

República Federativa do Brasil
Ministério do Desenvolvimento, Indústria
e do Comércio Exterior
Instituto Nacional da Propriedade Industrial.

(11) **PI9612911-5 B1**

(22) Data de Depósito: 17/07/1996
(45) Data da Concessão: 14/12/2010
(RPI 2084)



* B R P I 9 6 1 2 9 1 1 B 1 *

(51) *Int.Cl.:*
G06F 7/00
G06F 7/38
G06F 7/52
G06F 7/50
G06F 9/30

(54) Título: **APARELHO E MÉTODO PARA REALIZAR OPERAÇÕES MULTIPLICAÇÃO-ADIÇÃO EM DADOS EM PACOTE.**

(30) Prioridade Unionista: 31/08/1995 US 521360

(73) Titular(es): Intel Corporation

(72) Inventor(es): Alexander D. Peleg, Andrew F. Glew, Benny Eitan, Carole Dulong, Eiichi Kowashi, Larry M. Mennemeier, Millind Mittal, Wolf Witt, Yaacov Yaari

Relatório Descritivo da Patente de Invenção para "**APARELHO E MÉTODO PARA REALIZAR OPERAÇÕES MULTIPLICAÇÃO-ADIÇÃO EM DADOS EM PACOTE**".

Pedido Dividido do PI 9610095-8 de 17/07/1996.

5 Antecedentes da Invenção

1. Campo da Invenção

Em particular, a invenção refere-se ao campo de sistemas de computador. Mais especificamente, a invenção refere-se à área de operações de dados compactados.

10 2. Descrição da Técnica Relacionada

Nos sistemas de computador típicos são implementados processadores para operarem nos valores representados por um grande número de bits (por exemplo, 64) com o uso de instruções que produzem um resultado. Por exemplo, a execução de uma instrução de soma acrescentará entre si um primeiro valor de 64 bits e um segundo valor de 64 bits e armazenará o resultado como um terceiro valor de 64 bits. Entretanto, aplicações de múltiplos meios/multimídia (por exemplo, aplicações com alvo na cooperação sustentada por computador (CSC -- a integração da teleconferência com a manipulação de dados de meios mistos), gráficos 2D/3D, processamento de imagem, compressão/descompressão de vídeo, algoritmos de reconhecimento e manipulação de áudio) exigem a manipulação de grandes quantidades de dados que podem ser representados em um pequeno número de bits. Por exemplo, os dados gráficos tipicamente exigem de 8 ou 16 bits e os dados de som tipicamente exigem 8 ou 16 bits. Cada uma destas aplicações de múltiplos meios exige um ou mais algoritmos, cada qual exigindo diversas operações. Por exemplo, um algoritmo pode exigir uma operação de soma, de comparação e de deslocamento.

Para aperfeiçoar a eficiência das aplicações de múltiplos meios (bem como outras aplicações que possuam as mesmas características), os processadores da técnica anterior apresentam formatos de dados compactados. Um dado compactado é aquele no qual os bits tipicamente usados para representar um único valor são subdivididos em vários elementos de

dados de tamanho fixo, cada um dos quais representando um valor separado. Por exemplo, um registro de 64 bits pode ser subdividido em dois elementos de 32 bits, cada um dos quais representando um valor separado de 32 bits. Adicionalmente, estes processadores da técnica anterior fornecem 5 instruções para separadamente manipular cada elemento nestes tipos de dados compactados em paralelo. Por exemplo, uma instrução de soma compactada soma entre si elementos de dados correspondentes originários de primeiros dados compactados e de segundos dados compactados. Dessa maneira, se um algoritmo de múltiplos meios exigir um circuito contendo cinco 10 operações que precisam ser executadas em um grande número de elementos de dados, é desejável compactar os dados e executar estas operações em paralelo com o uso de instruções de dados compactados. Dessa forma, estes processadores podem processar de uma maneira mais eficiente as aplicações de múltiplos meios.

15 Entretanto, se o circuito de operações contiver uma operação que não possa ser executada pelo processador em dados compactados (isto é, o processador carece da instrução apropriada), os dados terão que ser descompactados a fim de executar a operação. Por exemplo, se o algoritmo de múltiplos meios exigisse uma operação de soma e a instrução de soma 20 compactada descrita anteriormente não estiver disponível, então o programador tem que descompactar tanto os primeiros dados compactados quanto os segundos dados compactados (isto é, separar os elementos que compreendem tanto os primeiros dados compactados quanto os segundos dados compactados), somar os elementos separados entre si de maneira individual, e depois compactar os resultados em um resultado compactado para o 25 processamento compactado adicional. Por isso, é desejável incorporar em um processador de finalidades gerais um jogo de instruções de dados compactados que forneça todas as operações exigidas para os algoritmos típicos de múltiplos meios. Contudo, devido à área de matriz limitada nos microprocessadores atuais, mostra-se limitado o número de instruções que podem 30 ser acrescentadas. Portanto, é desejável inventar instruções que garantam tanto versatilidade (isto é, instruções que possam ser utilizadas em uma

grande variedade de algoritmos) e a maior vantagem de performance.

Uma técnica anterior para prover operações para o uso em algoritmos de múltiplos meios é unir um processador de sinalização digital (DSP) separado a um processador de propósitos gerais existente (por exemplo, o Intel.RTM. 486 produzido por Intel Corporation of Santa Clara, Califórnia). O processador de propósitos gerais aloca tarefas que podem ser realizadas usando pacotes de dados (por exemplo, processamento de vídeo) no DSP.

Essa técnica DSP anterior inclui uma instrução de multiplicação acumulada que soma a um valor acumulado os resultados da multiplicação de dois valores juntos (veja Kawakami, Yuichi, et al. "Aplicações de um processador de sinais digitais de um único chip para banda de voz" IEEE conferência internacional de circuitos de estado sólido, 1980, pg 40-41). Um exemplo da operação de multiplicação acumulada para esse DSP é demonstrada abaixo na tabela 1, em que as instruções são realizadas nos valores de dados A_1 e B_1 acessados como fonte 1 e fonte 2, respectivamente.

Tabela 1

Acumular para Multiplicar Fonte 1 . Fonte 2

A_1	Fonte 1
B_1	Fonte 2
=	
$A_1B_1 + \text{Valor de Acumulação}$	Resultado 1

Uma limitação dessa técnica anterior é a sua eficiência limitada, isto é, ela só trabalha com dois valores e um valor de acumulador. Para multiplicar e acumular dois conjuntos de dois valores é necessário a realização serial das seguintes instruções: 1) multiplicação acumulada do primeiro valor do primeiro conjunto, primeiro valor do segundo conjunto e a acumulação do valor zero para gerar um valor de acumulador intermediário; 2) multiplicação acumulada do segundo valor do primeiro conjunto, segundo valor do segundo conjunto e o valor de acumulador intermediário para gerar o resultado.

Uma outra técnica DSP anterior inclui uma instrução multiplica-

ção acumulada que operar em dois conjuntos de dois valores e um valor de acumulador (veja “Processador de sinal digital com multiplicadores paralelos”, U.S Pat. No. 4.771. 379 – referida aqui pela referência “Ando et. Al”). Um exemplo da instrução de multiplicação acumulada é demonstrado abaixo na tabela 2, em que as instruções são realizadas nos valores de dados A_1 , A_2 , B_1 e B_2 respectivamente acessados como fontes 1 a 4.

Tabela 2

Fonte 1		Fonte 3
A_1		A_2
	Acumular para multiplicar	
Fonte 2		Fonte 4
B_1		B_2
=		Resultado 1
$A_1 \cdot B_1 + A_2 \cdot B_2 + \text{Valor de Acumulação}$		

Usando essa técnica anterior, dois conjuntos de dois valores são multiplicados e então somados em um valor de acumulador em uma instrução.

Essa instrução de multiplicação acumulada tem versatilidade limitada porque sempre é somada ao valor de acumulador. Como resultado, é difícil utilizar a instrução para operações que não sejam multiplicação acumulada. Por exemplo, a multiplicação de dois números complexos (r_1 i_1 e r_2 e i_2) é realizada de acordo com a seguinte equação:

$$\text{Componente real} = r_1 \cdot r_2 - i_1 \cdot i_2$$

$$\text{Componente imaginário} = r_1 \cdot i_2 + r_2 \cdot i_1$$

Essa técnica DSP anterior não pode realizar a função de multiplicar dois números complexos juntos usando uma instrução de multiplicação acumulada.

As limitações dessa instrução de multiplicação acumulada podem ser mais facilmente vistas quando o resultado dessa multiplicação é necessário em uma operação de multiplicação subsequente ao invés de uma operação de acumulação. Por exemplo, se o valor real fosse calculado utili-

zando essa técnica DSP anterior, o valor de acumulador precisaria ser inicializado em zero para computar o resultado corretamente. Então o valor de acumulador precisaria novamente ser inicializado em zero para poder calcular a componente imaginária. Para realizar outra multiplicação complexa entre o número complexo resultante e um terceiro número complexo (por exemplo, r_3 , i_3), o número complexo resultante deve ser redimensionado e armazenado no formato de memória aceitável e o valor de acumulador de ser novamente inicializado em zero. Então, a multiplicação complexa pode ser realizada como descrita acima. Em cada uma dessas operações a ULA (unidade lógica e aritmética), que é ligada ao valor do acumulador, é hardware supérfluo e são necessárias instruções extras para reinicializar o valor do acumulador. Essas instruções extras de outra forma seriam desnecessárias.

Uma outra limitação dessa técnica anterior é que os dados devem ser acessados através de custosas memórias multi-portas. Isso acontece porque os multiplicadores estão conectados diretamente com as memórias de dados. Portanto a quantidade de paralelismo que pode ser explorada é limitada um pequeno número pelo custo da interconexão, e o fato de que essa interconexão não é desligada da instrução.

A referência Ando e outros. também descreve que uma alternativa para essa custosa interconexão é introduzir um atraso para cada par de dados subsequente para ser multiplicado. Essa solução diminui qualquer vantagem de performance em relação àquela solução apresentada anteriormente na tabela 1.

Finalmente, a noção de memória multi-portas ou de acesso de memória por pipeline é vinculada ao uso de endereços múltiplos. Esse uso explícito de um endereço por dado, claramente demonstra que a noção crítica de pacote de dados não é empregada na técnica anterior.

Sumário da invenção

São descritos método e aparelho para incluir em um processador instruções para realizar operações multiplicação-adição em pacotes de dados. Em uma modalidade, o processador é ligado a uma memória. A me-

mória possui armazenados nela um primeiro pacote de dados e um segundo pacote de dados. O processador realiza operações em elementos de dados no primeiro pacote de dados e no segundo pacote de dados para gerar um terceiro pacote de dados em resposta ao recebimento de uma instrução.

- 5 Pelo menos dois dos elementos de dados nesse terceiro pacote de dados armazenam o resultado da realização de operações multiplicação-adição nos elementos de dados do primeiro pacote de dados e do segundo pacote de dados.

Breve Descrição dos Desenhos

- 10 A invenção é ilustrada por meio de exemplo, e não de limitação, nas figuras. Referências semelhantes indicam elementos similares.

A Figura 1 ilustra um sistema de computador exemplificativo, de acordo com uma concretização da invenção.

- 15 A Figura 2 ilustra um arquivo de registro geral do processador, de acordo com uma concretização da invenção.

A Figura 3 é um diagrama de fluxo que ilustra as etapas gerais usadas pelo processador para manipular os dados, de acordo com uma concretização da invenção.

- 20 A Figura 4 ilustra os tipos de dados em pacote, de acordo com uma concretização da invenção.

A Figura 5a ilustra as representações de dados em pacote em registro, de acordo com uma concretização da invenção.

A Figura 5b ilustra as representações de dados em pacote em registro, de acordo com uma concretização da invenção.

- 25 A Figura 5c ilustra as representações de dados em pacote em registro, de acordo com uma concretização da invenção.

A Figura 6a ilustra um formato de sinal de controle para indicar o uso de dados em pacote, de acordo com uma concretização da invenção.

- 30 A Figura 6b ilustra um formato de sinal de controle para indicar o uso de dados em pacote, de acordo com uma concretização da invenção.

SOMA/SUBTRAÇÃO EM PACOTE

A Figura 7a ilustra um processo para a execução de adição em

pacote, de acordo com uma concretização da invenção,

A Figura 7b ilustra um processo para a execução de subtração em pacote, de acordo com uma concretização da invenção.

5 A Figura 8 ilustra um circuito para a execução de adição em pacote e de subtração em pacote em bits individuais de dados em pacote, de acordo com uma concretização da invenção.

A Figura 9 ilustra um circuito para a execução de adição em pacote e de subtração em pacote nos dados de byte em pacote, de acordo com uma concretização da invenção.

10 A Figura 10 é uma vista lógica de um circuito para a execução de adição em pacote e de subtração em pacote nos dados de palavra em pacote, de acordo com uma concretização da invenção.

A Figura 11 é uma vista lógica de um circuito para a execução de adição em pacote e de subtração em pacote nos dados de palavra dupla em pacote, de acordo com uma concretização da invenção.

MULTIPLICAÇÃO EM PACOTE

A Figura 12 é um diagrama de fluxo que ilustra um processo para a execução de operações de multiplicação em pacote nos dados em pacote, de acordo com uma concretização da invenção.

20 A Figura 13 ilustra um circuito para a execução de multiplicação em pacote, de acordo com uma concretização da invenção.

MULTIPLICAÇÃO-SOMA/SUBTRAÇÃO

A Figura 14 é um diagrama de fluxo que ilustra um processo para a execução de operações de multiplicação-soma e de multiplicação-subtração nos dados em pacote, de acordo com uma concretização da invenção.

A Figura 15 ilustra um circuito para a execução de operações de multiplicação-soma e/ou multiplicação-subtração nos dados em pacote, de acordo com uma concretização da invenção.

30 DESLOCAMENTO EM PACOTE

A Figura 16 é um diagrama de fluxo que ilustra um processo para a execução de uma operação de deslocamento em pacote nos dados

em pacote, de acordo com uma concretização da invenção.

A Figura 17 ilustra um circuito para a execução de um deslocamento em pacote nos bytes individuais dos dados em pacote, de acordo com uma concretização da invenção.

5 EMPACOTAMENTO

A Figura 18 é um diagrama de fluxo que ilustra um processo para a execução de operações de empacotamento nos dados em pacote, de acordo com uma concretização da invenção.

10 A Figura 19a ilustra um circuito para a execução de operações de empacotamento nos dados de byte em pacote, de acordo com uma concretização da invenção.

A Figura 19b ilustra um circuito para a execução de operações de empacotamento nos dados de palavra em pacote, de acordo com uma concretização da invenção.

15 DESEMPACOTAMENTO

A Figura 20 é um diagrama de fluxo que ilustra um processo para a execução de operações desempacotadas nos dados em pacote, de acordo com uma concretização da invenção.

20 A Figura 21 ilustra um circuito para a execução de operações desempacotadas nos dados em pacote, de acordo com uma concretização da invenção.

CONTAGEM DE OCUPAÇÃO

25 A Figura 22 é um diagrama de fluxo que ilustra um processo para a execução de uma operação de contagem de ocupação nos dados em pacote, de acordo com uma concretização da invenção.

30 A Figura 23 é um diagrama de fluxo que ilustra um processo para a execução de uma operação de contagem de ocupação em um elemento de dados dos dados em pacote e para a geração de um único elemento de dados de resultado para dados em pacote de resultado, de acordo com uma concretização da invenção.

A Figura 24 ilustra um circuito para a execução de uma operação de contagem de ocupação nos dados em pacote possuindo quatro ele-

mentos de dados de palavras, de acordo com uma concretização da invenção.

A Figura 25 ilustra um circuito detalhado para a execução de uma operação de contagem de ocupação em um elemento de dados de palavra de dados em pacote, de acordo com a concretização da invenção.

OPERAÇÕES LÓGICAS EM PACOTE

A Figura 26 é um diagrama de fluxo que ilustra um processo para a execução de diversas operações lógicas nos dados em pacote, de acordo com uma concretização da invenção.

A Figura 27 ilustra um circuito para a execução de operações lógicas nos dados em pacote, de acordo com a concretização da invenção.

COMPARAÇÃO EM PACOTE

A Figura 28 é um diagrama de fluxo que ilustra um processo para a execução de operações de comparação em pacote nos dados em pacote, de acordo com uma concretização da invenção.

A Figura 29 ilustra um circuito para a execução de operações de comparação em pacote nos bytes individuais dos dados em pacote, de acordo com uma concretização da invenção.

Descrição Detalhada

Este pedido descreve um processo e um aparelho para incluir em um processador um conjunto de instruções que suportem as operações nos dados em pacote exigidos pelas aplicações típicas de múltiplos meios. Na seguinte descrição, são explicados numerosos detalhes específicos, de maneira a fornecerem um completo entendimento da invenção. Entretanto, é entendido que a invenção pode ser praticada sem estes detalhes específicos. Em outros casos, circuitos, estruturas e técnicas bem conhecidos não foram mostrados em detalhes, a fim de não obscurecer desnecessariamente a invenção.

Definições

Para prover um fundamento para o entendimento da descrição das concretizações da invenção, são fornecidas as seguintes definições.

Bit X até Bit Y:

define um subcampo de número binário. Por exemplo, bit seis até bit zero do byte 00111010_2 (mostrado na base dois) representam o subcampo 111010_2 . O $_2$ que segue um número binário indica a base 2. Por isso, 1000_2 é igual a 8_{10} , enquanto F_{16} é igual a 15_{10} .

- 5 R_x : é um registro. Um registro é qualquer dispositivo capaz de armazenar e fornecer dados. A funcionabilidade adicional de um registro é descrita abaixo. Um registro não é necessariamente parte do pacote do processador.

SRC1, SRC2, e DEST:

- 10 identificam áreas de armazenamento (por exemplo, endereços de memória, registros, etc.)

Fonte1-i e Resultado1-i:

representam dados.

SISTEMA DE COMPUTADOR

- 15 A Figura 1 ilustra um sistema de computador exemplificativo 100, de acordo com uma concretização da invenção. O sistema de computador 100 inclui uma barra 101, ou outros hardware e software de comunicação, para comunicar a informação, e um processador 109 acoplado com a barra 101 para o processamento da informação. O processador 109 representa uma unidade de processamento central de qualquer tipo de arquitetura, incluindo uma arquitetura tipo CISC ou RISC. O sistema de computador 100 adicionalmente inclui uma memória de acesso aleatório (RAM) ou outro dispositivo de armazenamento dinâmico (mencionado como memória principal 104), acoplado à barra 101 para armazenar a informação e as instruções a serem executadas pelo processador 109. A memória principal 104 também pode ser usada para armazenar variáveis temporárias ou outra informação intermediária durante a execução de instruções através do processador 109. O sistema de computador 100 também inclui uma memória de leitura apenas (ROM) 106, e/ou outro dispositivo de armazenamento estático, acoplado à barra 101 para armazenar a informação estática e as instruções para o processador 109. O dispositivo de armazenamento de dados 107 é acoplado à barra 101 para armazenar informação e instruções.
- 20
- 25
- 30

A Figura 1 também ilustra que o processador 109 inclui uma unidade de execução 130, um arquivo de registro geral 150, um cache 160, um decodificador 165, e uma barra interna 170. Naturalmente, o processador 109 contém circuitos adicionais que não são mostrados para que não obscureçam a invenção.

A unidade de execução 130 é usada para executar instruções recebidas pelo processador 109. Em adição ao reconhecimento de instruções tipicamente implementadas nos processadores de finalidades gerais, a unidade de execução 130 reconhece as instruções no conjunto de instruções em pacote 140 para a execução de operações nos formatos de dados em pacote. Em uma concretização, o conjunto de instruções em pacote 140 inclui instruções para suportar a(s) operação(ções) de empacotamento, a(s) operação(ções) de desempacotamento, a(s) operação(ções) de soma em pacote, a(s) operação(ções) de subtração em pacote, a(s) operação(ções) de multiplicação em pacote, a(s) operação(ções) de deslocamento em pacote, a(s) operação(ções) de comparação, a(s) operação(ções) de multiplicação-adição, a(s) operação(ções) de multiplicação-subtração, a(s) operação(ções) de contagem de ocupação, e um conjunto de operações lógicas em pacote (incluindo AND em pacote, ANDNOT em pacote, OR em pacote, e XOR em pacote), na maneira descrita posteriormente aqui. Enquanto uma concretização é descrita, na qual o conjunto de instruções em pacote 140 inclui estas instruções, a concretização alternativa pode conter um subconjunto ou um superconjunto destas instruções.

Com a inclusão destas instruções, as operações exigidas por muitos dos algoritmos usados nas aplicações de múltiplos meios podem ser executadas com o uso de dados em pacote. Dessa forma, estes algoritmos podem ser escritos para empacotar os dados necessários e para executar as operações necessárias nos dados em pacote, sem exigir que os dados em pacote sejam desempacotados para executar uma ou mais operações de um elemento de dados de cada vez. Conforme anteriormente descrito, isto proporciona vantagens de desempenho sobre os processadores de finalidades gerais da técnica anterior que não suportam as operações de dados em pa-

cote exigidas por certos algoritmos de múltiplos meios -- isto é, se um algoritmo de múltiplos meios exigir uma operação que não possa ser executada nos dados em pacote, o programa tem que desempacotar os dados, executar a operação nos elementos separados individualmente, e depois empacotar os resultados em um resultado em pacote para o processamento em pacote adicional. Em adição, a maneira descrita, na qual são executadas várias das instruções, aperfeiçoa o desempenho das muitas aplicações de múltiplos meios.

A unidade de execução 130 é acoplada ao arquivo de registro geral 150 através da barra interna 170. O arquivo de registro geral 150 representa uma área de armazenamento no processador 109 para armazenar a informação, incluindo os dados. É entendido que um aspecto da invenção é o conjunto de instruções descrito para operação em dados em pacote. De acordo com este aspecto da invenção, a área de armazenamento usada para armazenar os dados em pacote não é crítica. Entretanto, uma concretização do arquivo de registro geral 150 é descrita mais tarde com referência à Figura 2. A unidade de execução 130 é acoplada ao cache 160 e ao decodificador 165. O cache 160 é usado para melhorar o desempenho dos dados e/ou sinais de controle originários, por exemplo, da memória principal 104. O decodificador 165 é usado para decodificar as instruções recebidas pelo processador 109 em sinais de controle e/ou pontos de entrada de microcódigo. Em resposta a estes sinais de controle e/ou pontos de entrada de microcódigo, a unidade de execução 130 executa as operações adequadas. Por exemplo, se uma instrução de soma for recebida, o decodificador 165 faz com que a unidade de execução 130 execute a adição exigida; se uma instrução de subtração for recebida, o decodificador 165 faz com que a unidade de execução 130 execute a subtração exigida; etc. O decodificador 165 pode ser implementado com o uso de qualquer número de diferentes mecanismos (por exemplo, uma tabela de consulta, uma implementação de hardware, um PLA (Matriz Lógica Programável), etc.). Dessa forma, enquanto a execução das diversas instruções pelo decodificador e pela unidade de execução é representada por uma série de declarações se/então, é entendido que a

execução de uma instrução não requer um processamento em série destas declarações de se/então. Ao invés disso, qualquer mecanismo para executar este processamento se/então com lógica é considerado como estando dentro do escopo da invenção.

5 A Figura 1 adicionalmente mostra um dispositivo de armazenamento de dados 107, tal como um disco magnético ou disco ótico, e sua unidade de disco correspondente. O sistema de computador 100 pode também ser acoplado através da barra 101 a um dispositivo de vídeo 121 para exibir a informação a um usuário do computador. O dispositivo de vídeo 121 pode
10 incluir um buffer de quadro, dispositivos de finalização de gráficos, um tubo de raios catódicos (CRT), e/ou um vídeo de painel plano. Um dispositivo de entrada alfanumérico 122, incluindo teclas alfanuméricas e outras teclas, é tipicamente acoplado à barra 101 para comunicar a informação e as seleções de comando ao processador 109. Outro tipo de dispositivo de entrada
15 de usuário é o controle de cursor 123, tal como um mouse(dispositivo para apontar), um trackball uma caneta, uma tela sensível ao toque, ou teclas de direção de cursor para comunicar a informação de direção e as seleções de comando ao processador 109, e para controlar o movimento do cursor no dispositivo de vídeo 121. Este dispositivo de entrada tipicamente possui dois
20 graus de liberdade em dois eixos, um primeiro eixo (por exemplo, x) e um segundo eixo (por exemplo, y), que permitem que o dispositivo especifique as posições em um plano. Entretanto, esta invenção não deve ser limitada a dispositivos de entrada com apenas dois graus de liberdade.

Outro dispositivo que pode ser acoplado à barra 101 é um dispositivo de cópia impressa 124 o qual pode ser usado para imprimir as instruções, dados ou outra informação em um meio, tal como papel, filme, ou tipos similares de meios. Adicionalmente, o sistema de computador 100 pode ser acoplado a um dispositivo para a gravação do som, e/ou a repetição de disco 125, tal como um digitalizador acoplado a um microfone para
30 gravar a informação. Adicionalmente, o dispositivo pode incluir um altofalante que é acoplado a um conversor digital a analógico (D/A) para a repetição dos sons digitalizados.

Também o sistema de computador 100 pode ser um terminal em uma rede de computador (por exemplo, uma LAN (Rede Local)). O sistema de computador 100 seria então um subsistema de computador de uma rede de computador. O sistema de computador 100 opcionalmente inclui o dispositivo de digitalização de vídeo 126. O dispositivo de digitalização de vídeo 126 pode ser usado para capturar imagens de vídeo que podem ser transmitidas a outros na rede de computador.

Em uma concretização, o processador 109 adicionalmente suporta um conjunto de instruções que é compatível com o conjunto de instruções x86 (o conjunto de instruções usado pelos microprocessadores existentes, tais como o microprocessador Pentium[®], fabricado pela Intel Corporation of Santa Clara, Califórnia). Dessa maneira, em uma concretização, o processador 109 suporta todas as operações suportadas na IA[®] - Arquitetura da Intel, como definido pela Intel Corporation of Santa Clara, Califórnia (vide Microprocessadores, Livros de Dados da Intel, volume 1 e volume 2, 1992 e 1993, disponível pela Intel of Santa Clara, Califórnia). Como resultado, o processador 109 pode suportar as operações x86 existentes, em adição às operações da invenção. Enquanto a invenção é descrita como sendo incorporada ao conjunto de instruções com base em x86, concretizações alternativas poderiam incorporar a invenção em outros conjuntos de instruções. Por exemplo, a invenção poderia ser incorporada a um processador de 64 bits com o uso de um novo conjunto de instruções.

A Figura 2 ilustra o arquivo de registro geral do processador, de acordo com uma concretização da invenção. O arquivo de registro geral 150 é usado para armazenar informações, incluindo a informação de controle/estado, os dados inteiros, os dados de vírgula flutuante, e os dados em pacote. Na concretização mostrada na Figura 2, o arquivo de registro geral 150 inclui registros inteiros 201, registros 209, registros de estado 208 e registro de indicador de instrução 211. Os registros de estado 208 indicam o estado do processador 109. O registro de indicador de instrução 211 armazena o endereço da próxima instrução a ser executada. Os registros inteiros 201, os registros 209, os registros de estado 208, e o registro de indicador

de instrução 211 são todos acoplados à barra interna 170. Quaisquer registros adicionais seriam também acoplados à barra interna 170.

Em uma concretização, os registros 209 são usados tanto para os dados em pacote como para os dados de vírgula flutuante. Em tal concretização, o processador 109, em qualquer momento, tem que tratar os registros 209 como sendo registros de vírgula flutuante com referência de empilhamento ou registros de dados em pacote com referência de não-empilhamento. Nesta concretização, um mecanismo é incluído para permitir que o processador 109 seja comutado entre a operação nos registros 209 como registros de vírgula flutuante mencionados no empilhamento e registros de dados em pacote mencionados no não-empilhamento. Em outra tal concretização, o processador 109 pode simultaneamente operar nos registros 209 como registros de dados em pacote e de vírgula flutuante mencionados no não-empilhamento. Como outro exemplo, em outra concretização, os mesmos registros podem ser usados para armazenar dados inteiros.

Naturalmente, concretizações alternativas podem ser implementadas para conter mais ou menos conjuntos de registros. Por exemplo, uma concretização alternativa pode incluir um conjunto separado de registros de vírgula flutuante para armazenar os dados de vírgula flutuante. Como outro exemplo, uma concretização alternativa pode incluir um primeiro conjunto de registros, cada qual para armazenar a informação de controle/estado, e um segundo conjunto de registros, cada um capaz de armazenar, dados inteiros, de vírgula flutuante e em pacote. Como finalidade de clareza, os registros de uma concretização não devem ter seu significado limitado a um tipo específico de circuito. Ao invés disso, um registro de uma concretização precisa apenas ser capaz de armazenar e fornecer dados, e de executar as funções descritas aqui.

Os diversos conjuntos de registros (por exemplo, os registros interiores 201, os registros 209) podem ser implementados para incluírem diferentes números de registros e/ou registros de tamanho diferente. Por exemplo, em uma concretização, os registros interiores 201 são implementados para armazenar trinta e dois bits, enquanto os registros 209 são im-

plementados para armazenar oitenta bits (todos os oitenta bits sendo usados para armazenar os dados de vírgula flutuante, enquanto apenas sessenta e quatro são usados para dados em pacote). Em adição, os registros 209 contêm oito registros, R₀ 212a a R₇ 212h, R₁ 212a, R₂ 212b e R₃ 212c sendo exemplos de registros individuais nos registros 209. Trinta e dois bits de um registro nos registros 209 podem ser movidos para um registro interior nos registros inteiros 201. Similarmente, um valor em um registro inteiro pode ser movido para trinta e dois bits de um registro nos registros 209. Em outra concretização, os registros inteiros 201 contêm, cada qual, 64 bits, e 64 bits de dados podem ser movidos entre o registro inteiro 201 e os registros 209.

