UMA ARQUITETURA DE CONTROLE DE UMA CENTRAL TELEFÔNICA BASEADA EM COMPUTADORES A FLUXO DE DADOS

Ildeberto de Genova Bugatti
DCEs - UFSCar
C.P. - 676
13.560 - São Carlos - S.P.

Shusaburo Motoyama
Telemática FEE-UNICAMP
C.P. - 6101
13.081 - Campinas - S.P.

RESUMO

Neste trabalho é proposta uma arquitetura a Fluxo de dados para controlar uma central de comutação telefônica. No fluxo de dados o mecanismo de execução de instruções é semelhante ao mecanismo de controle de uma central de comutação convencional. Isto permite realizar eficientemente o processamento de uma chamada telefônica, possibilitando assim, uma possível solução para o problema de engarrafamento de software hoje existentes nos grandes sistemas de comutação eletrônica (SPC) comerciais.

A arquitetura a fluxo de dados proposta está organizada em forma de anel. Esta organização permite flexibilidade para expansões e dimensionamento da central.

Avaliações preliminares mostram que esse tipo de arquitetura oferece boas perspectivas na obtenção de centrais telefônicas com alto desempenho.

I- Introdução

Os principios utilizados nos projetos de arquitetura de computadores permaneceram durante muito tempo inalterados, baseados na organização conceitual de Von Neumann (1945). Entretanto, as máquinas baseadas nos conceitos de Von Neumann possuem uma limitação intrinseca que é a busca e execução sequencial das instruções. Desse modo para aumentar a velocidade de processamento dessas máquinas é necessário aumentar a velocidade dos componentes eletrônicos que compõem o sistema. Porém, a velocidade desses componentes já estão chegando a um patamar teóricamente possível de se alcançar. Mas, a exigência de computação mais rápida continua ainda crescendo, principalmente em aplicações como previsão de tempo, por exemplo.

Desse modo surgiu na década passada o conceito de processamento paralelo, em que as instruções seriam executadas na forma paralela, eliminando, assim o principal ponto de estrangulamento das máquinas tipo Von Neumann.

Atualmente todos os grandes computadores comerciais e experimentais reunem inovações de arquitetura dirigidas para aumento de desempenho. essas inovações são sempre baseadas em alguma forma de obter a capacidade de executar um grande número de atividades em paralelo (concorrentemente).

Essas inovações na área de arquitetura de computadores tiveram consequencias imediatas em atividades que fazem uso intensivo de computadores, controle de processos em tempo real por exemplo. E em particular houve um avanço quase que em paralelo no desenvolvimento de novas centrais de comutação Dentre as máquinas com processamento paralelo que surgiram, destacamos o fluxo de dados por esta mostrar boas perspectivas na aplicação no controle de processos e portanto com possibilidades de utilizá-la para realizar o controle de uma central de comutação. Eliminando dessa maneira os problemas hoje existentes nos grandes sistemas de comutação eletrônica que usam processadores tipo Von Neumann.

Existem várias centrais com controle distribuído e processamento paralelo propostas na literatura [CPqD85, In81, Sp77]. A central com processamento paralelo utilizando o conceito de computador a fluxo de dados [AYM84] parece ser bastante promissora. Isto porque o processamento de uma chamada telefônica, pode ser programado com simplicidade numa máquina a fluxo de dados, além de sua capacidade de executar processamento paralelo.

O objetivo desse trabalho é propor uma arquitetura a fluxo de dados para realizar o controle de uma central de comutação telefônica, que soma eficiência com flexibilidade para expansão, e dimensionamento.

Na seção II são resumidos os principais conceitos de computadores a fluxo de dados. Na seção III é descrita a arquitetura a fluxo de dados proposta. Na seção IV é feita uma análise de desempenho da arquitetura e um estudo de dimensionamento do número de processadores necessários para que uma central de comutação ofereça um determinado tráfego telefônico. Finalmente a seção V mostra as principais conclusões do trabalho.

II- Conceitos de Fluxo de Dados

O termo fluxo de dados é utilizado em sistemas de computação que possuem operações concorrentes e a linguagem de programação é representada de uma forma paralela [De74, DM75, De80, GW80, Ru77, TBH82]. A execução de instruções em computadores a fluxo de dados é conduzida por dados, isto é, cada instrução é habilitada à execução, logo após o fornecimento dos operandos pelas instruções predecessoras. Isto possibilita um processamento altamente concorrente, divergindo da máquina de Von Neumann que possui um registrador contador que sequencia a execução da instrução.

