

Boletim Técnico da Escola Politécnica da USP
Departamento de Engenharia de Sistemas
Eletrônicos

ISSN 1517-3542

BT/PSI/0005

Modelamento em Linguagem VHDL
de uma Unidade de Policiamento
para Redes Locais ATM

Édson Takeshi Nakamura
Marius Strum

São Paulo – 2000

FICHA CATALOGRÁFICA

Nakamura, Édson Takeshi

Modelamento em linguagem VHDL de uma unidade de policiamento para redes locais ATM / E.T. Nakamura, M. Strum. -- São Paulo : EPUSP, 2000.

p. -- (Boletim Técnico da Escola Politécnica da USP, Departamento de Engenharia de Sistemas Eletrônicos, BT/PSI/0005)

1. Comutador ATM 2. Projeto ASIC I. Strum, Marius II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos III. Título IV. Série

ISSN 1517-3542

CDD 004.68
621.3815

MODELAMENTO EM LINGUAGEM VHDL DE UMA UNIDADE DE POLICIAMENTO PARA REDES LOCAIS ATM

Édson Takeshi Nakamura ⁽¹⁾

Marius Strum ⁽²⁾

Escola Politécnica da Universidade de São Paulo
Laboratório de Microeletrônica
Caixa Postal 8174
01065-970 – São Paulo – SP
Fax: (011) 818-5585

Resumo

Este trabalho apresenta o modelamento, em linguagem VHDL, da Unidade de Policiamento de Tráfego de um comutador ATM para redes locais ⁽³⁾. Esta unidade realiza a verificação do tráfego utilizando o algoritmo do Balde Furado de Estado Contínuo. Após um estudo da interação entre esta Unidade de Policiamento e o restante do sistema, são mostradas as simulações realizadas para a validação desta descrição. Finalmente, discute-se o uso do modelo realizado para a obtenção de uma arquitetura apropriada para o circuito.

Abstract

This work presents the behavioral description of the Traffic Policing Unit of a ATM switch for Local Area Networks, in the VHDL language. This unit performs the traffic enforcement by using the Continuous-State Leaky Bucket Algorithm. After a study about the handshaking between this Policing Unit and the other blocks of the ATM switch, simulations performed to validate this description are shown. Finally, the use of this model to obtain the architecture of the circuit is discussed.

⁽¹⁾ Bolsista DTI do CNPq. Engenheiro Eletricista, ênfase em Microeletrônica, pela Escola Politécnica da USP.

E-mail: enaka@lme.poli.usp.br

⁽²⁾ Professor associado, Escola Politécnica da USP.

E-mail: strum@lme.poli.usp.br

⁽³⁾ Este trabalho vem sendo patrocinado pelo CNPq através do projeto ProTem/COMATM.

MODELAMENTO EM LINGUAGEM VHDL DE UMA UNIDADE DE POLICIAMENTO PARA REDES LOCAIS ATM

1. Introdução

O Modo de Transferência Assíncrono (ATM) foi originalmente concebido como uma técnica para multiplexação e comutação em redes públicas. Esta técnica se baseia no conceito de comutação por células e na multiplexação por divisão no tempo assíncrona.

Nos últimos anos, o ATM vem encontrando grande aplicação em redes locais de alta velocidade. É para redes locais que se destina o comutador ATM deste projeto.

O hardware deste comutador ATM consiste em uma placa de circuito impresso conectado ao barramento PCI de um microcomputador IBM-PC ou compatível, que constitui o sistema hospedeiro [HSS94]. Esta placa tem como componentes o roteador RCube, desenvolvido pelo laboratório MASI/CAO&VLSI da Universidade de Paris VI, em cooperação com a empresa francesa Bull [MAS93], e um conjunto de um a oito exemplares do ASIC periférico denominado ABSE (ATM Basic Switching Element - Elemento Básico de Comutação ATM); uma das etapas do projeto do Comutador ATM passa então pela obtenção do leiaute do ABSE.

O roteador RCube realiza o roteamento de células ATM. O ABSE executa funções complementares de comutação ATM, entre as quais se encontra o policiamento de tráfego.

Haja vista a necessidade de que o algoritmo de policiamento seja executado muito rapidamente, optou-se pela sua implementação em hardware, através de uma Unidade de Policiamento que se comunica com o Processador de Controle do ABSE.

Este trabalho apresenta o modelamento, em linguagem VHDL, desta Unidade de Policiamento.

2. Mecanismos para Policiamento de Tráfego em Redes ATM

O mecanismo ideal de policiamento é aquele que detecta qualquer situação ilegal de tráfego, descartando todas as células que violem a especificação de tráfego (ditas células mal comportadas), e somente elas, deixando passar todas as ditas células bem comportadas.

Dos mecanismos propostos na literatura, alguns descartam células bem comportadas, outros deixam passar células mal comportadas [Rat91], [Sur94]. A referência [Sur94] apresenta uma descrição detalhada de alguns mecanismos de policiamento existentes, que se encontra resumida a seguir.