A Figura 3 é um diagrama de fluxo que ilustra as etapas gerais usadas pelo processador para manipular os dados, de acordo com uma concretização da invenção. Por exemplo, tais operações incluem uma operação de carregamento para carregar um registro no arquivo de registro geral 150 com dados originários do cache 160, da memória principal 104, da memória de leitura apenas (ROM) 106, ou do dispositivo de armazenamento de dados 107.

Na etapa 301, o decodificador 202 recebe um sinal de controle 207 originário do cache 160 ou da barra 101. O decodificador 202 decodifica o sinal de controle para determinar as operações a serem executadas.

Na etapa 302, o Decodificador 202 acessa o arquivo de registro geral 150, ou uma localização na memória. Os registros no arquivo de registro geral 150, ou as localizações de memória na memória, são acessados dependendo do endereço de registro especificado no sinal de controle 207. Por exemplo, para uma operação em dados em pacote, o sinal de controle 207 pode incluir os endereços de registro SRC1, SRC2 e DEST. SRC1 é o endereço do primeiro registro de fonte. SRC2 é o endereço do segundo registro de fonte. Em alguns casos, o endereço SRC2 é opcional, na medida em que nem todas as operações exigem dois endereços de fonte. Se o endereço SRC2 não for exigido para uma operação, então apenas o endereço SRC1 é usado. DEST é o endereço do registro de destino, onde são armazenados os dados de resultado. Em uma concretização, SRC1 e SRC2 são

também usados como DEST. SRC1, SRC2 e DEST são descritos mais completamente em relação à Figura 6a e Figura 6b. Os dados armazenados nos registros correspondentes são mencionados como Fonte1, Fonte2, e Resultado, respectivamente. Cada um destes dados possui sessenta e quatro bits de comprimento.

Em outra concretização da invenção, qualquer um, ou todos os SRC1, SRC2 e DEST podem definir uma localização de memória no espaço de memória endereçável do processador 109. Por exemplo, SRC1 pode identificar uma localização de memória na memória principal 104, enquanto SRC2 identifica um primeiro registro nos registros inteiros 201 e DEST identifica um segundo registro nos registros 209. Para simplicidade de descrição aqui, a invenção será descrita em relação ao acesso do arquivo de registro geral 150. Contudo, estes acessos poderiam ser feitos, ao invés disso, para a memória.

Na etapa 303, a unidade de execução 130 é possibilitada de executar a operação nos dados acessados. Na etapa 304, o resultado é armazenado de volta para o arquivo de registro geral 150, de acordo com as exigências do sinal de controle 207.

Formatos de Dados e de Armazenamento

A Figura 4 ilustra os tipos de dados em pacote, de acordo com uma concretização da invenção. São ilustrados três formatos de dados em pacote: byte em pacote 401, palavra em pacote 402 e a palavra dupla em pacote 403. O byte em pacote, em uma concretização da invenção, tem sessenta e quatro bits de comprimento contendo oito elementos de dados. Cada elemento de dados tem um byte de comprimento. De maneira geral, um elemento de dados é uma peça individual de dados que é armazenada em um único registro (ou localização de memória) com outros elementos de dados de comprimento igual. Em uma concretização da invenção, o número de elementos de dados armazenados em um registro é de sessenta e quatro bits divididos pelo comprimento em bits de um elemento de dados.

A palavra em pacote 402 tem sessenta e quatro bits de comprimento e contém quatro elementos de dados de palavra 402. Cada elemento

de dados de palavra 402 contém dezesseis bits de informação.

A palavra dupla em pacote 403 tem sessenta e quatro bits de comprimento e contém dois elementos de dados de palavra dupla 403. Cada elemento de dados de palavra dupla 403 contém trinta e dois bits de informação.

As Figuras de 5a a 5c ilustram a representação de armazenamento de dados em pacote em registro, de acordo com uma concretização da invenção. A representação em registro de byte em pacote sem sinal algébrico 510 ilustra o armazenamento de um byte em pacote sem sinal algébrico 401 em um dos registros R₀ 212a a R₇ 212h. A informação para cada elemento de dados de byte é armazenado do bit sete até o bit zero para byte zero, do bit 15 até o bit oito para byte um, do bit vinte e três até o bit dezesseis para byte dois, do bit trinta e um até o bit vinte e quatro para byte três, do bit trinta e nove até o bit trinta e dois para byte quatro, do bit quarenta e sete até o bit quarenta para byte cinco, do bit cinquenta e cinco até o bit quarenta e oito para byte seis e do bit sessenta e três até o bit cinquenta e seis para byte sete. Desse modo, todos os bits disponíveis são usados no registro. Esta disposição de armazenamento aumenta a eficiência de armazenamento do processador. Da mesma forma, com oito elementos de dados acessados, uma operação pode agora ser executada nos oito elementos de dados simultaneamente. A representação em registro de byte em pacote com sinal algébrico 511 ilustra o armazenamento de um byte em pacote com sinal algébrico 401. Deve ser notado que apenas o oitavo bit de cada elemento de dados de byte é necessário para o indicador de sinal.

Representação em registro de palavra em pacote sem sinal algébrico 512 ilustra como as palavras de três a zero são armazenadas em um registro dos registros 209. Os bits de quinze a zero contêm a informação de elemento de dados para a palavra zero, os bits de trinta e um a dezesseis contêm a informação para a palavra um do elemento de dados, os bits de quarenta e sete a trinta e dois contêm a informação para a palavra dois do elemento de dados e os bits de sessenta e três a quarenta e oito contêm a informação para a palavra três do elemento de dados. A representação em

registro de palavra em pacote com sinal algébrico 513 é similar à representação em registro de palavra em pacote sem sinal algébrico 512. Deve ser notado que apenas o décimo-sexto bit de cada elemento de dados de palavra é o necessário para o indicador de sinal.

5 A representação em registro de palavra dupla em pacote sem sinal algébrico 514 mostra como os registros 209 armazenam dois elementos de dados de palavra dupla. A palavra dupla zero é armazenada do bit trinta e um até o bit zero do registro. A palavra dupla um é armazenada do bit sessenta e três até o bit trinta e dois do registro. A representação em
10 registro de palavra dupla em pacote com sinal algébrico 515 é similar à representação em registro de palavra dupla em pacote sem sinal algébrico 514. Deve ser notado que o bit de sinal necessário é o trigésimo segundo bit do elemento de dados de palavra dupla.

Conforme mencionado anteriormente, os registros 209 podem
15 ser usados tanto para os dados em pacote como para os dados de vírgula flutuante. Nesta concretização da invenção, o processador de programação individual 109 pode ser exigido de trilhar, se um registro endereçado, R_0 212a por exemplo, estiver armazenando os dados em pacote ou os dados de vírgula flutuante. Em uma concretização alternativa, o processador 109
20 poderia trilhar o tipo de dados armazenados nos registros individuais dos registros 209. Esta concretização alternativa poderia então gerar erros, se, por exemplo, fosse feita uma tentativa para uma operação de adição em pacote nos dados de vírgula flutuante.

Formatos de Sinal de Controle

25 O seguinte descreve uma concretização dos formatos de sinal de controle usados pelo processador 109 para manipular os dados em pacote. Em uma concretização da invenção, os sinais de controle são representados como trinta e dois bits. O decodificador 202 pode receber o sinal de controle 207 originário da barra 101. Em outra concretização, o decodifi-
30 cador 202 pode também receber tais sinais de controle originários do cache 160.

A Figura 6 ilustra um formato de sinal de controle para indicar o

uso de dados em pacote, de acordo com uma concretização da invenção. O campo de operação OP 601, do bit trinta e um até o bit vinte e seis, fornece a informação a cerca da operação a ser executada pelo processador 109; por exemplo, a adição em pacote, a subtração em pacote, etc. SRC1 602, do bit vinte e cinco até o bit vinte, fornece o endereço de registro de fonte de um registro nos registros 209. Este registro de fonte contém os primeiros dados em pacote, Fonte1, a serem usados na execução do sinal de controle. Similarmente, SRC2 603, do bit dezenove até o bit quatorze, contém o endereço de um registro nos registros 209. Este segundo registro de fonte contém os dados em pacote, Fonte2, a serem usados durante a execução da operação. DEST 605, do bit cinco até o bit zero, contém o endereço de um registro nos registros 209. Este registro de destino armazenará os dados em pacote de resultado, Resultado, da operação de dados em pacote.

Os bits de controle SZ 610, bit doze e bit treze, indicam o comprimento dos elementos de dados nos primeiro e segundo registros de fonte de dados em pacote. Se SZ 610 for igual a 01_2 , então os dados em pacote serão formatados como byte em pacote 401. Se SZ 610 for igual a 10_2 ; então os dados em pacote serão formatados como palavra em pacote 402. SZ 610 que é igual a 00_2 ou 11_2 é reservado; contudo, em outra concretização, um destes valores poderia ser usado para indicar a palavra dupla em pacote 403.

O bit de controle T 611, bit onze, indica se a operação deve ser executada com o modo de saturação. Se T 611 for igual a um, então uma operação de saturação será executada. Se T 611 for igual a zero, então uma operação de não-saturação será executada. Operações de saturação serão descritas mais tarde.

O bit de controle S 612, bit dez, indica o uso de uma operação com sinal algébrico. Se S 612 for igual a um, então uma operação com sinal algébrico será executada. Se S612 for igual a zero, então uma operação sem sinal algébrico será executada.

A Figura 6b ilustra um segundo formato de sinal de controle para indicar o uso de dados em pacote, de acordo com uma concretização da

invenção. Este formato corresponde ao formato de código de operação interior geral descrito no "Manual do Usuário da Família de Processador Pentium", disponível pela Intel Corporation, Vendas Literárias, P.O. Box 7641, prospecto Mt., IL, 60056-7641. Deve ser notado que OP 601, SZ 610, T 611 e S 612 são todos combinados em um grande campo. Para alguns sinais de controle, os bits de três a cinco são SRC1 602. Em uma concretização, onde haja um endereço SRC1 602, então os bits de três a cinco também correspondem ao DEST 605. Em uma concretização alternativa, onde haja um endereço SRC2 603, então os bits de zero a dois também correspondem ao DEST 605. Para outros sinais de controle, como uma operação imediata de deslocamento em pacote, os bits de três até cinco representam uma extensão ao campo de código de operação. Em uma concretização, esta extensão permite que um programador inclua um valor imediato com o sinal de controle, tal como um valor de contagem de deslocamento. Em uma concretização, o valor imediato segue o sinal de controle. Isto é descrito em maiores detalhes no "Manual do Usuário da Família de Processador Pentium", no apêndice F, nas páginas F-1 a F-3. Os bits de zero a dois representam SRC2 603. Este formato geral permite o endereçamento de registro a registro, de memória a registro, de registro por memória, de registro por registro, de registro por imediato e de registro à memória. Também, em uma concretização, este formato geral pode suportar o endereçamento de registro inteiro a registro, e de registro a registro inteiro.

Descrição de Saturação/Insaturação

Conforme mencionado anteriormente, T 611 indica se as operações opcionalmente saturam. No caso de o resultado de uma operação, com possibilidade de saturação, saturar a capacidade ou saturar negativamente a faixa dos dados, o resultado será retido. Meios de retenção ajustam o resultado em um valor máximo ou mínimo, na eventualidade de um resultado exceder o valor máximo ou mínimo da faixa. No caso de estouro negativo, a saturação retém o resultado no valor mais baixo na faixa e, no caso de saturação de capacidade, no valor mais alto. A faixa permissível para cada formato de dados é mostrada na Tabela 7.

Tabela 7

Formato dos Dados	Valor Mínimo	Valor Máximo
Byte sem sinal algébrico	0	255
Byte com sinal algébrico	-128	127
Palavra sem sinal algébrico	0	65535
Palavra com sinal algébrico	-32768	32767
Palavra dupla sem sinal algébrico	0	$2^{64}-1$
Palavra dupla com sinal algébrico	-2^{63}	$2^{63}-1$

Conforme mencionado acima, T 611 indica se as operações de saturação estão sendo executadas ou não. Por isso, com o uso de um formato de dados de byte sem sinal algébrico, no caso de um resultado de operação = 258 e a saturação ter sido ativada, então o resultado seria retido em 255, antes de ser armazenado no registro de destino da operação. Similarmente, se o resultado de uma operação = -32999 e o processador usou um formato de dados de palavra com sinal algébrico com a possibilidade de saturação, então o resultado seria retido em -32768, antes de ser armazenado no registro de destino da operação.

ADIÇÃO EM PACOTE

Operação de Adição em pacote

Uma concretização da invenção permite que as operações de adição em pacote sejam executadas na unidade de Execução 130. Isto é, a invenção permite que cada elemento de dados dos primeiros dados em pacote seja somado individualmente a cada elemento de dados dos segundos dados em pacote.

A Figura 7a ilustra um processo para a execução de adição em pacote, de acordo com uma concretização da invenção. Na etapa 701, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa forma, o decodificador 202 decodifica o código de operação para adição em pacote, os endereços SRC1 602, SRC2 603 e DEST 605 nos registros 209, a saturação/insaturação, a aritmética com sinal algébrico/sem sinal algébrico, e o comprimento dos elementos de dados nos dados em pacote. Na etapa 702, através da barra interna 170, o decodificador 202 aces-

sa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 e SRC2 603. Os registros 209 fornecem a unidade de Execução 130 com os dados em pacote armazenados nos registros nestes endereços, Fonte1 e Fonte2, respectivamente. Isto é, os registros 209 comunicam os dados em pacote à unidade de Execução 130 através da barra interna 170.

Na etapa 703, o decodificador 202 permite que a unidade de Execução 130 execute uma operação de adição em pacote. O decodificador 202 adicionalmente comunica, através da barra interna 170, o comprimento dos elementos de dados em pacote, na eventualidade de a saturação vir a ser usada, e na eventualidade da aritmética com sinal algébrico vir a ser usada também. Na etapa 704, o comprimento do elemento de dados determina qual a etapa que deve ser executada a seguir. Se o comprimento dos elementos de dados nos dados em pacote for de oito bits (dados de byte), então a unidade de Execução 130 executa a etapa 705a. Entretanto, se o comprimento dos elementos de dados nos dados em pacote for de dezesseis bits (dados de palavras), então a unidade de Execução 130 executa a etapa 705b. Em uma concretização da invenção, apenas é suportada a adição em pacote com comprimento de elemento de dados de oito bits e de dezesseis bits. Entretanto, concretizações alternativas podem suportar comprimentos diferentes e/ou outros comprimentos. Por exemplo, uma concretização alternativa poderia adicionalmente suportar uma adição em pacote do comprimento de elemento de dados de trinta e dois bits.

Assumindo-se que o comprimento dos elementos de dados é de oito bits, então é executada a etapa 705a. A unidade de Execução 130 soma os bits de sete até zero da Fonte1 aos bits de sete até zero de SRC2, produzindo os bits de sete até zero dos dados em pacote do Resultado. Em paralelo com esta adição, a unidade de Execução 130 soma os bits de quinze até oito da Fonte1 aos bits de quinze até oito da Fonte 2, produzindo os bits de quinze até oito dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de vinte e três até dezesseis da Fonte 1 aos bits de vinte e três até dezesseis da Fonte2, produzindo

os bits de vinte e três até dezesseis dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de trinta e um até vinte e quatro da Fonte1 aos bits de trinta e um até vinte e quatro da Fonte2, produzindo os bits de trinta e um até vinte e quatro dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de trinta e nove até trinta e dois da Fonte1 aos bits de trinta e nove até trinta e dois da Fonte2, produzindo os bits de trinta e nove até trinta e dois dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de quarenta e sete até quarenta da Fonte1 aos bits de quarenta e sete até quarenta da Fonte 2, produzindo os bits de quarenta e sete até quarenta dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de cinquenta e cinco até quarenta e oito da Fonte 1 aos bits de cinquenta e cinco até quarenta e oito da Fonte2, produzindo os bits de cinquenta e cinco até quarenta e oito dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de sessenta e três até cinquenta e seis da Fonte1 aos bits de sessenta e três até cinquenta e seis da Fonte2, produzindo os bits de sessenta e três até cinquenta e seis dos dados em pacote do Resultado.

Assumindo-se que o comprimento dos elementos de dados é de dezesseis bits, então é executada a etapa 705b. A unidade de execução 130 soma os bits de quinze até zero da Fonte1 aos bits de quinze até zero de SRC2, produzindo os bits de quinze até zero dos dados em pacote do Resultado. Em paralelo com esta adição, a unidade de Execução 130 soma os bits de trinta e um até dezesseis da Fonte1 aos bits de trinta e um até dezesseis da Fonte 2, produzindo os bits de trinta e um até dezesseis dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 13 soma os bits de quarenta e sete até trinta e dois da Fonte1 aos bits de quarenta e sete até trinta e dois da Fonte2, produzindo os bits de quarenta e sete até trinta e dois dos dados em pacote do Resultado. Em paralelo com estas adições, a unidade de Execução 130 soma os bits de sessenta e três até quarenta e oito da Fonte1 aos bits de sessenta e três até

quarenta e oito da Fonte2, produzindo os bits de sessenta e três até quarenta e oito dos dados em pacote do Resultado.

Na etapa 706, o decodificador 202 permite um registro nos registros 209 com o endereço DEST 605 do registro de destino. Dessa maneira, o Resultado é armazenado no registro endereçado por DEST 605.

A Tabela 8a ilustra a representação em registro da operação de adição em pacote. A primeira fileira de bits é a representação de dados em pacote dos dados em pacote da Fonte1. A segunda fileira de bits é a representação de dados em pacote dos dados em pacote da Fonte2. A terceira fileira de bits é a representação de dados em pacote dos dados em pacote do Resultado. O número abaixo de cada bit de elemento de dados é o número do elemento de dados. Por exemplo, o elemento de dados 0 da Fonte1 é 10001000_2 . Por isso, se os elementos de dados tiverem oito bits de comprimento (dados de byte), e sem sinal algébrico, é executada a adição não-saturada, a unidade de Execução 130 fornecendo os dados em pacote do Resultado, conforme mostrado.

Deve ser notado que em uma concretização da invenção, onde um resultado satura a capacidade ou satura negativamente e a operação que é usada é insaturada, esse resultado é simplesmente truncado. Isto é, o bit de transporte é ignorado. Por exemplo na Tabela 8a, na representação em registro do elemento de dados um do resultado seria: $10001000_2 + 10001000_2 = 00001000_2$. Similarmente, para as saturações negativas, o resultado é truncado. Esta forma de truncamento permite que um programador facilmente execute a aritmética de módulo. Por exemplo, uma equação para um elemento de dados do resultado pode ser expressa como: (elemento de dados um da Fonte1 + elemento de dados um da Fonte2) mod 256 = um elemento de dados de resultado. Adicionalmente, aquele versado na técnica entenderia a partir desta descrição que as saturações de capacidade e as saturações negativas poderiam ser detectadas através do ajuste dos bits de erro em um registro de estado.

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
<u>+</u> ²	<u>+</u> ⁶	<u>+</u> ²	<u>+</u> ⁴	<u>+</u> ²	<u>+</u> ²	<u>+</u> ²	<u>+</u> ²
10101010	01010101	10101010	10000001	10000000	11110000	11001111	10001000
<u>=</u> ²	<u>=</u> ⁶	<u>=</u> ²	<u>=</u> ⁴	<u>=</u> ²	<u>=</u> ²	<u>=</u> ²	<u>=</u> ²
11010100	10101010	11111111	Overflow	Overflow	Overflow	Overflow	Overflow
7	6	5	4	3	2	1	0

onde Overflow = Saturação de capacidade

Tabela 8a

A Tabela 8b ilustra a representação em registro de uma operação de adição de dados de palavra em pacote. Por isso, se os elementos de dados apresentarem dezesseis bits de comprimento (dados de palavra), e sem sinal algébrico, é executada a adição não-saturada, a unidade de Execução 130 produzindo os dados em pacote do Resultado, conforme mostrado. Deve ser notado que no elemento de dados dois de palavra, o transporte a partir do bit sete (vide bits 1 enfatizado abaixo) propagado para o bit oito, fazendo com que o elemento de dados dois sature sua capacidade (vide saturação de capacidade enfatizada abaixo).

00101010 01010101	01010101 11111111	10000000 01110000	10001111 10001000
<u>+</u> ²	<u>+</u> ²	<u>+</u> ²	<u>+</u> ²
10101010 01010101	10101010 10000001	10000000 11110000	11001111 10001000
<u>=</u> ²	<u>=</u> ²	<u>=</u> ²	<u>=</u> ²
11010100 10101010	Overflow	Overflow	Overflow
3	2	1	0

onde Overflow = Saturação de capacidade

Tabela 8b

A Tabela 8c ilustra a representação em registro da operação de adição de dados de palavra dupla em pacote. Esta operação é suportada em uma concretização alternativa da invenção. Por isso, se os elementos de dados tiverem trinta e dois bits de comprimento (isto é, dados de palavra

dupla), e sem sinal algébrico, é executada a adição insaturada, a unidade de Execução 130 produzindo os dados em pacote do Resultado, conforme mostrado. Deve ser notado que os transportes a partir do bit sete e do bit quinze do elemento de dados um de palavra dupla propagaram para o bit 5 oito e o bit dezesseis, respectivamente.

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
+				+			
10101010	01010101	10101010	10000001	10000000	11110000	11001111	10001000
=				=			
11010100	10101011	00000000	10000000	Overflow			

onde Overflow = Saturação de capacidade

Tabela 8c

Para melhor ilustrar a diferença entre a adição em pacote e a adição ordinária, os dados originários do exemplo acima são duplicados na Tabela 9. Entretanto, neste caso, a adição ordinária (sessenta e quatro bits) é executada nos dados. Deve ser notado que os transportes a partir do bit sete, do bit quinze, do bit vinte e três, do bit trinta e um, do bit trinta e nove e do bit quarenta e sete foram conduzidos para o bit oito, o bit dezesseis, o bit vinte e quatro, o bit trinta e dois, o bit quarenta e o bit quarenta e oito, respectivamente.

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
±							
10101010	01010101	10101010	10000001	10000000	11110000	11001111	10001000
≡							
11010100	10101011	00000000	10000001	00000001	01100001	01011111	00010000

Tabela 9

Adição em pacote Com Sinal Algébrico/Insaturada

A Tabela 10 ilustra um exemplo de uma adição em pacote com

sinal algébrico, onde o comprimento do elemento de dados dos dados em pacote é de oito bits. A saturação não é usada. Por isso, os resultados podem saturar a capacidade ou saturar negativamente. A Tabela 10 usa dados diferentes dos das Tabelas de 8a-8c e Tabela 9.

00101010	01010101	01010101	01111111	00000000	11110000	00001111	10001000
$\frac{+}{\Sigma}^7$	$\frac{+}{\Sigma}^6$	$\frac{+}{\Sigma}$	$\frac{+}{\Sigma}^4$	$\frac{+}{\Sigma}^3$	$\frac{+}{\Sigma}^2$	$\frac{+}{\Sigma}^1$	$\frac{+}{\Sigma}^0$
10101010	01010101	10101010	00000001	00000000	11110000	00001111	10001000
$\frac{=}{\Sigma}^7$	$\frac{=}{\Sigma}^6$	$\frac{=}{\Sigma}$	$\frac{=}{\Sigma}^4$	$\frac{=}{\Sigma}^3$	$\frac{=}{\Sigma}^2$	$\frac{=}{\Sigma}^1$	$\frac{=}{\Sigma}^0$
11010100	Overflow	11111111	Overflow	00000000	Underflow	00011110	Underflow
7	6	5	4	3	2	1	0

- 5 onde Overflow = Saturação de capacidade e
Underflow = Saturação negativa

Tabela 10

Adição em pacote Com Sinal Algébrico/Saturada

A Tabela 11 ilustra um exemplo de uma adição em pacote com
10 sinal algébrico, onde o comprimento do elemento de dados dos dados em
pacote é de oito bits. A saturação é usada; conseqüentemente, a saturação
de capacidade será retida no valor máximo, e a saturação negativa será reti-
da no valor mínimo. A Tabela 11 usa os mesmos dados que a Tabela 10.
Aqui, o elemento de dados zero e o elemento de dados dois são retidos no
15 valor mínimo, enquanto o elemento de dados quatro e o elemento de dados
seis são retidos no valor máximo.

00101010	01010101	01010101	01111111	00000000	11110000	00001111	10001000
$\begin{array}{r} + \\ \hline \end{array}$ ^Z	$\begin{array}{r} + \\ \hline \end{array}$ ⁶	$\begin{array}{r} + \\ \hline \end{array}$ ⁵	$\begin{array}{r} + \\ \hline \end{array}$ ⁴	$\begin{array}{r} + \\ \hline \end{array}$ ³	$\begin{array}{r} + \\ \hline \end{array}$ ²	$\begin{array}{r} + \\ \hline \end{array}$ ¹	$\begin{array}{r} + \\ \hline \end{array}$ ⁰
10101010	01010101	10101010	00000001	00000000	11110000	00001111	10001000
$\begin{array}{r} = \\ \hline \end{array}$ ^Z	$\begin{array}{r} = \\ \hline \end{array}$ ⁶	$\begin{array}{r} = \\ \hline \end{array}$ ⁵	$\begin{array}{r} = \\ \hline \end{array}$ ⁴	$\begin{array}{r} = \\ \hline \end{array}$ ³	$\begin{array}{r} = \\ \hline \end{array}$ ²	$\begin{array}{r} = \\ \hline \end{array}$ ¹	$\begin{array}{r} = \\ \hline \end{array}$ ⁰
11010100	01111111	11111111	01111111	00000000	10000000	00011110	10000000
7	6	5	4	3	2	1	0

Tabela 11

SUBTRAÇÃO EM PACOTE

Operação de Subtração Em pacote

Uma concretização da invenção permite que as operações de subtração em pacote sejam executadas na unidade de Execução 130. Isto é, a invenção permite que cada elemento de dados dos segundos dados em pacote seja subtraído individualmente a partir de cada elemento de dados dos primeiros dados em pacote.

A Figura 7b ilustra um processo para a execução da subtração em pacote, de acordo com uma concretização da invenção. Deve ser notado que as etapas 710-713 são similares às etapas 701-704.

Na presente concretização da invenção, apenas é suportada a subtração em pacote do comprimento do elemento de dados de oito bits e de dezesseis bits. Entretanto, concretizações alternativas podem suportar comprimentos diferentes e/ou outros comprimentos. Por exemplo, uma concretização alternativa poderia adicionalmente suportar a subtração em pacote do comprimento do elemento de dados de trinta e dois bits.

Assumindo-se que o comprimento do elemento de dados é de oito bits, são executadas as etapas 714a e 715a. A unidade de Execução 130 executa a aritmética dos complementos a 2 do bit sete até o bit zero da Fonte2. Em paralelo com este complemento a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit quinze até o bit oito da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit vinte e três até o bit

dezesseis da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit trinta e nove até o bit trinta e dois da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit quarenta e sete até o bit quarenta da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit cinquenta e cinco até o bit quarenta e oito da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit sessenta e três até o bit cinquenta e um da Fonte2. Na etapa 715a, a unidade de Execução 130 executa a adição dos bits de complementos a 2 da Fonte2 aos bits da Fonte1, conforme descrito, de maneira geral, para a etapa 705a.

Assumindo-se que o comprimento do elemento de dados é de dezesseis bits, são executadas as etapas 714b e 715b. A unidade de Execução 130 executa a aritmética dos complementos a 2 do bit quinze até o bit zero da Fonte2. Em paralelo com este complemento a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit trinta e um até o bit dezesseis da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit quarenta e sete até o bit trinta e dois da Fonte2. Em paralelo com estes complementos a 2, a unidade de Execução 130 executa a aritmética dos complementos a 2 do bit sessenta e três até o bit quarenta e oito da Fonte2. Na etapa 715b, a unidade de Execução executa a adição dos bits de complementos a 2 da Fonte2 aos bits da Fonte1, conforme descrito, de maneira geral, para a etapa 705b.

Deve ser notado que as etapas 714 e 715 são o processo usado em uma concretização da invenção para subtrair um primeiro número a partir de um segundo número. Entretanto, outras formas de subtração são conhecidas na técnica e esta invenção não deve ser considerada limitada ao uso da aritmética do complemento a 2.

Na etapa 716, o decodificador 202 ativa os registros 209 com o endereço de destino do registro de destino. Desse modo, os dados em pa-

cote do resultado são armazenados no registro DEST dos registros 209.

A Tabela 12 ilustra a representação em registro da operação de subtração em pacote. Assumindo-se que os elementos de dados se apresentam com oito bits de comprimento (dados de byte), e sem sinal algébrico, é executada a subtração insaturada; a seguir, a unidade de Execução 130 produz os dados em pacote do resultado, conforme mostrado.

00101010	01010101	01010101	01111111	00000000	11110000	00001111	10001000
<u> </u> ²	<u> </u> ⁶	<u> </u> ²	<u> </u> ⁴	<u> </u> ²	<u> </u> ²	<u> </u> ⁴	<u> </u> ²
10101010	01010101	10101010	00000001	00000000	11110000	00001111	10001000
<u> </u> ²	<u> </u> ²	<u> </u> ²	<u> </u> ⁴	<u> </u> ²	<u> </u> ²	<u> </u> ⁴	<u> </u> ²
Underflow	00000000	Underflow	01111110	00000000	00000000	00000000	00000000
7	6	5	4	3	2	1	0

Underflow = Saturação negativa

Tabela 12

Circuitos de Adição/Subtração de Dados em pacote

10 A Figura 8 ilustra um circuito para a execução da adição em pacote e subtração em pacote nos bits individuais de dados em pacote, de acordo com uma concretização da invenção. A Figura 8 mostra os somador/subtrator modificados de fração de bit 800. Os somador/subtrator 801a-d ativam dois bits originários da Fonte2 para serem somados à Fonte1 ou subtraídos da Fonte1. O controle de operação e de transporte 803 transmite ao controle 809a sinais de controle para ativar uma operação de adição ou subtração. Dessa forma, os somador/subtrator 801a somam ou subtraem o bit i recebido na Fonte2 _{i} , 805a ao/do bit i recebido na Fonte1 _{i} , 804a, produzindo um bit de resultado transmitido no Resultado _{i} , 806a. C_{in} 807a-b e C_{out} 808a-b representam circuitos de controle de transporte, como é comumente encontrado nos somadores/subtratores.