Um progama numa linguagem elementar de fluxo de dados é um grafo orientado, no qual os nós são operadores conectados por enlaces, através dos quais podme fluir valores (senhas). Um operador fluxo de dado (Fig. 1) possui enlaces de entrada e saída e especifica uma função (adição, multiplicação, comparação, etc) entre os valores de dados contidos em seus enlaces de saída. Os enlaces especificam as dependências de dados num programa. Um operador é habilitado (pronto para execução) quando valores estiverem presentes em todos seus enlaces de entrada. O operador habilitado pode disparar (iniciar a execução) a qualquer instante, removendo os valores de seus

enlaces de entrada, computando esses valores e enviando o resultado para seu enviado de saída. Um resultado pode ser enviado a mais de um destino através de um operador copiador (Fig. 1F.), que remove um valor de seu enlace de entrada e coloca uma cópia deste valor em cada um de seus enlaces de saída. Um operador não pode disparar caso exista um dado em algum arco de saída do operador. A execução de um operador depende somente da informação local para o operador; não existem variáveis globais ou efeitos secundários. Operadores não possuem memória interna entre execuções.Desta forma a execução de um operador pode levar a 4 possíveis estados mostrados na Fig. 2.

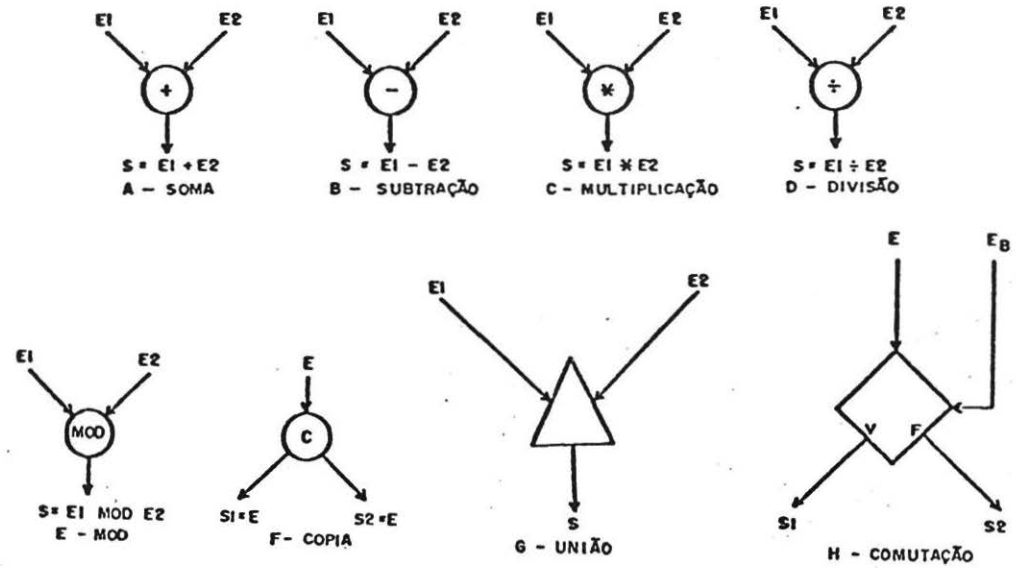


FIGURA 1 – ALGUNS OPERADORES FLUXO DE DADOS

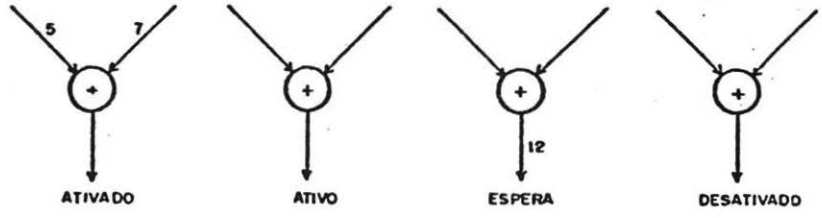


FIGURA 2 – EXECUÇÃO DO OPERADOR SOMA

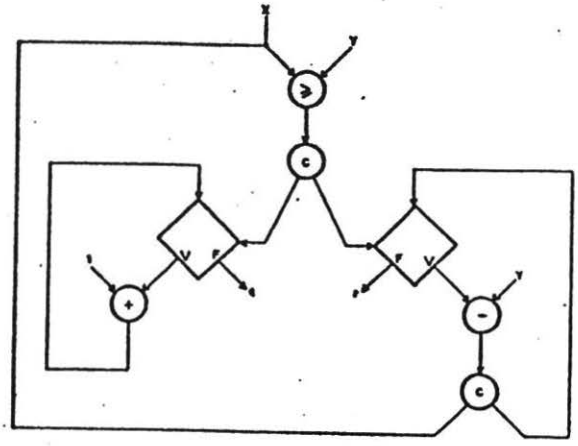
A Fig. 1 mostra também dois operadores de controle: comutação e união. Esses operadores permitem a realização de iterações e comutações. O operador de comutação (Fig. 1H),

seleciona um entre dois enlaces de saída, para transferir o dado contido no enlace de entrada (E), de acordo com o estado do enlace de entrada lógico (EB). O operador união (Fig. 1G) transfere o dado de um dos seus enlaces de entrada para seu enlace de saída.

