO	MULTICAST	TABELA DE ROTAS	FLEXIBILIDADE	ELEMENTO COMUTADOR	ÁREA DO CORE (mm^2)	ÁREA DO CHIP (mm^2)	NO. DE CÉLULAS/ TRANSISTORES	FREQÜÊNCIA (MHz)	TECNOLOGIA	NO. DE PINOS
Merayo	Não	Sim (de caminho)	Externa	Não	Específico	–	–	8300 transistores (parte ECL) 300000 transistores (parte CMOS)	311 (parte ECL) 74 (parte CMOS)	ECL/CMOS	319
Katevenis	Sim	Não	Interna (4 K)	Não	Específico	120	225	–	50	CMOS (0,5µm)	–
Banniza (2 ASICSs)	Sim	Sim (de caminho)	Externa	Só para o policiamento	Específico	–	–	28000 células (ASIC - TMC_{IN}) 28000 células (ASIC - TMC_{OUT})	38,88	CMOS (0,5µm)	–
Rathgeb	Não	Não	Externa (8 K)	Não	Específico	–	–	–	–	–	–
CMCA	Sim	Sim (de caminho e de canal)	Externa (128 K)	Sim	Qualquer	98,34	115,02	29438 células	70	CMOS (0,7µm)	166

Tabela 24 - Tabela comparativa

A construção de um Sistema Comutador ATM usando o CMCA é indicada na Figura 65 abaixo:

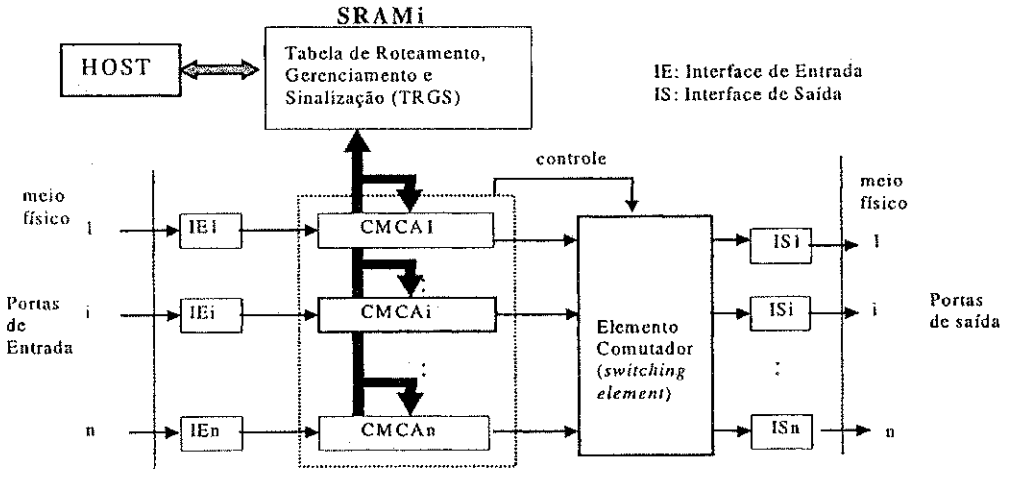


Figura 65 - Comutador ATM construído com múltiplos CMCAs

O processamento das funções da Camada ATM inclui 23 acessos à Tabela de Roteamento, Gerenciamento e Sinalização (TRGS) para leitura/escrita dos parâmetros. Uma memória SRAM comercial com tempo de acesso de 14 ns usada para implementar a TRGS, garante os acessos necessários em 280 ns. A implementação da TRGS como um banco de SRAM *dual-port* para cada porta de entrada (apenas o CMCAi específico acessa o banco SRAMi) elimina a restrição quanto ao número máximo de portas do comutador. Desta forma Comutadores ATM com múltiplas portas podem ser construídos usando o CMCA, sendo a quantidade de portas limitada apenas pela capacidade do hospedeiro de interface com múltiplos bancos SRAM e do elemento comutador.

Trabalhando-se com um relógio de 70 MHz, todas as operações, incluindo os acessos à SRAM e envio da célula para o elemento comutador são executadas em 644 ns (46 ciclos x 14 ns), desta forma, os dois padrões

comerciais de taxa de transmissão de Comutadores ATM, 155 Mbps e 622 Mbps (intervalo mínimo entre células de 2735 ns e 680 ns, respectivamente), são atendidos.

O aumento da freqüência de operação do circuito, conduz a um aumento da taxa de transmissão de células, este aumento na freqüência de operação pode ser conseguido através da implementação do circuito usando tecnologias mais recentes (0,35 μm; 0,25 μm; 0,2 μm ...). A Tabela 25 mostra a taxa de transmissão de célula em função da freqüência de operação do circuito.

Freqüência de operação do circuito (MHz)	Taxa de transmissão máxima
20	155 Mbps
50	460 Mbps
70	622 Mbps
100	921 Mbps
150	1,22 Gbps
200	1,84 Gbps

Tabela 25 - Freqüência de transmissão x taxa de transmissão de células

Capítulo

7

Conclusão

7.1 Contribuição deste trabalho

A implementação de um circuito específico que executa as funções relacionadas com a Camada ATM de um Comutador ATM é justificada pelas altas velocidades de transmissão exigidas por essas redes. Grande parte dos trabalhos sobre o processamento das funções da Camada ATM, se caracterizam pela implementação em um ou mais ASICs, desenvolvidos a partir de um conjunto próprio de especificações. Essas arquiteturas executam as funções da Camada ATM sem flexibilidade para suportar mudanças causadas pela absorção de novas padronizações, criação de novos serviços ou necessidade de otimização, que podem requerer, por exemplo, a inclusão de novos parâmetros no cabeçalho da célula e a reorganização (ordem) destes parâmetros dentro do cabeçalho; o aumento do número de portas de saída do comutador; alteração no tipo de gerenciamento das filas de células nas portas de saída, implicando na redefinição dos parâmetros associados a este gerenciamento; a modificação do algoritmo de policiamento praticado, ou

inclusão de um novo algoritmo; redefinição da estrutura e organização da tabela de rotas, alterando a forma de acesso e o manuseio dos dados da tabela; alteração no elemento comutador tendo como conseqüência uma redefinição da interface com este elemento. A incorporação dessas mudanças só é possível através de um novo ciclo de projeto e fabricação, devido à pouca ou nenhuma característica de reprogramação de tais arquiteturas.

Foi possível mostrar que é viável a implementação de um circuito que execute estas funções de forma flexível, de modo a torná-lo adaptável a mudanças sem alteração no *hardware*. Estas características foram conseguidas através do uso de microprogramação e paralelismo.

O circuito pode ser usado para implementar Sistemas Comutadores ATM com qualquer número de portas, devido a existência de sinais de interface específicos com a tabela de rotas do comutador e com o elemento comutador externo.

A comparação com trabalhos similares existentes indica que os resultados obtidos em termos de funcionalidade e desempenho do circuito atendem aos requisitos exigidos hoje pelas redes ATM.

7.2 Trabalhos futuros

Para diminuir a área de roteamento, sugere-se que seja otimizado a quantidade de ligações das saídas dos módulos "ICF", "UCP", "CONTADOR" e "registradores rascunho" para os barramentos internos A e B dos Módulos Microprocessadores (MM), esta otimização pode ser feita através da melhoria do microprograma no que diz respeito ao fluxo de dados interno, identificando ligações redundantes que podem ser retiradas. Esta otimização deve

avaliar cuidadosamente o impacto da retirada dos barramentos de forma a não prejudicar a flexibilidade do circuito.