15

20

O controle de bit 802 é ativado a partir do controle de operação e transporte 803 através da ativação de dados em pacote 811 para o controle C_{ini+1} 807b e C_{outi}. Por exemplo, na Tabela 13a, é executada uma adição de

byte em pacote sem sinal algébrico. Se os somador/subtrator 801a somarem o bit sete da Fonte1 ao bit sete da Fonte2, então o controle de operação e transporte 803 ativará o controle de bit 802, detendo a propagação de um transporte a partir do bit sete para o bit oito.

...	00001111	10001000
$\frac{+}{2}$	$\frac{+}{6}$	$\frac{+}{5}$	$\frac{+}{4}$	$\frac{+}{3}$	$\frac{+}{2}$	$\frac{+}{1}$	$\frac{+}{0}$
...	00001111	10001000
$\frac{=}{2}$	$\frac{=}{6}$	$\frac{=}{5}$	$\frac{=}{4}$	$\frac{=}{3}$	$\frac{=}{2}$	$\frac{=}{1}$	$\frac{=}{0}$
...	00011110	Overflow
7	6	5	4	3	2	1	0

5 onde Overflow = Saturação de capacidade

Tabela 13a

Entretanto, se for executada uma adição de palavra em pacote sem sinal algébrico, e os somador/subtrator 801 forem similarmente usados para somar o bit sete da Fonte1 ao bit sete da Fonte2, o controle de bit 802 propagará o transporte para o bit oito. A Tabela 13b ilustra este resultado. Esta propagação seria permitida para a adição de palavra dupla em pacote, bem como para a adição não-em pacote.

...	00001111	10001000
$\frac{+}{2}$	$\frac{+}{6}$	$\frac{+}{5}$	$\frac{+}{4}$	$\frac{+}{3}$	$\frac{+}{2}$	$\frac{+}{1}$	$\frac{+}{0}$
...	00001111	10001000
$\frac{=}{2}$	$\frac{=}{6}$	$\frac{=}{5}$	$\frac{=}{4}$	$\frac{=}{3}$	$\frac{=}{2}$	$\frac{=}{1}$	$\frac{=}{0}$
...	00011111	00010000
3	2	1	0				

Tabela 13b

Os somador/subtrator 801a subtraem a Fonte2_i 805a da Fonte1_i 804a de bit através primeiramente da formação do complemento a 2 da Fonte2_i 805a invertindo a Fonte2_i 805a e somando um. Depois, os soma-

15

dor/subtrator 801a somam este resultado à Fonte1_i 804a. As técnicas de complementação a 2 de fração de bit são bem conhecidas na técnica, e aquele versado na técnica entenderia como projetar tal circuito de complementação a 2 de fração de bit. Deve ser notado que a propagação de transportes é controlada pelo controle de bit 802 e pelo controle de operação e de transporte 803.

A Figura 9 ilustra um circuito para a execução da adição em pacote e subtração em pacote nos dados de byte em pacote, de acordo com uma concretização da invenção. A barra da Fonte1 901 e a barra da Fonte2 902 transportam os sinais de informação para os somadores/subtratores 908a-h através da Fonte_{in} 906a-h e da Fonte2_{in} 905a-h, respectivamente. Desse modo, os somador/subtrator 908a somam/subtraem do bit sete até o bit zero da Fonte2 ao/do bit sete até o bit zero da Fonte1; os somador/subtrator 908b somam/subtraem do bit quinze ao bit oito da Fonte2 ao/do bit quinze ao bit oito da Fonte1, etc. CTRL 904a-h recebe, a partir do Controle de Operação 903, através do controle em pacote 911, os sinais de controle que desativam a propagação dos transportes, a saturação de ativação/desativação, e a aritmética de ativação/desativação com sinal algébrico/sem sinal algébrico. O Controle de Operação 903 desativa a propagação dos transportes através do recebimento da informação de transporte originária do CTRL 904a-h e não a propaga para os somador/subtrator 908a-h seguintes mais significativos. Dessa maneira, o Controle de Operação 903 executa as operações de controle de operação e de transporte 803 e de controle de bit 802 para os dados em pacote de 64 bits. Aquele versado na técnica seria capaz de criar tal circuito, uma vez fornecidas as ilustrações nas Figuras de 1-9 e a descrição acima.

Os somadores/subtratores 908a-h comunicam a informação do resultado, através da saída de resultado 907a-h, das diversas adições em pacote ao registro de resultado 910a-h. Cada registro de resultado 910a-h armazena e, em seguida, transmite a informação de resultado para a barra de resultado 909. Esta informação de resultado é então armazenada no registro inteiro especificado pelo endereço de registro DEST 605.

A Figura 10 é uma vista lógica de um circuito para executar a adição em pacote e a subtração em pacote nos dados de palavra em pacote, de acordo com uma concretização da invenção. Aqui, as operações de palavra em pacote estão sendo executadas. A propagação de transportes entre o bit oito e o bit sete, o bit vinte e quatro e o bit vinte e três, o bit quarenta e o bit trinta e nove, o bit cinqüenta e seis e o bit cinqüenta e cinco é ativada pelo Controle de Operação 903. Desse modo, os somador/subtrator 908a e 908b, mostrados como somador/subtrator virtuais 1008a, atuarão juntos para somar/subtrair a primeira palavra dos dados de palavra em pacote da Fonte2 (do bit quinze até o bit zero) a/da primeira palavra dos dados de palavra em pacote da Fonte1 (do bit quinze até o bit zero); somador/subtrator 908c e 908d, mostrados como somador/subtrator virtuais 1008b, atuarão juntos para somar/subtrair a segunda palavra dos dados de palavra em pacote da Fonte2 (do bit trinta e um até o bit dezesseis) à/da segunda palavra dos dados de palavra em pacote da Fonte 1 (do bit trinta e um até o bit dezesseis), etc..

Os somadores/subtratores virtuais 1008a-d comunicam a informação de resultado, através da saída de resultado 1007a-d (saídas de resultado combinadas 907a-b, 907c-d, 907e-f e 907g-h), aos registros de resultado virtuais 1010a-d. Cada registro de resultado virtual 1010a-d (registros de resultado combinados 910a-b, 910c-d, 910e-f e 910g-h) armazena um elemento de dados de resultado de dezesseis bits para ser comunicado à barra de Resultado 909.

A Figura 11 é uma vista lógica de um circuito para a execução de adição em pacote e de subtração em pacote nos dados de palavra dupla em pacote, de acordo com uma concretização da invenção. A propagação dos transportes entre o bit oito e bit sete, o bit dezesseis e o bit quinze, o bit vinte e quatro e o bit vinte e três, o bit quarenta e o bit trinta e nove, o bit quarenta e oito e o bit quarenta e sete, e o bit cinqüenta e seis e o bit cinqüenta e cinco é ativada pelo Controle de Operação 903. Desse modo, os somadores/subtratores 908a-d, mostrados como somadores/subtratores virtuais 1108a-d, atuam juntos para somar/subtrair a primeira palavra dupla dos dados de palavra dupla em pacote da Fonte2 (bit trinta e um até o bit zero)

à/da primeira palavra dupla dos dados de palavra dupla em pacote da Fonte1 (bit trinta e um até bit zero); somadores/subtratores 908e-h, mostrados como somadores/subtratores virtuais 1108b, atuam juntos para somar/subtrair a segunda palavra dupla dos dados de palavra dupla em pacote da Fonte2 (bit sessenta e três até o bit trinta e dois) à/da segunda palavra dupla dos dados de palavra dupla em pacote da Fonte 1 (bit sessenta e três até o bit trinta e dois).

Os somadores/subtratores virtuais 1108a-b comunicam a informação de resultado, através da saída de resultado 1107a-b (saídas de resultado combinadas 907a-d e 907e-h), aos registros de resultado virtuais 1110a-b. Cada registro de resultado virtual 110a-b (registros de resultado combinados 910a-d e 910e-h) armazena um elemento de dados de resultado de trinta e dois bits para ser comunicado à barra de Resultado 909.

MULTIPLICAÇÃO EM PACOTE

15 Operação de Multiplicação em pacote

Em uma concretização da invenção, o registro SRC1 contém dados do multiplicando (Fonte1), o registro SRC2 contém dados do multiplicador (Fonte2), e o registro DEST conterá uma porção do produto da multiplicação (Resultado). Isto é, a Fonte1 apresentará cada elemento de dados independentemente multiplicado pelo respectivo elemento de dados da Fonte2. Dependendo do tipo de multiplicação, o Resultado incluirá bits de ordem mais superior ou de ordem mais inferior do produto.

Em uma concretização da invenção, são suportadas as seguintes operações de multiplicação: multiplicação em pacote superior sem sinal algébrico, multiplicação em pacote superior com sinal algébrico e multiplicação em pacote inferior. As multiplicações superior/inferior indicam quais os bits originários do produto de multiplicação que devem ser incluídos no Resultado. Isto se faz necessário porque uma multiplicação de dois números de N bits resulta em um produto que apresenta 2N bits. Como cada elemento de dados de resultado possui o mesmo tamanho que os elementos de dados do multiplicando e do multiplicador, apenas metade do produto pode ser representada pelo resultado. A multiplicação superior faz com que bits de or-

dem mais superior sejam emitidos como resultado. A multiplicação inferior faz com que bits de ordem inferior sejam emitidos como resultado. Por exemplo, a multiplicação superior em pacote sem sinal algébrico da Fonte1[7:0] pela Fonte2[7:0] armazena os bits de ordem superior do produto no

5 Resultado[7:0].

Em uma concretização da invenção, o uso do modificador de operação superior/inferior afasta a possibilidade de uma saturação de capacidade originária de um elemento de dados no próximo elemento de dados mais alto. Isto é, este modificador permite que o programador selecione

10 quais os bits do produto que devem estar no resultado sem a preocupação de saturações de capacidade. O programador pode gerar um produto completo de $2N$ bits com o uso de uma combinação de operações de multiplicação em pacote. Por exemplo, o programador pode usar uma operação de multiplicação em pacote superior sem sinal algébrico e, em seguida, usando

15 a mesma Fonte1 e Fonte2, uma operação de multiplicação em pacote inferior para obter os produtos completos ($2N$). A operação superior de multiplicação é provida porque, geralmente, os bits de ordem superior do produto são a única parte do produto. O programador pode obter os bits de ordem superior do produto sem primeiro ter que executar qualquer truncamento, conforme

20 é freqüentemente exigido por uma operação de dados não empacotados.

Em uma concretização da invenção, cada elemento de dados na Fonte2 pode ter um valor diferente. Isto dá ao programador a flexibilidade de ter um valor diferente como o multiplicador para cada multiplicando na Fonte

1.

25 A Figura 12 é um diagrama de fluxo que ilustra um processo de execução das operações de multiplicação em pacote nos dados em pacote, de acordo com uma concretização da invenção.

Na etapa 1201, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa maneira, o decodificador

30 202 decodifica o código de operação para a operação de multiplicação adequada, os endereços SRC1 602, SRC2 603 e DEST 605 nos registros 209, as multiplicações com sinal algébrico/sem sinal algébrico, superior/inferior, e

o comprimento dos elementos de dados nos dados em pacote.

Na etapa 1202, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 e SRC2 603. Os registros 209 apresentam a unidade de execução 130 com os dados em pacote armazenados no registro SRC1 602 (Fonte1), e os dados em pacote armazenados no registro SRC2 603 (Fonte2). Isto é, os registros 209 comunicam os dados em pacote à unidade de execução 130 através da barra interna 170.

Na etapa 1130, o decodificador 202 ativa a unidade de execução 130 para executar a operação adequada de multiplicação em pacote. O decodificador 202 adicionalmente comunica, através da barra interna 170, o tamanho dos elementos de dados e a superioridade/inferioridade com relação à operação de multiplicação.

Na etapa 1210, o tamanho do elemento de dados determina qual a etapa que deve ser executada a seguir. Se o tamanho dos elementos de dados for de oito bits (dados de byte), então a unidade de execução 130 executa a etapa 1212. Entretanto, se o tamanho dos elementos de dado nos dados em pacote for de dezesseis bits (dados de palavra), então a unidade de execução 130 executa a etapa 1214. Em uma concretização, apenas são suportadas as multiplicações em pacote com tamanho de elemento de dados de dezesseis bits. Em outra concretização, são suportadas as multiplicações em pacote com tamanho de elemento de dados de oito bits e de dezesseis bits. Entretanto, em outra concretização, é também suportada a multiplicação em pacote com tamanho de elemento de dados de trinta e dois bits.

Assumindo-se que o tamanho dos elementos de dados é de oito bits, então é executada a etapa 1212. Na etapa 1212, é executado o seguinte. Os bits de sete a zero da Fonte1 são multiplicados pelos bits de sete a zero da Fonte2, gerando os bits de sete a zero do Resultado. Os bits de quinze a oito da Fonte1 são multiplicados pelos bits de quinze a oito da Fonte 2, gerando os bits de quinze a oito do Resultado. Os bits de vinte e três a dezesseis da Fonte 1 são multiplicados pelos bits de vinte e três a de-

zesseis da Fonte 2, gerando os bits de vinte e três a dezesseis do Resultado. Os bits de trinta e um até vinte e quatro da Fonte 1 são multiplicados pelos bits de trinta e um até vinte e quatro da Fonte2, gerando os bits de trinta e um até vinte e quatro do Resultado. Os bits de trinta e nove até trinta e dois da Fonte1 são multiplicados pelos bits de trinta e nove até trinta e dois da Fonte2, gerando os bits de trinta e nove até trinta e dois do Resultado. Os bits de quarenta e sete até quarenta da Fonte1 são multiplicados pelos bits de quarenta e sete até quarenta da Fonte2, gerando os bits de quarenta e sete até quarenta do Resultado. Os bits de cinqüenta e cinco até quarenta e oito da Fonte1 são multiplicados pelos bits de cinqüenta e cinco até quarenta e oito da Fonte2, gerando os bits de cinqüenta e cinco até quarenta e oito do Resultado. Os bits de sessenta e três até cinqüenta e seis da Fonte1 são multiplicados pelos bits de sessenta e três até cinqüenta e seis da Fonte 2.

Assumindo-se que o tamanho dos elementos de dados é de dezesseis bits, então é executada a etapa 1214. Na etapa 1214, é executado o seguinte. Os bits de quinze a zero da Fonte 1 são multiplicados pelos bits de quinze até zero da Fonte2, gerando os bits de quinze a zero do Resultado. Os bits de trinta e um até dezesseis da Fonte1 são multiplicados pelos bits de trinta e um até dezesseis da Fonte2, gerando os bits de trinta e um até dezesseis do Resultado. Os bits de quarenta e sete até trinta e dois da Fonte1 são multiplicados pelos bits de quarenta e sete até trinta e dois da Fonte2, gerando os bits de quarenta e sete até trinta e dois do Resultado. Os bits de sessenta e três até quarenta e oito da Fonte1 são multiplicados pelos bits de sessenta e três até quarenta e oito da Fonte 2, gerando os bits de sessenta e três até quarenta e oito do Resultado.

Em uma concretização, as multiplicações da etapa 1212 são executadas simultaneamente. Entretanto, em outra concretização, estas multiplicações são executadas em série. Em outra concretização, algumas destas multiplicações são executadas simultaneamente e algumas são executadas em série. Esta discussão também se aplica igualmente à multiplicações da etapa 1214.

Na etapa 1220, o Resultado é armazenado no registro DEST.

A Tabela 14 ilustra a representação em registro da operação de multiplicação superior em pacote sem sinal algébrico nos dados de palavra em pacote. A primeira fileira de bits é a representação de dados em pacote da Fonte1. A segunda fileira de bits é a representação de dados da Fonte2.

- 5 A terceira fileira de bits é a representação de dados em pacote do Resultado. O número abaixo de cada elemento de dados é o número do elemento de dados. Por exemplo, o elemento de dados dois da Fonte1 é 11111111 00000000₂.

11111111 11111111	11111111 00000000	11111111 00000000	00001110 00001000
Multiplicação ³	Multiplicação ²	Multiplicação ¹	Multiplicação ⁰
00000000 00000000	00000000 00000001	10000000 00000000	00001110 10000001
=	=	=	=
00000000 00000000	00000000 00000000	01111111 10000000	00000000 11001011
3	2	1	0

Tabela 14

- 10 A Tabela 15 ilustra a representação em registro da operação de multiplicação superior em pacote com sinal algébrico nos dados de palavra em pacote.

11111111 11111111	11111111 00000000	11111111 00000000	00001110 00001000
Multiplicação ³	Multiplicação ²	Multiplicação ¹	Multiplicação ⁰
00000000 00000000	00000000 00000001	10000000 00000000	00001110 10000001
=	=	=	=
00000000 00000000	11111111 11111111	00000000 10000000	00000000 11001011
3	2	1	0

Tabela 15

- 15 A Tabela 16 ilustra a representação em registro da operação de multiplicação inferior em pacote nos dados de palavra em pacote.

11111111 11111111	11111111 00000000	11111111 00000000	00001110 00001000
Multiplicação ³	Multiplicação ²	Multiplicação ¹	Multiplicação ⁰
00000000 00000000	00000000 00000001	10000000 00000000	00001110 10000001
=	=	=	=
00000000 00000000	11111111 00000000	00000000 00000000	10000010 00001000
3	2	1	0

Tabela 16

Circuitos de Multiplicação de Dados em pacote

Em uma concretização, a operação de multiplicação pode ocorrer nos múltiplos elementos de dados no mesmo número de ciclos de relógio, como uma única operação de multiplicação nos dados desempacotados. Para se alcançar a execução do mesmo número de ciclos de relógio, é usado o paralelismo. Isto é, os registros são simultaneamente instruídos para executar a operação de multiplicação nos elementos de dados. Isto é discutido em maiores detalhes abaixo.

A Figura 13 ilustra um circuito para executar uma multiplicação em pacote, de acordo com uma concretização da invenção. O controle de operação 1300 controla os circuitos que executam a multiplicação. O controle de operação 1300 processa o sinal de controle para a operação de multiplicação, apresentando as seguintes saídas: ativação superior/inferior 1380; ativação de byte/palavra 1381 e ativação de sinal 1382. A ativação superior/inferior 1380 identifica se os bits de ordem superior ou inferior do produto devem ser incluídos no resultado. A ativação de byte/palavra 1381 identifica se uma operação de multiplicação de dados em pacote de byte e de dados em pacote de palavra deve ser executada. A ativação de sinal 1382 indica se a multiplicação com sinal algébrico deve ser usada.

O multiplicador de palavra em pacote 1301 multiplica quatro elementos de dados de palavra simultaneamente. O multiplicador de byte em pacote 1302 multiplica oito elementos de dados de byte. O multiplicador de palavra em pacote 1301 e o multiplicador de byte em pacote 1302 possuem ambos as seguintes entradas: Fonte1[63:0]1331, Fon-

te2[63:0]1333, ativação de sinal 1382, e ativação superior/inferior 1380.

O multiplicador de palavra em pacote 1301 inclui quatro circuitos multiplicadores de 16x16: multiplicador A 1310 de 16x16, multiplicador B 1311 de 16x16, multiplicador C 1312 de 16x16 e multiplicador D 1313 de 16x16. O multiplicador A 1310 de 16x16 possui como entradas a Fonte1[15:0] e a Fonte2[15:0]. O multiplicador A 1311 de 16x16 possui como entradas a Fonte1[31:16] e a Fonte2[31:16]. O multiplicador C 1312 de 16x16 possui como entradas a Fonte1[47:32] e a Fonte2[47:32]. O multiplicador D 1313 de 16x16 possui como entradas a Fonte1[63:48] e a Fonte2[63:48]. Cada multiplicador de 16x16 é acoplado à ativação de sinal 1382. Cada multiplicador de 16x16 produz um produto de trinta e dois bits. Para cada multiplicador, um multiplexor (Mx0 1350, Mx1 1251, Mx2 1352 e Mx3 1353, respectivamente) recebe o resultado de trinta e dois bits. Dependendo do valor da ativação superior/inferior 1380, cada multiplexor emite dezesseis bits de ordem superior ou dezesseis bits de ordem inferior do produto. As saídas dos quatro multiplexores são combinadas em um resultado de sessenta e quatro bits. Este resultado é opcionalmente armazenado em um registro de resultado 1 1371.

O multiplicador de byte em pacote 1302 inclui oito circuitos multiplicadores de 8x8: multiplicador A 1320 de 8x8 até multiplicador H 1327 de 8x8. Cada multiplicador de 8x8 possui uma entrada de oito bits a partir de cada uma da Fonte1[63:0]1331 e da Fonte2[63:0]1333. Por exemplo, o multiplicador A 1320 de 8x8 possui como entradas a Fonte1[7:0] e a Fonte2[7:0], enquanto o multiplicador H 1327 de 8x8 possui como entradas a Fonte1[63:56] e a Fonte2[63:56]. Cada multiplicador de 8x8 é acoplado à ativação de sinal 1382. Cada multiplicador de 8x8 produz um produto de dezesseis bits. Para cada multiplicador, um multiplexor (por exemplo, Mx4 1360 e Mx11 1367) recebe o resultado de dezesseis bits. Dependendo do valor da ativação superior/inferior 1380, cada multiplexor emite oito bits de ordem superior ou oito bits de ordem inferior do produto. As saídas dos oito multiplexores são combinadas em um resultado de sessenta e quatro bits. Este resultado é opcionalmente armazenado em um registro de resultado 2 1372.

A ativação de byte/palavras 1381 ativa o registro de resultado específico, dependendo do tamanho do elemento de dados que exige a operação.

Em uma concretização, a área usada para realizar as multiplicações é reduzida criando circuitos que possam multiplicar tanto dois números de 8x8 como um número de 16x16. Isto é, dois multiplicadores de 8x8 e um multiplicador de 16x16 são combinados em um multiplicador de 8x8 e 16x16. O controle de operação 1300 ativaria o tamanho adequado para a multiplicação. Em tal concretização, a área física usada pelos multiplicadores seria reduzida; contudo, seria difícil executar uma multiplicação de byte em pacote e uma multiplicação de palavra em pacote. Em outra concretização que suporta as multiplicações de palavra dupla em pacote, um multiplicador pode executar quatro multiplicações de 8x8, dois multiplicações de 16x16 ou uma de 32x32.

Em uma concretização, é apenas provida uma operação de multiplicação de palavra em pacote. Nesta concretização, o multiplicador de byte em pacote 1302 e o registro de resultado 1 1372 não seriam incluídos.

Vantagens de Incluir a Operação de Multiplicação em pacote Descrita no Conjunto de Instruções

Dessa maneira, a instrução de multiplicação em pacote descrita apresenta a multiplicação independente de cada elemento de dados na Fonte1 através de seu respectivo elemento de dados na Fonte2. Naturalmente, algoritmos que exijam que cada elemento na Fonte1 seja multiplicado pelo mesmo número podem ser executados através do armazenamento do mesmo número em cada elemento da Fonte2. Em adição, esta instrução de multiplicação assegura-se contra as saturações de capacidade através da interrupção das cadeias de transporte, isentando assim o programador desta responsabilidade, afastando a necessidade de instruções para preparar os dados com a finalidade de impedir as saturações de capacidade, e resultando em um código mais robusto.

Em contraste, os processadores com finalidades gerais da técnica anterior que não suportam tal instrução são exigidos para executarem esta operação através do desempacotamento dos elementos de dados,

através da execução das multiplicações, e, em seguida, através do empacotamento dos resultados para o processamento em pacote adicional. Dessa maneira, o processador 109 pode multiplicar diferentes elementos de dados dos dados em pacote através de diferentes multiplicadores em paralelo com o uso de uma instrução.

Os algoritmos típicos de múltiplos meios executam um grande número de operações de multiplicação. Dessa maneira, com a redução do número de instruções exigido para a execução destas operações de multiplicação, o desempenho destes algoritmos de múltiplos meios é aumentado. Assim, com a provisão desta instrução de multiplicação no conjunto de instruções suportado pelo processador 109, o processador 109 pode executar os algoritmos exigindo esta funcionalidade em um nível de desempenho mais alto.

MULTIPLICAÇÃO-SOMA/SUBTRAÇÃO

15 Operações de Multiplicação-Soma/Subtração

Em uma concretização, são executadas duas operações de multiplicação-soma com o uso de uma única instrução de multiplicação-soma, conforme mostrado abaixo na Tabela 17a e Tabela 17b. A Tabela 17a mostra uma representação simplificada da instrução descrita de multiplicação-soma, enquanto a Tabela 17b mostra um exemplo de nível de bit da instrução descrita de multiplicação-soma.

Multiplicação-soma da Fonte1, Fonte2

A ₁	A ₂	A ₃	A ₄	Fonte 1
B ₁	B ₂	B ₃	B ₄	Fonte 2
=				
A ₁ B ₁ +A ₂ B ₂		A ₃ B ₃ +A ₄ B ₄		Resultado 1

Tabela 17a



11111111 11111111	11111111 00000000	01110001 11000111	01110001 11000111
Multiplicação ³	Multiplicação ²	Multiplicação ¹	Multiplicação ⁰
00000000 00000000	00000000 00000001	10000000 00000000	00000100 00000000
↓	↓	↓	↓
Resultado Intermediário 4 de 32 bits	Resultado Intermediário 3 de 32 bits	Resultado Intermediário 2 de 32 bits	Resultado Intermediário 1 de 32 bits
 Soma		 Soma	
11111111 11111111 11111111 00000000		11001000 11100011 10011100 00000000	
1		0	

Tabela 17b

A operação de multiplicação-subtração é a mesma que a operação de multiplicação-soma, exceto pelo fato de que a soma é substituída por uma subtração. A operação de uma instrução de multiplicação-subtração exemplificativa que executa duas operações de multiplicação-subtração é mostrada abaixo na Tabela 12.

Multiplicação-subtração da Fonte1, Fonte2

A ₁	A ₂	A ₃	A ₄	Fonte 1
=				
B ₁	B ₂	B ₃	B ₄	Fonte 2
=		=		
A ₁ B ₁ -A ₂ B ₂		A ₃ B ₃ -A ₄ B ₄		Resultado 1

Tabela 12

Em uma concretização da invenção, o registro SRC1 contém dados em pacote (Fonte1), o registro SRC2 contém dados em pacote (Fonte2), e o registro DEST irá conter o resultado (Resultado) da execução da instrução de multiplicação-soma ou de multiplicação-subtração na Fonte1 ou Fonte2. Na primeira etapa da instrução de multiplicação-soma ou multiplicação-subtração, a Fonte2 irá apresentar cada elemento de dados indepen-

dentemente multiplicado pelo respectivo elemento de dados da Fonte2, de modo a gerar um conjunto de respectivos resultados intermediários. Quando da execução da instrução de multiplicação-soma, estes resultados intermediários são somados por pares, produzindo dois elementos de dados resultantes que são armazenados como elementos de dados do Resultado. Em contraste, quando da execução da instrução de multiplicação-subtração, estes resultados intermediários são subtraídos por pares, produzindo dois elementos de dados resultantes que são armazenados como elementos de dados do Resultado.

Concretizações alternativas podem variar o número de bits nos elementos de dados, nos resultados intermediários, e/ou nos elementos de dados no Resultado. Em adição, a concretização alternativa pode variar o número de elementos de dados na Fonte1, na Fonte2, e no Resultado. Por exemplo, se a Fonte1 e a Fonte2 possuírem, cada qual, 8 elementos de dados, as instruções de multiplicação-soma/subtração podem ser implementadas para produzirem um Resultado com 4 elementos de dados (cada elemento de dados no Resultado representando a adição de dois resultados intermediários), 2 elementos de dados (cada elemento de dados no resultado representando a adição dos quatro resultados intermediários), etc.

A Figura 14 é um diagrama de fluxo que ilustra um processo para executar as operações de multiplicação-soma e multiplicação-subtração nos dados em pacote, de acordo com uma concretização da invenção.

Na etapa 1401, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa maneira, o decodificador 202 decodifica o código de operação para uma instrução de multiplicação-soma ou multiplicação-subtração.

Na etapa 1402, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 e SRC2 603. Os registros 209 apresentam a unidade de execução 130 com os dados em pacote armazenados no registro SRC1 602 (Fonte1), e os dados em pacote armazenados no registro SRC2 603 (Fonte2). Isto é, os registros 209 comunicam os dados em pacote

à unidade de execução 130 através da barra interna 170.

Na etapa 1403, o decodificador 202 ativa a unidade de execução 130 para executar a instrução. Se a instrução for uma instrução de multiplicação-soma, o fluxo passa para a etapa 1414. Entretanto, se a instrução for
5 uma instrução de multiplicação-subtração, o fluxo passa para a etapa 1415.

Na etapa 1414, é executado o seguinte. Os bits de quinze até zero da Fonte1 são multiplicados pelos bits de quinze até zero da Fonte2, gerando um primeiro resultado intermediário de 32 bits (Resultado Intermediário 1). Os bits de trinta e um até dezesseis da Fonte1 são multiplicados
10 pelos bits de trinta e um até dezesseis da Fonte2, gerando um segundo resultado intermediário de 32 bits (Resultado Intermediário 2). Os bits de quarenta e sete até trinta e dois da Fonte1 são multiplicados pelos bits de quarenta e sete até trinta e dois da Fonte2, gerando um resultado intermediário de 32 bits (Resultado Intermediário 3). Os bits de sessenta e três até qua-
15 renta e oito da Fonte1 são multiplicados pelos bits de sessenta e três até quarenta e oito da Fonte2, gerando um quarto resultado intermediário de 32 bits (Resultado Intermediário 4). O Resultado Intermediário 1 é somado ao Resultado Intermediário 2, gerando os bits de trinta e um até 0 do Resultado, e o Resultado Intermediário 3 é somado ao Resultado Intermediário 4, ge-
20 rando os bits de sessenta e três até trinta e dois do Resultado.