2.1. Balde Furado (Leaky Bucket)

O mecanismo do Balde Furado é baseado na pseudo-fila. A pseudo-fila consiste em um contador que é decrementado a uma taxa constante, e incrementado quando da chegada de uma célula da conexão que está sendo policiada; esse contador possui um valor máximo que corresponde ao comprimento máximo da pseudo-fila. Quando uma célula chega e encontra o contador no valor máximo, é descartada.

Há algumas variações desse mecanismo. Uma delas é o Balde Furado Virtual, em que as células excessivas, em vez de serem descartadas, são marcadas com baixa prioridade.

Outra variação é o Balde Furado com Memória. Neste esquema, são geradas permissões a uma taxa constante até um valor máximo, e cada célula que chega consome uma dessas permissões; células que não encontram permissões são armazenadas num buffer até que haja permissões disponíveis. Células que chegam quando o buffer está cheio são descartadas.

Existe também o Balde Furado Generalizado, com um limite no número de células em excesso e um espreijador de células para suavizar o tráfego.

2.2. Janelas Saltitantes e Janelas Deslizantes

O mecanismo de Janela Saltitante (Jumping Window) determina um número máximo N de células que podem ser transmitidas num intervalo de tempo (janela) T . Para isso, um contador é incrementado sempre que chega uma célula; quando o contador atingir o valor N , as células seguintes serão descartadas até que termine a janela T ; então, o contador é zerado, e inicia-se nova janela de duração T .

O mecanismo de Janela Deslizante (Sliding Window) difere do anterior pelo fato de que a chegada de uma célula deve ser lembrada por um período de tempo igual à duração da janela T ; passado o período T , o contador é decrementado em uma unidade, não havendo reinicialização da contagem.

2.3. Contadores de Pico

O mecanismo do Contador de Pico é formado por dois contadores, o primeiro análogo ao balde furado, e dito contador de média, e o segundo dito contador de pico.

No mecanismo do Contador de Pico sem Limite, o contador de média não é limitado. Assim, mesmo após passar o valor N , ele continua sendo incrementado com a chegada de novas células. Sempre que o contador de média supera o valor N , o contador de pico começa a ser incrementado numa determinada taxa, e é decrementado na mesma taxa quando o contador de média está abaixo do valor N . Sempre que o contador de pico atinge um valor máximo L_{pc} , as células são marcadas e não mais contadas no contador de média.

No Contador de Pico com Limite, há um valor máximo para o contador de média L_a , e as células marcadas continuam a ser contadas no contador de média se seu valor for menor que L_a .

2.4. Algoritmo Genérico de Controle de Taxa

O Algoritmo Genérico de Controle de Taxa (GCRA – Generic Cell Rate Algorithm), apresentado na Especificação da Interface Usuário-Rede (UNI Specification), versão 3.1, do Fórum ATM [For94], possui duas versões equivalentes: Escalonamento Virtual e Balde Furado de Estado Contínuo. Os fluxogramas para essas versões do Algoritmo Genérico de Controle de Taxa são mostrados na figura 1. Em ambos os casos, há dependência de dois parâmetros: o incremento I (entendido como espaçamento ideal entre duas células consecutivas) e o limite de antecipação L (tolerância admitida nesse espaçamento).

No Escalonamento Virtual, há uma variável que corresponde ao instante teórico de chegada da próxima célula (TAT — Theoretical Arrival Time). Quando a célula chega no instante $ta(k)$, verifica-se se essa célula chegou após o esperado ($ta(k) > TAT$); se isso ocorreu, a célula não desrespeitou a especificação de tráfego, e o valor de TAT é atualizado com o valor de $ta(k)$; se não ocorreu, então é preciso checar se a célula está dentro do limite de antecipação admitido; se $TAT > ta(k) + L$, então esse limite não foi respeitado e a célula é excessiva, caso contrário, a célula está dentro da especificação. Se a célula for excessiva, o valor de TAT não é alterado. Se a célula for válida, TAT é atualizado, somando-se o incremento I ao valor anterior de TAT .

No Balde Furado de Estado Contínuo, considera-se um balde furado de comprimento $L+I$ unidades de tempo, decrementado à taxa de uma unidade por unidade de tempo. O conteúdo do balde furado é representado por X . Quando uma célula chega em $ta(k)$, o valor do balde furado tem de ser decrementado do valor correspondente ao espaço de tempo $ta(k) - LCT$, onde LCT (Last Conformance Time) é o instante de chegada da célula válida anterior. O novo valor do balde furado é armazenado na variável X' . Se $X' < 0$, então o balde furado está vazio, e faz-se $X' = 0$, sendo a célula aceita. Se $X' > 0$, é preciso verificar se $X' > L$; se for, o conteúdo do balde furado irá atingir seu valor máximo, e a célula é considerada excessiva; se não, a célula é aceita. Finalmente, o conteúdo do balde furado deve ser incrementado pela chegada da célula válida, fazendo-se $X = X' + I$, e o valor de LCT também é atualizado ($LCT = ta(k)$). No caso de célula excessiva, os valores de X e LCT não se alteram.