Sugere-se também a implementação do circuito usando tecnologias mais recentes (0,35 μm, 0,25 μm, 0,20 μm, ...) de forma a se obter menor área de silício e maior freqüência de operação para a atingir taxas de transmissão na ordem de Gbps.

A implementação de um único *chip*, para executar as funções da Camada ATM e a comutação da célula, com pequena quantidade de portas de entrada e saída poderia ser obtida através da inclusão de um elemento comutador interno baseado, por exemplo, em memória compartilhada à arquitetura original, de forma a atender aos requisitos de pequenas redes locais ATM.

A característica de flexibilidade obtida com a unidade de controle microprogramada do CMCA poderia ser utilizada na concepção e implementação de circuitos para o gerenciamento de filas nas portas de saída, dotando a arquitetura de meios para alterar o controle e gerenciamento das células durante o funcionamento do sistema.

Finalmente, com relação a implementação FPGA, sugere-se que seja feita uma otimização no código VHDL, originalmente desenvolvido para a ferramenta CADENCE, de forma a adaptá-lo à ferramenta FPGA usada, garantindo uma diminuição dos caminhos críticos do circuito (atrasos) aumentando consequentemente o seu desempenho.

Capítulo

8

Referências Bibliográficas

[1] K. Habara "ISDN: A look at the future through the past," *IEEE Communications Magazine*, v.26, pp.25-32, November 1988.

[2] E. Y. Rocher "Information outlet, ULAN versus ISDN," *IEEE Communications Magazine*, v.25, pp.18-32, April 1987.

[3] A. S. Acampora, "An Introduction to Broadband Network," – New York, Plenum, 1994

[4] M. Kawarasaki and B. Jabbari, "B-ISDN Architecture and Protocol," *IEEE Journal on Select Areas in Communications*, vol. 9, N°9, pp. 1405 –1415, December 1991.

[5] W. Fisher and E. Wallmeier, "Data Communications using ATM-Architectures, protocols and resource management," *IEEE Communication Magazine*, vol. 32, August 1994.

[6] R. Y. Awdeh and H. T. Mouftah, "Survey of ATM Switch architectures," *Computer Networks and ISDN System*, vol. 27, pp. 1567-1613, 1995.

[7] K. Oshima, "A New ATM Switch Architecture Based on STS-Type Shared Buffering and its LSI Implementation," *Proceedings XIX ISS'92*, pp. 359-363, October, 1992, Yokohama, Japan

[8] H. Yamanaka and H. Saito, "Scalable Shared-Buffering ATM Switch with a Versatile Searchable Queue," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 773-784 June 1997

[9] H. J. Chao, B. Choe, Jin-Soo Park and N. Uzun., "Design and Implementation of Abacus Switch: A Scalable Multicast ATM Switch," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 830-842, June 1997.

[10] S. F. Oktug and M. U. Çaglayan, "Design and Performance Evaluation of a Banyan Network Based Interconnection Structure for ATM Switches," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 807-815, June 1997.

[11] K. Genda, "A 160 Gb/s ATM switching system using an internal speed-up crossbar switch," *Proceedings GLOBECOM'94*, pp. 123-133, 1994.

[12] M. R. Hashemi and A. L. Garcia, "The Single-Queue Switch: A Building Block for Switches with Programmable Scheduling," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 785-793, June 1997.

[13] D. Basak, A. K. Choudhury and E. L. Hahne, "Sharing Memory in Banyan-Based ATM Switches," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 881 –891, June 1997.

[14] M. R. Hashemi and L. A. Garcia, "The Single - Queue Switch: A Building Block for Switches with Programmable Scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[15] H. J. Chao, H. Cheng, Y. Jenq and D. Jeong, "Design of a Generalized Priority Queue Manager for ATM Switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[16] C. Fulton, San-qi Li and A. Lin, "Measurement-Based Performance Evaluation of an ATM Switch with External Multicasting Engine and Multiple Priority Classes," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[17] H. J. Chao and N. Uzun, "A VLSI sequencer chip for ATM traffic shaper and queue manager," *IEEE J. Solid-State Circuits*, vol. 27, pp. 1634-1643, November 1992.

[18] T. Worster and W. Fisher, "Buffering and flow control for statistical multiplexing in an ATM switch," *Proceedings ISS'95*, 1995.

[19] H. J. Chao and B. S. Choe, "Design and analysis of a large-scale multicast output buffered ATM switch," *IEEE/ACM Trans. Networking*, vol. 3, pp. 112-138, April 1995.

[20] H. J. Chao, "A Novel Architecture for Queue Management in the ATM Network," *IEEE Journal on Select Areas in Communications*, vol. 9, N°7, pp. 1110 –1118, September 1991.

[21] M. Katevenis and D. Serpanos ,"ATLAS I: A General-Purpose Single-Chip ATM Switch with Credit-Based Flow Control," *IEEE Hot Interconnects IV Symposium Proceedings*, Stanford, Ca, USA, 15-17 August 1996.

[22] E. P. Rathgeb and W. Fischer, "The MainStreetXpressCore Services Node - A Versatile ATM Switch Architecture for the Full Service Network," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[23] H. S. Silva, R. P. Ferreira e A. C. Cavalcanti, "Implementação de um Comutador ATM Usando como Matriz Básica de Comutação o Roteador de Uso Geral Reube e um Conjunto de ASICs Companions," *IX SBMICRO*, vol. 1, pp. 101 – 109, Agosto de 1994, Rio de Janeiro, RJ, Brasil.

[24] E. Nakamura e M. Strum, "Modelamento em Linguagem VHDL de uma Unidade de Policiamento para redes Locais ATM," *IX SBCCI*, vol. 1, pp. 345 – 351, Março 1996, Recife , PE , Brasil.

[25] T. R. Banniza and G. J. Eilenberger, "Design and Technology Aspects of VLSI's for ATM Switches," *IEEE Journal on Select Areas in Communications*, vol. 9, N°8, pp. 1255-1264, October 1991.

[26] L. A. Merayo, "Technology for ATM Multigigabits/s Switches," *Proceedings GLOBECOM'94*, vol. 2, pp. 117-121, November 1994.

[27] H. Yamanaka, and H. Saito, "Scalable Shared-Buffering ATM Switch with a Versatile Searchable Queue," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 830-842, June 1997.

[28] H. Saito, H. and H. Kondoh, "A 622 Mb/s 32x32 scalable shared buffer ATM switch with searchable address queue," *Proceedings GLOBECOM'96*, vol. 2, pp. 1363-1368, November 1996.

[29] T. M. Chen, M. Thomas and S. S. Liu, "Management and Control Functions in ATM Switching System," *IEEE Network*, pp. 27 – 40, July/August 1994.

[30] A. Itoh W. and Takahash, "Practical Implementation and Packaging Technologies for a Large-Scale ATM Switching System," *IEEE Journal on Select Areas in Communications*, vol. 9, N°8, pp. 1280-1288, October 1991.

[31] T. Kozaki, N. Endo and O. Matsubara, "32x32 Shared Buffer Type ATM Switch VLSI's for B-ISDN's, " *IEEE Journal on Select Areas in Communications*, vol. 9, N°8, pp. 1239-1247, October 1991.

[32] J. A. G. Lima, H. S. Silva, A. C. Cavalcanti and E. U. K. Melcher, "A Flexible Microprogrammable Controller for ATM Switch," *XIII SBMicro -International Conference on Microelectronics and Packaging (ICMP'98)* vol. 1, pp 221- 228, August 1998, Curitiba, PR, Brazil.