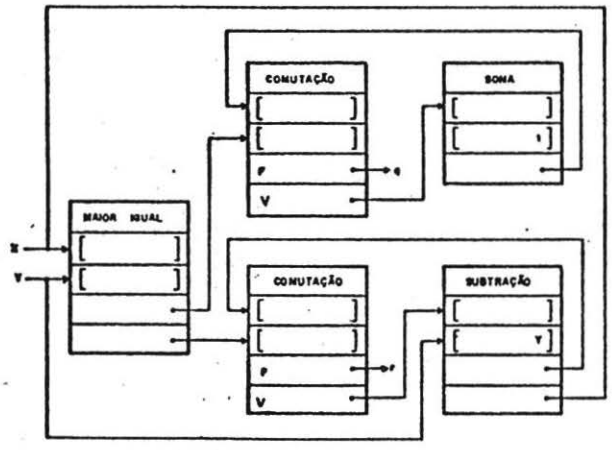
A Fig. 3a mostra um trecho de programa (divisão através de subtrações sucessivas) em linguagem convencional, a Fig. 3b mostra esse mesmo programa numa linguagem fluxo de dados, e a Fig. 3c mostra a representação do programa na memória de uma computador a fluxo de dados. A representação da instrução na memória (cédula de instrução) é, em geral, formada pelos campos: código de operação, operandos e destinos.

```
" Begin
   q:= 0;
   r:= x;
   While r)=y do
    Begin
     q:= q+1;
     r:= r-y
    End
  End "
```

(a)



(b)



(c)

FIGURA 3 - ESQUEMA DE UMA ATIVIDADE ITERATIVA QUE REALIZA A DIVISÃO DE X POR Y E SUA REPRESENTAÇÃO NA MEMÓRIA

III- Arquitetura Fluxo de Dados proposta

Como visto acima, uma máquina fluxo de dados é baseada num sistema de multi-processadores com funções partilhadas; onde instruções são executadas de acordo com "regras conduzidas por dados", pelas quais, "cada instrução torna-se executável quando e somente quando todos os dados necessários estão disponíveis". Portanto o mecanismo de funcionamento de uma máquina à fluxo de dados é semelhante à seção de controle de uma sistema de comutação convencional. Pois num sistema de comutação, as tarefas serão executadas (disparadas) somente quando estiverem disponíveis todos os recursos e dados necessários para realizá-la. Por exemplo, a procura de caminho na matriz de comutação só pode ser realizado se forem previamente fornecidos (obtidos) os números do assinante (usuário) chamador, assinante chamado e a ligação será efetuada somente se houver caminho disponível na matriz de comutação e o assinante chamado estiver no estado não ocupado.

Assim um sistema de comutação controlado por um computador à fluxo de dados que explore as semelhanças inerentes aos dois sistemas (controle da central e fluxo de dados) possui vantagens tais como:

- Simplificação de software:- não é necessário nenhuma técnica sofisticada de multiprogramação. Possibilita-se a construção de programas [Ac82] (numa linguagem de fluxo de dados) simplificados e legíveis, onde a sequência de execução de instruções estará bem próxima da sequencia lógica dos sinais de controle necessários ao processamento de uma chamada telefônica.
- Concorrência na execução de instruções:- utilizando o paralelismo do sistema a fluxo de dados é possivel reduzir o tempo de processamento viabilizando a realização de um processamento de chamada eficiente.

A arquitetura a fluxo de dados proposta possui a forma de anel [Bu86]. Essa estrutura possibilita maior velocidade de processamento, pois além do paralelismo dos processadores (Ver Fig. 4), utiliza processamento concorrente no anel (paralelismo "pipeline"). Ela permite facilidades de expansão, pois os anéis podem ser colocados em paralelo, e a interligação entre os anéis é feita através de um módulo de comutação.

A Fig. 4 mostra em diagrama de blocos a arquitetura de controle da central de comutação com configuração em anel. O anel é formado por quatro blocos básicos, sendo que as atividades exercidas por cada bloco são descritas abaixo:

Comutador de Entrada e Saída (CES) - Permite realizar a comunicação do anel com o exterior; possibilita a expansão da arquitetura e também o carregamento de programas.

Unidade de Ativação (UA) - Ela contém programas em fluxo de dados e tem a função de verificar se os operandos estão disponíveis para o processamento. Para isto ela recebe pacotes de dados provenientes de uma instrução predecessora que foi processada na Unidade de Processamento, ou do exterior através do CEs.