A etapa 1415 é a mesma que a etapa 1414, com a exceção de que o Resultado Intermediário 1 e o Resultado Intermediário 2 são subtraídos para gerarem os bits de trinta e um a 0 do Resultado, o Resultado Intermediário 3 e o Resultado Intermediário 4 sendo subtraídos para gerarem
25 os bits de sessenta e três até trinta e dois do Resultado.

Diferentes concretizações podem executar as multiplicações e somas/subtrações em série, em paralelo, ou em alguma combinação das operações em série e em paralelo.

Na etapa 1420, o Resultado é armazenado no registro DEST.

30 Circuitos de Multiplicação-Soma/Subtração de Dados em Pacote

Em uma concretização, cada uma das instruções de multiplicação-soma e multiplicação-subtração pode ocorrer nos múltiplos elementos

de dados no mesmo número de ciclos de relógio como uma única multiplicação nos dados desempacotados. Para se alcançar a execução no mesmo número de ciclos de relógio, é usado o paralelismo. Isto é, os registros são simultaneamente instruídos para executarem as operações de multiplicação-soma ou multiplicação-subtração nos elementos de dados. Isto é discutido em maiores detalhes abaixo.

A Figura 15 ilustra um circuito para executar as operações de multiplicação-soma e/ou multiplicação-subtração nos dados em pacote, de acordo com uma concretização da invenção. O controle de operação 1500 processa o sinal de controle para as instruções de multiplicação-soma e multiplicação-subtração. O controle de operação 1500 emite sinais sobre a Ativação 1580 para controlar a Multiplicação em pacote dos Somador/Subtrator 1501.

A Multiplicação em pacote dos Somador/Subtrator 1501 possui as seguintes entradas: Fonte1[63:0]1531, Fonte2[63:0]1533, e Ativação 1580. A Multiplicação em pacote dos Somador/Subtrator 1501 inclui quatro circuitos multiplicadores de 16x16: multiplicador A 1510 de 16x16, o multiplicador B 1511 de 16x16, o multiplicador C 1512 de 16x16, e o multiplicador D 1513 de 16x16. O multiplicador A 1510 de 16x16 possui como entradas a Fonte1[15:0] e a Fonte2[15:0]. O multiplicador B 1511 de 16x16 possui como entradas a Fonte1[31:16] e a Fonte2[31:16]. O multiplicador C 1512 de 16x16 possui como entradas a Fonte1[47:32] e a Fonte2[47:32]. O multiplicador D 1513 de 16x16 possui como entradas a Fonte1[63:48] e a Fonte2[63:48]. Os resultados intermediários de 32 bits gerados pelo multiplicador A 1510 de 16x16 e pelo multiplicador B 1511 de 16x16 são recebidos pelos Somador/Subtrator Virtuais 1550, enquanto os resultados intermediários de 32 bits gerados pelo multiplicador C 1512 de 16x16 e pelo multiplicador D 1513 de 16x16 são recebidos pelos Somador/Subtrator Virtuais 1551.

Com base na eventualidade da instrução comum ser uma instrução de multiplicação-soma ou multiplicação-subtração, os Somador/ Subtrator Virtuais 1550 e os Somador/Subtrator Virtuais 1551 somam ou subtraem suas respectivas entradas de 32 bits. A saída dos Somador/Subtrator

Virtuais 1550 (isto é, os bits de trinta e um até zero do Resultado) e a saída dos Somador/Subtrator Virtuais 1551 (isto é, os bits de 63 a trinta e dois do Resultado) são combinadas no Resultado de 64 bits e comunicadas ao Registro de Resultado 1571.

5 Em uma concretização, os Somador/Subtrator Virtuais 1551 e os Somador/Subtrator Virtuais 1550 são implementados de forma similar como os Somador/Subtrator Virtuais 1108b e os Somador/Subtrator Virtuais 1108a (isto é, cada um dos Somador/Subtrator Virtuais 1551 e Somador/Subtrator Virtuais 1550 sendo composto de quatro somadores de 8 bits com retardos
10 de propagação apropriados). Entretanto, concretizações alternativas poderiam implementar os Somador/Subtrator Virtuais 1551 e os Somador/Subtrator Virtuais 1550 em diversas maneiras.

 Para executar o equivalente destas instruções de multiplicação-soma ou multiplicação-subtração nos processadores da técnica anterior que
15 operam nos dados desempacotados, seriam necessárias quatro operações de multiplicação separadas de 64 bits e duas operação de soma ou subtração de 64 bits, bem como as operações necessárias de carregamento e armazenamento. Isto faz perder circuitos e linhas de dados que são usados para os bits que são maiores do que o bit dezesseis para a Fonte1 e Fonte2,
20 e maiores do que o bit trinta e dois para o Resultado. Igualmente, o resultado completo de 64 bits gerado por tais processadores da técnica anterior pode não ter uso para o programador. Por isso, o programador teria que truncar cada resultado.

25 Vantagens de Incluir a Operação Descrita de Multiplicação-Soma no Conjunto de Instruções

 As instruções descritas de multiplicação-soma/subtração podem ser usadas para diversas finalidades. Por exemplo, a instrução de multiplicação-soma pode ser usada para a multiplicação de números complexos e para a multiplicação e acúmulo de valores. Diversos algoritmos que utilizam
30 a instrução de multiplicação-soma serão posteriormente descritos aqui.

 Dessa maneira, com a inclusão das instruções de multiplicação-soma e/ou de multiplicação-subtração descritas no conjunto de instruções

suportado pelo processador 109, podem ser executadas muitas funções em um menor número de instruções do que os processadores de finalidades gerais da técnica anterior, que carecem destas instruções.

DESLOCAMENTO EM PACOTE

5 Operação de Deslocamento Em pacote

Em uma concretização da invenção, o registro SRC1 contém os dados (Fonte1) a serem deslocados, o registro SRC2 contém os dados (Fonte2) que representam a contagem de deslocamento, e o registro DEST irá conter o resultado do deslocamento (Resultado). Isto é, a Fonte1 irá
10 apresentar cada elemento de dados independentemente deslocado pela contagem de deslocamento. Em uma concretização, a Fonte2 é interpretada como um escalar de 64 bits sem sinal algébrico. Em outra concretização, a Fonte2 são os dados em pacote e ela contém as contagens de deslocamento para cada elemento de dados correspondente na Fonte1.

15 Em uma concretização da invenção, são suportados tanto os deslocamentos aritméticos como os deslocamentos lógicos. Um deslocamento aritmético desloca descendentemente os bits de cada elemento de dados através de um número específico, e preenche o bit de ordem superior de cada elemento de dados com o valor inicial do bit de sinal. Uma conta-
20 gem de deslocamento maior do que sete para os dados de byte em pacote, maior do que quinze para os dados de palavra em pacote, ou maior do que trinta e um para os dados de palavra dupla em pacote, faz com que cada elemento de dados do Resultado seja preenchido com o valor inicial do bit de sinal. Um deslocamento lógico pode operar através do deslocamento de
25 bits para cima e para baixo. Em um deslocamento lógico para a direita, os bits de ordem superior de cada elemento de dados são preenchidos com zeros. Um deslocamento lógico para a esquerda faz com que os bits menos significativos de cada elemento de dados sejam preenchidos com zeros.

Em uma concretização da invenção, as operações de desloca-
30 mento aritmético para a direita, de deslocamento lógico para a direita e de deslocamento lógico para a esquerda são suportadas para os bytes em pacotes e para as palavras em pacote. Em outra concretização da invenção,

estas operações são sustentadas para as palavras duplas em pacote também.

A Figura 16 é um diagrama de fluxo que ilustra um processo para executar uma operação de deslocamento em pacote nos dados em pacote, de acordo com uma concretização da invenção.

Na etapa 1601, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa forma, o decodificador 202 decodifica o código de operação para a operação de deslocamento adequada, os endereços SRC1 602, SRC2 603 e DEST 605 nos registros 209, a saturação/insaturação (não necessariamente obrigatórias para as operações de deslocamento), a aritmética com sinal algébrico/sem sinal algébrico (novamente não necessariamente obrigatórias), e o comprimento dos elementos de dados nos dados em pacote.

Na etapa 1602, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 e SRC2 603. Os registros 209 apresentam a unidade de execução 130 com os dados em pacote armazenados no registro SRC1 602 (Fonte1), e a contagem de deslocamento escalar no registro SRC2 603 (Fonte2). Isto é, os registros 209 comunicam os dados em pacote à unidade de execução 130 através da barra interna 170.

Na etapa 1603, o decodificador 202 ativa a unidade de execução 130 para executar a operação apropriada de deslocamento em pacote. O decodificador 202 adicionalmente comunica, através da barra interna 170, o tamanho dos elementos de dados, o tipo de operação de deslocamento, e a direção do deslocamento (para deslocamentos lógicos).

Na etapa 1610, o tamanho do elemento de dados determina qual a etapa que deve ser executada a seguir. Se o tamanho dos elementos de dados for de oito bits (dados de byte), então a unidade de execução 130 executa a etapa 1612. Entretanto, se o tamanho dos elementos de dados nos dados em pacote for de dezesseis bits (dados de palavras), então a unidade de execução 130 executa a etapa 1614. Em uma concretização, apenas são suportados deslocamentos em pacote com tamanho de elemento de

dados de oito bits e de dezesseis bits. Entretanto, em outra concretização, é também suportado um deslocamento em pacote com tamanho de elemento de dados de trinta e dois bits.

Assumindo-se que o tamanho dos elementos de dados é de oito bits, então é executada a etapa 1612. Na etapa 1612, é executado o seguinte. Os bits de sete até zero da Fonte1 são deslocados pela contagem de deslocamento (Bits de sessenta e três até zero da Fonte2), gerando os bits de sete até zero do Resultado. Os bits de quinze até oito da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de quinze até oito do Resultado. Os bits de vinte e três até dezesseis da Fonte 1 são deslocados pela contagem de deslocamento, gerando os bits de vinte e três até dezesseis do Resultado. Os bits de trinta e um até vinte e quatro da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de trinta e um até vinte e quatro do Resultado. Os bits de trinta e nove até trinta e um da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de trinta e nove até trinta e dois do Resultado. Os bits de quarenta e sete até quarenta da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de quarenta e sete até quarenta do Resultado. Os bits de cinquenta e cinco até quarenta e oito da Fonte 1 são deslocados pela contagem de deslocamento, gerando os bits de cinquenta e cinco até quarenta e oito do Resultado. Os bits de sessenta e três até cinquenta e seis da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de sessenta e três até cinquenta e seis do Resultado.

Assumindo-se que o tamanho dos elementos de dados é de dezesseis bits, então é executada a etapa 1614. Na etapa 1614, é executado o seguinte. Os bits de quinze até zero da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de quinze até zero do Resultado. Os bits de trinta e um até dezesseis da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de trinta e um até dezesseis do Resultado. Os bits de quarenta e sete até trinta e dois da Fonte1 são deslocados pela contagem de deslocamento, gerando os bits de quarenta e sete até trinta e dois do Resultado. Os bits de sessenta e três até quarenta e oito da Fonte1

são deslocados pela contagem de deslocamento, gerando os bits de sessenta e três até quarenta e oito do Resultado.

Em uma concretização, os deslocamentos da etapa 1612 são executados simultaneamente. Entretanto, em outra concretização, estes deslocamentos são executados em série. Em outra concretização, alguns destes deslocamentos são executados simultaneamente e alguns são executados em série. Esta discussão aplica-se igualmente aos deslocamentos da etapa 1614.

Na etapa 1620, o Resultado é armazenado no registro DEST.

A Tabela 19 ilustra a representação em registro da operação em pacote de byte de deslocamento aritmético para a direita. A primeira fileira de bits é a representação de dados em pacote da Fonte1. A segunda fileira de bits é a representação de dados da Fonte2. A terceira fileira de bits é a representação de dados em pacote do Resultado. O número abaixo de cada bit de elemento de dados é o número do elemento de dados. Por exemplo, o elemento três dos dados da Fonte1 é 10000000_2 .

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
Shift ⁷	Shift ⁶	Shift ⁵	Shift ⁴	Shift ³	Shift ²	Shift ¹	Shift ⁰
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000100
=	=	=	=	=	=	=	=
00000010	00000101	00000101	11111111	11110000	00000111	11111000	11111000
7	6	5	4	3	2	1	0

onde Shift = Deslocamento

Tabela 19

A Tabela 20 ilustra a representação em registro da operação em pacote de deslocamento lógico para a direita nos dados de byte em pacote.

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
Shift ⁷	Shift ⁶	Shift ⁵	Shift ⁴	Shift ³	Shift ²	Shift ¹	Shift ⁰
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000011							
= = = = = = = =							
00000101	00001010	00001010	00011111	00010000	00001110	00010001	00010001
7	6	5	4	3	2	1	0

onde Shift = Deslocamento

Tabela 20

A Tabela 21 ilustra a representação em registro da operação em pacote de deslocamento lógico para a esquerda nos dados de byte em pacote.

5

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
Shift ⁷	Shift ⁶	Shift ⁵	Shift ⁴	Shift ³	Shift ²	Shift ¹	Shift ⁰
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000011							
= = = = = = = =							
01010000	10101000	10101000	11111000	00000000	10000000	01111000	01000000
7	6	5	4	3	2	1	0

onde Shift = Deslocamento

Tabela 21

Circuitos de Deslocamento de Dados em pacote

Em uma concretização, a operação de deslocamento pode ocorrer nos múltiplos elementos de dados no mesmo número de ciclos de relógio como uma única operação de deslocamento nos dados desempacotados. Para se alcançar a execução no mesmo número de ciclos de relógio, é usado o paralelismo. Isto é, os registros são simultaneamente instruídos para executarem a operação de deslocamento nos elementos de dados. Isto é discutido em maiores detalhes abaixo.

10

15

A Figura 17 ilustra um circuito para a execução de um deslocamento em pacote nos bytes individuais dos dados em pacote, de acordo com uma concretização da invenção. A Figura 17 ilustra o uso de um circuito modificado de deslocamento de fração de byte, o estágio_i de fração de byte

1799. Cada fração de byte, exceto para a fração de byte do elemento de dados mais significativo, inclui um controle de bit e unidade de deslocamento. A fração de byte de elemento de dados mais significativo precisa apenas de uma unidade de deslocamento.

5 A unidade_i de deslocamento 1711 e a unidade_{i+1} de deslocamento 1771 permitem, cada qual, que oito bits da Fonte1 sejam deslocados pela contagem de deslocamento. Em uma concretização, cada unidade de deslocamento opera como um circuito de deslocamento de oito bits conhecido. Cada unidade de deslocamento possui uma entrada de Fonte1, uma entrada de Fonte2, uma entrada de controle, um próximo estágio de sinal, um
10 último estágio de sinal, e uma saída de resultado. Por isso, a unidade_i de deslocamento 1711 possui a entrada da Fonte1_i 1731, a entrada da Fonte2[63:0]1733, a entrada de controle_i 1701, o próximo estágio de sinal_i 1713, a última entrada de estágio_i 1712, e um resultado armazenado no registro de
15 resultado_i 1751. Por isso, a unidade_{i+1} de deslocamento 1771 possui a entrada de Fonte1_{i+1} 1732, a entrada de Fonte2[63:0] 1733, a entrada de controle_{i+1} 1702, o próximo estágio de sinal_{i+1} 1773, a última entrada de estágio_{i+1} 1772, e um resultado armazenado no registro_{i+1} 1752 de resultado.

 A entrada de Fonte1 é tipicamente uma porção de oito bits da
20 Fonte1. Os oito bits representam o menor tipo de elemento de dados, um elemento de dados de byte em pacote. A entrada de Fonte2 representa a contagem de deslocamento. Em uma concretização, cada unidade de deslocamento recebe a mesma contagem de deslocamento da Fonte2[63:0] 1733. O controle de operação 1700 transmite sinais de controle para ativar cada
25 unidade de deslocamento para a execução do deslocamento exigido. Os sinais de controle são determinados a partir do tipo de deslocamento (aritmético/lógico) e a direção do deslocamento. O próximo estágio de sinal é recebido a partir do controle de bit para essa unidade de deslocamento. A unidade de deslocamento irá deslocar a saída/entrada do bit mais significati-
30 vo no próximo estágio de sinal, dependendo da direção do deslocamento (esquerda/direita). Similarmente, cada unidade de deslocamento irá deslocar a saída/entrada do bit menos significativo no último estágio de sinal, depen-

dendo da direção do deslocamento (direita/esquerda). O último estágio de sinal é recebido a partir da unidade de controle de bit do estágio anterior. A saída do resultado representa o resultado da operação de deslocamento na porção da Fonte1 que a unidade de deslocamento está operando.

5 O controle_i de bit 1720 é ativado a partir do controle de operação 1700 através da ativação_i1706 dos dados em pacote. O controle_i de bit 1720 controla o próximo estágio_i1713 e o último estágio_{i+1} 1772. Assume-se, por exemplo, que a unidade_i de deslocamento 1711 seja responsável pelos oito bits menos significativos da Fonte1, e a unidade_{i+1} de deslocamento 1771
10 seja responsável pelos próximos oito bits da Fonte1. Se um deslocamento for executado nos bytes em pacote , o controle_i de bit 1720 não irá permitir que o bit menos significativo originário da unidade_{i+1} de deslocamento 1771 tenha comunicação com o bit mais significativo da unidade_i de deslocamento 1711. Entretanto, é executado um deslocamento nas palavras em pacote;
15 então o controle_i de bit 1720 irá permitir que o bit menos significativo originário da unidade_{i+1} de deslocamento 1771 tenha comunicação com o bit mais significativo da unidade_i de deslocamento 1711.

 Por exemplo, na Tabela 22, é executado um deslocamento aritmético direito de byte em pacote. Assume-se que a unidade_{i+1} de deslocamento 1771 opere no elemento um de dados, e a unidade_i de deslocamento
20 1711 opere no elemento zero de dados. A unidade_{i+1} de deslocamento 1771 desloca seu bit menos significativo para fora. Entretanto, o controle de operação 1700 irá fazer com que o controle_i de bit 1720 detenha a propagação desse bit, recebido a partir do último estágio_{i+1} 1721, para o último estágio_i
25 1713. Ao invés disso, a unidade_i de deslocamento 1711 irá encher os bits de ordem superior com o bit de sinal, Fonte1[7].

...	00001110	10001000
Shift 7	Shift 6	Shift 5	Shift 4	Shift 3	Shift 2	Shift 1	Shift 0
...							00000001
=	=	=	=	=	=	=	=
...	00001111	01000100
7	6	5	4	3	2	1	0

onde Shift = Deslocamento

Tabela 22

Entretanto, se for executado um deslocamento aritmético de palavra em pacote, então o bit menos significativo da unidade_{i+1} de deslocamento 1771 será comunicado ao bit mais significativo da unidade_i de deslocamento 1711. A Tabela 23 ilustra este resultado. Esta comunicação seria igualmente permitida para os deslocamentos de palavra dupla em pacote.

...	00001110 10001000
Shift 3	Shift 2	Shift 1	Shift 0
...			00000001
=	=	=	=
...	00000111 01000100
3	2	1	0

onde Shift = Deslocamento

Tabela 23

10 Cada unidade de deslocamento é opcionalmente acoplada a um registro de resultado. O registro de resultado temporariamente armazena o resultado da operação de deslocamento até que o resultado completo, Resultado[63:0] 1760 possa ser transmitido ao registro DEST.

15 Para um circuito completo de deslocamento em pacote de sessenta e quatro bits, são usadas oito unidades de deslocamento e sete unidades de controle de bit. Tal circuito pode também ser usado para executar um deslocamento em dados desempacotados de sessenta e quatro bits, usando assim o mesmo circuito para executar a operação de deslocamento

desempacotado e a operação de deslocamento em pacote.

Vantagens de Incluir a Operação de Descolamento Descrita no Conjunto de Instruções

5 A instrução descrita de deslocamento em pacote faz com que cada elemento da Fonte1 seja deslocado pela contagem de deslocamento indicada. Com a inclusão desta instrução no conjunto de instruções, cada elemento de dados em pacote pode ser deslocado com o uso de uma única instrução. Em contraste, os processadores de finalidades gerais da técnica anterior, que não suportam tal operação, têm que executar numerosas ins-
10 truções para a Fonte1 desempacotada, individualmente deslocar cada elemento de dados desempacotados, e depois empacotar os resultados em um formato de dados em pacote para o processamento adicional em pacote.

OPERAÇÃO DE MOVIMENTO

15 A operação de movimento transfere os dados para e a partir dos registros 209. Em uma concretização, SRC2 603 é o endereço que contém os dados de fonte e DEST 605 é o endereço para onde devem ser transferidos os dados. Nesta concretização SRC1 602 não seria usado. Em outra concretização, SRC1 602 é igual a DEST 605.

20 Para fins de explicação da operação de movimento, é feita uma distinção entre um registro e uma localização de memória. Registros são encontrados no arquivo de registro geral 150, enquanto a memória pode estar, por exemplo, no cache 160, na memória principal 104, na ROM (Memória de Leitura Apenas) 106, no dispositivo de armazenamento de dados 107.

25 A operação de movimento pode mover os dados da memória para os registros 209, dos registros 209 para a memória, e de um registro nos registros 209 para um segundo registro nos registros 209. Em uma concretização, os dados em pacote são armazenados em diferentes registros do que aqueles usados para armazenar dados inteiros. Nesta concretização, a
30 operação de movimento pode mover os dados dos registros inteiros 201 para os registros 209. Por exemplo, no processador 109, se os dados em pacote estiverem armazenados nos registros 209 e os dados inteiros estive-

rem armazenados nos registros interiores 201, então uma instrução de movimento pode ser usada para mover os dados dos registros inteiros 201 para os registros 209, e vice-versa.

Em uma concretização, quando um endereço de memória for indicado para o movimento, os oito bytes de dados na localização da memória (a localização da memória contendo o byte menos significativo) são carregados para um registro nos registros 209 ou armazenados a partir desse registro. Quando um registro nos registros 209 for indicado, os conteúdos desse registro são movidos ou carregados a partir de um segundo registro nos registros 209. Se os registros inteiros 201 tiverem sessenta e quatro bits de comprimento, e um registro inteiro for especificado, então os oito bytes de dados no registro inteiro são carregados para um registro nos registros 209 ou armazenados a partir desse registro.

Em uma concretização, os inteiros são representados como trinta e dois bits. Quando uma operação de movimento é executada a partir dos registros 209 para os registros inteiros 201, então apenas os trinta e dois bits baixos dos dados em pacote são movidos para o registro inteiro específico. Em uma concretização, os trinta e dois bits de ordem superior são zerados. Similarmente, apenas os trinta e dois bits baixos de um registro nos registros 209 são carregados, quando um movimento é executado a partir dos registros inteiros 201 para os registros 209. Em uma concretização, o processador 109 suporta uma operação de movimento de trinta e dois bits entre um registro nos registros 209 e na memória. Em outra concretização, um movimento de apenas trinta e dois bits é executado em trinta e dois bits de ordem superior dos dados em pacote.

OPERAÇÃO DE EMPACOTAMENTO

Em uma concretização da invenção, o registro SRC1 602 contém os dados (Fonte1), o registro SRC2 603 contém os dados (Fonte2), e o registro DEST 605 irá conter os dados do resultado (Resultado) da operação. Isto é, as partes da Fonte1 e as partes da Fonte2 serão empacotadas juntas para gerarem o Resultado.

Em uma concretização, uma operação de empacotamento con-

verte as palavras (ou palavras duplas) em pacote em bytes (ou palavras) em pacote através do empacotamento dos bytes (ou palavras) de ordem inferior das palavras (ou palavras duplas) em pacote da fonte nos bytes (ou palavras) do Resultado. Em uma concretização, a operação de empacotamento
5 converte as palavras em pacote quádruplas em palavras duplas em pacote. Esta operação pode ser opcionalmente executada com os dados contendo sinal algébrico. Adicionalmente, esta operação pode ser opcionalmente executada com a saturação. Em uma concretização alternativa são incluídas as operações de empacotamento adicionais, as quais operam nas porções de
10 ordem superior de cada elemento de dado.

A Figura 18 é um diagrama de fluxo que ilustra um processo para a execução de operações em pacote nos dados em pacote, de acordo com uma concretização da invenção.

Na etapa 1801, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa forma, o decodificador 202
15 decodifica o código de operação para a operação de empacotamento apropriada, os endereços SRC1 602, SRC2 603 e DEST 605 nos registros 209, a saturação/ insaturação, a operação com sinal algébrico/sem sinal algébrico, e o comprimento dos elementos de dados nos dados em pacote. Conforme
20 anteriormente mencionado. SRC1 602 (ou SRC2 603) pode ser usado com DEST 605.

Na etapa 1802, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 SRC2 603. Os registros 209 apresentam a
25 unidade de execução 130 com os dados em pacote armazenados no registro SRC1 602 (Fonte1), e os dados em pacote armazenados no registro SRC2 603 (Fonte2). Isto é, os registros 209 comunicam os dados em pacote à unidade de execução 130 através da barra interna 170.

Na etapa 1803, o decodificador 202 ativa a unidade de execução
30 130 para executar a operação de empacotamento apropriada. O decodificador 202 adicionalmente comunica, através da barra interna 170, a saturação e o tamanho dos elementos de dados na Fonte1 e na Fonte2. A saturação é

opcionalmente usada para maximizar o valor dos dados no elemento de dados do resultado. Se o valor dos elementos de dados na Fonte1 ou Fonte2 for maior ou menor do que a faixa de valores que os elementos de dados do Resultado podem representar, então o elemento de dados do resultado correspondente é ajustado no valor mais alto ou mais baixo. Por exemplo, se os valores com sinal algébrico nos elementos de dados de palavra da Fonte1 e da Fonte2 forem menos do que 0x80 (ou 0x8000 para palavras duplas), então os elementos de dados do byte (palavra) de resultado são retidos em 0x80 (ou 0x8000 para palavras duplas). Se os valores com sinal algébrico nos elementos de dados de palavra da Fonte1 e da Fonte2 forem maiores do que 0x7F (ou 0x7FFF para palavras duplas), então os elementos de dados do byte (ou palavra) do resultado são retidos em 0x7F (ou 0x7FFF).

Na etapa 1810, o tamanho do elemento de dados determina qual a etapa que deve ser executada a seguir. Se o tamanho dos elementos de dados for de dezesseis bits (dados de palavra em pacote 402), então a unidade de execução 130 executa a etapa 1812. Entretanto, se o tamanho dos elementos de dados nos dados em pacote for de trinta e dois bits (dados de palavra dupla em pacote 403), a unidade de execução 130 irá executar a etapa 1814.

Assumindo-se que o tamanho dos elementos de dados da fonte é de dezesseis bits, então é executada a etapa 1812. Na etapa 1812, é executado o seguinte. Os bits de sete até zero da Fonte1 são os bits de sete até zero do Resultado. Os bits de vinte e três até dezesseis da Fonte, são os bits de quinze até oito do Resultado. Os bits de trinta e nove até trinta e dois da Fonte1 são os bits de vinte e três até dezesseis do Resultado. Os bits de sessenta e três até cinquenta e seis da Fonte1 são os bits de trinta e um até vinte e quatro do Resultado. Os bits de sete até zero da Fonte2 são dos bits de trinta e nove até trinta e dois do Resultado. Os bits de vinte e três até dezesseis da Fonte2 são os bits de quarenta e sete até quarenta do Resultado. Os bits de trinta e nove até trinta e dois da Fonte2 são os bits de cinquenta e cinco até quarenta e oito do Resultado. Os bits de sessenta e três até cinquenta e seis da Fonte2 são os bits de trinta e um até vinte e quatro do Re-

sultado. Se a saturação for estabelecida, então os bits de ordem superior de cada palavra são testados para determinarem se o elemento de dados do Resultado deve ser retido.

5 Assumindo-se que o tamanho dos elementos de dado da fonte é de trinta e dois bits, então é executada a etapa 1814. Na etapa 1814, é executado o seguinte: os bits de quinze até zero da Fonte1 são os bits de quinze até zero do Resultado. Os bits de quarenta e sete até trinta e dois da Fonte1 são os bits de trinta e um até dezesseis do Resultado. Os bits de quinze até zero da Fonte2 são os bits de quarenta e sete até trinta e dois do
10 Resultado. Os bits de quarenta e sete até trinta e dois da Fonte2 são os bits de sessenta e três até quarenta e oito do Resultado. Se a saturação for estabelecida, então os bits de ordem superior de cada palavra dupla são testados para determinarem se o elemento de dados do Resultado deve ser retido.

15 Em uma concretização, o empacotamento da etapa 1812 é executada simultaneamente. Entretanto, em outra concretização, esta empacotamento é executada em série. Em outra concretização, parte do empacotamento é executada simultaneamente e parte é executada em série. Esta discussão também aplica-se à empacotamento da etapa 1814.

20 Na etapa 1820, o Resultado é armazenado no registro DEST 605.

 A Tabela 24 ilustra a representação em registro de uma operação de palavra em pacote. Os H_S e L_S subscritos representam os bits de ordem superior e de ordem inferior, respectivamente, de cada elemento de
25 dados de 16 bits na Fonte1 e na Fonte2. Por exemplo, A_L representa os 8 bits de ordem inferior do elemento de dados A na Fonte 1.