[33] J. A. G. Lima, A. C. Cavalcanti and E. U. K. Melcher, "MCA; One-Port Scalable Microprogrammable ATM Layer Controller," Accepted in *The 6th IEEE International Conference on Electronics, Circuits and Systems (ICECS'99)*, September 1999, Paphos, Cypros, Greece.

[34] J. A. G. Lima, E. U. K. Melcher and A. C. Cavalcanti "MCA: A Single Chip One-Port Scalable ATM Layer Controller," Accepted in *XII Symposium on Integrated Circuits and System Design (SBCCI'99)*, September 1999, Natal, Brazil.

[35] H. S. Silva, J. A. G. Lima e J. A. M. Suruagy, "Arquitetura Funcional do Comutador ATM," *Protem/Projeto COMATM (UFPb/UFPe/USP) Relatório Técnico RT 01/95, versão 1.0*, Campina Grande, 1995.

[36] H. S. Silva, J. A. G. Lima e J. A. M. Suruagy, "Arquitetura Básica e Descrição Funcional do Comutador ATM," *Protem/Projeto COMATM (UFPb/UFPe/USP) Relatório Técnico RT 01/96, versão 3.0*, Campina Grande, Fevereiro 1996.

[37] J. A. G. Lima, "Estudo sobre a especificação UTOPIA," *Protem/Projeto COMATM (UFPb/UFPe/USP) Relatório Técnico RT 04/95, versão 1.0*, Campina Grande, Outubro 1995.

[38] J. A. G. Lima, "Modelo VHDL do bloco Interface com Camada Física," *Protem/Projeto COMATM (UFPb/UFPe/USP) Relatório Técnico RT 01/96, versão 2.0*, Campina Grande, Fevereiro 1996.

[39] J. F. V. Câmara, M. L Cornélio, H. S. Silva e J. A. G. Lima, "Implementação de um Comutador ATM," *III Encontro de Iniciação Científica da UFPB*, Outubro de 1995.

[40] H. J. R Dutton and P. Lenhard, "High-Speed Networking Technology An Introductory Survey," McGraw-Hill, Third Edition – 1995

[41] CCITT, Geneva, *Recommendation I.121*: "Broadband Aspects of ISDN," 1991.

[42] Y. Kim and San-qi Li, "Timescale of interest in traffic measurement for link bandwidth allocation design," *Proceedings IEEE INFOCOM'96*, pp. 738-748.

[43] ITU-T: *Recommendation I.211*: "B-ISDN services aspects," March 1993.

[44] CCITT, *Draft Recommendation I.413*: "B-ISDN user-network interface," June 1990.

[45] ITU-T, *Recommendation I.361*, "B-ISDN ATM Layer Specification," Geneva, June 1992.

[46] ATM Forum, "ATM User-Network Interface Specification," Version 3.0, September 1993.

[47] CCITT, Geneva, *Recommendation I.321*: "B-ISDN Protocol Reference Model and its application," 1991.

[48] ITU-T, *Recommendation I.362*, "B-ISDN ATM Adaptation Layer (AAL) Functional Description," Geneva, June 1992.

[49] L. F. G. Soares, G. Lemos e S. Colcher, "Redes de Computadores das LANs MANs e WANs às Redes ATM," Editora Campos, 1ª Edição, 1995.

[50] CCITT, *Revised Recommendation I.150*: "B-ISDN ATM functional characteristics," 1992.

[51] ATM Forum, "UTOPIA, An ATM Physical Specification Level 1," version 2.01 March 1994, USA.

[52] CCITT, *Draft Recommendation I.432*: "B-ISDN user-network interface – Physical Layer specification," June 1990.

[53] A. S. Tanenbaum, "Computer Networks," – Prentice Hall, 3rd Edition , 1994.

[54] José Augusto Suruagy Monteiro "Rede Digital de Serviços Integrados de Faixa Larga (RDSI-FL), " *IV Escola de Computação*, Recife, Agosto 1994.

[55] M. Kawarasaki and B. Jabbari, "B-ISDN Architecture and Protocol," *IEEE Journal on Selected Areas in Communications*, vol 15, No. 5, June 1997.

[56] P. Newman, "ATM Technology for Corporate Networks," *IEEE Communications Magazine*, pp. 90-101, April 1992.

[57] M. de Marco and A. Pattavina, "Distributed Routing Protocols for ATM Extended Banyan Networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[58] M. De MarcoDe Marco and A. Pattavina, "Distributed Routing Protocols for ATM Extended Banyan Networks," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 925 –937, June 1997.

[59] J. Duato, S. Yalamanchili and L. Vi, "Interconnection Networks an Engineering Approach," *IEEE Computer Society*, 1st Edition, 1997.

[60] H. J. Siegel and J. Howard, "Interconnection Networks for Large-Scale Parallel Processing," McGraw-Hill, 2nd Edition, 1990.

[61] M. Dècina, "Open issues regarding the universal application of ATM for multiplexing and switching in the B-ISDN," *Proceedings ICC'91*, pp. 1258 – 1264, 1991.

[62] ITU-T: *Recommendation I.371*: "Traffic Control and Congestion Control in B-ISDN," 1993.

[63] M. W. Garret and M. Vetterli, "Congestion control strategies for packet video," *4th International Workshop Packet Video*, Kyoto, Japan, August 1991.

[64] K. Kawahara, K. Kitajima and T. Takine, "Packet Loss Performance of Selective Cell Discard Schemes in ATM Switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[65] M. Casoni and J. S. Turner, "On the Performance of Early Packet Discard," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[66] K. Kawahara, K. Kitajima and T. Takine, "Packet Loss Performance of Selective Cell Discard Schemes in ATM Switches," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 903 –913, June 1997.

[67] E. Rathgeb, "Modeling and Performance Comparison of Policing Mechanisms for ATM Networks," *IEEE Journal on Select Areas in Communications*, vol. 9, N°3, pp. 325-334, April 1991.

[68] J. A. S. Monteiro, M. Gerla and L. Fratta, "Input rate control for ATM networks – Queueing, Performance and Control in ATM," pp. 117-122, *Copenhagem*, June, 1991.

[69] Martin DePrycker, "Asynchronous Transfer Mode - Solutions for Broadband," Editora Ellies Horwood - 2ª. edição 1993

[70] ATM Forum, "ATM User-Network Interface Specification," Version 3.1, 1994.

[71] K. Law and A. L. Garcia, "A Large Scalable ATM Multicast Switch," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 844–854, June 1997.

[72] C. Fulton, San-qi Li and A. Lin, "Measurement-Based Performance Evaluation of an ATM Switch with External Multicasting Engine and Multiple–Priority Classes," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 951 –959, June 1997.

[73] J. Y. Hui and T. Renner, "Queueing analysis for multicast packet switching," *IEEE Trans. Commum.*, vol. 42, pp. 723-731, April 1994.

[74] S. Kumar and D. P. Agrawal, "On Multicast Support for Shared Memory Based ATM Switch Architecture," *IEEE Network* - January/February 1996.

[75] B. Prabhakar, N. McKeown and R. Ahuja, "Multicast Scheduling for Input - Queed Switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[76] B. Prabhakar, N. McKeown and R. Ahuja, "Multicast Scheduling for Input-Queued Switches," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 855-865, June 1997.