Unidade de Instruções Executáveis (UIE) - Desempenha uma atividade moderadora de tráfego. Ela contém uma memória com a função de armazenar instruções executáveis provenientes da Unidade de Ativação quando estas encontram a unidade de Processamento sobrecarregada.

Unidade de Processamento (UP) - A Unidade de Processamento possui tres funções: distribuição, processamento e arbitragem. A função de distribuição tem por finalidade distribuir as instruções provenientes da UIE para os vários processadores elementares. Os processadores elementares (microprogramados) processam as instruções (função de processamento) e enviam os resultados na forma de pacotes de dados, através de um esquema de arbitragem, para o comutador de entrada e saída. No comutador, o pacote de dados pode dirigir-se à unidade de ativação, originado as instruções sucessoras ou dirigem-se à saída do sistema de comutação (anel).

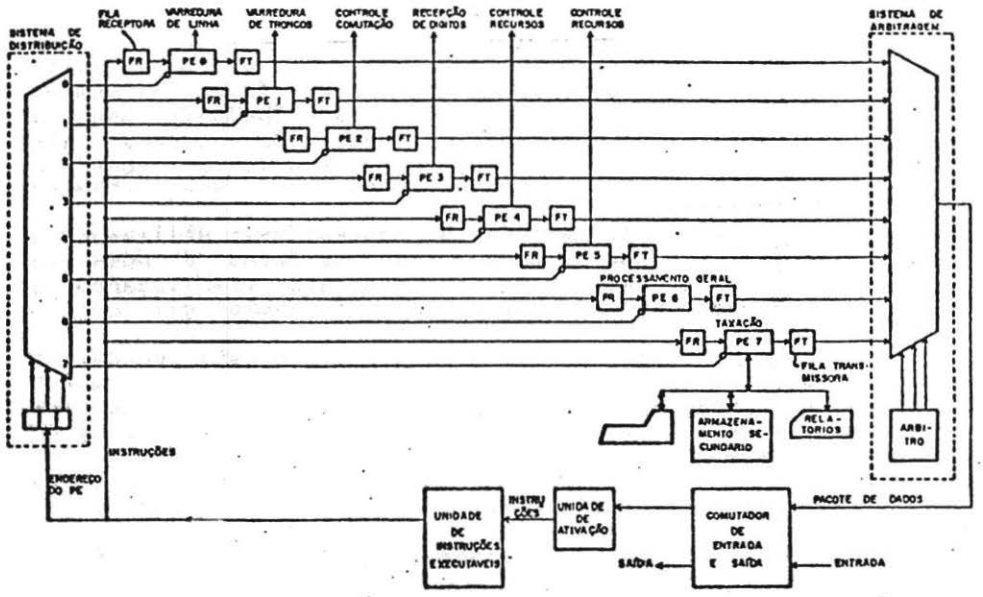


FIGURA 4 - ORGANIZAÇÃO DO SISTEMA DE CONTROLE

III.1 - Formatos de Instruções e Dados

Um programa fluxo de dados é um conjunto de instruções contidas na unidade de Ativação. Cada instrução contém a função a ser executada sobre os operandos e os endereços das instruções sucessoras. Assim sendo, cada instrução

é a descrição de um nó do programa fluxo de dados; onde os operandos estão contidos nos arcos de entrada e o resultado é colocado em seus arcos de saída (instruções sucessoras).

O formato das Instruções contidas na Unidade de Ativação é mostrado na Fig. 5.

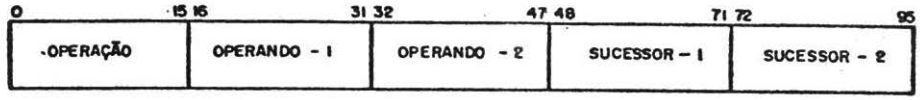


FIGURA 5 - FORMATO DE UMA PALAVRA DE INSTRUÇÃO FLUXO DE DADOS

Sendo:

Sucessor-1 e **Sucessor-2** os endereços das instruções sucessoras, que foram limitadas no máximo a duas, com o objetivo de encontrarmos facilidades na implementação da arquitetura (largura das vias de comunicação e da Unidade de Ativação).
Operando-1 e **Operando-2** respectivamente, os operandos da esquerda e da direita para instruções binárias, e sempre o operando da esquerda para as instruções unárias.

III.2 - Unidade de Ativação

A Unidade de ativação é o bloco mais importante e também crítico desse sistema fluxo de dados em anel. O desempenho deste bloco influenciará de maneira preponderante o desempenho do sistema de controle como um todo.

Esta unidade tem as funções de :

- armazenar os programas fluxo de dados que geram o controle da central de comutação.

- realizar o mecanismo de disparo para processamento de instruções executáveis (instruções que tenham todos os seus operandos disponíveis).