2.5. Comparação entre os Mecanismos

Encontra-se na literatura uma comparação entre alguns dos mecanismos expostos (Janela Saltitante, Balde Furado e Contador de Pico)[Sur94], o que serve de base para uma escolha no âmbito deste projeto.

Um critério de suma importância é o tempo de reação do mecanismo, definido como o tempo que o mecanismo leva para detectar uma situação de violação de tráfego. Observa-se que os mecanismos de Janela Saltitante e Balde Furado apresentam um bom desempenho nesse aspecto, enquanto o desempenho do Contador de Pico foi ruim. Como o Contador de Pico também apresenta uma complexidade de implementação em hardware muito grande, ele não se mostra adequado a este projeto. O Balde Furado e a Janela Saltitante apresentaram uma baixa complexidade de implementação (como a Janela Deslizante é nitidamente mais complexa do que a Janela Saltitante, ela também pode ser descartada).

O desempenho do Balde Furado foi superior ao da Janela Saltitante no aspecto de conformidade. A conformidade é a proximidade que o mecanismo em estudo apresenta do mecanismo ideal. O Balde furado apresenta uma probabilidade de rejeição de células mais baixa para taxas de transmissão não excessivas, e alta probabilidade de rejeição de células em taxas acima da especificada.

3. Descrição VHDL

Para a Unidade de Policiamento do Comutador ATM deste projeto, foi escolhido o Algoritmo Genérico de Controle de Taxa (GCRA), versão Balde Furado de Estado Contínuo. Foi utilizado o compilador para VHDL V-System/PC, versão 1.3 (rodando em computadores IBM-PC e compatíveis) [Mod92].

Primeiramente, devem ser feitas algumas considerações sobre a interação existente entre a Unidade de Policiamento e o Processador de Controle do ABSE [COM95]. A figura 2 mostra o diagrama de blocos do ABSE.

As células ATM não passam pela Unidade de Policiamento. Isto para que o policiamento não altere o fluxo de células e não introduza atrasos. A única função da Unidade de Policiamento é verificar o cumprimento das especificações de tráfego. Assim, é necessário que o Processador de Controle informe à Unidade de Policiamento a chegada de uma célula ATM; esta informação é passada pelo sinal CHEGA (ativo em nível '1'), que é ativado no instante de chegada de uma célula ao ABSE.

Uma célula pode sofrer mais de uma verificação de tráfego. Em primeiro lugar, a célula deve passar por uma verificação de tráfego para taxa de pico; caso não ocorra violação, ela deverá sofrer uma nova verificação, desta vez para taxa média. O número de verificações também depende de sua prioridade. Caso uma célula de alta prioridade viole a especificação de tráfego, ela tem sua prioridade rebaixada e deve ser feito um novo policiamento, agora para o tráfego de prioridade mais baixa. Como a prioridade é marcada no cabeçalho da célula, que não passa pela Unidade de Policiamento, quem determina se devem ou não ser feitas novas verificações é o Processador de Controle. Para isso, existe o sinal POL (ativo em nível '1'). Assim, o Processador de Controle requisita uma nova verificação ativando o sinal POL; sempre que POL for ativado, o algoritmo de policiamento será executado.

As especificações de tráfego são totalmente determinadas pelos parâmetros X, LCT, I e L, que ficam nas tabelas armazenadas na memória do ABSE. Como a Unidade de Policiamento se comunica apenas com o Processador de Controle, como pode ser visto na figura 2, ela não tem acesso à memória. Assim, os dados são passados da memória para o Processador, e daí para a Unidade de Policiamento. É preciso definir entradas para esses valores (vetores de bits): Xin, LCTin, Iin e Lin.

Da mesma forma, as tabelas precisam ser atualizadas após cada verificação de tráfego. Assim, a Unidade de Policiamento precisa fornecer ao Processador de Controle os novos valores de LCT e X; estes são fornecidos pelas saídas LCTout e Xout (vetores de bits).

Os vetores de bits, tanto para entrada como para saída, não têm ainda seus tamanhos especificados no âmbito do projeto do Computador ATM. Para este trabalho, a fim de simplificar as simulações, foram usados vetores de quatro bits.

Feito o policiamento, a Unidade de Policiamento precisa indicar ao Processador de Controle se houve ou não violação de tráfego, para que a célula tenha sua prioridade rebaixada ou seja descartada. Isto é indicado pela saída EXCESSO: quando ocorrer violação, EXCESSO='1'; se não, EXCESSO='0'.

Há também uma entrada de relógio, clk, para incrementar o contador interno, que serve de referência para marcar o tempo de chegada das células.

São sinais internos da Unidade de Policiamento: LCT, X, I, L, e tak (equivalente a $ta(k)$), além de cont (valor do contador interno). Estes sinais internos foram definidos como decimais (tipo INTEGER), pois a versão de VHDL utilizada só efetua operações aritméticas com este tipo de sinais e variáveis.

Isto levou à necessidade de criar duas funções: uma que converte números binários em valores decimais (função F1), e outra que converte valores decimais em números binários (função F2).