[77] M. H. Guo and R. S. Chang, "Multicast ATM Switches: Survey and Performance Evaluation," *Computer Communication Review*, pp. 98-130, September 1998.

[78] H. J Chao, "Overview of Implementing ATM - Based Enterprise Local Area Network for Desktop Multimedia Computing," *IEEE Communications Magazine*, April 1996.

[79] H. Kondoh, "An efficient self-timed queue architecture for ATM switch LSI's," *Proceedings IEEE CICC'94*, pp. 637-640, 1994.

[80] ATM Forum, "UTOPIA, An ATM Physical Specification Level 2," version 1.0, June 1996, USA.

[81] ITU-T, *Recommendation G.709*, "Synchronous Multiplexing Structure," Melbourne, November 1988.

[82] "Fore Systems," *htpp://www.fore.com/products/swtch/asx4000.html.*

[83] "Bay Networks," *htpp://www.baynetworks.com/products/datasheets/3169.html.*

[84] G. Veciana and G. Kesidis, "Bandwidth allocation for multiple qualities of service using generalized processor sharing," *IEEE Trans. Inform. Theory*, vol. 42, pp. 268-272, January 1996.

[85] K. Genda and N. Yamanaka, "TORUS: Terabit-per-Second ATM Switching System Architecture Based on Distributed Internal Speed-up ATM Switch," *IEEE Journal on Select Areas in Communications*, vol. 15, N°5, pp. 817-828, June 1997.

[86] D. Basak and E. L. Hahne, "Sharing Memory in Banyan - Based ATM Switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[87] T. T. Lee and C. H. Lam, "Path Switching - A Quasi-Static Routing Scheme for Large-Scale ATM Packet Switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[88] J. Rexford, F. Bonomi, A. Greenberg and A. Wong, "Scalable Architecture for Integrated Traffic Shaping and Link Scheduling in High-Speed ATM Switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, No. 5, June 1997.

[89] T. Kozaki and N. Endo, "32x32 shared buffer type ATM switch VLSI's for B-ISDN's," *IEEE Journal on Select Areas in Communications*, vol. 9, pp. 1239-1247, October 1991.

[90] William Stalling, "Computer Organization and Architecture Designing for Performance," Prentice Hall, 4th Edition, 1996

[91] M. Wilkes, "The best way to design an automatic calculating machine," *Proceedings , Manchester University Computer, Inaugural Conference*, July 1951.

[92] Z. Navabi, "VHDL Analysis and Modeling of Digital System," McGraw Hill, 1st Edition, 1993.

[93] D. L. Perry, "VHDL," McGraw Hill, 1st Edition, 1991.

[94] L. Carro, "Application Specific Systems Design and Prototyping," *XIII SBMicro -International Conference on Microelectronics and Packaging (ICMP'98)* vol. 2, pp. 59 - 97, August 1998, Curitiba, PR, Brasil.

[95] D. S. Brown and J. F. Roberts, "FPGA's Application Handbook," Kluwer Academic Publishers, 1st. Edition, 1993.

[96] "Synergy VHDL Synthesizer and Optimizer Modeling Style Guide," *Cadence System Inc.*, Version 2.0, June 1994.

[97] "Design Framework Manual," Cadence System Inc., 1988

[98] J. A. G Lima, H. S. Silva e I. R. Almeida, "Implementação em FPGA de um Controlador para Comutadores ATM," *VI Encontro de Iniciação Científica da UFPB*, 25 a 27 de Novembro de 1998.

Anexo

A

Glossário

AAL – Camada de adaptação (*ATM Adaptation Layer*)

ASIC – Circuito Integrado de Aplicação Específica (*Application Specific Integrated Circuit*)

AT – Adaptador de Terminais

ATM – Modo de Transferência Assíncrono (*Asynchronous Transfer Mode*)

BC – *Buffer* de Células

B-ISDN – *Broadband Integrated Services Digital Network* (vide RDSI-FL)

CBR – Classe de Tráfego Constante (*Constant Bit Rate*)

CCITT – Comitê Consultivo Internacional de Telegrafia e Telefonia

CLP – Bit de prioridade de perda da célula (*Cell Loss Priority*)

CMCA – Controlador Microprogramável para Comutadores ATM

FPGA – (*Field Programmable Gate Array*)

GCRA – Algoritmo Genérico de Controle de Taxa (*Generic Cell Rate Algorithm*)

GFC – Controle de fluxo genérico (*Generic Flow Control*)

HEC – Controle de erro de cabeçalho (*Header Error Control*)

ICF – Interface com Camada Física

IE – Interface de Entrada

IEC – Interface com Elemento Comutador

ILF – Interface de Linha Física

IS – Interface de Saída

ISDN – *Integrated Services Digital Network* (vide RDSI)

ITU-T– *International Telecommunication Union Telecommunication Standardization Sector*

LAN – Rede Local (*Local Area Network*)

LCT – Instante da última atualização (*Last Conformance Time*)

LS – Lógica de Sequenciamento

MAN – Rede metropolitana (*Metropolitan Area Network*)

MM – Módulo Microprocessador

N– ISDN – *Narrowband Integrated Services Digital Network* (vide RDSI-FE)

NNI – Interface rede-rede (*Network-Network Interface*)

NT – Terminação de rede (*Network Termination*)

OAM – Operação e manutenção (*Operation And Maintenance*)

PHY – Camada física

PM – Subcamada do meio físico (*Physical Medium*)

PT – Tipo do conteúdo de informação de uma célula ATM (*Payload Type*)

QOS – Qualidade do serviço (*Quality Of Service*)

RDM – Registrador de Dados da Memória

RDSI – Rede Digital de Serviços Integrados

RDSI- FE – Rede Digital de Serviços Integrados de Faixa Estreita

RDSI- FL – Rede Digital de Serviços Integrados de Faixa Larga

REM – Registrador de Endereço da Memória

RIE – Registrador Incrementador de Endereço

RMI – Registrador de Microinstrução

STM – Modo de transferência síncrono (*Synchronous Transfer Mode*)

TA – Adaptador Terminal (*Terminal Adapter*)

TC – Subcamada de convergência de transmissão (*Transmission Convergence*)

TE – Equipamento Terminal (*Terminal Equipment*)

TRGS – Tabela de Roteamento, Gerenciamento e Sinalização

UCM – Unidade de Controle Microprogramável

UCP – Unidade de Controle de Parâmetros

UD – Unidade de Decodificação

UNI – *User Network Interface* (interface usuário-rede)

UTOPIA – Protocolo padrão de sinais entre as camadas física e ATM (*Universal Test & Operations PHY Interface for Atm*)

VBR – Classe de Tráfego Variável (*Variable Bit Rate*)

VCC – Conexão de canais virtuais (*Virtual Channel Connection*)

VCI – Identificador de canal virtual (*Virtual Channel Identifier)*

VCL – Enlace de canal virtual (*Virtual Channel Link*)

VHDL – Linguagem de Descrição de Hardware (*VHSIC (Very High Speed Integrated Circuits) Hardware Description Circuits).*

VLSI – Integração em Muito Larga Escala (*Very Large Scale Integration*)