A Fig. 6 mostra uma possível implementação desta unidade; seu desempenho será tanto melhor quanto menor o tempo de ciclo de leitura e escrita das memórias de acesso aleatório usados na implementação.

Este esquema de implementação da Unidade de Ativação usando vários bancos de memória com controles independentes, reproduzindo os campos de uma palavra de instrução da máquina a fluxo de dados facilitará sobremaneira a implementação do mecanismo de disparo de instruções executáveis, embora em algumas situações reproduza-se desnecessáriamente os campos de operandos.

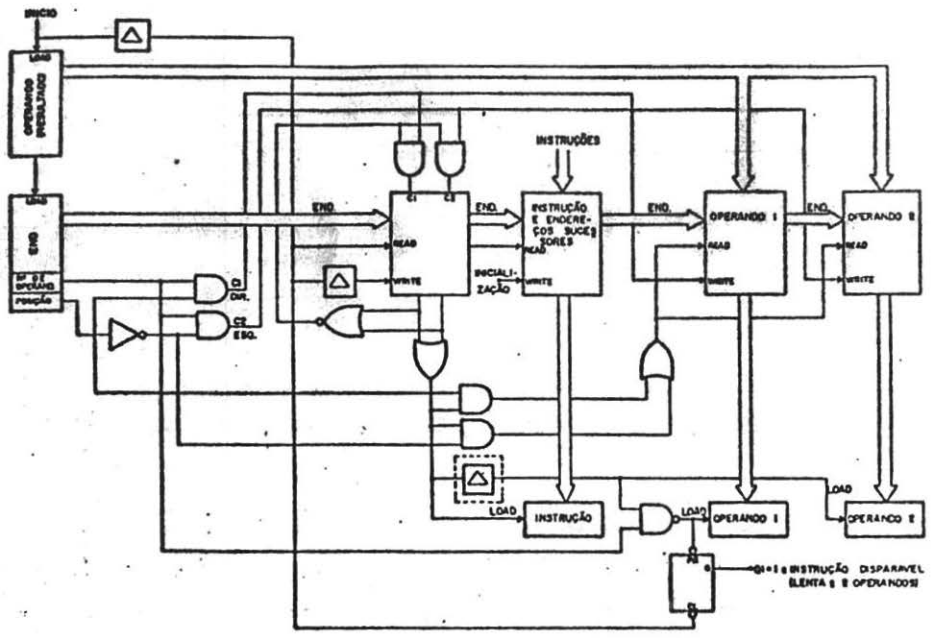


FIGURA 6 - UNIDADE DE ATIVAÇÃO

A arquitetura até aqui explicada refere-se a um único anel. A vantagem do esquema proposto é que se pode expandir a sua capacidade de processamento facilmente. A Fig. 7 mostra a maneira de conseguir essa expansão. Os vários anéis são colocados em paralelo e são interligados através do comutador de entrada e saída. Desse modo o paralelismo de execução é conseguido dentro do anel e de anel para anel.

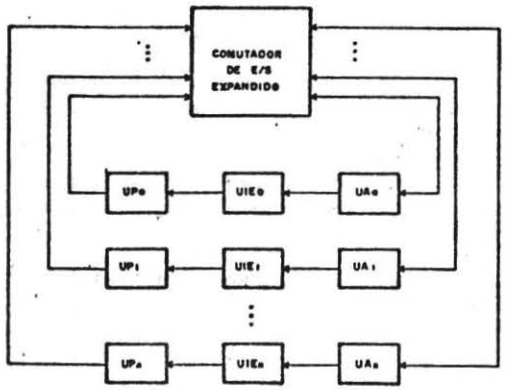


FIGURA 7 - ARQUITETURA EXPANDIDA

IV- Análise de desempenho e dimensionamento da Central

Utilizando um método de análise proposto em GWG78, foi feita uma avaliação da eficiência da arquitetura em anel proposta. O método preocupa-se basicamente com a determinação do número de passos necessários para executar um programa (paralelismo do programa versus paralelismo da arquitetura), e na obtenção do tempo médio gasto para executar uma instrução na arquitetura em anel (tempo de atraso no "pipeline").

A seguir serão relatadas algumas considerações sobre o método:

A eficiência (Ej) de um sistema com "j" processadores é sempre maior que 50%, e é dada por:

$$Ej = \frac{Coo}{Cj}$$ onde:

Cj - é o comprimento de um programa, ou seja, o número de passos necessários para executar um programa com a disponibilidade de "j" processadores para realizar a execução de até "j" instruções em cada passo.
Coo - número de passos para a execução paralela de um programa. (isto pode requerer a disponibilidade de um número muito grande de processadores elementares).