A descrição está dividida em três processos. Sua listagem se encontra no anexo 1. O processo "conta" é o contador interno, usado como referência de tempo, incrementando o valor do sinal cont a cada borda positiva de clk. O processo "chegou" é ativado pelo sinal CHEGA, e atualiza o valor do sinal tak igualando-o ao sinal cont na borda positiva de CHEGA.

O processo "teste" é ativado pela borda positiva de POL, e executa o algoritmo de policiamento propriamente dito. Este processo apresenta três variáveis internas, todas definidas como decimais (tipo INTEGER). A variável Xlinha equivale a X' no algoritmo do Balde Furado de Estado Contínuo. A variável Xis equivale ao novo valor de X, que é convertido por F2, dando o valor da saída Xout, no caso de célula não excessiva; ela se fez necessária porque não é possível forçar valores para um sinal (no caso, o sinal X) duas vezes durante a descrição. Finalmente, há a variável temp, que foi incluída prevendo a futura utilização desta descrição na síntese lógica da Unidade de Policiamento; pretende-se evitar a necessidade de um circuito aritmético com três operadores no seu hardware, para executar a operação $X - (ta(k) - LCT)$.

As simulações realizadas são discutidas a seguir.

Para um clock de frequência de 20 MHz (período de 50 ns), que é o passo do contador, foram fixados como especificações de tráfego: incremento (espaçamento ideal entre as células) de 150 ns, o que equivale a $I=3$; e limite de antecipação de 50 ns, ou $L=1$.

Cada célula é policiada uma única vez nesta simulação; isto corresponde a uma verificação de tráfego para uma sequência de células de mesma prioridade, para um único tipo de especificação de tráfego (por exemplo, para taxa de pico); para cada célula policiada, as saídas LCTout e Xout fornecem os novos valores de LCT e X; estes valores devem ser passados às entradas LCTin e Xin para o policiamento da próxima célula.

Esta simulação é suficiente para a validação do modelo. O caso real, em que uma célula sofre várias verificações de tráfego, corresponde a repetir o procedimento desta simulação para cada especificação de tráfego (alta e baixa prioridade, pico e média) de maneira entrelaçada, de modo que cada célula sofra todas as verificações necessárias antes da chegada da próxima célula. O controle de todas essas verificações fica a cargo do Processador de Controle.

As células chegaram nos instantes de tempo mostrados na tabela 1.

Célula	Instante de Chegada
1	100 ns
2	250 ns
3	300 ns
4	350 ns
5	450 ns
6	550 ns

Tabela 1: fluxo de células na simulação da Unidade de Policiamento.

A célula 1 é aceita. A célula 2 chegou 150 ns após a anterior, respeitando o incremento, e portanto deve ser aceita. A célula 3 chega 50 ns após a célula 2, violando o incremento e o limite de antecipação ($I - L = 2$, o que equivale a 100 ns); portanto, a célula 3 deve ser descartada.

A célula 4 chega 100 ns depois da célula 2, que é a última válida; isto desrespeita o incremento, mas está dentro do limite de antecipação, e a célula 4 deve ser aceita, fazendo no entanto com que o conteúdo do balde furado seja maior que o incremento ($X > I$). A célula 5 chega 100 ns após a célula 4, o que também viola o incremento, mas está dentro do limite de antecipação; entretanto, como o balde furado tinha um conteúdo maior que o incremento, seu valor atingirá o limite com a chegada desta célula; logo, a célula 5 será descartada.

A célula 6 chega 200 ns após a célula 4, tempo maior que o incremento; portanto, a célula 6 deve ser aceita.

Resumindo, as células 3 e 5 devem ser descartadas, e as demais aceitas. As simulações realizadas concordam com os resultados esperados, o que permite concluir que o modelamento está correto. Os resultados das simulações se encontram no Anexo 2.

Feita a validação da descrição comportamental, a próxima etapa do projeto será sua utilização para realizar a síntese de arquitetura da Unidade de Policiamento, seguida da síntese lógica dos blocos que a compõem. O objetivo final será a síntese de leiaute desta unidade.

O modelamento em VHDL foi utilizado para extração de grafos de fluxo de controle e de fluxo de dados (CFG e DFG), o que constitui a primeira etapa da síntese de arquitetura. O estudo desse grafos indica que a Unidade de Policiamento será composta por um somador/subtrator, um comparador, nove registradores, uma unidade de controle e um buffer de saída. Para esta arquitetura, é esperada a execução do algoritmo em sete estados de controle.

4. Conclusões

A implementação em hardware de um circuito que efetue o policiamento de uma rede ATM se justifica pelas altas velocidades de operação exigidas por essas redes. Numa aplicação em redes locais, além da velocidade, a baixa complexidade de implementação desse mecanismo é fundamental. O mecanismo do Balde Furado, além de preencher todos esses requisitos, é o que apresenta um desempenho mais próximo do mecanismo ideal no aspecto de probabilidade de rejeição de células. Isto justifica seu uso na Unidade de Policiamento proposta neste trabalho.