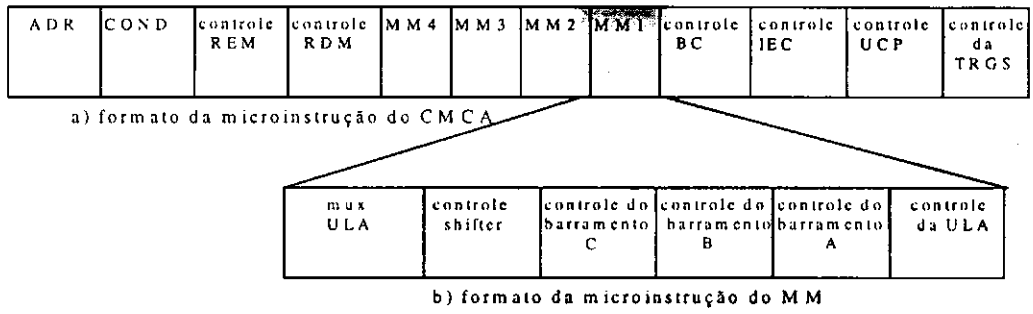
VPC – Conexão de caminhos virtuais (*Virtual Path Connection*)

VPI – Identificador de caminho virtual (*Virtual Path Identifier)*

Anexo

B

Formato da Microinstrução



B.1 - Descrição dos campos da microinstrução do CMCA

ADR (10 bits)

Endereço da próxima microinstrução

COND (5 bits)

Controle de sequenciamento das microinstruções

COND	Escolha
0	Não desvia
1	Desvia para ADR se c = 1
2	Desvia para ADR se z = 1
3	Desvia para ADR se eq = 1
4	Desvia para ADR se gt = 1
5	Desvia para ADR se lt = 1
6	Desvia para ADR se n = 1
7 até 12	Mesmo acima para MM2
13 até 18	Mesmo acima para MM3
19 até 24	Mesmo acima para MM4
25	Desvie para ADR se cell = 1
26	Desvio incondicional

Controle REM (4 bit)

Controle do registrador de endereço da memória

Campo formado por três sub-campos (**HABREM,REM e INREM**)

HABREM (2 bits)	**Função**	**INREM**	**Função**	**REM (1 bit)**	**Função**
0	escrita do MM1	0	não incrementa REM	0	ativa controle do REM
1	escrita do MM2	1	incrementa REM	1	desativa controle do REM
3	escrita do MM3				
4	escrita do MM4				

Controle RDM (3 bit)

Controle do registrador de dados da memória (RDM)

Campo formado por dois sub-campos (**HABRDM** e **RDM**)

HABRDM (2 bits)	**Função**	**RDM (1 bit)**	**Função**
0	escrita do MM1	0	ativa controle do RDM
1	escrita do MM2	1	desativa controle do RDM
3	escrita do MM3		
4	escrita do MM4		

Controle BC (2 bits)

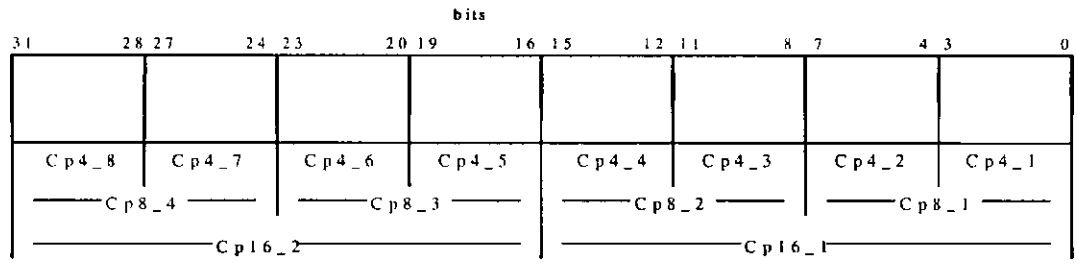
Buffer de Células

bit **hospedeiro**	**Função**	bit **transmite**	**Função**
1	ativa transmissão de célula para o hospedeiro	1	ativa transmissão de célula para o elemento comutador
0	desativa transmissão de célula para o hospedeiro	0	desativa transmissão de célula para o elemento comutador

Controle UPC (8 bits)

Unidade de Controle de Parâmetros

Campo formado por dois sub-campos (**CNBA** e **CNBD**)



CNBA (4 bits) (controle do (barramento A)	**operação**	**CNBD (4 bits)** controle do (barramento D)	**operação**
0	escreve campo Cp16_1 no barramento A	0	escreve campo Cp16_1 no barramento D
1	escreve campo Cp16_2 no barramento A	1	escreve campo Cp16_2 no barramento D
2	idem campo Cp8_1	2	idem campo Cp8_1
3	idem campo Cp8_2	3	idem campo Cp8_2
4	idem campo Cp8_3	4	idem campo Cp8_3
5	idem campo Cp8_4	5	idem campo Cp8_4
6	idem campo Cp4_1	6	idem campo Cp4_1
7	idem campo Cp4_2	7	idem campo Cp4_2
8	idem campo Cp4_3	8	idem campo Cp4_3
9	idem campo Cp4_4	9	idem campo Cp4_4
10	idem campo Cp4_5	10	idem campo Cp4_5
11	idem campo Cp4_6	11	idem campo Cp4_6
12	idem campo Cp4_7	12	idem campo Cp4_7
13	idem campo Cp4_8	13	idem campo Cp4_8

Controle IEC (2 bits)

Controle da Interface com Elemento Comutador

Controle IEC	Função
0	não transmite célula
1	transmite célula *unicast*
2	transmite célula *multicast*

Controle shifter (2 bits)

Controle do registrador de deslocamento

bit shift right	Função	bit shift left	Função
1	ativa deslocamento à direita do registrador de deslocamento	1	ativa deslocamento à esquerda do registrador de deslocamento
0	desativa deslocamento à direita do registrador de deslocamento	0	desativa deslocamento à esquerda do registrador de deslocamento

Controle da TRGS (6 bits)

Controle da memória externa (Tabela de Roteamento, Gerenciamento e Sinalização)

bit	Função
csrotn	Seleciona módulo de roteamento
cspoln	Seleciona módulo de policiamento
csgern	Seleciona módulo de gerenciamento
csceln	Seleciona memória de células de sinalização e gerenciamento
Rd_wrn	Controle de leitura/escrita dos módulos de roteamento, policiamento e gerenciamento
rdwrn	Controle de leitura/escrita da memória de células

Controle da ULA (4 bits)

Operações da Unidade Lógica e Aritmética

Operação da ULA	Controle
A + B	0000
A-B	0001
A comp B	0010
A and B	0011
A or B	0100
saída ULA: = A	0101
saída ULA: = B	0110
saída ULA: = not A	0111
saída ULA: = not B	1000
flag c := not flag c	1001
flag c : = 0	1011
flag c := 1	1100
nenhuma operação	1100

mux ULA (1 bit)

Controle do mux da ULA

Mux ULA	Função
0	Seleciona o barramento B como entrada esquerda da ULA
1	Seleciona o RDM como entrada esquerda da ULA

Controle do barramento A (4 bits)

Controla o fluxo de dados no barramento A

Controle	Operação
0	(barramento A recebe valor do contador) A ← contador
1	A ← **VPI**
2	A ← **VCI**
3	A ← **CLP**
4	A ← **PT**
5	A ← **GFC**
6	A ← **HEC**
7	A ← **RA**[1]
8	A ← **RB**[1]
9	A ← **RC**[1]
10	A ← **RD**[1]
11	A ← **RE**[1]
12	A ← **RF**[1]
13	A ← **UCP**

[1] *Registradores Rascunho:* ***(RA, RB, RC, RD, RE e RF)***

Controle do barramento B (4 bits)