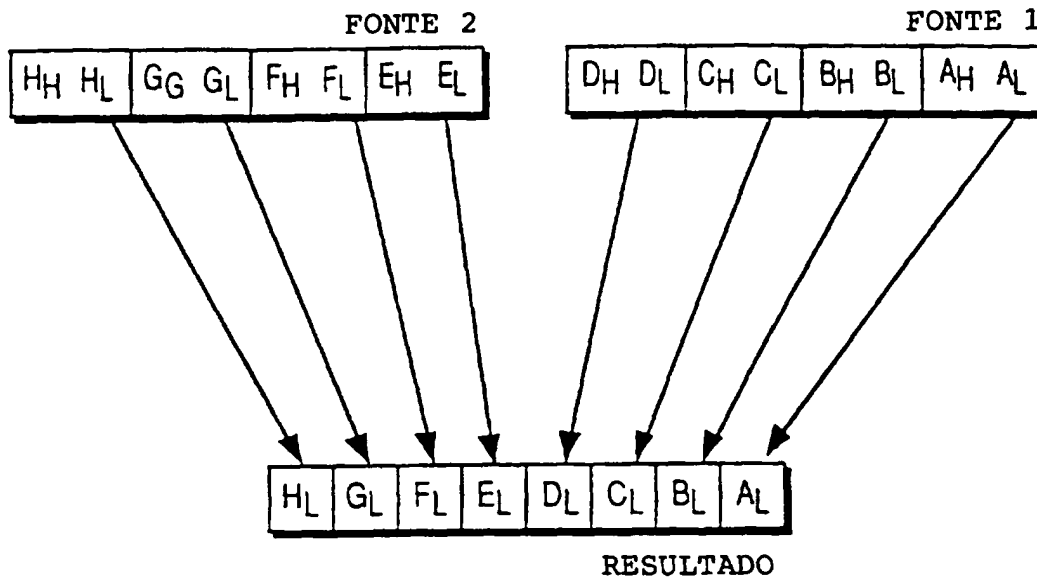


Tabela 24

A Tabela 25 ilustra a representação em registro de uma operação de palavra dupla em pacote, onde os H_S e L_S subscritos representam os bits de ordem superior e inferior, respectivamente, de cada elemento de dados de 32 bits na Fonte1 e Fonte2.

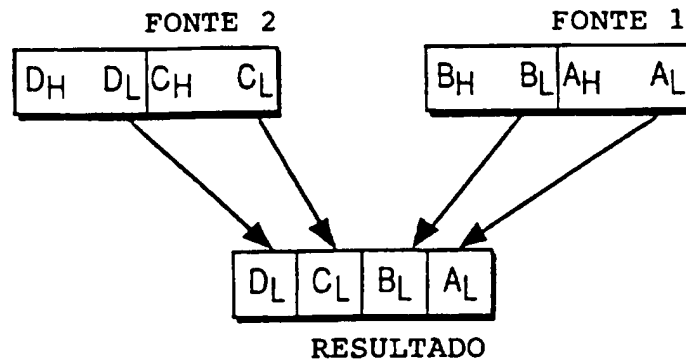


Tabela 25

Circuitos de Empacotamento

Em uma concretização da invenção, para se alcançar a execução eficiente das operações de empacotamento é usado o paralelismo. As Figuras 19a e 19b ilustram um circuito para executar as operações de empacotamento nos dados em pacote, de acordo com uma concretização da invenção. O circuito pode opcionalmente executar a operação de empacotamento com saturação.

O circuito das Figuras 19a e 19b inclui um controle de operação

1900, um registro de resultado 1952, um registro de resultado 1953, oito circuitos de saturação de teste de dezesseis bits até 8 bits, e quatro circuitos de saturação de teste de trinta e dois bits até dezesseis bits.

5 O controle de operação 1900 recebe a informação do decodificador 202 para possibilitar uma operação de empacotamento. O controle de operação 1900 usa o valor de saturação para ativar os testes de saturação para cada um dos circuitos de saturação de teste. Se o tamanho dos dados em pacote da fonte for dados em pacote de palavra 503, então a ativação de saída 1931 é estabelecida pelo controle de operação 1900. Isto possibilita a
10 saída do registro de resultado 1952. Se o tamanho dos dados em pacote de fonte for dados em pacote de palavra dupla 504, então a ativação da saída 1932 é estabelecida pelo controle de operação 1900. Isto possibilita a saída do registro de saída 1953.

15 Cada circuito de saturação de teste pode seletivamente testar a saturação. Se um teste para saturação for desativado, então cada circuito de saturação de teste meramente passa os bits de ordem inferior para uma posição correspondente em um registro de resultado. Se um teste para saturação for ativado, então cada circuito de saturação de teste testa os bits de ordem superior para determinar se o resultado deve ser retido.

20 A saturação de teste 1910 até a saturação de teste 1917 possuem dezesseis entradas de bit e oito saídas de bit. As oito saídas de bit são os oito bits inferiores das entradas, ou, opcionalmente, são um valor retido (0x80, 0x7F, ou 0xFF). A saturação de teste 1910 recebe os bits de quinze até zero da Fonte1 e emite os bits de sete até zero para o registro de resultado 1952. A saturação de teste 1911 recebe os bits de trinta e um até dezesseis da Fonte1 e emite os bits de quinze até oito para o registro de resultado 1952. A saturação de teste 1912 recebe os bits de quarenta e sete até trinta e dois da Fonte1 e emite os bits de vinte e três até dezesseis para o registro de resultado 1952. A saturação de teste 1913 recebe os bits de
25 sessenta e três até quarenta e oito da Fonte 1 e emite os bits de trinta e um até vinte e quatro para o registro de resultado 1952. A saturação de teste 1914 recebe os bits de quinze até zero da Fonte2 e emite os bits de trinta e

nove até trinta e dois para o registro de resultado 1952. A saturação de teste 1915 recebe os bits de trinta e um até dezesseis da Fonte2 e emite os bits de quarenta e sete até quarenta para o registro de resultado 1952. A saturação de teste 1916 recebe os bits de quarenta e sete até trinta e dois da

5 Fonte2 e emite os bits cinquenta e cinco até quarenta e oito para o registro de resultado 1952. A saturação de teste 1917 recebe os bits de sessenta e três até quarenta e oito da Fonte2 e emite os bits de sessenta e três até cinquenta e seis para o registro de resultado 1952.

A saturação de teste 1920 a saturação de teste 1923 possuem

10 trinta e duas entradas de bit e dezesseis saídas de bit. As dezesseis saídas de bit são os dezesseis bits inferiores das entradas, ou, opcionalmente, são um valor retiro (0x8000, 0x7FFF, ou 0xFFFF). A saturação de teste 1920 recebe os bits de trinta e um até zero da Fonte1 e emite os bits de quinze até zero para o registro de resultado 1953. A saturação de teste 1921 recebe

15 os bits de sessenta e três até trinta e dois da Fonte1 e emite os bits de trinta e um até dezesseis para o registro de resultado 1953. A saturação de teste 1922 recebe os bits de trinta e um até zero da Fonte2 e emite os bits de quarenta e sete até trinta e dois para o registro de resultado 1953. A saturação de teste 1923 recebe os bits de sessenta e três até trinta e dois da Fonte2 e

20 emite os bits de sessenta e três até quarenta e oito para o registro de resultado 1953.

Por exemplo, na Tabela 26, é executada uma palavra em pacote sem sinal algébrico sem qualquer saturação. O controle de operação 1900 irá ativar o registro de resultado 1952 a emitir o resultado[63:0] 1960.

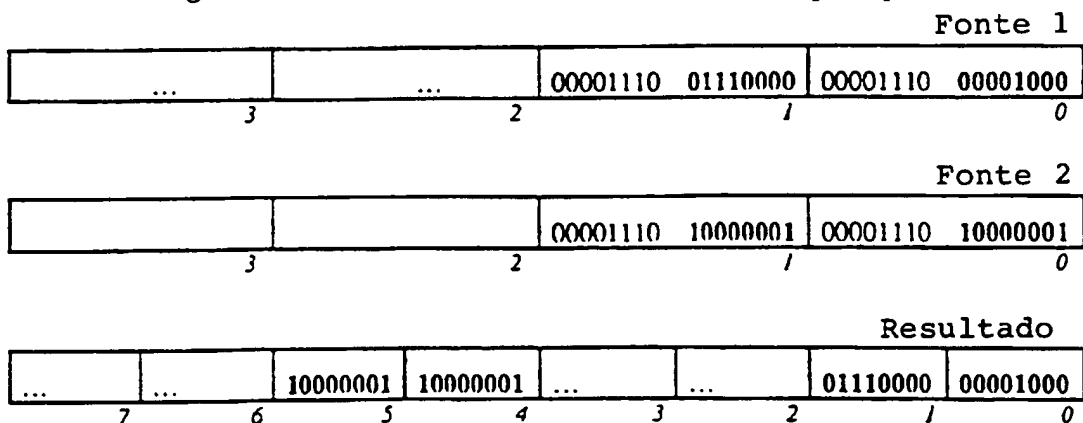


Tabela 26

Entretanto, se uma palavra dupla em pacote sem sinal algébrico e sem qualquer saturação for executada, o controle de operação 1900 irá ativar o registro de resultado 1953 para emitir o resultado [63:0] 1960. A Tabela 27 ilustra este resultado.

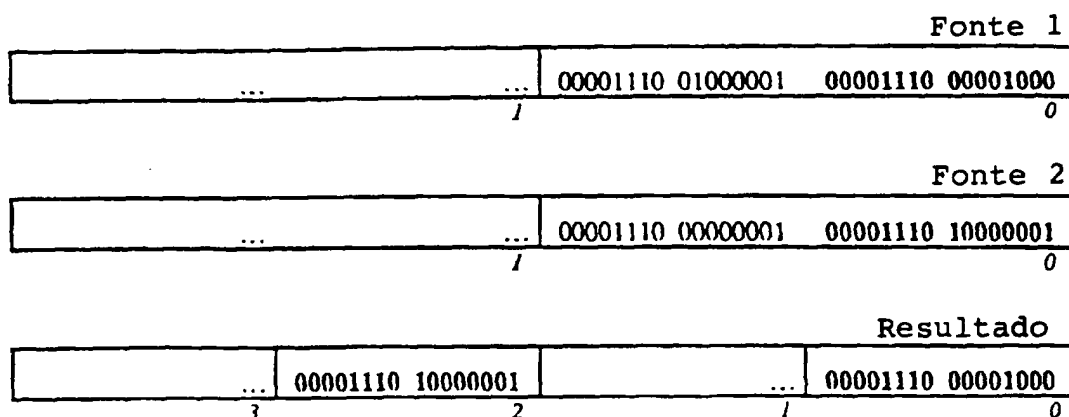


Tabela 27

Vantagens de Incluir a Operação de Empacotamento Descrita no Conjunto de Instruções

A instrução de empacotamento descrita empacota um número predefinido de bits originário de cada elemento de dados na Fonte1 e Fonte2 para gerar o Resultado. Desta maneira, o processador 109 pode empacotar os dados em tão pouco quanto a metade das instruções exigidas pelos processadores de finalidades gerais da técnica anterior. Por exemplo, a geração de um resultado que contenha quatro elementos de dados de 16 bits originários de quatro elementos de dados de 32 bits requer apenas uma instrução (em oposição a 2 instruções), conforme mostrado abaixo:

Empacotamento Superior da Fonte1, Fonte2

A ₀	.A ₀	C ₀	.C ₀	Fonte 1
G ₀	.G ₀	B ₀	.B ₀	Fonte 2
A ₀	C ₀	G ₀	B ₀	Resultado 1

Tabela 28

As aplicações típicas de múltiplos meios empacotam grandes

quantidades de dados. Desse modo, com a redução do número de instruções exigidas para empacotar estes dados em tanto quanto a metade, é intensificado o desempenho destas aplicações de múltiplos meios.

OPERAÇÃO DE DESEMPACOTAMENTO

5 Operação de Desempacotamento

Em uma concretização, uma operação de desempacotamento intercala os bytes, palavras ou palavras duplas em pacote de ordem inferior de dois dados em pacote de fonte para gerar os bytes, palavras ou palavras duplas em pacote do resultado. Esta operação é mencionada aqui como
10 uma operação de desempacotamento inferior. Em outra concretização, uma operação de desempacotamento poderia também intercalar os elementos de ordem superior (mencionados como a operação de empacotamento superior).

A Figura 20 é um diagrama de fluxo que ilustra um processo de
15 execução das operações de desempacotamento nos dados em pacote, de acordo com uma concretização da invenção.

A etapa 2001 e a etapa 2002 são executadas primeiro. Na etapa 2003, o decodificador 202 ativa a unidade de execução 130 para executar a operação de desempacotamento. O decodificador 202 comunica, através da
20 barra interna 170, o tamanho dos elementos de dados na Fonte1 e Fonte2.

Na etapa 2010, o tamanho do elemento de dados determina qual a etapa que deve ser executada a seguir. Se o tamanho dos elementos de dados foi de oito bits (dados do byte em pacote 401), então a unidade de execução 130 executa a etapa 2012. Entretanto, se o tamanho dos elementos
25 de dados nos dados em pacote for de dezesseis bits (dados de palavra em pacote 402), então a unidade de execução 130 executa a etapa 2014. Entretanto, se o tamanho dos elementos de dados nos dados em pacote for de trinta e dois bits (dados de palavra dupla em pacote 503), então a unidade de execução 130 executa a etapa 2016.

30 Assumindo-se que o tamanho dos elementos de dados de fonte é de oito bits, então é executada a etapa 2012. Na etapa 2012, é executado o seguinte. Os bits de sete até zero da Fonte 1 são os bits de sete até zero

do Resultado. Os bits de sete até zero da Fonte2 são os bits de quinze até oito do Resultado. Os bits de quinze até oito da Fonte 1 são os bits de vinte e três até dezesseis do Resultado. Os bits de quinze até oito da Fonte2 são os bits de trinta e um até vinte e quatro do Resultado. Os bits de vinte e três até dezesseis da Fonte1 são os bits de trinta e nove até trinta e dois do Resultado. Os bits de vinte e três até dezesseis da Fonte2 são os bits de quarenta e sete até quarenta do Resultado. Os bits de trinta e um até vinte e quatro da Fonte1 são os bits de cinquenta e cinco até quarenta e oito do Resultado. Os bits de trinta e um até vinte e quatro da Fonte2 são os bits de sessenta e três até cinquenta e seis do Resultado.

Assumindo-se que o tamanho dos elementos de dados de fonte é de dezesseis bits, então é executada a etapa 2012. Na etapa 2014, é executado o seguinte. Os bits de quinze até zero da Fonte1 são os bits de quinze até zero do Resultado. Os bits de quinze até zero da Fonte2 são os bits de trinta e um até dezesseis do Resultado. Os bits de trinta e um até dezesseis da Fonte1 são os bits de quarenta e sete até trinta e dois do Resultado. Os bits de trinta e um até dezesseis da Fonte2 são os bits de sessenta e três até quarenta e oito do Resultado.

Assumindo-se que o tamanho dos elementos de dados da fonte é de trinta e dois bits, então é executada a etapa 2016. Na etapa 2016, é executado o seguinte. Os bits de trinta e um até zero da Fonte1 são os bits de trinta e um até zero do Resultado. Os bits de trinta e um até zero da Fonte2 são os bits de sessenta e três até trinta e dois do Resultado.

Em uma concretização, o desempacotamento da etapa 2012 é executado simultaneamente. Entretanto, em outra concretização, este desempacotamento é executado em série. Em outra concretização, parte deste desempacotamento é executado simultaneamente e parte é executada em série. Esta discussão também se aplica a desempacotamento da etapa 2014 e da etapa 2016.

Na etapa 2020, o Resultado é armazenado no registro DEST 605.

A Tabela 29 ilustra a representação em registro da operação de

desempacotamento de palavra dupla (cada um dos elementos de dados A_{0-1} e B_{0-1} contém 32 bits).

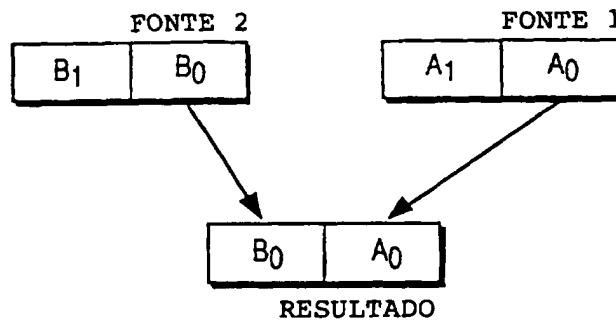


Tabela 29

A Tabela 30 ilustra a representação em registro de uma operação de desempacotamento de palavra (cada um dos elementos de dados A_{0-3} e B_{0-3} contém 16 bits).

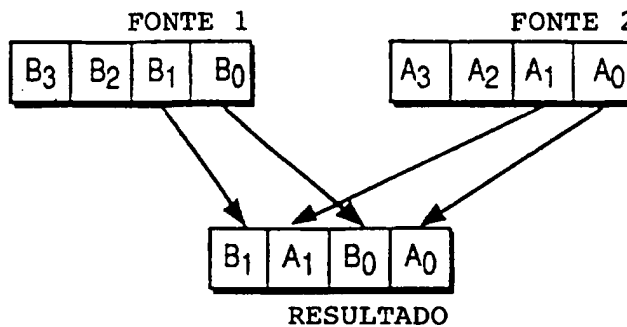


Tabela 30

A Tabela 31 ilustra a representação em registro de uma operação de desempacotamento de byte (cada um dos elementos de dados A_{0-7} e B_{0-7} contém 8 bits).

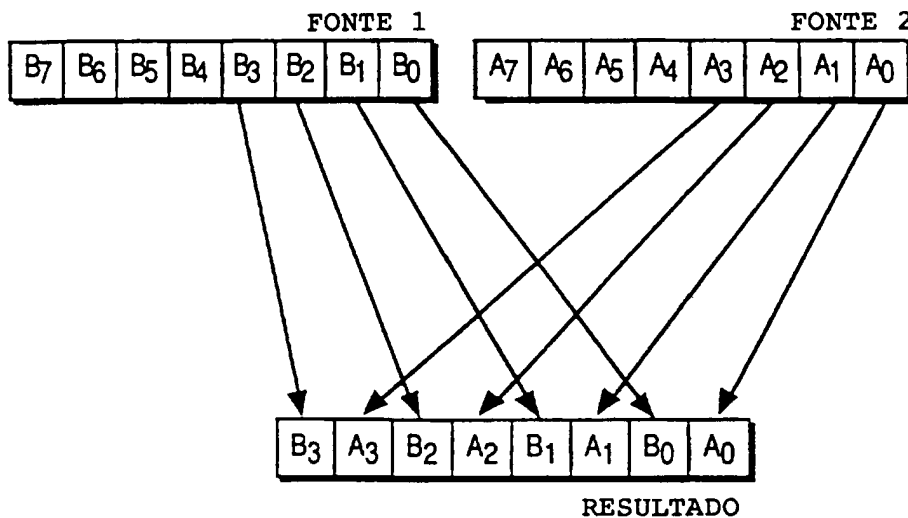


Tabela 31

Circuitos de Desempacotamento

A Figura 21 ilustra um circuito para a execução das operações de desempacotamento nos dados em pacote, de acordo com uma concreti-
5 zação da invenção. O circuito da Figura 21 inclui o circuito de controle de operação 2100, um registro de resultado 2152, um registro de resultado 2153 e um registro de resultado 2154.

O controle de operação 2100 recebe a informação originária do decodificador 202 para ativar uma operação de desempacotamento. Se o
10 tamanho dos dados em pacote da fonte for dados em pacote de byte 502, então uma ativação de saída 2132 é estabelecida pelo controle de operação 2100. Isto ativa a saída do registro de resultado 2152. Se o tamanho dos dados em pacote da fonte for dados em pacote de palavra 503, então a ativação de saída 2133 é estabelecida pelo controle de operação 2100. Isto
15 ativa a saída do registro de saída 2153. Se o tamanho dos dados em pacote de fonte for dados em pacote de palavra dupla 504, então a ativação de saída 2134 é estabelecida pelo controle de operação 2100. Isto ativa a saída do registro de resultado de saída 2154.

O registro de resultado 2152 possui as seguintes entradas. Os
20 bits de sete até zero da Fonte1 são os bits de sete até zero para o registro de resultado 2152. Os bits de sete até zero da Fonte 2 são os bits de quinze até oito para o registro de resultado 2152. Os bits de quinze até oito da Fonte1 são os bits de vinte e três até dezesseis para o registro de resultado 2152. Os bits de quinze até oito da Fonte2 são os bits de trinta e um até
25 vinte e quatro para o registro de resultado 2152. Os bits de vinte e três até dezesseis da Fonte1 são os bits de trinta e nove até trinta e dois para o registro de resultado 2152. Os bits de vinte e três até dezesseis da Fonte2 são os bits de quarenta e sete até quarenta para o registro de resultado 2152. Os bits de trinta e um até vinte e quatro da Fonte 1 são os bits cinquenta e cinco
30 até quarenta e oito para o registro de resultado 2152. Os bits de trinta e um até vinte e quatro da Fonte 2 são os bits de sessenta e três até cinquenta e seis para o registro de resultado 2152.

O registro de resultado 2153 possui as seguintes entradas. Os bits de quinze até zero da Fonte1 são os bits de quinze até zero para o registro de resultado 2153. Os bits de quinze até zero da Fonte2 são os bits de trinta e um até dezesseis para o registro de resultado 2153. Os bits de trinta e um até dezesseis da Fonte1 são os bits de quarenta e sete até trinta e dois para o registro de resultado 2153. Os bits de trinta e um até dezesseis da Fonte2 são os bits de sessenta e três até quarenta e oito do registro de resultado 1953.

O registro de resultado 2154 possui as seguintes entradas. Os bits de trinta e um até zero da Fonte1 são os bits de trinta e um até zero para o registro de resultado 2154. Os bits de trinta e um até zero da Fonte2 são os bits de sessenta e três até trinta e dois do registro de resultado 2154.

Por exemplo, na Tabela 32, é executada uma operação de desempacotamento de palavras. O controle de operação 2100 irá ativar o registro de resultado 2153 para emitir o resultado[63:0] 2160.

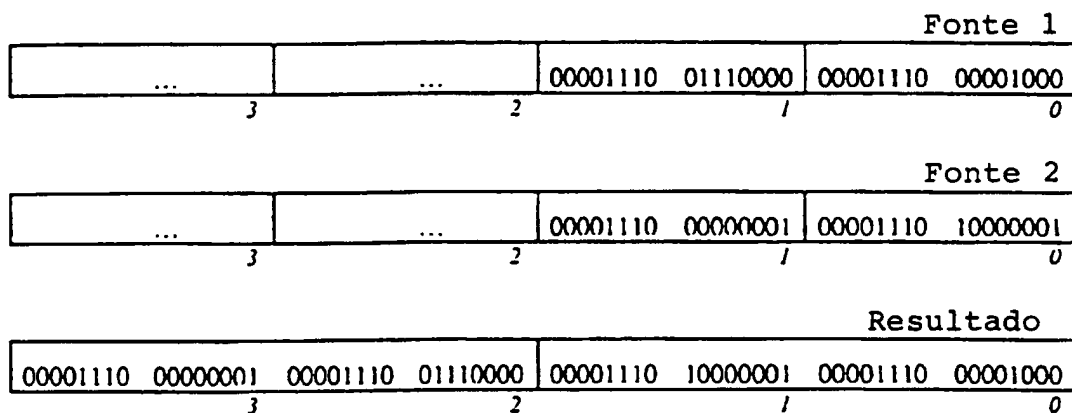


Tabela 32

Entretanto, se for executada uma operação de desempacotamento de palavra dupla, o controle de operação 2100 irá ativar o registro de resultado 2154 para emitir o resultado[63:0] 2160. A Tabela 33 ilustra este resultado.

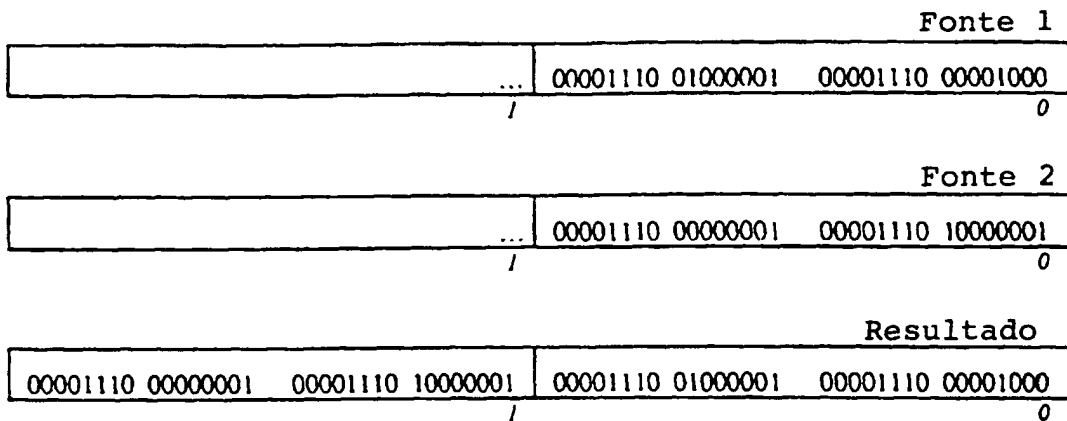
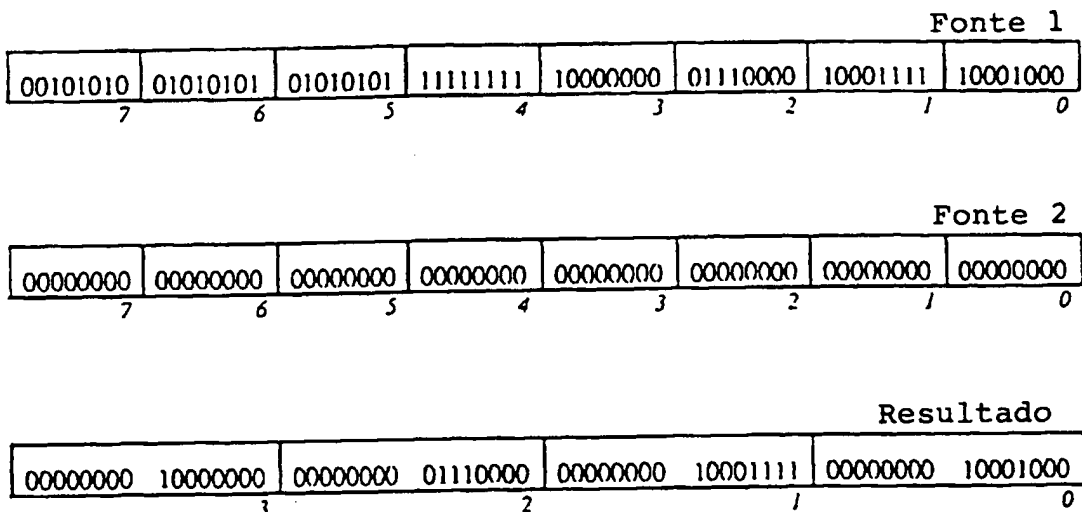


Tabela 33

Vantagens de Incluir a Instrução de Desempacotamento Descrita no Conjunto de Instruções

- 5 Com a inclusão da instrução de desempacotamento descrita no conjunto de instruções, os dados em pacote podem ser intercalados ou desempacotados. Esta instrução de desempacotamento pode ser usada para desempacotar os dados em pacote através da criação de todos os elementos de dados em todos os os da Fonte2. Um exemplo de bytes de desempacotamento é mostrado abaixo na Tabela 34a.



10 Tabela 34a

Esta mesma instrução de desempacotamento pode ser usada para intercalar os dados, conforme mostrado na Tabela 34b. A intercalação é útil em inúmeros algoritmos de múltiplos meios. Por exemplo, a intercalação é útil para transpor as matrizes e os pixels de interpolação.

Fonte 1							
00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
7	6	5	4	3	2	1	0

Fonte 2							
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
7	6	5	4	3	2	1	0

Resultado							
00000000	10000000	00000000	01110000	00000000	10001111	00000000	10001000
	3		2		1		0

Tabela 34b

Dessa forma, com a provisão desta instrução de desempacotamento no conjunto de instruções suportado pelo processador 109, o processador 109 torna-se mais versátil e pode executar os algoritmos que exijam esta funcionalidade em um nível mais alto de desempenho.

5

CONTAGEM DE OCUPAÇÃO

Contagem de Ocupação

Uma concretização da invenção permite que as operações de contagem de ocupação sejam executadas nos dados em pacote. Isto é, a invenção gera um elemento de dados de resultado para cada elemento de dados de primeiros dados em pacote. Cada elemento de dados de resultado representa o número de bits estabelecido em cada elemento de dados correspondente dos primeiros dados em pacote. Em uma concretização, o número total de bits estabelecido em um é contado.

10

15

A Tabela 35a ilustra uma representação em registro de uma operação de contagem de ocupação nos dados em pacote. A primeira fileira de bits é a representação de dados em pacote dos dados em pacote da Fonte1. A segunda fileira de bits é a representação de dados em pacote dos dados em pacote do Resultado. O número abaixo de cada bit do elemento de dados é o número do elemento de dados. Por exemplo, o elemento 0 dos dados da Fonte1 é 100011110001000_2 . Por isso, se os elementos de dados tiverem dezesseis bits de comprimento (dados de palavra), e uma ope-

20

ração de contagem de ocupação for executada, a unidade de Execução 130 irá produzir os dados em pacote do Resultado, conforme mostrado.

01110010 00000101	11111111 11111111	01111111 11111111	10001111 10001000
= 2	= 2	= 1	= 0
00000000 00000110	00000000 00010000	00000000 00001111	00000000 00000111
3	2	1	0

Tabela 35a

Em outra concretização, as contagens de ocupação são execu-
5 tadas nos elementos de dados de oito bits. A Tabela 35b ilustra uma repre-
sentação em registro de uma contagem de ocupação nos dados em pacote
que possuem oito elementos de dados em pacote de oito bits.