O modelamento desta Unidade de Policiamento em linguagem VHDL, no nível comportamental, teve por objetivo testar e validar, através de simulações, a escolha do Algoritmo do Balde Furado de Estado Contínuo, e definir a sua interação com as demais

partes constituintes do Sistema Comutador ATM no qual está inserido, servindo para validação desse sistema como um todo. As simulações realizadas abrangeram os casos de: célula aceita por não violar o incremento I; célula aceita por não violar o limite de antecipação L, mesmo violando o incremento I; célula rejeitada por violar o incremento I, respeitar o limite de antecipação L, mas encontrar o balde furado com conteúdo maior que I; e célula rejeitada por violar o incremento I e o limite de antecipação L.

Esta descrição comportamental também permite a obtenção do hardware da Unidade de Policiamento, servindo de base para um processo de síntese de arquitetura e posterior síntese lógica dos blocos componentes. O processo de síntese de arquitetura vem sendo feito manualmente.

5. Referências Bibliográficas

[HSS94] Da Silva, Hamilton Soares; Cavalcanti, Antonio Carlos; Ferreira, Romulo Pires Coelho; Giozza, William Ferreira. "Implementação de um Comutador ATM Usando como Matriz Básica de Comutação o Roteador de Uso Geral *RCube* e um conjunto de ASIC's *Companions*". Anais do IX Congresso da Sociedade Brasileira de Microeletrônica, Rio de Janeiro, 1994.

[MAS93]MASI. Internal Technical Report. "RCube: A High Performance Reconfigurable Packet Switching Router". Laboratório MASI/CAO&VLSI, UPMC, Paris, França, 1993.

[Rat91] Rathgeb, Erwin P. "Modeling and Performance Comparison of Policing Mechanisms for ATM Networks". IEEE Journal on Selected Areas in Communications, vol. 9, no. 3, pp. 325-334, Apr. 1991.

[Sur94] Monteiro, José Augusto Suruagy. "Rede Digital de Serviços Integrados de Faixa Larga (RDSI-FL)". IX Escola de Computação, Recife, 1994.

[For94] The ATM Forum Technical Committee. "User-Network Interface (UNI) Specification, Version 3.1". 1994.

[Mod 92] Model Technology Incorporated. "V-System/PC - User's Manual". 1992.

[COM95] ProTem/Projeto COMATM (UFPb/UFPe/USP/Quickchip). Relatório Técnico RT-01/95, Versão 1.0. "Arquitetura Básica e Descrição Funcional do Comutador ATM". Campina Grande, 1995.