Controla o fluxo de dados no barramento B

Controle	Operação
0	(barramento B recebe valor do contador) B ← **contador**
1	B ← **VPI**
2	B ← **VCI**
3	B ← **CLP**
4	B ← **PT**
5	B ← **GFC**
6	B ← **HEC**
7	B ← **RA**[1]
8	B ← **RB**[1]
9	B ← **RC**[1]
10	B ← **RD**[1]
11	B ← **RE**[1]
12	B ← **RF**[1]

[1]*Registradores Rascunho:* ***(RA, RB, RC, RD, RE e RF)***

Controle do barramento C (3 bits)

Controla a carga de registradores a partir do barramento C

Controle	Operação
0	Carga de **RA**[1] a partir do barramento **C**) **RA**[1] ← **C**
1	**RB**[1] ← **C**
2	**RC**[1] ← **C**
3	**RD**[1] ← **C**
4	**RE**[1] ← **C**
5	**RF**[1] ← **C**
6	**UCP** ← **C**
7	**nenhuma carga**

[1]*Registradores Rascunho:* ***(RA, RB, RC, RD, RE e RF)***

Anexo

C

Pinagem do CMCA

Sinal	Dimensão	E/S[1]	D/C[2]	Função	
reset_n	bit	E	C	Sinal de reset geral	**Controle Externo**
master	bit	E	C	Sinal de relógio geral	
test	bit	E	C	Indica modo de operação/teste do CMCA (test = 1 → teste)	
pol_clockin	bit	E	C	Sinal de relógio para contador de células	
RxSOC	bit	E	C	Indica a presença do primeiro dado válido da célula em RxData.	**Interface com ILF (padrão UTOPIA)**
RxEmpty_n	bit	E	C	Indica que a ILF não está habilitada a enviar células	
RxEnb_n	bit	S	C	Indica que o CMCA está habilitada a receber células.	
RxClk	bit	S	C	Cada pulso neste sinal durante a ativação de RxEnb_n, transfere os dados presentes em RxData.	
RxData	16 bits	E	D	Corresponde aos dados das células originadas da Interface de Linha Física	
scanin	bit	E	C	Entrada de teste	**Sinais de scanpath**
scanout	bit	S	C	Saída de teste	
scanmode	bit	E	C	Seleção de teste	

Convenção: Todos os sinais são ativos alto, a menos daqueles com a terminação **_n**
Ex: sinal **test** → *ativo alto; sinal* **reset_n** → *ativo baixo*

[1] **E/S** ; **E** - Entrada, **S** - Saída

[2] **D/C**; **D** - Dado, **C** - Controle

Sinal	Dimensão	E/S	D/C	Função	
dados	32 bits	E/S	D	Barramento de dados para troca de informações com a TRGS	**Interface com a TRGS (roteamento, policiamento e gerenciamento)**
Rd_Wr_n	bit	S	C	Sinal de controle de escrita/leitura na TRGS	
csrot_n	bit	S	C	Seleção do módulo de roteamento da TRGS	
cspol_n	bit	S	C	Seleção do módulo de policiamento da TRGS	
csger_n	bit	S	C	Seleção do módulo de gerenciamento da TRGS	
endereco	17 bits	S	C	Barramento de endereço	
cscel_n	bit	S	C	Seleção da área de células da TRGS	**Interface com a TRGS (área de células do hospedeiro)**
rdwr_n	bit	S	C	Controle de escrita/leitura na TRGS (área de células)	
endereco1	8 bits	S	C	Barramento de endereço	
cell_hosp	32 bits	E/S	D	Barramento de dados para troca de células com a TRGS	
strobe	bit	S	C	Relógio para transferência de dados do CMCA, para o Elemento Comutador	**Interface com Elemento Comutador**
habilita	bit	S	C	Indica que os dados em "cell_switch" são válidos	
cell_switch	32 bits	S	D	Célula para o Elemento Comutador	
pwait	bit	S	C	Informa ao hospedeiro que a TRGS está sendo acessada pelo CMCA	**Interface com o hospedeiro**
bist_test	bit	E	C	Ativa modo teste	**Sinais para teste das memórias**
bist_clk	bit	E	C	Relógio para teste	
bist_result_micro	bit	S	D	Resultado memória1	
bist_result_bc1	bit	S	D	Resultado memória2	
bist_result_bc2	bit	S	D	Resultado memória3	
bist_result_bc3	bit	S	D	Resultado memória4	
clk_carga	bit	E	C	Relógio para armazenamento do microprograma	**Sinais para carga da RAM de controle**
In_carga	bit	E	D	Entrada serial para armazenamento do microprograma	
carga	bit	E	C	Ativa armazenamento	

Anexo

D

Microprograma

Endereço	Instrução	Microinstrução	Descrição
0	1	RD; RA_1 <= RDM	Estado de reset REM com o valor 0. Leitura da posição zero da TRGS que contém o valor de estados da tabela. Armazenamento do dado lido no registrador RA.
1	2	CMP(RA, RB) if z1 =1 then goto 0	Compara RA com RB e desvia para o endereço 0 se RA = RB, o que indica que a tabela não está montada.
2	3	INREM	Incrementa REM, apontando para o endereço 1 da memória de estados, onde se localiza o índice da tabela de roteamento.
3	4	INREM	idem acima.
4	5	RD; RA_1 <= RDM; INREM;	Leitura da posição 1 da memória de estados, que contém o índice da memória de roteamento. Armazenamento do dado lido no RA e incremento do REM apontando para o endereço 2 da TRGS, onde se localiza o endereço do módulo de policiamento.
5	6	RD; RA_2 <= RDM; RA_3 <= RDM; INREM	Leitura da posição 2 e incremento para a posição 3.
6	7	RD; RA_4<= RDM;	Leitura da posição 3.
7	8	if cel = 1 then goto 8	Se não tiver chegado célula mantenha-se neste loop.
8	9	1 - RB_1 <= VPI + RA_1 2 - RB_2 <= VPI + RA_2 3 - RB_4 <= 2*VPI 4 - RB_3 <= VCI + RA_3	1- Calcula endereço físico do módulo de roteamento e armazena no RB_1 do MM1. 2- Calcula endereço físico do módulo de policiamento (modos 1 ou 2) e armazena resultado no RB_2 do processador 2. 3- Desloca o campo VPI para a esquerda. 4- Calcula endereço físico do módulo de policiamento (modos 3 ou 4) e armazena resultado no RB_3 do processador 3.