01111111	01010101	10101010	10000001	10000000	11111111	11001111	00000000
= 2	= 6	= 2	= 4	= 3	= 2	= 1	= 0
00000111	00000100	00000100	00000010	00000001	00001000	00000110	00000000
7	6	5	4	3	2	1	0

Tabela 35b

Em outra concretização, as contagens de ocupação são execu-
10 tadas nos elementos de dados de trinta e dois bits. A Tabela 35c ilustra uma
representação em registro de uma contagem de ocupação nos dados em
pacote que possuem dois elementos de dados em pacote de trinta e dois
bits.

11111111 11111111 11111111 11111111	10000000 11110000 11001111 10001000
= 1	= 0
00000000 00000000 00000000 00100000	00000000 00000000 00000000 00001101
1	0

Tabela 35c

15 As contagens de ocupação podem também ser executadas nos
dados inteiros de sessenta e quatro bits. Isto é, o número de bits estabeleci-

do em um é totalizado em sessenta e quatro bits de dados. A Tabela 35d ilustra uma representação em registro de uma contagem de ocupação nos dados inteiros de sessenta e quatro bits.

11111111 11111111 11111111 11111111 10000000 11110000 11001111 10001000
=
00000000 00000000 00000000 00100000 00000000 00000000 00000000 00101101

Tabela 35d

5 Processo de Execução de uma Contagem de Ocupação

A Figura 22 é um diagrama de fluxo que ilustra um processo para executar uma operação de contagem de ocupação nos dados em pacote, de acordo com uma concretização da invenção. Na etapa 2201, em resposta ao recebimento de um sinal de controle 207, o decodificador 202 decodifica esse sinal de controle 207. Em uma concretização, o sinal de controle 207 é suprido através da barra 101. Em outra concretização, o sinal de controle 207 é suprido pelo cachê 160. Dessa maneira, o decodificador 202 decodifica o código de operação para a contagem de ocupação, os endereços SRC1 602 e DEST 605 nos registros 209. Deve ser notado que SRC2 603 não é usado nesta presente concretização da invenção. Da mesma forma, não são usados nesta presente concretização a saturação/insaturação, a aritmética com sinal algébrico/sem sinal algébrico e o comprimento dos elementos de dados nos dados em pacote. Na presente concretização da invenção, é suportada apenas uma adição em pacote com comprimento de elemento de dados de dezesseis bits. Entretanto, aquele versado na técnica entenderia que as contagens de ocupação não podem ser executadas nos dados em pacote que possuem oito elementos de dados de byte em pacote ou dois elementos de dados de palavra dupla em pacote.

Na etapa 2202, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecido o endereço SRC1 602. Os registros 209 apresentam a unidade de Execução 130 com os dados em pacote, Fonte1, armazenados no registro

neste endereço. Isto é, os registros 209 comunicam os dados em pacote à unidade de Execução 130 através da barra interna 170.

Na etapa 2130, o decodificador 202 ativa a unidade de Execução 130 para executar uma operação de contagem de ocupação. Em uma concretização alternativa, o decodificador 202 adicionalmente comunica, através da barra interna 170, o comprimento dos elementos de dados.

Na etapa 2205, assumindo-se que o comprimento dos elementos de dados é de dezesseis bits, então a unidade de Execução 130 totaliza o número de bits estabelecido do bit quinze até o bit zero da Fonte1, produzindo do bit quinze até o bit zero dos dados em pacote do Resultado. Em paralelo com esta totalização, a unidade de Execução 130 soma os totais do bit trinta e um até o bit dezesseis da Fonte1, produzindo do bit trinta e um até bit dezesseis dos dados em pacote do Resultado. Em paralelo com a geração destes totais, a unidade de Execução 130 totaliza dos bit quarenta e sete até o bit trinta e dois da Fonte1, produzindo do bit quarenta e sete até o bit trinta e dois dos dados em pacote do Resultado. Em paralelo com a geração destes totais, a unidade de Execução 130 totaliza do bit sessenta e três até o bit quarenta e oito da Fonte1, produzindo do bit sessenta e três até o bit quarenta e oito dos dados em pacote do Resultado.

Na etapa 2206, o decodificador 202 ativa um registro nos registros 209 com o endereço DEST 605 do registro de destino. Dessa maneira, os dados em pacote do Resultado são armazenados no registro endereçado pelo DEST 605.

Processo de Execução de uma Contagem de Ocupação em um Elemento de Dados

A Figura 23 é um diagrama de fluxo que ilustra um processo para a execução de uma operação de contagem de ocupação em um elemento de dados de dados em pacote e para a geração de um único elemento de dados de resultado para dados em pacote de resultado, de acordo com uma concretização da invenção. Na etapa 2310a, uma soma de coluna, CSum 1a, e um transporte de coluna, CCarry 1a, são gerados a partir dos bits quinze, quatorze, treze e dois da Fonte1. Na etapa 2310b, uma soma de

coluna, CSum 1b, e um transporte de coluna, CCarry 1b, são gerados a partir dos bits onze, dez, nove e oito da Fonte1. Na etapa 2310c, uma soma de coluna, CSum 1c, e um transporte de coluna, CCarry 1c, são gerados a partir dos bits sete, seis, cinco e quatro da Fonte1. Na etapa 2310d, uma soma de coluna, CSum 1d, e um transporte de coluna CCarry 1d, são gerados a partir dos bits três, dois, um e zero da Fonte 1. Em uma concretização da invenção, as etapas 2310a-d são executadas em paralelo. Na etapa 2320a, uma soma de coluna, CSum 2a, e um transporte de coluna, CCarry 2b, são gerados a partir de CSum 1a, CCarry 1a, CSum 1b e CCarry 1b. Na etapa 2320b, uma soma de coluna, CSum 2b, e um transporte de coluna, CCarry 2b, são gerados a partir de CSum 1c, CCarry1, CSum1d, e CCarry1d. Em uma concretização da invenção, as etapas 2320a-b são executadas em paralelo. Na etapa 2330, uma soma de coluna CSum3, e um transporte de coluna, CCarry3 são gerados a partir de CSum2a, CCarry2a, CSum2b e CCarry2b. Na etapa 2340, um Resultado é gerado a partir de CSum3 e CCarry3. Em uma concretização, o Resultado é representado em dezesseis bits. Nesta concretização, na medida em que apenas os bits de quatro até zero são necessários para representarem o número máximo de bits estabelecido em uma Fonte1, os bits de quinze até cinco são estabelecidos em zero. O número máximo de bits para a Fonte1 é de dezesseis. Isto ocorre quando a Fonte1 equaliza 1111111111111111_2 . O Resultado seria dezesseis e seria representado por 0000000000010000_2 .

Desse modo, para calcular quatro elementos de dados do resultado para uma operação de contagem de ocupação em dados em pacote de sessenta e quatro bits, as etapas da Figura 23 seriam executadas para cada elemento de dados nos dados em pacote. Em uma concretização, os quatro elementos de dados de resultado de dezesseis bits seriam calculados em paralelo.

Circuito para Executar uma Contagem de Ocupação

A Figura 24 ilustra um circuito para executar uma operação de contagem de ocupação nos dados em pacote que apresentam quatro elementos de dados de palavra, de acordo com uma concretização da inven-

ção, a Figura 25 ilustra um circuito detalhado de execução de uma operação de contagem de ocupação em um elemento de dados de palavra de dados em pacote, de acordo com uma concretização da invenção.

A Figura 24 ilustra um circuito, no qual a barra 2401 da Fonte1
 5 conduz os sinais de informação para os circuitos popcnt (de contagem de ocupação) 2408a-d através da Fonte1_{IN} 2406a-d. Dessa maneira, o circuito popcnt (de contagem de ocupação) 2408a totaliza o número de bits estabelecido do bit quinze até o bit zero da Fonte1, produzindo do bit quinze até o bit zero do Resultado. O circuito popcnt (de contagem de ocupação) 2408b
 10 totaliza o número de bits estabelecido do bit trinta e um até o bit dezesseis da Fonte1, produzindo do bit trinta e um até o bit dezesseis do Resultado. O circuito popcnt (de contagem de ocupação) 2408c totaliza o número de bits estabelecido do bit quarenta e sete até o bit trinta e dois da Fonte1, produzindo do bit quarenta e sete até o bit trinta e dois do Resultado. O circuito popcnt (de contagem de ocupação) 2408d totaliza o número de bits estabelecido do bit sessenta e três para o bit quarenta e oito da Fonte1, produzindo do bit sessenta e três até o bit quarenta e oito do Resultado. A ativação 2404a-d recebe, a partir do Controle de Operação 2410, através do controle 2403, através dos sinais de controle que ativam os circuitos popcnt (de contagem de ocupação) 2408a-d para executar as operações de contagem de ocupação, e para colocar um Resultado na Barra de Resultado 2409. Aquele versado na técnica seria capaz de criar tal circuito, uma vez fornecidas a descrição acima e as ilustrações nas Figuras 1-6b e 23-25.

Os circuitos popcnt (de contagem de ocupação) 2408a-d comunicam a informação de resultado de uma operação de contagem de ocupação em pacote para a barra de Resultado 2409, através da saída de resultado 2407a-d. Esta informação de resultado é então armazenada no registro interior especificado pelo endereço de registro DEST 605.

Circuito para a Execução de uma Contagem de Ocupação em Um Elemento de Dados

A Figura 25 ilustra um circuito detalhado para a execução de uma operação de contagem de ocupação em um elemento de dados de pa-

lavra de dados em pacote. Em particular, a Figura 25 ilustra uma porção do circuito popcnt (de contagem de ocupação) 2408a. Para alcançar o desempenho máximo de aplicações que empregam uma operação de contagem de ocupação, a operação deve ser completa dentro de um ciclo de relógio. Por
5 isso, uma vez fornecido o acesso a um registro e o armazenamento de um resultado exige uma certa porcentagem de ciclo de relógio, o circuito da Figura 24 irá completar sua operação dentro de aproximadamente 80% de um período de relógio. Este circuito possui a vantagem de permitir que o processador 109 execute uma operação de contagem de ocupação em quatro
10 elementos de dados de dezesseis bits em um ciclo de relógio.

O circuito popcnt (de contagem de ocupação) 2408a emprega somadores de transporte-reserva 4->2 (a menos que de outra maneira especificado, os somadores de transporte-reserva (CSA) irão se referir a um somador de transporte-reserva 4->2). Os somadores de transporte-reserva
15 4->2, conforme podem ser empregados no circuito popcnt (de contagem de ocupação) 2408a-d, são bem conhecidos na técnica. Um somador de transporte-reserva 4->2 é um somador que soma quatro operandos, resultando em duas somas. Uma vez que a operação de contagem de ocupação no
20 circuito popcnt (de contagem de ocupação) 2408a envolver dezesseis bits, o primeiro nível irá incluir quatro somadores de transporte-reserva 4->2. Estes quatro somadores de transporte-reserva 4->2 transformam os dezesseis operandos de um bit em oito somas de dois bits. O segundo nível transforma as oito somas de dois bits em quatro somas de três bits, e o terceiro nível
25 transforma as quatro somas de três bits em duas somas de quatro bits. Em seguida, um somador completo de quatro bits soma as duas somas de quatro bits para gerar um resultado final.

Embora sejam usados os somadores de transporte-reserva 4->2, uma concretização alternativa poderia empregar somadores de transporte-reserva 3->2. Alternativamente, diversos somadores completos poderiam ser
30 usados; entretanto, esta configuração não apresentaria um resultado tão rapidamente quanto a concretização mostrada na Figura 25.

A Fonte1_{IN 15-0 24}06a conduz do bit quinze até o bit zero da Fon-

te1. Os primeiros quatro bits são acoplados às entradas de um somador de transporte-reserva 4->2 (CSA 2510a). Os próximos quatro bits são acoplados às entradas de CSA 2510b. Os próximos quatro bits são acoplados às entradas de CSA 2510c. Os quatro bits finais são acoplados às entradas de

5 CSA 2510d. Cada CSA 2510a-d gera duas saídas de dois bits. As duas saídas de dois bits de CAS 2510a são acopladas as duas entradas de CSA 2520a. As duas saídas de dois bits de CAS 2510b são acopladas às outras duas entradas de CSA 2520a. As duas saídas de dois bits de CSA 2510a são acopladas às duas entradas de CSA 2520b. As duas saídas de dois bits

10 de CSA 2510d são acopladas às outras duas entradas de CSA 2530b. Cada CSA 2520a-d gera duas saídas de três bits. As duas saídas de três bits de 2520a são acopladas as duas entradas de CSA 2530. As duas saídas de três bits de 2520b são acopladas às outras duas entradas de CSA 2530. CSA 2530 gera duas saídas de quatro bits.

15 Estas duas saídas de quatro bits são acopladas as duas entradas de um somador completo (FA 2550). FA 2550 soma as duas entradas de quatro bits e comunica do bit três até o bit zero da Saída de Resultado 2407a como um total da adição de duas entradas de quatro bits. FA 2550 gera o bit quatro da Saída de Resultado 2407a através da saída de transporte (CO 2552). Em uma concretização alternativa, um somador completo

20 de cinco bits é usado para gerar do bit quatro até o bit zero da Saída de Resultado 2407a. Em cada caso, dos bits de quinze até cinco da Saída de Resultado 2407a são ligados a zero.

Embora não-mostrado na Figura 25, aquele versado na técnica

25 entenderia que a Saída de Resultado 2407a poderia ser multiplexada ou separada na barra de Resultado 2409. O multiplexador seria controlado através da Ativação 2404a. Isto iria permitir que outros circuitos da unidade de Execução gravassem os dados na barra de Resultado 2409.

30 Vantagens de Incluir a Operação de Contagem de Ocupação Descrita no Conjunto de Instruções

A instrução de contagem de ocupação descrita calcula o número de bits estabelecido em cada um dos elementos de dados de dados em pa-

cote, tal como a Fonte1. Dessa maneira, com a inclusão desta instrução no conjunto de instruções, pode ser executada uma operação de contagem de ocupação nos dados em pacote em uma única instrução. Em contraste, os processadores de finalidades gerais da técnica anterior têm que executar

5 numerosas instruções para desempacotar a Fonte1, têm que executar a função individualmente em cada elemento de dados desempacotados, e depois empacotar os resultados para o processamento em pacote adicional.

Desse modo, com a provisão esta instrução de contagem de ocupação no conjunto de instruções suportado pelo processador 109, é intensificado o desempenho dos algoritmos exigindo esta funcionabilidade.

10

OPERAÇÕES LÓGICAS

Operações Lógicas

Em uma concretização da invenção, o registro SRC1 contém os dados em pacote (Fonte1), o registro SRC2 contém os dados em pacote

15 (Fonte2) e o registro DEST irá conter o resultado (Resultado) da execução da operação lógica selecionada na Fonte1 e Fonte2. Por exemplo, se a operação AND lógica for selecionada, a Fonte1 será operada com lógica em AND com a Fonte2.

Em uma concretização da invenção, as seguintes operações

20 lógicas são suportadas: AND lógica, ANDN lógica, OR lógica, e XOR lógica. As operações lógicas AND, OR e XOR são bem conhecidas na técnica. A operação lógica ANDN faz com que a Fonte2 seja operada com lógica em AND com a inversão lógica da Fonte1. Enquanto a invenção é descrita em relação a estas operações lógicas, concretizações alternativas poderiam im-

25 plementar outras operações lógicas.

A Figura 26 é um diagrama de fluxo que ilustra um processo para a execução de diversas operações lógicas nos dados em pacote, de acordo com uma concretização da invenção.

Na etapa 2601, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa maneira, o decodificador

30 202 decodifica o código de operação para a operação lógica apropriada (isto é, AND, ANDN, OR ou XOR), e os endereços SRC1 602, SRC2 603 e DEST

605 nos registros 209.

Na etapa 2602, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 e SRC2 603. Os registros 209 apresentam a unidade de execução 130 com os dados em pacote armazenados no registro SRC1 602 (Fonte1) e nos dados em pacote armazenados no registro SRC2 603 (Fonte2). Isto é, os registros 209 comunicam os dados em pacote à unidade de execução 130 através da barra interna 170.

Na etapa 2603, o decodificador 202 ativa a unidade de execução 130 para executar a operação selecionada das operações lógicas em pacote.

Na etapa 2610, a operação selecionada das operações lógicas em pacote determina qual a etapa que deve ser executada a seguir. A unidade de Execução 130 executa a etapa 2612, se a operação AND lógica fosse selecionada; a unidade de Execução 130 executa a etapa 2613, se a operação ANDN lógica fosse selecionada; a unidade de Execução 130 executa a etapa 2614, se a operação OR lógica fosse selecionada; e a unidade de Execução 130 executa a etapa 2615, se a operação lógica XOR fosse selecionada.

Assumindo-se que a operação AND lógica fosse selecionada, é executada a etapa 2612. Na etapa 2612, os bits de sessenta e três até zero da Fonte1 são operados com lógica em AND com os bits de sessenta e três até zero da Fonte2 para gerar os bits de sessenta e três até zero do Resultado.

Assumindo-se que a operação ANDN lógica fosse selecionada, é executada a etapa 2613. Na etapa 2613, os bits de sessenta e três até zero da Fonte1 são operados com lógica em ANDN com os bits de sessenta e três até zero da Fonte2 para gerar os bits de sessenta e três até zero do Resultado.

Assumindo-se que a operação OR lógica fosse selecionada, é executada a etapa 2614. Na etapa 2614, os bits de sessenta e três até zero da Fonte um são operados com lógica em OR com os bits de sessenta e três

até zero da Fonte2 para gerar os bits de sessenta e três até zero do Resultado.

Assumindo-se que a operação XOR lógica fosse selecionada, a etapa 2615 é executada. Na etapa 2615, os bits de sessenta e três até zero da Fonte1 são exclusivamente operados com lógica em OR com os bits de sessenta e três até zero da Fonte2 para gerar os bits de sessenta e três até zero do Resultado.

Na etapa 2620, o Resultado é armazenado no registro DEST.

A Tabela 36 ilustra a representação em registro de uma operação ANDN lógica nos dados em pacote. A primeira fileira de bits é a representação de dados em pacote da Fonte1. A segunda fileira de bits é a representação de dados em pacote da Fonte2. A terceira fileira de bits é a representação de dados em pacote do Resultado. O número abaixo de cada bit de elemento de dados é o número do elemento de dados. Por exemplo, o elemento dois de dados da Fonte1 é $11111111\ 00000000_2$.

11111111 11111111	11111111 00000000	11111111 00000000	00001110 00001000
ANDN Lógica ³	ANDN Lógica ²	ANDN Lógica ¹	ANDN Lógica ⁰
00000000 00000000	00000000 00000001	10000000 00000000	00001110 10000001
=	=	=	=
00000000 00000000	00000000 00000001	00000000 00000000	00000000 10000001
₃	₂	₁	₀

Tabela 36

Enquanto a invenção é descrita em relação a mesma operação lógica que é executada nos elementos de dados correspondentes na Fonte1 e Fonte2, concretizações alternativas poderiam suportar as instruções que possibilitaram a operação lógica executada nos elementos de dados correspondentes a serem selecionados em uma base por elemento.

Circuitos Lógicos de Dados em pacote

Em uma concretização, as operações lógicas descritas podem ocorrer nos múltiplos elementos de dados no mesmo número de ciclos de relógio que a única operação lógica nos dados desempacotados. Para se

alcançar a execução no mesmo número de ciclos de relógio, é usado o paralelismo.

A Figura 27 ilustra um circuito para executar as operações lógicas nos dados em pacote, de acordo com uma concretização da invenção. O controle de operação 2700 controla os circuitos executando as operações lógicas. O controle de operação 2700 processa o sinal de controle e emite os sinais de seleção nas linhas de controle 2780. Estes sinais de seleção comunicam ao Circuito de Operações Lógicas 2701 a operação selecionada das operações AND, ANDN, OR e XOR.

10 O Circuito de Operações Lógicas 2701 recebe a Fonte1[63:0] e a Fonte2[63:0] e executa a operação lógica indicada pela seleção de sinais para gerar o Resultado. O Circuito de Operações Lógicas 2701 comunica o Resultado[63:0] ao Registro de Resultado 2731.

15 Vantagens de Incluir as Operações Lógicas Descritas no Conjunto de Instruções

As instruções lógicas descritas executam uma operação lógica AND, uma operação lógica AND NOT, uma operação lógica OR e uma operação lógica OR NOT. Estas instruções são úteis em qualquer aplicação que exija a manipulação lógica de dados. Com a inclusão destas instruções no conjunto de instruções suportado pelo processador 109, estas operações lógicas podem ser executadas nos dados em pacote em uma instrução.

COMPARAÇÃO EM PACOTE

Operação de Comparação em Pacote

25 Em uma concretização da invenção, o registro SRC1 602 contém os dados (Fonte1) a serem comparados, o registro SRC2 603 contém os dados (Fonte2) a serem comparados, e o registro DEST 605 irá conter o resultado de comparação (Resultado). Isto é, a Fonte1 terá cada elemento de dados independentemente comparado por cada elemento de dados da Fonte2, de acordo com uma relação indicada.

30 Em uma concretização da invenção, são suportadas as seguintes relações de comparação: igual, com sinal algébrico maior do que; com sinal algébrico maior do que ou igual; sem sinal algébrico maior do que; ou

sem sinal algébrico maior do que ou igual. A relação é testada em cada par de elementos de dados correspondentes. Por exemplo, a Fonte1[7:0] é maior do que a Fonte2[7:0], com o resultado sendo Resultado[7:0]. Se o resultado da comparação satisfizer a relação, então, em uma concretização, o elemento de dados correspondente no Resultado é estabelecido a todos aqueles. Se o resultado de comparação não satisfizer a relação, então o elemento de dados correspondente no Resultado é estabelecido para todos os zeros.

A Figura 28 é um diagrama de fluxo que ilustra um processo para a execução de operações de comparação em pacote nos dados em pacote, de acordo com uma concretização da invenção.

Na etapa 2801, o decodificador 202 decodifica o sinal de controle 207 recebido pelo processador 109. Dessa forma, o decodificador decodifica o código de operação para a operação de comparação apropriada, os endereços SRC1 602, SRC2 603 e DEST 605 nos registros 209, a saturação/insaturação (não necessariamente obrigatórias para as operações de comparação), a aritmética com sinal algébrico/sem sinal algébrico, e o comprimento dos elementos de dados nos dados em pacote. Conforme anteriormente mencionado, SRC1 602 (ou SRC2 603) pode ser usado como DEST 605.

Na etapa 2802, através da barra interna 170, o decodificador 202 acessa os registros 209 no arquivo de registro geral 150, uma vez fornecidos os endereços SRC1 602 e SRC2 603. Os registros 209 apresentam a unidade de execução 130 com os dados em pacote armazenados no registro SRC1 602 (Fonte1), e os dados em pacote armazenados no registro SRC2 603 (Fonte2). Isto é, os registros 209 comunicam os dados em pacote à unidade de execução 130 através da barra interna 170.

Na etapa 2803, o decodificador 202 ativa a unidade de execução 130 para executar a operação de comparação em pacote apropriada. O decodificador 202 adicionalmente comunica, através da barra interna 170, o tamanho dos elementos de dados e a relação para a operação de comparação.

Na etapa 2810, o tamanho do elemento de dados determina qual a etapa que deve ser executada a seguir. Se o tamanho dos elementos de dados for de oito bits (dados byte em pacote 401), então a unidade de execução 130 executa a etapa 2812. Entretanto, se o tamanho dos elementos de dados nos dados em pacote for de dezesseis bits (dados de palavra em pacote 402), então a unidade de execução 130 executa a etapa 2814. Em uma concretização, apenas são suportadas as comparações em pacote do tamanho de elemento de dados de oito bits de dezesseis bits. Entretanto, em outra concretização, uma comparação em pacote do tamanho do elemento de dados de trinta e dois bits é também suportada (palavra dupla em pacote 403).

Assumindo-se que o tamanho dos elementos de dados é de oito bits, então é executada a etapa 2812. Na etapa 2812, é executado o seguinte. Os bits de sete até zero da Fonte1 são comparados aos bits de sete até zero da Fonte2, gerando os bits de sete até zero do Resultado. Os bits de quinze até oito da Fonte1 são comparados aos bits de quinze até oito da Fonte2, gerando os bits de quinze até oito do Resultado. Os bits de vinte e três até dezesseis da Fonte1 são comparados aos de vinte e três até dezesseis da Fonte2, gerando os bits de vinte e três até dezesseis do Resultado. Os bits de trinta e um até vinte e quatro da Fonte1 são comparados aos bits de trinta e um até vinte e quatro, gerando os bits de trinta e um até vinte e quatro do Resultado. Os bits de trinta e nove até trinta e dois da Fonte1 são comparados aos bits de trinta e nove até trinta e dois da Fonte2, gerando os bits de trinta e nove até trinta e dois do Resultado. Os bits de quarenta e sete até quarenta da Fonte1 são comparados com os bits de quarenta e sete até quarenta da Fonte2, gerando os bits de quarenta e sete até quarenta do Resultado. Os bits de cinqüenta e cinco até quarenta e oito da Fonte1 são comparados com os bits de cinqüenta e cinco até quarenta e oito da Fonte2, gerando os bits de cinqüenta e cinco até quarenta e oito do Resultado. Os bits de sessenta e três até cinqüenta e seis da Fonte1 são comparados com os bits de sessenta e três até cinqüenta e seis da Fonte2, gerando os bits de sessenta e três até cinqüenta e seis do Resultado.

Assumindo-se que o tamanho dos elementos de dados é de dezesseis bits, então é executada a etapa 2814. Na etapa 2814, é executado o seguinte. Os bits de quinze até zero da Fonte1 são comparados com os bits de quinze até zero da Fonte2, gerando os bits de quinze até zero do Resultado. Os bits de trinta e um até dezesseis da Fonte1 são comparados aos bits de trinta e um até dezesseis da Fonte2, gerando os bits de trinta e um até dezesseis do Resultado. Os bits de quarenta e sete até trinta e dois da Fonte1 são comparados com os bits de quarenta e sete até trinta e dois da Fonte2, gerando os bits de quarenta e sete até trinta e dois do Resultado. Os bits de sessenta e três até quarenta e oito da Fonte1 são comparados aos bits de sessenta e três até quarenta e oito da Fonte2, gerando os bits de sessenta e três até quarenta e oito do Resultado.

Em uma concretização, as comparações da etapa 2812 são executadas simultaneamente. Entretanto, em outra concretização, estas comparações são executadas em série. Em outra concretização, algumas destas comparações são executadas simultaneamente e algumas são executadas em série. Esta discussão também aplica-se igualmente às comparações da etapa 2814.

Na etapa 2820, o Resultado é armazenado no registro DEST 605.

A Tabela 37 ilustra a representação em registro da comparação em pacote sem sinal algébrico maior do que a operação. A primeira fileira de bits é a representação dos dados em pacote da Fonte1. A segunda fileira de bits é a representação de dados em pacote da Fonte2. A terceira fileira é a representação de dados em pacote do Resultado. O número abaixo de cada bit do elemento de dados é o número do elemento de dados. Por exemplo, o elemento três dos dados da Fonte1 é 10000000_2 .

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
> ⁷	> ⁶	> ⁵	> ⁴	> ³	> ²	> ¹	> ⁰
00000000	00000000	10000000	00000000	11110011	00000000	10001110	10001000
↓	↓	↓	↓	↓	↓	↓	↓
11111111	11111111	00000000	11111111	00000000	11111111	11111111	00000000
⁷	⁶	⁵	⁴	³	²	¹	⁰

Tabela 37

A Tabela 38 ilustra a representação em registro da operação de comparação em pacote com sinal algébrico maior ou igual nos dados de byte em pacote.

00101010	01010101	01010101	11111111	10000000	01110000	10001111	10001000
>= ⁷	>= ⁶	>= ⁵	>= ⁴	>= ³	>= ²	>= ¹	>= ⁰
00000000	00000000	10000000	00000000	11110011	00000000	10001110	10001000
↓	↓	↓	↓	↓	↓	↓	↓
11111111	11111111	11111111	00000000	00000000	11111111	00000000	11111111
⁷	⁶	⁵	⁴	³	²	¹	⁰

5 Tabela 38Circuitos de Comparação de Dados em pacote

Em uma concretização, a operação de comparação pode ocorrer nos múltiplos elementos de dados no mesmo número de ciclos de relógio como uma única operação de comparação nos dados desempacotados.

- 10 Para se alcançar a execução no mesmo número de ciclos de relógio, é usado o paralelismo. Isto é, os registros são simultaneamente instruídos para executar a operação de comparação nos elementos de dados. Isto é discutido em maiores detalhes abaixo.

- 15 A Figura 29 ilustra um circuito para executar as operações de comparação em pacote nos bytes individuais dos dados em pacote, de acordo com uma concretização da invenção. A Figura 29 ilustra o uso de um circuito de comparação de fração de byte modificado, o estágio₂₉₉₉ da fração

de byte. Cada fração de byte, exceto para a fração de byte do elemento de dados mais significativo, inclui um controle de bit e unidade de controle. A fração de byte do elemento de dados mais significativo precisa apenas ter uma unidade de comparação.

5 A unidade_i2911 de comparação e a unidade_{i+1}2971 de comparação permitem, cada qual, que oito bits da Fonte1 sejam comparados a oito bits correspondentes da Fonte2. Em uma concretização, cada unidade de comparação opera como um circuito de comparação conhecido de oito bits. Tal circuito de comparação conhecido de oito bits inclui um circuito de fração
10 de byte que permite a subtração da Fonte2 a partir da Fonte1. Os resultados da subtração são processados para determinarem os resultados da operação de comparação. Em uma concretização, os resultados da subtração incluem uma informação de saturação de capacidade. Esta informação de saturação de capacidade é testada para determinar se o resultado da operação
15 de comparação é verdadeiro.