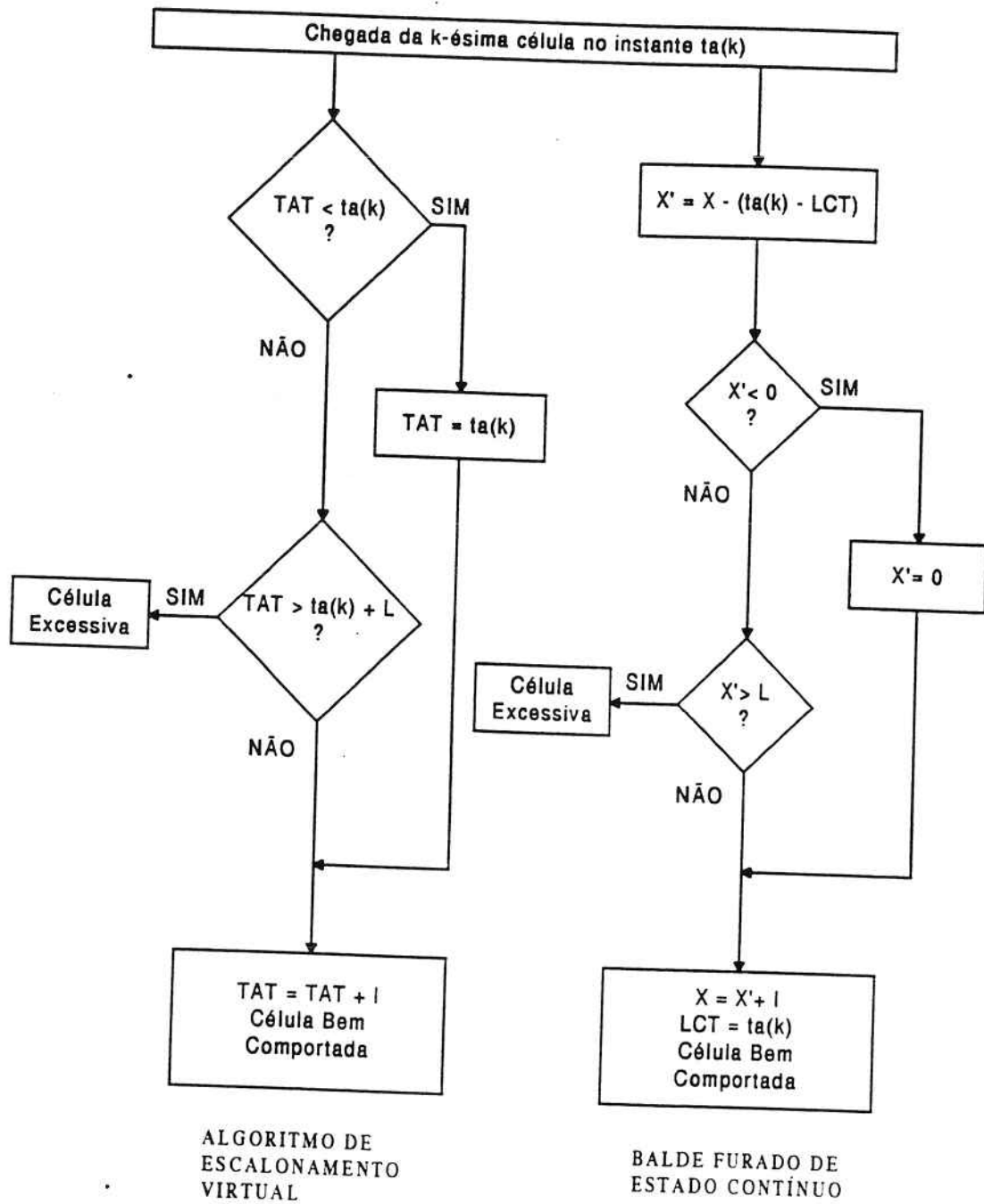


Figura 1 : Algoritmo Genérico de Controle de Taxa (GCRA).

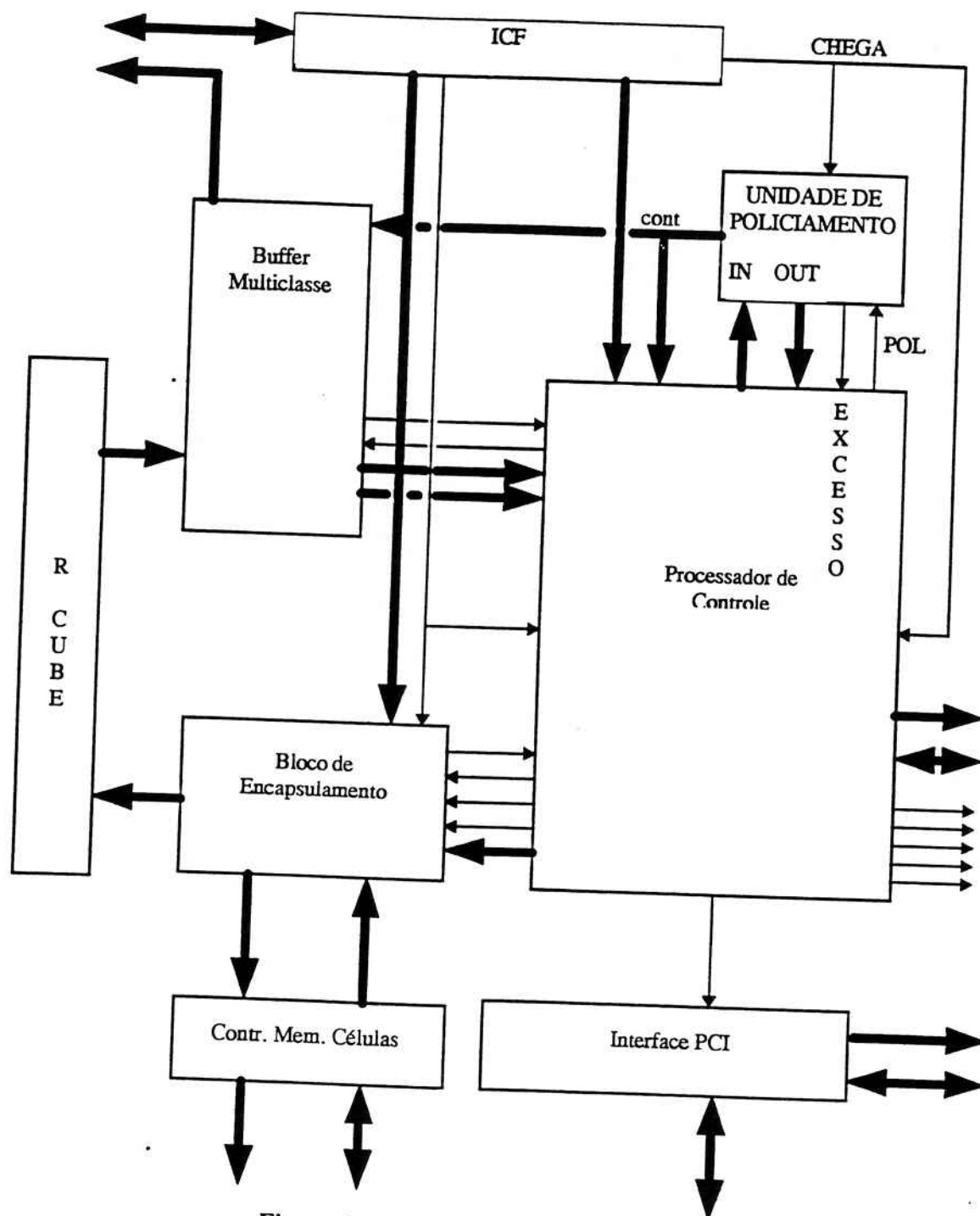


Figura 2: Diagrama de Blocos do ABSE.