Endereço	Instrução	Microinstrução	Descrição
9	10	RC_4 <= VCI + RA_4 REM <= RB_1	Calcula endereço físico do módulo de gerenciamento (modos 3 ou 4) armazena o resultado no RC_4 processador 4 e carrega REM com endereço físico da memória de roteamento.
10	11	RD; CP_1 <= RDM; RB_4 <= RB_4 + RA_4;	Lê palavra do módulo de roteamento e armazena palavra nos registradores de campo. Calcula endereço físico do módulo de gerenciamento (modos 1 ou 2) e armazena resultado no RB_4 do processador 4.
11	12	RD_1 <= shift-right(CP_{4-6}) if c1 = 1 then goto 7	Desloca registrador CP_{4-6} (que contém o parâmetro Flag VP Válido) para a direita, afetando o carry, processador1. Desvia para endereço 7 se não é célula válida.
12	13	RC_1 <= shift-right(RD_1) if c1 = 1 then goto 88	Desloca registrador RD_1 (que contém o parâmetro Flag VP Terminal) para a direita, afetando o carry, processador1. RC_1 recebe parâmetro de flag multicast. Desvia para endereço "não terminal" se c1=1.
13	14	RD_1 <= CP_{16-1} + RA_1	Inicia o cálculo do endereço da base de canal a partir do ponteiro de segmento de canal e do RB_1, armazenando o resultado em RD_1.
14	15	1- RE_1 <= RD_1 + VCI 2- RB_4 <= VCI + RA_4 3- RB_2 <= VCI + RA_2	Calcula o endereço da base de canal e armazena o resultado em RE_1. Calcula endereço físico do módulo de gerenciamento (modos 3 ou 4) armazena o resultado no RB_4 processador 4. Calcula endereço físico do módulo de policiamento (modos 3 ou 4) e armazena resultado no RB_2 do processador 2.
15	16	REM <= RE_1	Carrega REM com endereço físico do canal.
16	17	RD; CP_1 <= RDM; goto 96;	Carrega palavra do módulo de roteamento (segmento de canal).
17	18	goto 101	Desloca registrador CP_{4-7} (que contém o parâmetro Flag VCI Válido) para a direita, afetando o carry, processador1. Desvia para endereço 7 se não é célula válida.
18	19	CMP(PT, 3) if ma =1 then goto 20	Se PT é maior que 3, a célula vai para o sistema hospedeiro para ser processada.
19	20	goto 21	Vai para o procedimento de policiamento
20	21	flag_hosp = 1 goto 7	Seta flag_hosp em 1 e envia célula para controle de payload e espera por uma outra célula.
21	22	CMP(VCI, 5) if ma = 1 then goto 23	Compara o valor do VCI com 5, se for maior vai para a endereço 23, caso contrário vai a próxima microinstrução.
22	23	CMP(VCI, 0) if ma = 1 then goto 24	Compara o valor do VCI com 0, se for maior vai para a endereço 24, caso contrário vai a próxima microinstrução.
23	24	goto 25	Chama procedimento de policiamento.

Endereço	Instrução	Microinstrução	Descrição
24	25	flag_hosp = 1; goto 7	Seta flag_hosp em 1 e envia célula para controle de payload e espera por uma outra célula.
25	26	RB_2 <= 16*VPI + RA_2	Multiplica VPI por 16 para acessar cada canal da memória de policiamento.
26	27	REM <= RB_2	Carrega REM com endereço do módulo de policiamento que contém o parâmetro LCT (policiamento pelo pico).
27	28	RD; RE_2 <= RDM; INREM;	Lê parâmetro LCT (policiamento pelo pico). Armazena palavra no registrador RE_2 do processador2. Incrementa REM apontando para o endereço do parâmetro X (policiamento pelo pico).
28	29	RD; RF_2 <= RDM; INREM;	Lê parâmetro X (policiamento pelo pico). Armazena palavra no registrador RF_2 do processador2. Incrementa REM apontando para o endereço do parâmetro L (policiamento pelo pico).
29	30	RD; RC_2 <= RDM; INREM;	Lê parâmetro L (policiamento pelo pico) Armazena palavra no registrador RC_2 do processador2. Incrementa REM apontando para o endereço do parâmetro I (policiamento pelo pico).
30	31	RD; RD_2 <= RDM; INREM;	Lê parâmetro I (policiamento pelo pico). Armazena palavra no registrador RD_2 do processador2. Incrementa REM apontando para o endereço do parâmetro LCT (policiamento agregado pela média).
31	32	RD; RE_3 <= RDM; INREM; RE_2 <= contador - RE_2	Lê parâmetro LCT (policiamento agregado pela média) . Armazena palavra no registrador RE_3 do processador3. Incrementa REM apontando para o endereço do parâmetro (policiamento agregado pela média). Armazena resultado da operação $t_a(k)$ - LCT para o RE_2 do processador2.
32	33	RD; RF_3 <= RDM; INREM; RE_2 <= RF_2 - RE_2	Lê parâmetro X (policiamento agregado pela média). Armazena palavra no registrador RF_3 do processador3. Incrementa REM apontando para o endereço do parâmetro L (policiamento agregado pela média). Armazena o resultado de X'= X - ($t_{a(k)}$ - LCT), no RE_2 do processador2.
33	34	RD; RC_3 <= RDM; INREM; CMP(RE_2, RE_2) if n2 = 1 then goto 36	Lê parâmetro L (policiamento agregado pela média). Armazena palavra no registrador RC_3 do processador3. Incrementa REM apontando para o endereço do parâmetro I (policiamento agregado pela média). Desvia se X < 0.

Endereço	Instrução	Microinstrução	Descrição
34	35	RD; RD_3 <= RDM; CMP(RE_2, RC_2) if mc2 = 1 then goto 87;	Lê parâmetro I (policiamento agregado pela média). Armazena palavra no registrador RD_3 do processador3. Compara X' com L, se for menor vai ao endereço 37.
35	36	RB_2 <= RB_2 + 8; goto 71	Excessiva para policiamento de pico. Incrementa RB_2 para apontar para o endereço de LCT (PB).
36	37	RE_2 <= 0 RE_3 <= contador - RE_3; goto 70	X' do processador 2 <= 0 Armazena resultado da operação $t_{a(k)}$ - LCT para o RE_3 do processador3. Vai para a instrução de endereço 70.
37	38	REM <= RB_2 RE_3 <= RF_3 - RE_3;	Carrega REM com endereço do módulo de policiamento que contém o parâmetro LCT (policiamento pelo pico).Armazena o resultado de X'= X - ($t_{a(k)}$ - LCT), no RE_3 do processador3. Incrementa REM para a próxima posição de memória.
38	39	RB_2 <= RB_2 + 4; INREM; RDM <= contador; CMP(RE_3, RE_3) if n3 = 1 then goto 41	Escreve na memória de policiamento na posição que contém o parâmetro LCT, modificando o LCT com o valor $t_{a(k)}$ Incrementa REM para a próxima posição de memória. Compara X' com 0, se for negativo vai ao endereço 41.
39	40	RDM <= RD_2 + RE_2; CMP(RE_3, RC_3) if mc3 = 1 then goto 42	Escreve na memória de policiamento na posição que contém o parâmetro X, modificando o X com o valor X' + I.
40	41	RB_2 <= RB_2 + 8; goto 52	Vai para o endereço LCT(MB). Excessiva para agregado pela média.
41	42	RDM <= RD_2 + RE_2; RE_3 <= 0	Escreve na memória de policiamento na posição que contém o parâmetro X, modificando o X com o valor X' + I. X' do processador 3 <= 0
42	43	REM <= RB_2;	Carrega REM com endereço do módulo de policiamento que contém o parâmetro LCT (policiamento agregado pela média).
43	44	RDM <= contador INREM;	Escreve na memória de policiamento na posição que contém o parâmetro LCT, modificando o LCT com o valor $t_{a(k)}$
44	45	RDM <= RD_3 + RE_3	Escreve na memória de policiamento na posição que contém o parâmetro X, modificando o X com o valor X' + I.
45	46	Rdn_Wr <= 1; goto 90;	Escreve na memória de policiamento.
46	47	REM <= RB_4	Carrega REM com endereço do modulo de gerenciamento (modos 1 ou 2).
47	48	INREM; RB_4 <= RDM	Incrementa endereço de memória para apontar para a segunda palavra da memória de gerenciamento.
48	49	RB_4 <= RDM	Recebe palavra que contém o parâmetro de células rotcadas.