Cada unidade de comparação possui uma entrada de Fonte1, uma entrada de Fonte2, uma entrada de controle, um sinal de próximo estágio, um sinal de estágio final, e uma saída de resultado. Por isso, a unidade_i2911 possui a entrada Fonte1_i2931, a entrada Fonte2_i2933, a entrada de
20 controle_i2901, o próximo estágio de sinal_i 2913, a entrada de estágio_i2912 final, e um resultado armazenado no registro_i2951 de resultado. Por isso, a unidade_{i+1}2971 de comparação possui a entrada de Fonte1_{i+1}2932, a entrada de Fonte2_{i+1}2934, a entrada de controle_{i+1}2902, o próximo estágio de sinal_{i+1}2973, a entrada de estágio_{i+1}2972 final, e um resultado armazenado no
25 registro_{i+1}2952 do resultado.

A entrada de Fonte1_n é tipicamente uma porção de oito bits da Fonte1. Os oito bits representam o menor tipo de elemento de dados, um elemento de dados de byte em pacote 401. A entrada de Fonte2 é a porção correspondente de oito bits da Fonte2. O controle de operação 2900 transmite os sinais de controle para ativar cada unidade de comparação, com a
30 finalidade de executar a comparação exigida. Os sinais de controle são determinados a partir da relação para a comparação (por exemplo com sinal

algébrico maior do que) e o tamanho do elemento de dados (por exemplo, byte ou palavra). O sinal de próximo estágio é recebido a partir do controle de bit para essa unidade de comparação. As unidades de comparação são efetivamente combinadas pelas unidades de controle de bit, quando for usado um elemento de dados com tamanho de byte maior. Por exemplo, quando forem comparados os dados em pacote de palavra, a unidade de controle de bit entre a primeira unidade de comparação e a segunda unidade de comparação irá fazer com que as duas unidades de comparação atuem como uma unidade de comparação de dezesseis bits. Similarmente, a unidade de comparação entre as terceira e quarta unidades de comparação irá fazer com que estas duas unidades de comparação atuem como uma unidade de comparação. Isto continua para os quatro elementos de dados de palavra em pacote.

Dependendo da relação desejada e dos valores da Fonte1, e Fonte2, a unidade de comparação executa a comparação permitindo que o resultado da unidade de comparação de ordem mais superior seja propagado até a unidade de comparação de ordem inferior ou vice-versa. Isto é, cada unidade de comparação irá prover os resultados da comparação com o uso da informação comunicada pelo controle de bit. Se forem usados os dados em pacote de palavra dupla, então quatro unidades de comparação atuam entre si para formar uma unidade de comparação com um comprimento de trinta e dois bits para cada elemento de dados. A saída do resultado de cada unidade de comparação representa o resultado da operação de comparação na porção da Fonte1 e Fonte2 da unidade de comparação em que está operando.

O controle de bit é ativado a partir do controle de operação através da ativação dos dados em pacote. O controle de bit controla o próximo estágio e o último estágio. Assume-se, por exemplo, que a unidade de comparação seja responsável pelos oito bits menos significativos da Fonte1 e Fonte2, e uma unidade seja responsável pelos próximos oito bits da Fonte1 e Fonte2. Se for executada uma comparação nos dados de byte em pacote, o controle de bit não

irá permitir que a informação de resultado originária da unidade_{i+1}2971 seja comunicada com a unidade_i2911 de comparação, e vice-versa. Entretanto, se for executada uma comparação nas palavras em pacote, então o controle_i2920 de bit irá permitir que a informação do resultado (em uma concretização, uma saturação de capacidade) originária da unidade_i2911 de comparação seja comunicada à unidade_{i+1}de comparação e a informação do resultado (em uma concretização, uma saturação de capacidade) originária da unidade_{i+1}2971 seja comunicada à unidade_i2911 de comparação.

Por exemplo, na Tabela 39, é executada uma comparação de byte em pacote com sinal algébrico maior. Assume-se que a unidade_{i+1} 2971 de comparação opera no elemento de dados um, e a unidade_i2911 de comparação opera no elemento de dados zero. A unidade_{i+1} de comparação 2971 compara os oito bits mais significativos de uma palavra e comunica a informação do resultado através do último estágio_{i+1}2972. A unidade_i2911 de comparação compara os oito bits menos significativos de palavra e comunica a informação do resultado através do próximo estágio_i2913. Entretanto, o controle de operação 2900 irá fazer com que o controle_i2920 de bit detenha a propagação dessa informação de resultado, recebida a partir do último estágio_{i+1}2972 e do próximo estágio_i2913, entre as unidades de comparação.

...	00001110	00001000
> 7	> 6	> 5	> 4	> 3	> 2	> 1	> 0
...	00001110	10001000
↓	↓	↓	↓	↓	↓	↓	↓
...	00000000	11111111
7	6	5	4	3	2	1	0

20 Tabela 39

Entretanto, se for executada uma palavra em pacote com sinal algébrico maior do que a comparação, então o resultado da unidade_{i+1}2971 de comparação será comunicado à unidade_i de comparação 2911, e vice-versa. A Tabela 40 ilustra este resultado. Este tipo de comunicação seria igualmente permitido para as comparações de palavra dupla em pacote.

...	00001110 00001000
> 3	> 2	> 1	> 0
...	00001110 10000001
↓	↓	↓	↓
...	00000000 00000000
3	2	1	0

Tabela 40

Cada unidade de comparação é opcionalmente acoplada a um registro de resultado. O registro de resultado temporariamente armazena o resultado da operação de comparação até que o resultado completo, Resultado[63:0]2960, possa ser transmitido ao registro DEST 605.

Para um circuito completo de comparação em pacote de sessenta e quatro bits, são usadas oito unidades de comparação e sete unidades de controle de bit. Tal circuito pode também ser usado para executar uma comparação nos dados desempacotados de sessenta e quatro bits, usando assim o mesmo circuito para executar a operação de comparação desempacotada e a operação de comparação em pacote.

Vantagens de Incluir a Operação de Comparação em Pacote Descrita no Conjunto de Instruções

A instrução de comparação em pacote descrita armazena o resultado de comparação da Fonte1 e Fonte2 como uma máscara em pacote. Conforme anteriormente descrito, as bifurcações condicionais nos dados tornam-se imprevisíveis, e, conseqüentemente, o desempenho do processador de custo, porque elas interrompem os algoritmos de prognóstico de bifurcação. Entretanto, com a geração de uma máscara em pacote, esta instrução de comparação reduz o número de bifurcações condicionais com base nos dados. Por exemplo, a função (se $Y > A$, então $X = X + B$; ou $X = X$) pode ser executada nos dados em pacote, conforme mostrado abaixo na Tabela 41 (os valores mostrados na Tabela 41 são mostrados em notação hexadecimal).

Comparação Maior do que da Fonte1, Fonte2

00000001	00000000	Fonte1=Y ₀₋₁
>	>	
00000000	00000001	Fonte2=A ₀₋₁
=		
FFFFFFFF	00000000	Máscara

Empacotamento lógica AND em pacote da Fonte3, Máscara

00000005	0000000 ^A	Fonte3=B ₀₋₁
>	>	
FFFFFFFF	00000000	Máscara
=		
00000005	00000000	Resultado

Soma em pacote da Fonte4, Resultado

00000010	00000020	Fonte4=X ₀₋₁
>	>	
00000005	00000000	Resultado
=		
00000015	00000020	Novo valor X ₀₋₁

Tabela 41

- Conforme pode ser visto a partir do exemplo acima, as bifurcações condicionais não são mais exigidas. Uma vez que uma instrução de bifurcação não é exigida, os processadores que especulativamente predizem as bifurcações não apresentam um decréscimo do desempenho, quando do uso desta instrução de comparação para executar esta e outras operações similares. Dessa maneira, com a provisão desta instrução de comparação no conjunto de instruções suportado por um processador 109, o processador 109 pode executar os algoritmos que exijam esta funcionalidade em um nível maior de desempenho.

ALGORITMOS EXEMPLIFICATIVOS DE MÚLTIPLOS MEIOS

- Para ilustrar a versatilidade do conjunto de instruções descrito, são descritos abaixo diversos algoritmos exemplificativos de múltiplos meios. Em alguns casos, as instruções de dados em pacote similares poderiam ser

usadas para a execução certas etapas nestes algoritmos. Várias etapas que exigem o uso de instruções de processador de finalidades gerais para gerenciar o movimento de dados, a repetição e a bifurcação condicional tendo sido omitidas nos seguintes exemplos.

5 1) Multiplicação de Números Complexos

A instrução de multiplicação-soma descrita pode ser usada para multiplicar dois números complexos em uma única instrução, conforme mostrado na Tabela 42a. A multiplicação de dois números complexos (por exemplo, $r_1 i_1$ e $r_2 i_2$) é executada de acordo com a seguinte equação:

10 Componente Real = $r_1 \cdot r_2 \cdot i_1 \cdot i_2$

Componente Imaginário = $r_1 \cdot i_2 + r_2 \cdot i_1$

Se esta instrução for implementada para ser completada a cada ciclo de relógio, a invenção pode multiplicar dois números complexos a cada ciclo de relógio.

15 Multiplicação-Soma da Fonte1, Fonte2

r_1	i_2	r_1	i_1	Fonte1
r_2	$-i_2$	i_2	r_2	Fonte2
=				
Componente Real:		Componente Imaginário:		Resultado 1
$r_1 r_2 - i_1 i_2$		$r_1 i_2 + r_2 i_1$		

Tabela 42a

Como outro exemplo, a Tabela 42b mostra as instruções usadas para multiplicar entre si os três números complexos.

Multiplicação-Soma da Fonte1, Fonte2

r_1	i_1	r_1	i_1	Fonte1
r_2	$-i_2$	i_2	r_2	Fonte2
=				
Componente Real 1:		Componente Imaginário 1:		Resultado 1
$r_1 r_2 - i_1 i_2$		$r_1 i_2 + r_2 i_1$		

Deslocamento em pacote para a direita, da Fonte1, Fonte2

Componente Real		Componente Imaginário		Resultado1
16				Resultado2
=				
	Componente Real 1		Componente Imaginário 1	

Empacotamento do Resultado2, Resultado2

	Componente Real 1		Componente Imaginário 1	Resultado2
	Componente Real 1		Componente Imaginário 1	Resultado2
=				
Componente Real 1	Componente Imaginário 1	Componente Real 1	Componente Imaginário 1	Resultado3

Multiplicação-Soma do Resultado3, Fonte3

Componente Real 1: $r_1r_2-i_1i_2$	Componente Imaginário 1: $r_1i_2+r_2i_1$	Componente Real 1: $r_1r_2-i_1i_2$	Componente Imaginário 1: $r_1i_2+r_2i_1$	Resultado3
r_3	$-i_3$	i_3	r_3	Fonte3
=				
Componente Real 2		Componente Imaginário 2		Resultado4

Tabela 42b

5 2) Operações de Acúmulo de Multiplicação

As instruções descritas podem também ser usadas para multiplicar e acumular valores. Por exemplo, dois conjuntos de quatro elementos de dados (A_{1-4} e B_{1-4}) podem ser multiplicados e acumulados, conforme mostrado abaixo na Tabela 43. Em uma concretização, cada uma das instruções mostradas na Tabela 43 é implementada para completar cada ciclo de relógio.

Multiplicação-Soma da Fonte1, Fonte2

0	0	A_1	A_2	Fonte1
0	0	B_1	B_2	Fonte2
=				
0		$A_1B_1+A_2B_2$		Resultado1

Multiplicação-Soma da Fonte3, Fonte4

0	0	A_3	A_4	Fonte3
0	0	B_3	B_4	Fonte4
=				
0		$A_3A_4+B_3B_4$		Resultado2

Soma Desempacotada do Resultado1, Resultado2

0		$A_1B_1+A_2B_2$		Resultado1
0		$A_3A_4+B_3B_4$		Resultado2
=				
0		$A_1B_1+A_2B_2+ A_3A_4+B_3B_4$		Resultado3

Tabela 43

- 5 Se o número de elementos de dados em cada conjunto exceder a 8 e for um múltiplo de 4, a multiplicação e o acúmulo destes conjuntos exigem menos instruções, se executados conforme mostrado na Tabela 44 abaixo.

Multiplicação-Soma da Fonte1, Fonte2

A_1	A_2	A_3	A_4	Fonte1
B_1	B_2	B_3	B_4	Fonte2
=				
$A_1B_1+A_2B_2$		$A_3B_3+A_4B_4$		Resultado1

Multiplicação-Soma da Fonte3, Fonte4

A_5	A_6	A_7	A_8	Fonte3

B_5	B_6	B_7	B_8	Fonte4
=				
$A_5B_5+A_6B_6$		$A_7B_7+A_8B_8$		Resultado2

Soma em pacote do Resultado1, Resultado2

$A_1B_1+A_2B_2$	$A_3B_3+A_4B_4$	Resultado1
$A_5B_5+A_6B_6$	$A_7B_7+A_8B_8$	Resultado2
=		
$A_1B_1+A_2B_2+ A_5B_5+A_6B_6$	$A_3B_3+A_4B_4+ A_7B_7+A_8B_8$	Resultado3

Desempacotamento superior do Resultado3,Fonte5

$A_1B_1+A_2B_2+ A_5B_5+A_6B_6$	$A_3B_3+A_4B_4+ A_7B_7+A_8B_8$	Resultado3
0	0	Fonte5
=		
0	$A_1B_1+A_2B_2+ A_5B_5+A_6B_6$	Resultado4

Desempacotamento inferior do Resultado3,Fonte5

$A_1B_1+A_2B_2+ A_5B_5+A_6B_6$	$A_3B_3+A_4B_4+ A_7B_7+A_8B_8$	Resultado3
0	0	Fonte5
=		
0	$A_3B_3+A_4B_4+ A_5B_5+A_6B_6$	Resultado5

Soma em pacote do Resultado4, Resultado5 Resultado4

0	$A_1B_1+A_2B_2+ A_5B_5+A_6B_6$	Resultado4
0	$A_3B_3+A_4B_4+ A_7B_7+A_8B_8$	Resultado5
=		
0	TOTAL	Resultado6

5 Tabela 44

Como outro exemplo, a Tabela 45 mostra a multiplicação e o acúmulo dos conjuntos A e B e conjuntos C e D, onde cada um destes conjuntos inclui 2 elementos de dados.

Multiplicação-soma da Fonte1, Fonte2

A ₁	A ₂	C ₁	C ₂	Fonte1
B ₁	B ₂	D ₁	D ₂	Fonte2
=				
A ₁ B ₁ +A ₂ B ₂		C ₁ D ₁ +C ₂ D ₂		Resultado1

Tabela 45

- Como outro exemplo, a Tabela 46 mostra a multiplicação e o acúmulo dos conjuntos A e B e dos conjuntos C e D, onde cada um destes conjuntos inclui 4 elementos de dados.

Multiplicação-soma da Fonte1, Fonte2

A ₁	A ₂	C ₁	C ₂	Fonte1
B ₁	B ₂	D ₁	D ₂	Fonte2
=				
A ₁ B ₁ +A ₂ B ₂		C ₁ D ₁ +C ₂ D ₂		Resultado1

Multiplicação-soma da Fonte3, Fonte4

A ₃	A ₄	C ₃	C ₄	Fonte3
B ₃	B ₄	D ₃	D ₄	Fonte4
=				
A ₃ B ₃ +A ₄ B ₄		C ₃ D ₃ +C ₄ D ₄		Resultado2

Soma em pacote do Resultado1, Resultado2

A ₁ B ₁ +A ₂ B ₂		C ₁ D ₁ +C ₂ D ₂		Resultado1
A ₃ B ₃ +A ₄ B ₄		C ₃ D ₃ +C ₄ D ₄		Resultado2
=				
A ₁ B ₁ +A ₂ B ₂ + A ₃ B ₃ +A ₄ B ₄		C ₁ D ₁ +C ₂ D ₂ + C ₃ D ₃ +C ₄ D ₄		Resultado6

Tabela 46

10 3) Algoritmos de Produto de Pontos

O produto de pontos (também denominado de produto interno) é

usado no processamento de sinal e nas operações de matriz. Por exemplo, o produto de pontos é usado quando da computação do produto de matrizes, da filtragem digital de operações (tais a filtragem FIR e IIR), e da computação das seqüências de correlação. Uma vez que muitos algoritmos de compressão de fala (por exemplo, GSM, G,728, CELP e VSELP) e algoritmos de compressão de Alta Fidelidade (por exemplo, MPEG e codificação de subbanda) fazem uso extenso da filtragem digital e computações de correlação, intensificando o desempenho do produto de pontos, o desempenho destes algoritmos será também intensificado.

10 O produto de pontos de duas seqüências A e B de comprimento N é definido como:

$$\text{Resultado} = \sum_{i=0}^{N-1} A_i \cdot B_i$$

15 A execução de um cálculo de produto de pontos torna extenso o uso da operação de acúmulo de multiplicação, onde elementos correspondentes de cada uma das seqüências são multiplicados entre si, e os resultados são acumulados para formar o resultado de produto de pontos.

20 Com a inclusão das operações de movimento, de soma em pacote, de multiplicação-soma, e de deslocamento em pacote, a invenção permite que o cálculo do produto de pontos seja executado com o uso de dados em pacote. Por exemplo, se o tipo de dados em pacote contendo quatro elementos de dezesseis bits for usado, o cálculo do produto de pontos pode ser executado em duas seqüências, cada qual contendo quatro
25 valores através:

- 1) do acesso dos quatro valores de dezesseis bits originários da seqüência A para gerarem a Fonte1 com o uso de uma instrução de movimento;
 - 2) do acesso dos quatro valores de dezesseis bits originários da seqüência B para gerarem a Fonte2 com o uso de uma instrução de movimento; e
 - 3) da multiplicação e do acúmulo, conforme anteriormente descrito com o
- 30

uso de instruções de multiplicação-soma, soma em pacote e deslocamento.

Para os vetores com mais do que apenas alguns elementos, o processo mostrado na Tabela 46 é usado e os resultados finais são somados entre si no final. Outras instruções de suporte incluem as instruções em pacote OR e XOR para inicializar o registro acumulador, a instrução de deslocamento em pacote para deslocar os valores indesejados no estágio final da computação. As operações de controle de circuito são conseguidas com o uso de instruções já existentes no conjunto de instruções do processador

5
10

4) Filtro de Circuito Bidimensional

Filtros de circuito bidimensionais são usados em certos algoritmos de múltiplos meios. Por exemplo, os coeficientes de filtro mostrados abaixo na Tabela 47 podem ser usados nos algoritmos de conferência de vídeo para executar um filtro passa-baixo nos dados de pixel.

15

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Tabela 47

Para calcular o novo valor de um pixel em localização (x,y), é usada a seguinte equação:

$$\begin{aligned} \text{Pixel Resultante} &= (x-1,y-1) + 2(x,y-1) + (x+1,y-1) + 2(x-1,y) + 4(x,y) + \\ &20 \quad 2(x+1,y) \\ &+ \\ &(x-1,y+1) + 2(x,y+1) + (x+1,y+1) \end{aligned}$$

Com a inclusão de empacotamento, desempacotamento, movimento, deslocamento em pacote, e uma soma em pacote, a invenção permite que um filtro de circuito bidimensional seja executado com o uso de dados em pacote. De acordo com uma implementação do filtro de circuito anteriormente descrito, este filtro de circuito é aplicado como dois filtros unidimensionais simples; isto é, o filtro bidimensional acima pode ser aplicado como dois filtros 121. O primeiro filtro se encontra na direção horizontal, en-

25

quanto o segundo filtro encontra-se na direção vertical.

A Tabela 48 mostra uma representação de um bloco de 8x8 de dados de pixel.

←8→							
A ₀	A ₁	A ₂	A ₃	A ₄	A ₅	A ₆	A ₇
B ₀	B ₁	B ₂	B ₃	B ₄	B ₅	B ₆	B ₇
C ₀	C ₁	C ₂	C ₃	C ₄	C ₅	C ₆	C ₇
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
I ₀	I ₁	I ₂	I ₃	I ₄	I ₅	I ₆	I ₇

Tabela 48

- 5 As seguintes etapas são executadas para implementar a passagem horizontal do filtro neste bloco de 8x8 de dados de pixel:
- 1) acessar oito valores de oito bits como dados em pacote com o uso de uma instrução de movimento;
 - 2) desempacotar os oito pixels de oito bits em dados em pacote de 16 bits contendo quatro pixels de 8 bits (Fonte1) para manter a precisão durante os acúmulos;
 - 3) duplicar a Fonte1 duas vezes para gerar a Fonte2 e Fonte3;
 - 4) executar um deslocamento desempacotado para a direita através de 16 bits na Fonte1;
 - 15 5) executar um deslocamento desempacotado para a esquerda através de 16 bits na Fonte3;
 - 6) gerar $(\text{Fonte1} + 2 \cdot \text{Fonte2} + \text{Fonte3})$ através da execução das seguintes somas em pacote:
 - a) $\text{Fonte1} + \text{Fonte 1} + \text{Fonte2}$,
 - 20 b) $\text{Fonte 1} + \text{Fonte1} + \text{Fonte2}$,
 - c) $\text{Fonte1} + \text{Fonte1} + \text{Fonte3}$;
 - 7) armazenar os dados de palavra em pacote resultantes como

parte de uma disposição de resultado intermediária de 8x8; e

8) repetir estas etapas até que toda a disposição de resultado intermediária de 8x8 seja gerada, conforme mostrado na Tabela 49 abaixo (por exemplo, IA₀ representa o resultado intermediário para A₀ originário da Tabela 49).

←16→

IA ₀	IA ₁	IA ₂	IA ₃	IA ₄	IA ₅	IA ₆	IA ₇
IB ₀	IB ₁	IB ₂	IB ₃	IB ₄	IB ₅	IB ₆	IB ₇
IC ₀	IC ₁	IC ₂	IC ₃	IC ₄	IC ₅	IC ₆	IC ₇
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
II ₀	II ₁	II ₂	II ₃	II ₄	II ₅	II ₆	II ₇

Tabela 49

As seguintes etapas são executadas para implementarem a passagem vertical do filtro na disposição de resultado intermediário de 8x8:

- 1) acessar um bloco de dados de 4x4 a partir da disposição de resultado intermediário como dados em pacote usando uma instrução de movimento para gerar a Fonte1, Fonte2, e Fonte3 (por exemplo, vide Tabela 50, para um exemplo);

←16→

IA ₀	IA ₁	IA ₂	IA ₃	Fonte 1
IB ₀	IB ₁	IB ₂	IB ₃	Fonte 2
IC ₀	IC ₁	IC ₂	IC ₃	Fonte 3

Tabela 50

- 2) gerar (Fonte1 + 2*Fonte2 + Fonte3) através da execução das seguintes somas em pacote:

- a) Fonte1 = Fonte1 + Fonte2,
b) Fonte1 = Fonte1 + Fonte2,

c) $\text{Fonte1} = \text{Fonte1} + \text{Fonte3}$;

3) executar um deslocamento em pacote para a direita através de 4 bits na Fonte1 resultante para gerar a soma dos pesos -- isto é, efetivamente dividindo por 16;

5 4) empacotar a Fonte1 resultante com a saturação para converter os valores de 16 bits de volta para os valores de pixel de 8 bits;

6) armazenar os dados de byte em pacote resultantes como parte de uma disposição de resultado de 8x8 (com relação ao exemplo mostrado na Tabela 50, estes quatro bytes representam os novos valores de pixel para B_0 , B_1 , B_2 e B_3); e

10 7) repetir estas etapas até que toda a disposição de resultado de 8x8 seja gerada.

Vale por enquanto notar que as fileiras de topo e de base da disposição de resultado de 8x8 são determinadas com o uso de um algoritmo diferente que não está descrito aqui, de modo que não obscureça a invenção.

Desse modo, com a provisão no processador 109 das instruções de empacotamento, desempacotamento, movimento, deslocamento em pacote, e soma em pacote, a invenção apresenta um aumento significativo no desempenho sobre os processadores de finalidades gerais da técnica anterior, os quais têm que executar as operações exigidas por tais elementos de dados dos filtros 1 de uma vez.

5) Estimativa de Movimento

A estimativa de movimento é usada nas diversas aplicações de múltiplos meios (por exemplo, conferência de vídeo e MPEG (repetição de disco de vídeo de alta qualidade)). Em relação à conferência de vídeo, a estimativa de movimento é usada para reduzir a quantidade de dados que tem que ser transmitida entre os terminais. A estimativa de movimento trabalha com a divisão dos quadros de vídeo em blocos de vídeo de tamanho fixo. Para cada bloco no Quadro1, é determinado se há um bloco contendo uma imagem similar no Quadro2. Se tal bloco estiver contido no Quadro2, esse bloco pode ser descrito com uma referência de vetor de movimento no Qua-

dro1. Desse modo, além de transmitir todos os dados que representam esse bloco, apenas um vetor de movimento precisa ser transmitido ao terminal de recebimento. Por exemplo, se um bloco no Quadro1 for similar e na mesma tela coordenadas como um bloco no Quadro2, apenas um vetor de movimento de 0 precisa ser enviado para esse bloco. Contudo, se um bloco no Quadro1 for similar a um bloco no Quadro2, mas em diferentes coordenadas de tela do que um bloco no Quadro2, apenas um vetor de movimento que indica a nova localização desse bloco precisará ser enviado. De acordo com uma implementação, para determinar se um blocoA no Quadro1 é similar a um bloco B no Quadro2, será determinada a soma das diferenças absolutas entre os valores de pixel. Quanto mais inferior a soma, mais similar o bloco A é do bloco B (isto é, se a soma for 0), o bloco A será idêntico ao bloco B).

Com a inclusão das operações de movimento, de desempacotamento, de soma em pacote, de subtração em pacote com saturação, e operações lógicas, a invenção permite que a estimativa de movimento seja executada com o uso de dados em pacote. Por exemplo, se dois blocos de vídeo de 16x16 forem representados pelas duas disposições de valores de pixel de 8 bits armazenados como dados em pacote, a diferença absoluta dos valores de pixel nestes blocos pode ser calculada:

1) acessando oito valores de 8 bits originários do bloco A para gerarem a Fonte1 com o uso de uma instrução de movimento;

2) acessando oito valores de 8 bits originários do bloco B para gerarem a Fonte2 com o uso de uma instrução de movimento;

3) executando uma subtração em pacote com saturação para subtrair a Fonte1 da Fonte2 gerando a Fonte3-- Através da subtração com saturação, a Fonte 3 irá conter apenas os resultados positivos desta subtração (isto é, os resultados negativos serão zerados);

4) executando uma subtração em pacote com saturação para subtrair a Fonte2 da Fonte1 gerando a Fonte 4-- Com a subtração com saturação, a Fonte 4 irá conter apenas os resultados positivos desta subtração (isto é, os resultados negativos serão zerados);

5) executando uma operação OR em pacote na Fonte3 e Fonte4

para produzirem a Fonte5. Com a execução desta operação OR, a Fonte5 contém o valor absoluto da Fonte1 e Fonte2;

6) repetindo estas etapas até que os blocos de 16x16 tenham sido processados.

5 Os valores absolutos de 8 bits resultantes são desempacotados em elementos de dados de 16 bits para permitir a precisão de 16 bits, e depois somados com o uso de somas em pacote.

10 Dessa maneira, com a provisão no processador 109 das operações de movimento, de desempacotamento, de soma em pacote, de subtração em pacote com saturação e operações lógicas, a invenção proporciona um aumento significativo do desempenho sobre os processadores de finalidades gerais da técnica anterior que têm que executar as adições e as diferenças absolutas do cálculo de estimativa de movimento de um elemento de dados de cada vez.

15 6) Transformação Discreta de Cosseno

A Transformação Discreta de Cosseno (DCT) é uma função bem conhecida usada em muitos algoritmos de processamento de sinal. Os algoritmos de compressão de imagem e vídeo, em particular, tornam o uso desta transformação extensa.

20 Nos algoritmos de compressão de imagem e vídeo, DCT (Transformação Discreta de Cosseno) é usada para transportar um bloco de pixels da representação espacial para a representação de frequência. Na representação de frequência, a informação de modelo é dividida em componentes de frequência, alguns dos quais são mais importantes do que outros.

25 O algoritmo de compressão seletivamente quantiza ou descarta os componentes de frequência que não afetam contrariamente os conteúdos de modelo reconstruído. Desta maneira, é alcançada a compressão.

Há muitas implementações de DCT (Transformação Discreta de Cosseno), a mais popular sendo algum tipo de processo de transformação rápida modelado com base no fluxo de computação de Transformação Rápida de Fourier (FFT). Na transformação rápida, uma transformação de ordem

30 N é trazida até uma combinação de transformações de ordem $N/2$ e o resul-

tado combinado. Esta decomposição pode ser executada até que a transformação de ordem 2 menor seja alcançada. Este núcleo de transformação elementar 2 é freqüentemente mencionado como operação borboleta. A operação borboleta é expressa como segue:

$$\begin{aligned} X &= a*x+b*y \\ Y &= c*x-d*y \end{aligned}$$

onde, a, b, c e d são denominados de coeficiente, x e y são os dados de entrada, e X e Y são as saídas de transformação.

Com a inclusão das operações de movimento, multiplicação-soma, e de deslocamento em pacote, a invenção permite que o cálculo da DCT (Transformação Discreta de Cosseno) seja executado com o uso de dados em pacote, da seguinte maneira:

1) acessando os dois valores de 16 bits que representam x e y para gerar a Fonte1 (vide Tabela 51 abaixo) com o uso de instruções de movimento e de desempacotamento;

2) gerando a Fonte2, conforme mostrado na Tabela 51 abaixo. Deve ser notado que a Fonte2 pode ser reutilizada sobre diversas operações de borboleta; e

3) executando uma instrução de multiplicação-soma com o uso da Fonte1 e da Fonte2 para gerar o Resultado (vide Tabela 51 abaixo).

x	y	x	y	Fonte 1
---	---	---	---	---------

a	b	c	-d	Fonte 2
---	---	---	----	---------

$a \square x + b \square y$	$c \square x - d \square y$	Fonte 3
-----------------------------	-----------------------------	---------

Tabela 51

Em algumas situações, os coeficientes da operação borboleta é 1. Para estes casos, a operação borboleta degenera-se simplesmente em somas e subtrações que possam ser executadas com o uso de instruções de soma em pacote e de subtração em pacote.