Anexo 1: Listagem do modelamento em VHDL da Unidade de Policiamento.

```
-- Descricao VHDL
-- Unidade de Policiamento

entity policiamento is
port (
    phi1 : in bit;-----: primeira fase de clock
    phi2 : in bit;-----: segunda fase de clock
    MASTER_RESET : in bit;-----: sinal de inicializacao
    CHEGA : in bit;-----: instante de chegada da celula ATM
    POL : in bit;-----: dispara o algoritmo de policiamento
    LCTin : in bit_vector(0 to 3);---: valor de LCT
    Iin : in bit_vector(0 to 3);---: valor do intervalo I
    Xin : in bit_vector(0 to 3);---: valor de X
    Lin : in bit_vector(0 to 3);---: valor de L
    LCTout : out bit_vector(0 to 3);---: novo valor de LCT apos policiamento
    Xout : out bit_vector(0 to 3);---: novo valor de X apos policiamento
    EXCESSO : out bit;-----: indica violacao de trafego
);
end policiamento;

architecture behavioral of policiamento is
    signal I : integer;-----: I, variavel de entrada do algoritmo,
    signal L : integer;-----: intervalo de tempo ideal entre duas
    signal X : integer;-----: celulas consecutivas
    signal LCT : integer;-----: L, variavel de entrada do algoritmo,
    signal cont : integer := 0;---: limite de antecipacao da celula
    signal tak : integer := 0;---: X, variavel do algoritmo,
    signal LCTout : integer;-----: corresponde ao valor do balde furado
    signal Xout : integer;-----: no instante de chegada
    signal EXCESSO : integer;-----: LCT, variavel do algoritmo,
    signal Iin : integer;-----: corresponde ao tempo de chegada da
    signal Xin : integer;-----: ultima celula valida
    signal Lin : integer;-----: contador interno
    signal LCTin : integer;-----: ta(k), valor do contador no instante
    signal Xout : integer;-----: de chegada da celula a ser policiada
    signal EXCESSO : integer;-----:

-- Funcao F1: converte um sinal binario em valor decimal correspondente
function F1 (e: bit_vector(0 to 3)) return integer is
begin
    variable s : integer := 0;
    if e(0)='1' then s:=s+8; end if;
    if e(1)='1' then s:=s+4; end if;
    if e(2)='1' then s:=s+2; end if;
    if e(3)='1' then s:=s+1; end if;
    return s;
end;

-- Funcao F2: converte um valor decimal em um sinal binario
function F2 (e: integer) return bit_vector is
begin
    variable s: bit_vector(0 to 3);
    variable aux: integer;
    aux:=e;
    if aux>=8 then s(0):='1';
    aux:=aux-8;
    else s(0):='0';
    end if;
    if aux>=4 then s(1):='1';
    aux:=aux-4;
    else s(1):='0';
    end if;
    if aux>=2 then s(2):='1';
    aux:=aux-2;
    else s(2):='0';
    end if;
    if aux>=1 then s(3):='1';
    aux:=aux-1;
    else s(3):='0';
    end if;
    return s;
end;

begin
    X<=F1(Xin);-----[IN]
    LCT<=F1(LCTin);-----[IN]
    L<=F1(Lin);-----[IN]
    I<=F1(Iin);-----[IN]
    -- Leitura dos valores das variaveis de entrada X, LCT, I e L
    -- Bloco teste: executa o algoritmo de policiamento propriamente dito;
    teste : block
    begin
        process (POL)
            variable xlinha : integer;---: corresponde 'a variavel X'
            variable Xis : integer;---: corresponde ao novo valor de X,

```

```

variable temp : integer; -- apos a execucao do algoritmo
-- variavel temporaria para o
-- calculo de X'
begin
if POL='1' then
temp := X + LCT ; --(1)
Xlinha:= temp - tak; --(2)
X' = X - ( ta(k) - LCT ) --(3)
if Xlinha<0
--(4)
: ( X' < 0 ? )
then Xlinha:=0; --(5)
Sim => X' = 0
end if;
Nao => prossegue
if Xlinha>L --(6)
: ( X' > L ? )
then EXCESSO<='1'; --(7)
LCTout<=LCTin; --[OUT] --(8)
Xout<=Xin; --[OUT] --(9)
: Sim => Celula viola especificacao de trafego
LCT e X nao se alteram
else EXCESSO<='0'; --(10)
LCTout<=F2(tak); --[OUT] --(11)
Xis:=Xlinha+I; --[OUT] --(12)
Xout<=F2(Xis); --(13)
: Nao => Celula valida
LCT = ta(k) --[OUT]
X = X' + I
end if; end if;
end process;
end block;

-- Bloco chegou: atualiza o valor de tak, quando da chegada de uma nova celula
-- ativado pelo sinal CHEGA
chegou: block
begin
process (CHEGA)
begin
if (CHEGA='1') --(14)
then
tak<=cont; --(15)
end if;
end process;
end block;

-- Bloco conta : atualiza o contador interno (borda positiva de phil)
conta : block
begin
process (phil)
begin
if (phil='1') --(16)
then
cont<=cont+1; --(17)
end if;
end process;
end block;
end;