O paralelismo Médio (Cm) de um programa a fluxo de dados como sendo o número de passos necessários para a execução totalmente serial dividido pelo número de passos necessários para a ececução totalmente paralela do programa, ou seja:

$$Cm = \frac{C1}{Coo}$$

Um sistema de execução é dito de execução perfeita se:

i) todos processadores iniciam e terminam suas instruções simultaneamente.
ii) cada processador gasta uma unidade de tempo para realizar uma instrução.
iii) no final de qualquer passo, se "e" instruções são elegíveis, então todas "e" intruções serão executadas imediatamente se "e<j" e exatamente "j" instruções se "e)j".

A arquitetura à fluxo de dados elaborada neste trabalho é baseada num anel circular de blocos que desempenham atividades em paralelo de maneira sobreposta ("pipeline"). Cada bloco possui um tempo de atraso para processar os pacotes de dados ou pacotes de instruções. A avaliação de eficiência será feita através da aproximação da arquitetura proposta ao sistema de execução perfeito.

O comportamento do sistema em anel pode aproximar-se do sistema de execução perfeita se tivermos o seguinte:

I) As diferentes espécies de instruções obedecerem a uma distribuição uniforme,

II) Um tempo médio de execução de instrução (Tm).

Com uma distribuição uniforme das instruções pode-se determinar o tempo mínimo entre instruções no anel (Tmin) e assim estimar o grau de paralelismo no anel (K).

Define-se o grau de paralelismo K como sendo o menor inteiro maior ou igual a Tp/Tmin. O valor K representa o paralelismo da Unidade de Processamento e das atividades sobrepostas no anel (paralelismo "pipeline").

Aproximando-se o sistema em anel a um sistema de execução perfeita, pode-se avaliar a eficiência geral da arquitetura em anel determinando-se Tmin, Tp e K.

Para determinar-se Tmin, considera-se a execução de um programa completo que consome "x" pacotes de dados de entrada, executa Ci instruções, e produz "y" resultados de saída. Vamos determinar as relações entre o número total de pacotes passando através de cada parte do sistema durante o tempo de execução do programa (Te).

Através de uma análise da figura 8, que mostra a arquitetura em anel com símbolos representando o número de pacotes de instruções e dados ("tokens") que caminham através dos enlaces entre os módulos, podemos deduzir que o tempo médio de atraso entre instruções é dado por:

$$\frac{Te}{Ci} = (1 + P2e) * \frac{Te}{a}$$

Onde "a" é o número total de pacotes de dados necessários para executar um programa, "P2e" é a proporção de instruções que requerem dois pacotes de dados (instruções binárias) e "Te" é o tempo total de execução de um programa. O valor mínimo de atraso entre instruções (Tmin) é o valor mínimo de Te/Ci.

A unidade de ativação é o bloco que necessita de maior tempo para o processamento dos pacotes de dados. A máxima taxa de atividade é igual a 3 * a2 + ai ciclos de acesso à memória.

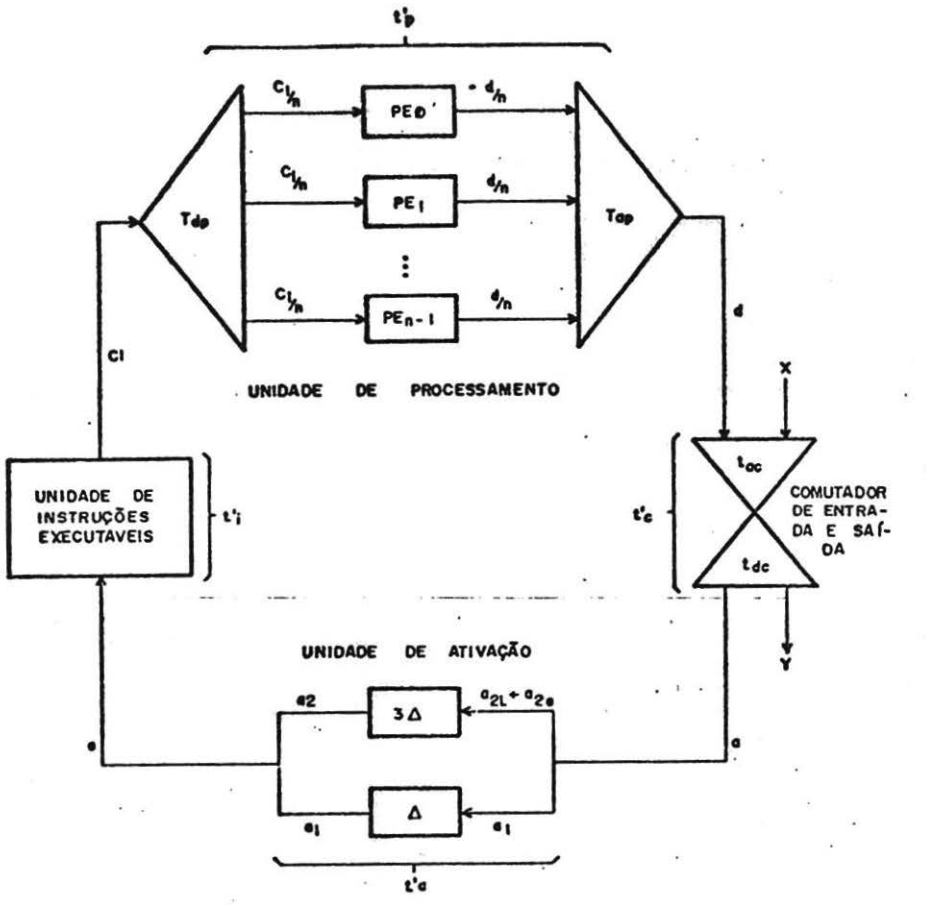
Portanto, o tempo mínimo entre os pacotes de dados no anel deve ser:

$$\frac{Te}{3 * a2 + ai}$$

Seja Tb, o tempo de atraso em um bloco do anel.

Em cada bloco, o Tb deve ser menor ou igual ao tempo mínimo assim:

$$Tb <= \frac{Tb}{3 * a2 + ai}$$



Onde:

- **a** - número total de pacotes de dados necessários para executar um programa.
- **x** - número de pacotes de dados que entram no sistema.
- **y** - número de pacotes de dados que saem do sistema.
- **a1** - quantidade de instruções unárias.
- **a2** - quantidade de instruções binárias.
- **a21** - quantidade de operandos que tornam instruções binárias elegíveis.
- **a2e** - quantidade de operandos que ficam armazenados na Unidade de Ativação (primeiro operando disponível de uma instrução binária) .
- **d** - quantidade de resultados (pacotes de dados) gerados pela Unidade de Processamento.
- **Ci** - quantidade de instruções contidas num programa fluxo de dados.

FIGURA 8 - NÚMERO DE PACOTES ATRAVÉS DO ANEL DURANTE A EXECUÇÃO DE UM PROGRAMA

Com o tempo de atraso entre instruções (Tb) no "pipeline" podemos deduzir o valor mínimo de Te/Ci como sendo:

$$Tmin = \left(\frac{Te}{Ci}\right)_{min} = Tb * (2 + P2e)$$

Fazendo os tempos de atrasos nos blocos do anel como sendo igual a "Tb" teremos:

$$Tp = Tb * (5 + \mathcal{C})$$

Sendo "$\mathcal{C}$" o número de microciclos requeridos para executar para executar uma operação de fluxo de dados.

Asim temos que o paralelismo "k" da arquitetura é dado por:

$$K = \frac{Tp}{Tmin} = \frac{5 + \mathcal{C}}{1 + P2e}$$

A Tabela1 mostra alguns valores de K em função de $\mathcal{C}$ e P2e. Com uma rápida análise da tabela 4.1 obtem-se para programas com $\mathcal{C}$ médio = 20 (número de microciclos para realizar uma instrução) e P2e = 3/4 (porcentagem de instruções binárias) necessita-se de no mínimo oito processadores elememtares na Unidade de Processamento.

Finalmente define-se a razão média de execução de instruções (Ø) como sendo:

$$Ø = \frac{Ci}{TJ}$$

onde TJ é o tempo de execução usando J processadores e

$$TJ = J * Cj * Tmin$$

assim:

$$Ø = \frac{Ci}{J * Cj * Tmin} = Ej * \frac{1}{Tmin}$$

sendo 1/Tmin a razão de execução máxima e para:

$$Pm \geq J \Longrightarrow Ej > \frac{1}{2}$$

Como Tmin=Tb*(1+2*P2e), para Tb=150 ns (nanosegundos) e P2e=0,5 tem-se a razão de execução em torno de 3,3 MIPS (Milhões de Intruções Por Segundo) e a razão média de execução maior que 1,66 MIPS, para programas com paralelismo médio (Pm) maior ou igual ao paralelismo do anel.

Para programas com P2e mínimo (P2e = Ø) tem-se a

razão de execução máxima em torno de 6,6 MIPS e a razão de execução média maior que 3,3 MIPS.

Para programas com P2e máximo (P2e = 1) tem-se a razão de execução máxima em torno de 2,2 MIPS e a razão de execução médja maior que 1,1 MIPS.

P2e \ @	5	10	15	20	25	30	35	40
0	5	10	15	20	25	30	35	40
1/4	4	7	10	14	17	20	24	27
1/2	3	5	8	10	13	15	18	20
3/4	2	4	6	8	10	12	14	16
1	2	4	5	7	9	10	12	14

TABELA 1 - Valores mínimos de n $)= \frac{@}{1 + 2 * P2e}$

Conclue-se que a razão de execução da arquitetura proposta estará entre 1,1 e 6,6 MIPS, para programas com Tm)= K (paralelismo do anel).