Endereço	Instrução	Microinstrução	Descrição
49	50	RB_4 <= RB_4 + 1	Incrementa contador de células roteadas.
50	51	RDM <= RB_4; Rdn_Wr <= 1	Escreve na memória de gerenciamento na palavra que contém o contador de células roteadas. Seta o flag de gerenciamento em 1.
51	52	goto 7	Vai para o endereço 7 esperar a chegada de uma nova célula.
52	53	REM <= RB_2	Carrega REM com endereço do módulo de policiamento que contém o parâmetro LCT (policiamento de média BP).
53	54	RD; RE_2 <= RDM; INREM;	Lê parâmetro LCT (policiamento de média BP). Armazena palavra no registrador RA_2 do processador2. Incrementa REM apontando para o endereço do parâmetro X (policiamento de média BP).
54	55	RD; RF_2 <= RDM; INREM;	Lê parâmetro X (policiamento de média BP). Armazena palavra no registrador RF_2 do processador2. Incrementa REM apontando para o endereço do parâmetro L (policiamento de média BP).
55	56	RD; RC_2 <= RDM; INREM;	Lê parâmetro L (policiamento de média BP) Armazena palavra no registrador RC_2 do processador2. Incrementa REM apontando para o endereço do parâmetro I (policiamento de média BP).
56	57	RD; RD_2 <= RDM;	Lê parâmetro I (policiamento de média BP) Armazena palavra no registrador RD_2 do processador2.
57	58	RE_2 <= contador - RE_2	Armazena resultado da operação $t_a(k)$ - LCT para o RE_2 do processador2.
58	59	RE_2 <= RF_2 - RE_2	Armazena o resultado de X'= X - ($t_a(k)$ - LCT), no RE_2 do processador2.
59	60	CMP(RE_2, RE_2) if n2 = 1 then goto 86	Compara X' com X', se for negativo vai ao endereço 86.
60	61	CMP(RE_2, RC_2) if mc2 = 1 then goto 62	Compara X' com L, se for menor vai ao endereço 62.
61	62	goto 65	Excessiva por média baixa prioridade.
62	63	REM <= RB_2	Carrega rem com o endereço do LCT do policiamento de média (BP).
63	64	RDM <= contador INREM;	Modifica LCT com o valor de contador.
64	65	RDM <= RD_2 + RE_2 goto 46	Modifica X com o valor X'+ I.
65	66	REM <= RB_4	Carrega REM com endereço do módulo de gerenciamento (modos 1 ou 2).
66	67	RB_4 <= RDM	Recebe palavra que contém o parâmetro de células não roteadas.
67	68	RB_4 <= RB_4 + 1	Incrementa contador de células não roteadas.

Endereço	Instrução	Microinstrução	Descrição
68	69	RDM <= RB_4; Rdn_Wr <= 1	Escreve na memória de gerenciamento na palavra que contém o contador de células roteadas. Seta o flag de gerenciamento em 1.
69	70	goto 7	Vai esperar a chegada de uma outra célula.
70	71	RD_3 <= RDM; goto 37	Lê parâmetro I (policiamento agregado pela média). Armazena palavra no registrador RD_3 do processador3.
71	72	REM <= RB_2	Carrega rem com o endereço do LCT do policiamento de pico (BP).
72	73	RD; RE_2 <= RDM; INREM;	Lê parâmetro LCT (policiamento de pico BP). Armazena palavra no registrador RA_2 do processador2. Incrementa REM apontando para o endereço do parâmetro X (policiamento de pico BP).
73	74	RD; RF_2 <= RDM; INREM;	Lê parâmetro X (policiamento de pico BP). Armazena palavra no registrador RF_2 do processador2. Incrementa REM apontando para o endereço do parâmetro L (policiamento de pico BP).
74	75	RD; RC_2 <= RDM; INREM;	Lê parâmetro L (policiamento de pico BP) Armazena palavra no registrador RC_2 do processador2. Incrementa REM apontando para o endereço do parâmetro I (policiamento de pico BP).
75	76	RD; RD_2 <= RDM; INREM;	Lê parâmetro I (policiamento de pico BP) Armazena palavra no registrador RD_2 do processador2. Aponta para o endereço de LCT do policiamento de média BP.
76	77	RE_2 <= contador - RE_2	Armazena resultado da operação $t_a(k)$ - LCT para o RE_2 do processador2.
77	78	RE_2 <= RF_2 - RE_2	Armazena o resultado de X'= X - ($t_a(k)$ - LCT), no RE_2 do processador2.
78	79	CMP(RE_2, RE_2) if n2 = 1 then goto 85	Compara X' com X', se for negativo vai ao endereço 85.
79	80	CMP(RE_2, RC_2) if me2 = 1 then goto 81	Compara X' com L, se for menor vai ao endereço 81.
80	81	goto 65	Excessiva por pico BP.
81	82	REM <= RB_2	
82	83	RDM <= contador	Modifica LCT com o valor de contador.
83	84	RDM <= RD_2 + RE_2 INREM;	Modifica X com o valor X'+ I.
84	85	RB_2 <= RB_2 + 4 goto 52	Vai para a instrução de endereço 52 . Carrega REM com endereço do módulo de policiamento que contém o parâmetro LCT (policiamento de média BP).
85	86	RE_2 <= 0 goto 81	X <= 0 e desvia para endereço 81.

Endereço	Instrução	Microinstrução	Descrição
86	87	RE_2 <= 0; goto 62	X <= 0 e desvia para endereço 62.
87	88	RE_3 <= contador - RE_3; goto 37	Armazena resultado da operação $t_{a(k)}$ - LCT para o RE_3 do processador3. Vai para a instrução de endereço 37.
88	89	RD_1 <= CP_{8-2};	RD_1 recebe o número de portas multicast.
89	90	RE_1 <= CP_{8-1} + RB_1; goto 18	Guarda o endereço físico do módulo de roteamento - segmento de caminho.
90	91	RC_1 <= shift-right(RC_1); if c1 = 1 then goto 92	Desvia para o endereço 92 se o flag multicast está em 1.
91	92	goto 98	É um célula unicast, então desvia para endereço 98.
92	93	REM <= RE_1	Carrega REM com o endereço físico do módulo de roteamento - segmento de caminho. Seta flag de transmite e multicast em 1.
93	94	CMP(RD_1, RD_1) if z =1 then goto 46	Verifica se o RD_1 está em zero. Se verdade vai para o gerenciamento.
94	95	CP_1 <= RDM	Lê dado da memória de roteamento. Coloca o bit Csrotn em 0.
95	96	RD_1 <= RD_1 – 1; INREM; goto 93;	Decreta o número de portas multicast e incrementa endereço de memória. Desvia para comparar no endereço 93.
96	97	RC_1 <=shift-right(CP_{4-7}) if c1 = 1 then goto 7	Desloca registrador CP_{4-7} (que contém o parâmetro FLAG VCI Válido) para a direita, afetando o carry do processador1. Desvia para endereço 7 se não é célula válida.
97	98	goto 17;	Desvia para endereço 17.
98	99	REM <= RE_1	Carrega REM com o endereço físico do módulo de roteamento - segmento de caminho. Seta flag de transmite em 1 e flag de multicast em 0.
99	100	INREM;	Incrementa endereço para apontar para o próximo endereço de memória, pois é uma célula unicast.
100	101	CP_1 <= RDM goto 46	Joga nos registradores de campo e vai para o gerenciamento.
101	102	RB_2 <= 16 * VCI + RA_2 goto 26;	Multiplica VCI por 16 para acessar cada canal da memória de policiamento.