```


ANEXO 2 - Resultados da Simulação
Unidade de Policiamento

ns	clk	CHEGA	POL	LCTin	Xin	Iin	Lin	LCTout	Xout	EXCESSO	
0	0	0	0	0000	0000	0000	0000	0000	0000	0	
0	0	0	0	0000	0000	0011	0001	0000	0000	0	LCT=0; X=0; I=3; L=
25	1	0	0	0000	0000	0011	0001	0000	0000	0	
50	0	0	0	0000	0000	0011	0001	0000	0000	0	
75	1	0	0	0000	0000	0011	0001	0000	0000	0	
100	0	0	0	0000	0000	0011	0001	0000	0000	0	
100	0	1	0	0000	0000	0011	0001	0000	0000	0	1a. célula 100ns
101	0	1	1	0000	0000	0011	0001	0000	0000	0	
101	0	1	1	0000	0000	0011	0001	0010	0011	0	LCT=2; X=3
125	1	1	1	0000	0000	0011	0001	0010	0011	0	
150	0	1	1	0000	0000	0011	0001	0010	0011	0	
175	1	1	1	0000	0000	0011	0001	0010	0011	0	
200	0	1	1	0000	0000	0011	0001	0010	0011	0	
200	0	0	0	0010	0011	0011	0001	0010	0011	0	
225	1	0	0	0010	0011	0011	0001	0010	0011	0	
250	0	0	0	0010	0011	0011	0001	0010	0011	0	
250	0	1	0	0010	0011	0011	0001	0010	0011	0	2a. célula 250ns
251	0	1	1	0010	0011	0011	0001	0010	0011	0	
251	0	1	1	0010	0011	0011	0001	0101	0011	0	LCT=5; X=3
275	1	1	1	0010	0011	0011	0001	0101	0011	0	
275	1	0	0	0101	0011	0011	0001	0101	0011	0	
300	0	0	0	0101	0011	0011	0001	0101	0011	0	
300	0	1	0	0101	0011	0011	0001	0101	0011	0	3a. célula 300ns
301	0	1	1	0101	0011	0011	0001	0101	0011	0	
301	0	1	1	0101	0011	0011	0001	0101	0011	0	
325	1	1	1	0101	0011	0011	0001	0101	0011	1	célula excessiva
325	1	0	0	0101	0011	0011	0001	0101	0011	1	
350	0	0	0	0101	0011	0011	0001	0101	0011	1	
350	0	1	0	0101	0011	0011	0001	0101	0011	1	4a. célula
351	0	1	1	0101	0011	0011	0001	0101	0011	1	
351	0	1	1	0101	0011	0011	0001	0111	0100	0	LCT=7; X=4
375	1	1	1	0101	0011	0011	0001	0111	0100	0	
400	0	1	1	0101	0011	0011	0001	0111	0100	0	
400	0	0	0	0111	0100	0011	0001	0111	0100	0	
425	1	0	0	0111	0100	0011	0001	0111	0100	0	
450	0	0	0	0111	0100	0011	0001	0111	0100	0	
450	0	1	0	0111	0100	0011	0001	0111	0100	0	5a. célula 450ns
451	0	1	1	0111	0100	0011	0001	0111	0100	0	
451	0	1	1	0111	0100	0011	0001	0111	0100	1	célula excessiva
475	1	1	1	0111	0100	0011	0001	0111	0100	1	
500	0	1	1	0111	0100	0011	0001	0111	0100	1	
500	0	0	0	0111	0100	0011	0001	0111	0100	1	
525	1	0	0	0111	0100	0011	0001	0111	0100	1	
550	0	0	0	0111	0100	0011	0001	0111	0100	1	
550	0	1	0	0111	0100	0011	0001	0111	0100	1	6a. célula 550ns
551	0	1	1	0111	0100	0011	0001	0111	0100	1	
551	0	1	1	0111	0100	0011	0001	1011	0011	0	LCT=11; X=3
575	1	1	1	0111	0100	0011	0001	1011	0011	0	
600	0	1	1	0111	0100	0011	0001	1011	0011	0	

BOLETINS TÉCNICOS - TEXTOS PUBLICADOS

- BT/PSI/0001 – Observabilidade Topológica de Osawa em Redes não Lineares – ARMANDO HANDAYA, FLÁVIO A. M. CIPPARRONE
- BT/PSI/0002 – Desenvolvimento de uma Microbalança de Quartzo para Detectar Gases – ROBERTO CHURA CHAMBI, FRANCISCO JAVIER RAMIREZ FERNANDEZ
- BT/PSI/0003 – Sistema para Desenvolvimento de Sensores Inteligentes – ANTONIO CARLOS GASPARETTI, FRANCISCO JAVIER RAMIREZ FERNANDEZ
- BT/PSI/0004 – A 1.6GHz Dual Modulus Prescaler Using the Extended True Single-Phase Clock CMOS Circuit Technique (E-TSPC) – JOÃO NAVARRO SOARES JÚNIOR, WILHELMUS ADRIANUS M. VAN NOIJE