De posse desses resultados pode-se dizer que o custo para aumentarmos a eficiência das arquiteturas fluxo de dados proposta nesta trabalho está na elaboração de programas altamente paralelos, que utilizem eficientemente os recursos disponíveis na arquitetura proposta.

Foram também feitos cálculos para o dimensionamento do número de processadores necessários à central para um determinado tráfego telefônico (medido em Erlangs) [Bmc86].

Para realizar-se este dimensionamento consideram-se os Processadores elementares como sendo o conjunto de servidores de uma fila e a Unidade de Instruções Executáveis que contém as instruções como sendo a sala de espera, formado a fila.

Para o dimensionamento do número de processadores necessários para atender a um determinado tráfego, foram consideradas as características da central de comutação D10 da NTT (Nippon Telephone and Telegraph).

Foi feito um exemplo de dimensionamento do número de processadores para um tráfego de 4.000 Erlangs e um tempo de retenção de chamada de 120 segundos. Isto implica em 33,33

chamadas por segundo e consequentemente 180.000 instruções por segundo (de acordo com as caracteísticas da D10-NTT). Conclui-se que com essas características são necessários 9 processadores para que a central de comutação à fluxo de dados atenda a um tráfego de 4.000 Erlangs, com tempo de retenção de 120 segundos, o número médio de instruções a espera de atendimento na UIE será em torno de 12.

V - conclusão

Com o intuito de realizar-se o controle de centrais de comutação de alto desempenho, utilizou-se uma máquina à fluxo de dados com características de programação que podem eliminar o problema de engarrafamento de software atualmente existentes nos grandes sistemas de comutação eletrônica.

Foi proposta uma arquitetura de fluxo de dados em anel cuja principais características são a eficiência na execução de instruções e flexibilidade para expansão e dimensionamento de uma central de comutação.

Os conceitos envolvidos na realização dessa central de comutação foram explicitados e foi feito também uma análise de desempenho de arquitetura a fluxo de dados proposta e um exemplo de dimensionamento do número de processadores necessários para atender um tráfego de 4.000 Erlangs.

VI - Referencias

Ac82 - ACKERMAN, W. B.. Data Flow Languages. IEEE Computer, Feb., 1982. p.15-25.

AYM84 - AKIYAMA,M.; YAMASHITA,M.; MISU,T.. DATAFLEX-1.. An Experimental Data Flow Computer Controlled Eletronic Switching Systems. ISS'84 May., 1984. 7p. (session 23B, paper 2).

BMc86 - BUGATTI, I. G. & MOTOYAMA, S. Central de Comutaçao Controlada por um Computador á Fluxo de Dados. Quarto Simpósio Brasileiro de Telecomunicações (SBT). Set., 1986. p.115-20.

Bu86 - BUGATTI, I. G. - "Computadores a Fluxo de Dados : Aplicação e central de Comutação Telefônica" DEE-FEC-UNICAMP., TM 054, 115p., out., 1986.

CPqD85- CENTRO DE PESQUISA E DESENVOLVIMENTO Trópico R: Central local Tandem digital do sistema trópico. CPqD TELEBRÁS, 1985. 163p.

De74 - DENNIS, J. B.. First version of a Data Flow Procedure Language. Lecture Notes in Computer Science, (19):362-76, 1974.

De80 - DENNIS, J. B.. Data Flow Supercomputers. IEEE Computer, Nov., 1980. p.48-56.

DM75 - DENNIS, J. B. & MISUMAS D. P.. A Preliminary Architecture for a Basic Data-Flow Processor. In: ANNUAL SYMP. COMPUTER ARCHITETURES, 2. Proceedings. 1975. p.126-32.

GW80 - GURD,J. & WATSON,I.. Structuring software for Parallel Execution I. In:Data Driven Systems for High Speed Parallel Computing. Computer Design, Jun., 1980. p.91-100.

GWG78 - GURD,J.; WATSON,I.; GLAUERT,J.. A Multilayered Data Flow Computer Architecture. University of Manchester, Departament of Computer Science, Jul., 1978. 56p.

In81 - INTERCONNECTION Networks. Computer, 14(12):8-65, Dec., 1981.

Ru77 - RUMBAUGH, J.. A Data Flow Multiprocessor. IEEE Transactions on Computer, Feb.,1977. p.138-46.

Sp77 - SPECIAL Issue in Telecomunications Circuit Switching. Procedings of the IEEE, 65(9) : 1235-1424, Sep., 1977.

TBH82 - TRELEVEAN,P.C.; BROWNBRIDGE,D.R.; HOPKINS, R.P.. Data - Driven and Demand - Driven Computer Architecture. ACM Computer Surveys, Mar.,1982. p.93-143.