Um documento do IEEE (Instituto de Engenheiros Elétricos e Eletrônicos) especifica a precisão com a qual a DCT (Transformação Dis-

creta de Cosseno) inversa deve ser executada para a conferência de vídeo. (Vide, Sociedade de Sistemas e Circuitos do IEEE, "Especificações Padrão do IEEE para as Implementações de Transformação Discreta de Cosseno Inversa de 8x8", Std. IEEE 1180-1990, IEEE Inc. 345 East 47th St., NY, NY 10017, USA, de 18 de março de 1991). A precisão exigida é encontrada através da instrução de multiplicação-soma descrita, porque ela usa entradas de 16 bits para gerar saídas de 32 bits.

Dessa maneira, com a provisão no processador 109 de operações de movimento, de multiplicação-soma e de deslocamento em pacote, a invenção apresenta um significativo aumento no desempenho sobre os processadores de finalidades gerais da técnica anterior, que têm que executar as adições e multiplicações do cálculo de DCT (Transformação Discreta de Cosseno) de um elemento de dados de cada vez.

Concretizações Alternativas

15 Enquanto a invenção foi descrita, na qual cada uma das diferentes operações possui circuitos separados, concretizações alternativas poderiam ser implementadas, de tal forma que certos circuitos fossem compartilhados pelas diferentes operações. Por exemplo, em uma concretização, são usados os seguintes circuitos: 1) uma única unidade lógica aritmética (ALU) para executar as operações de soma em pacote, de subtração em 20 pacote, de comparação em pacote e lógica em pacote; 2) uma unidade de circuitos para executar as operações de empacotamento, desempacotamento e de deslocamento em pacote; 3) uma unidade de circuitos para executar as operação de multiplicação em pacote e de multiplicação-soma; e 4) 25 uma unidade de circuitos para executar a operação de contagem de ocupação.

Os respectivos termos correspondentes são usados aqui para fazer referência à relação predeterminada entre os elementos de dados armazenados em um ou mais dados em pacote. Em uma concretização, esta 30 relação baseia-se nas posições de bit dos elementos de dados nos dados em pacote. Por exemplo, o elemento de dados 0 (por exemplo, armazenado nas posições de bit 0-7 no formato de byte em pacote) de primeiros dados

em pacote corresponde aos elementos de dados 0 (por exemplo, armazenados nas posições de bit 0-7 no formato de byte em pacote) de segundos dados em pacote. Entretanto, esta relação pode diferir nas concretizações alternativas. Por exemplo, elementos de dados correspondentes nos primeiro e segundo dados em pacote podem ser de diferentes tamanhos. Como outro exemplo, além do elemento de dados menos significativo dos primeiros dados em pacote que correspondem ao elemento de dados menos significativo dos segundos dados em pacote (e assim por diante), os elementos de dados nos primeiros e segundos dados em pacote podem se corresponder em alguma outra ordem. Como outro exemplo, ao invés de se ter uma correspondência de 1 para 1 dos elementos de dados nos primeiros e segundos dados em pacote, os elementos de dados podem se corresponder em uma proporção diferente (por exemplo, os primeiros dados em pacote podem ter um ou mais elementos de dados que correspondem a um ou mais elementos de dados diferentes nos segundos dados em pacote).

Enquanto a invenção foi descrita em termos das diversas concretizações, aqueles versados na técnica irão reconhecer que a invenção não é limitada às concretizações descritas. O processo e aparelho da invenção podem ser praticados com modificações e alterações que estejam dentro do espírito e escopo das reivindicações anexas. A descrição deve, dessa maneira, ser considerada ilustrativa e não como limitando a invenção.

REIVINDICAÇÕES

1. Método para realizar operações de multiplicação-adição em dados em pacote compreendendo as etapas de:

5 receber uma instrução que especifica as localizações de um primeiro dado em pacote possuindo uma primeira pluralidade de elementos de dados e um segundo dado em pacote possuindo uma segunda pluralidade de elementos de dados, cada um dos elementos de dados do primeiro dado em pacote correspondendo a um elemento de dado diferente do segundo dado em pacote; e

10 gerar um resultado de dado em pacote pela execução da instrução, **caracterizado pelo fato de que** o dado em pacote resultante possui uma pluralidade de elementos de dados resultantes, cada um dos elementos de dados resultantes compreendendo um somatório de produtos dos elementos de dados correspondentes ao primeiro dado em pacote e ao segundo dado em pacote.

15

2. Método, de acordo com a reivindicação 1, **caracterizado pelo fato de que** ainda compreende a etapa de armazenar o resultado de dados em pacote sobre os primeiros dados em pacote ou sobre os segundos dados em pacote.

20 3. Método, de acordo com a reivindicação 1 ou 2, **caracterizado pelo fato de que** cada um dos elementos de dados resultantes possui uma precisão maior do que os elementos de dados dos primeiros dados em pacote.

25 4. Método, de acordo com qualquer uma das reivindicações 1 a 3, **caracterizado pelo fato de que** os elementos de dados dos primeiro e segundo dados em pacote compreendem elementos de dados de palavras em pacote, e em que os elementos de dados resultantes compreendem elementos de dados de palavras duplas em pacote.

30 5. Método, de acordo com qualquer uma das reivindicações 1 a 4, **caracterizado pelo fato de que** ainda compreende a etapa de completar a execução da instrução sem somar a pluralidade de elementos de dados resultantes.

6. Método, de acordo com qualquer uma das reivindicações 1 a 5, **caracterizado pelo fato de que** ainda compreende as etapas de:

acessar o segundo dado em pacote a partir de uma localização especificada pelos bits zero a dois da instrução;

5 acessar o primeiro dado em pacote a partir de uma localização especificada pelos bits três a cinco da instrução;

armazenar o dado em pacote resultante em um destino especificado pelos bits três a cinco da instrução.

7. Método, de acordo com qualquer uma das reivindicações 1 a 10 **6, caracterizado pelo fato de que** o primeiro dado em pacote possui um A_1 , um A_2 , um A_3 , e um A_4 como elementos de dados, o segundo pacote de dados possui um B_1 , um B_2 , um B_3 e um B_4 como elementos de dados e a geração do dado em pacote resultante compreende gerar um primeiro elemento de dados resultante incluindo $(A_1 \times B_1 + A_2 \times B_2)$, e um segundo elemento de dados resultante incluindo $(A_3 \times B_3 + A_4 \times B_4)$.

8. Aparelho (109) para realizar operações de multiplicação-adição em dados em pacote compreendendo:

um decodificador (165) para receber e decodificar uma instrução que especifica localizações (1531,1533) de um primeiro dado em pacote possuindo uma primeira pluralidade de elementos de dados e um segundo dado em pacote possuindo uma segunda pluralidade de elementos de dados, cada um dos elementos de dados do primeiro dado em pacote correspondendo a um elemento de dados diferente do segundo dado em pacote; e
20 uma unidade de execução (130) em comunicação com o decodificador (165) para gerar um dado em pacote resultante em resposta ao decodificador (165) decodificando a instrução, **caracterizado pelo fato de que** o dado em pacote resultante tem uma pluralidade de elementos de dados resultantes, cada um dos elementos de dados resultantes compreendendo um somatório de produtos dos elementos de dados correspondentes dos
25 primeiro e segundo dados em pacote.
30

9. Aparelho, de acordo com a reivindicação 8, **caracterizado pelo fato de que** a unidade de execução (130) armazena os dados em pa-

cote resultantes sobre os primeiro ou segundo dados em pacote.

5 10. Aparelho, de acordo com a reivindicação 8 ou 9, **caracterizado pelo fato de que** a unidade de execução (130) gera elementos de dados resultantes que possuem uma precisão maior que os elementos de dados do primeiro dado em pacote.

11. Aparelho, de acordo com qualquer uma das reivindicações 8 a 10, **caracterizado pelo fato de que** a unidade de execução (130) completa a execução da instrução sem somar a pluralidade de elementos de dados resultantes.

10 12. Aparelho, de acordo com qualquer uma das reivindicações 8 a 11, **caracterizado pelo fato de que** o decodificador (165) decodifica uma instrução tendo um formato de instrução de 24 bits, em que os bits zero até dois da instrução especificam uma localização do segundo dado em pacote e em que os bits três até cinco da instrução especificam uma locação do primeiro dado em pacote.

15 13. Aparelho, de acordo com qualquer uma das reivindicações 8 a 12, **caracterizado pelo fato de que** a unidade de execução (130) gera um dado em pacote resultante tendo um elemento de dados resultante compreendendo um somatório de dois produtos de elementos de dados correspondentes aos primeiro e segundo dados em pacote.

20 14. Aparelho, de acordo com qualquer uma das reivindicações 8 a 13, **caracterizado pelo fato de que** o primeiro dado em pacote possui um A_1 , um A_2 , um A_3 , e um A_4 como elementos de dados, em que o segundo dado em pacote possui um B_1 , um B_2 , um B_3 e um B_4 como elementos de dados e em que a unidade de execução (130) gera um dado em pacote resultante tendo um primeiro elemento de dados resultante incluindo $(A_1 \times B_1 + A_2 \times B_2)$ e um segundo elemento de dados resultante incluindo $(A_3 \times B_3 + A_4 \times B_4)$.

25 15. Aparelho, de acordo com qualquer uma das reivindicações 8 a 14, **caracterizado pelo fato de que** o primeiro dado em pacote possui um A_1 , um A_2 , um A_3 , e um A_4 como elementos de dados, em que o segundo dado em pacote possui um B_1 , um B_2 , um B_3 e um B_4 como elementos de

dados e em que a unidade de execução (130) gera um dado em pacote resultante tendo um ou mais selecionados a partir de um primeiro elemento de dados resultante incluindo $(A_1 \times B_1 - A_2 \times B_2)$ e um segundo elemento de dados resultante incluindo $(A_3 \times B_3 - A_4 \times B_4)$.

5 16. Aparelho, de acordo com a reivindicação 8, **caracterizado pelo fato de que** um dado em pacotes compreende uma unicidade de dados que consiste em uma pluralidade de elementos de dados do mesmo tamanho.

10 17. Aparelho, de acordo com a reivindicação 8, **caracterizado pelo fato de que** um dado em pacotes compreende uma unicidade de dados tendo todos os bits disponíveis usados para uma pluralidade de elementos de dados do mesmo tamanho.

15 18. Aparelho, de acordo com a reivindicação 8, **caracterizado pelo fato de que** um registro usado para armazenar os dados em pacotes é implementado para armazenar um número de bits que é maior do que o número de bits usado para os dados em pacote.

19. Método para realizar operações de multiplicação-adição em dados em pacote compreendendo as etapas de:

20 receber uma primeira instrução, a primeira instrução de um formato de instrução compreendendo um primeiro código operacional (601), um primeiro campo (602) para indicar um primeiro operando tendo uma primeira pluralidade de elementos de dados incluindo pelo menos os elementos de dados A_1, A_2, A_3 e A_4 , e um segundo campo (603) para indicar um segundo operando tendo uma segunda pluralidade de elementos de dados incluindo
25 pelo menos os elementos de dados B_1, B_2, B_3 e B_4 , cada um dos elementos de dados da primeira e da segunda pluralidade de elementos de dados tendo um comprimento de N bits; e

30 armazenar o dado em pacote tendo um comprimento de pelo menos $4N$ bits em um operando de destino arquiteturalmente visível em resposta à primeira instrução **caracterizado pelo fato de que** o dado em pacote é gerado pela realização da operação $(A_1 \times B_1) + (A_2 \times B_2)$ para gerar um primeiro elemento de dado do dado em pacote, e pela realização da opera-

ção $(A_3 \times B_3) + (A_4 \times B_4)$ para gerar um segundo elemento de dados do dado em pacote, cada um dos primeiro e segundo elementos de dados tendo um comprimento de pelo menos $2N$ bits.

5 20. Método, de acordo com a reivindicação 19, **caracterizado pelo fato de que** ainda compreende a etapa de sobregravar a primeira pluralidade de elementos de dados do primeiro operando com o dado em pacote do operando de destino.

10 21. Método, de acordo com a reivindicação 19 ou 20, **caracterizado pelo fato de que** o operando de destino é indicado pelo primeiro campo (602).

22. Método, de acordo com qualquer uma das reivindicações 19 a 21, **caracterizado pelo fato de que** o primeiro campo (602) compreende os bits três a cinco do formato de instrução e em que o segundo campo (603) compreende os bits zero a dois do formato de instrução.

15 23. Aparelho (109) para realizar operações de multiplicação-adição em dados em pacote compreendendo:

um decodificador (165) para decodificar uma instrução tendo um primeiro formato, o primeiro formato sendo operável para identificar um primeiro conjunto de dados em pacote incluindo quatro elementos A_1 , A_2 , A_3 e A_4 , e para identificar um segundo conjunto de dados em pacote incluindo quatro elementos B_1 , B_2 , B_3 e B_4 , o decodificador (165) iniciando um primeiro conjunto de operações no primeiro e no segundo conjunto de dados em pacote em resposta à decodificação da instrução; e

20 uma unidade de execução (130) para realizar um primeiro conjunto de operações iniciado pelo decodificador (165), **caracterizado pelo fato de que** a unidade de execução (130) realiza uma primeira operação para produzir um primeiro resultado incluindo um somatório de produtos, $(A_1 \times B_1) + (A_2 \times B_2)$, e para realizar uma segunda operação do primeiro conjunto de operações iniciado pelo decodificador para produzir um segundo resultado incluindo um somatório de produtos $(A_3 \times B_3) + (A_4 \times B_4)$.

30 24. Aparelho, de acordo com a reivindicação 23, **caracterizado pelo fato de que** a unidade de execução (130) armazena o primeiro e o se-

segundo resultados sobre o primeiro conjunto de dados em pacote ou sobre o segundo conjunto de dados em pacote.

25. Aparelho, de acordo com qualquer uma das reivindicações 23 ou 24, **caracterizado pelo fato de que** a unidade de execução (130) gera um primeiro resultado que possui uma precisão maior do que a dos elementos de dados do primeiro conjunto de dados em pacote.

26. Aparelho, de acordo com qualquer uma das reivindicações 23 a 25, **caracterizado pelo fato de que** a unidade de execução (130) completa a execução da instrução sem somar o primeiro resultado com o segundo resultado.

27. Aparelho, de acordo com qualquer uma das reivindicações 23 a 26, **caracterizado pelo fato de que** o decodificador (165) decodifica uma instrução que possui um formato de 24 bits, em que os bits zero a dois da instrução especificam uma localização do segundo conjunto de dados em pacote (603) e em que os bits três a cinco da instrução especificam uma localização do primeiro conjunto de dados em pacote (602).

28. Aparelho, de acordo com qualquer uma das reivindicações 23 a 27, **caracterizado pelo fato de que** o primeiro conjunto de dados em pacote também inclui mais quatro elementos A_5 , A_6 , A_7 e A_8 , e o segundo conjunto de dados em pacote também inclui mais quatro elementos B_5 , B_6 , B_7 e B_8 , e a unidade de execução (130) realiza uma terceira operação do primeiro conjunto de operações iniciado pelo decodificador (165) para produzir um terceiro resultado incluindo um somatório de produtos $(A_5 \times B_5) + (A_6 \times B_6)$ e para realizar uma quarta operação do primeiro conjunto de operações iniciado pelo decodificador para produzir um quarto resultado incluindo um somatório dos produtos $(A_7 \times B_7) + (A_8 \times B_8)$.

29. Aparelho, de acordo com a reivindicação 23, **caracterizado pelo fato de que** um dado em pacotes compreende uma unicidade de dados que consiste em uma pluralidade de elementos de dados do mesmo tamanho.

30. Aparelho, de acordo com a reivindicação 23, **caracterizado pelo fato de que** um dado em pacotes compreende uma unicidade de da-

dos tendo todos os bits disponíveis usados para uma pluralidade de elementos de dados do mesmo tamanho.

31. Aparelho, de acordo com a reivindicação 23, **caracterizado pelo fato de que** um registro usado para armazenar os dados em pacotes é implementado para armazenar um número de bits que é maior do que o número de bits usado para os dados em pacote.

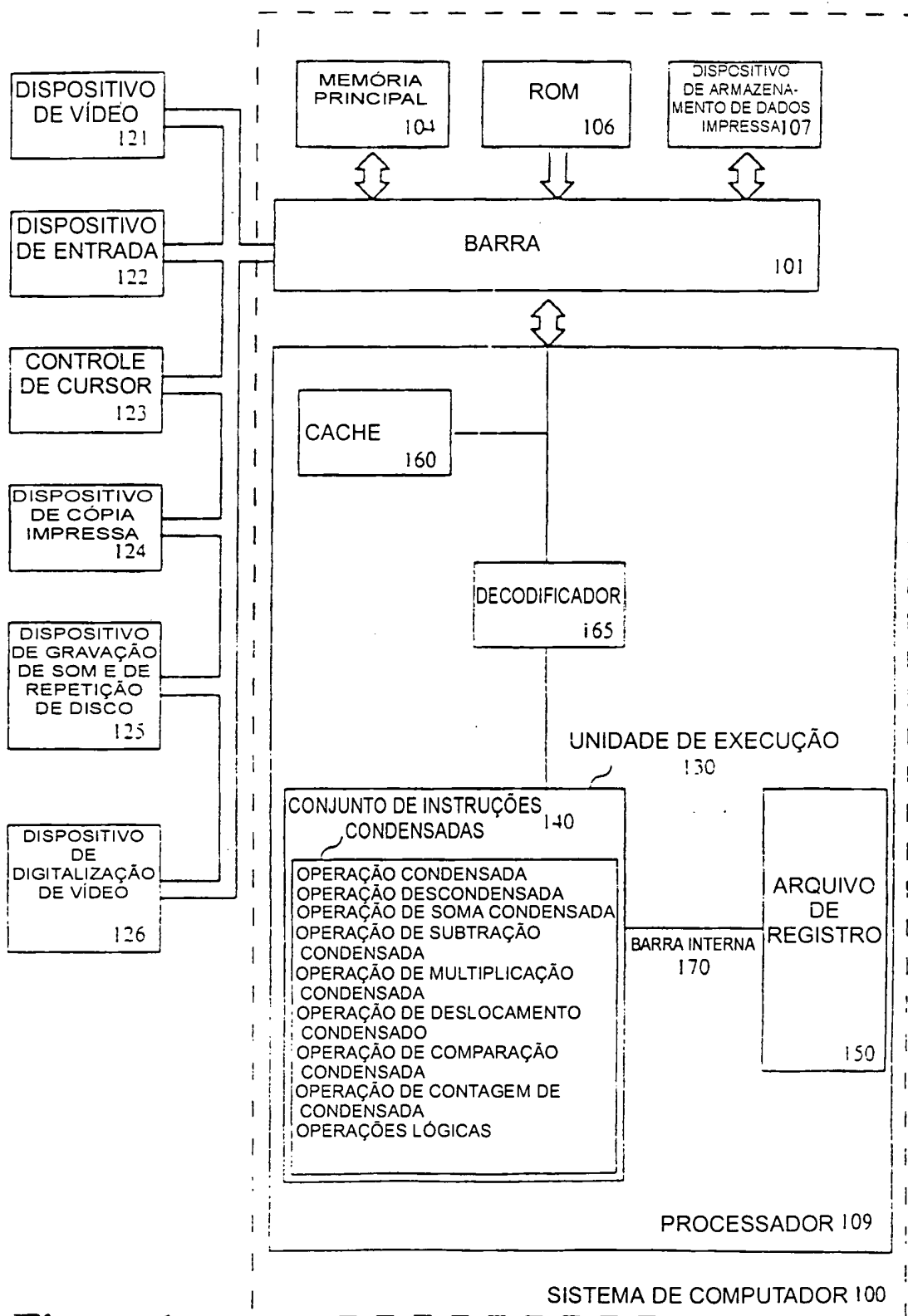


Fig 1

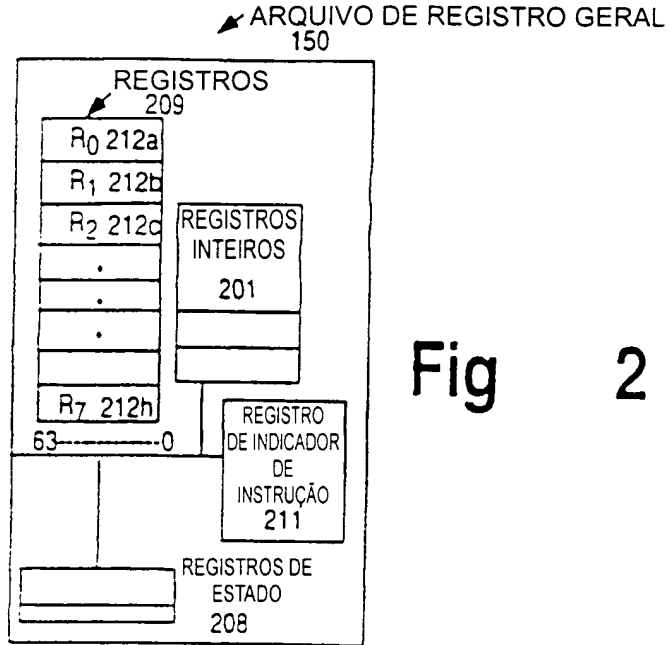


Fig 2

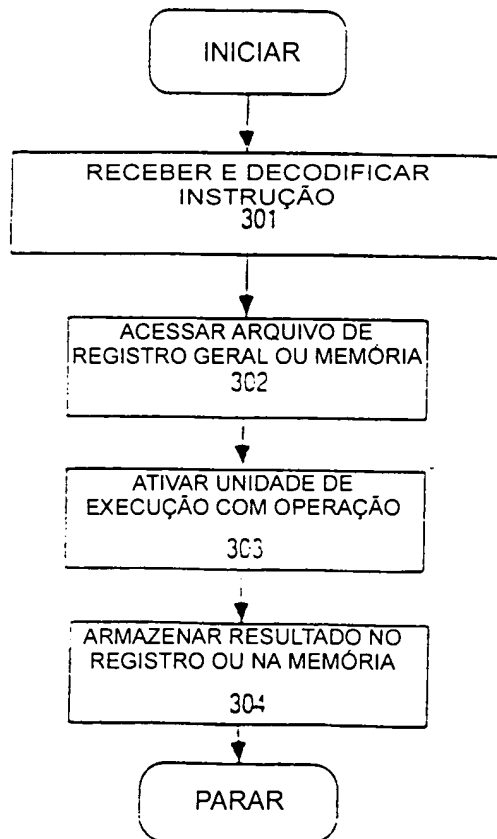


Fig 3

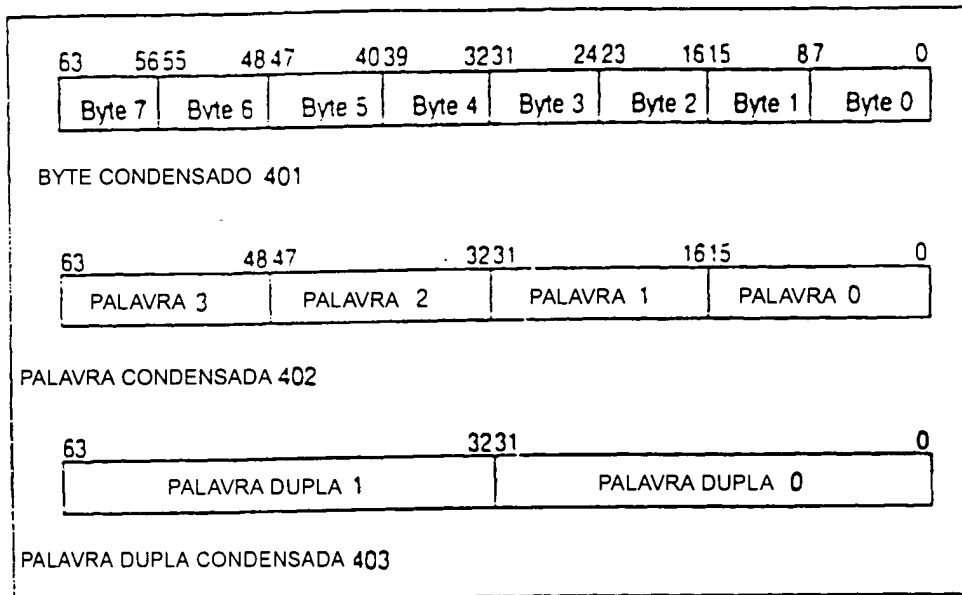


Fig 4

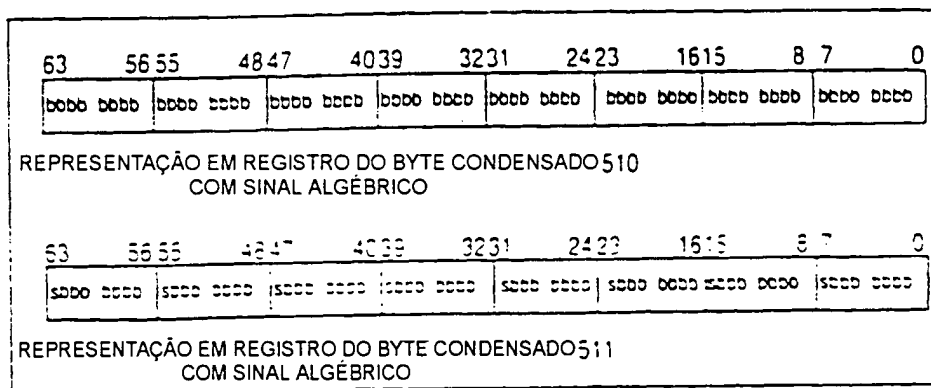


Fig 5a

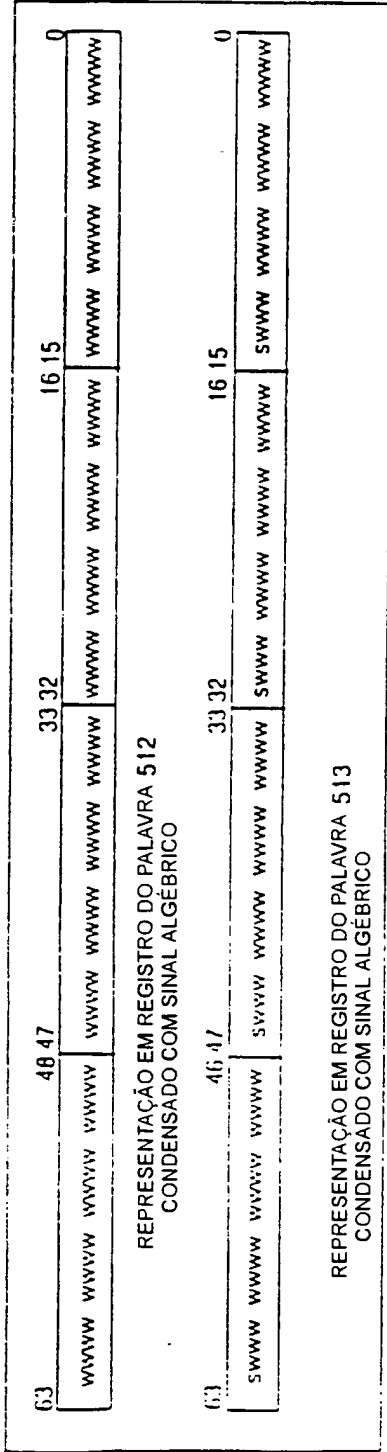


Fig 5b

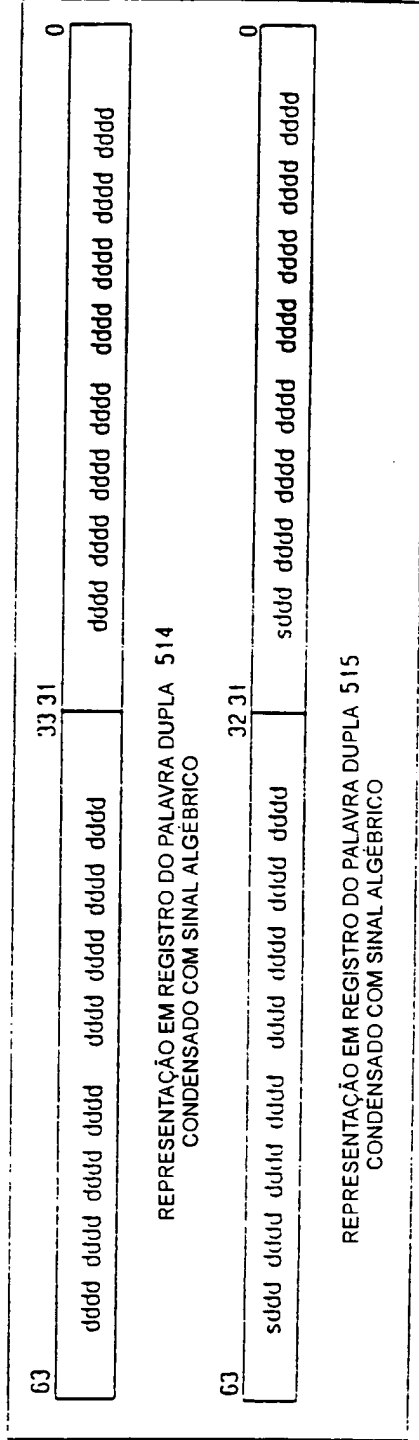


Fig 5c

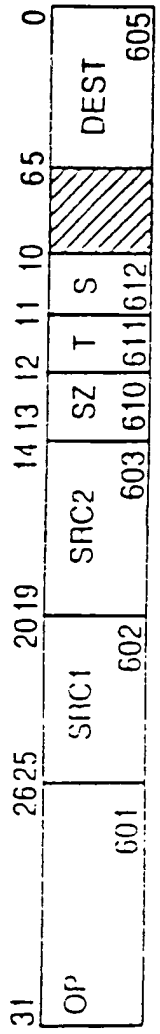


Fig 6a

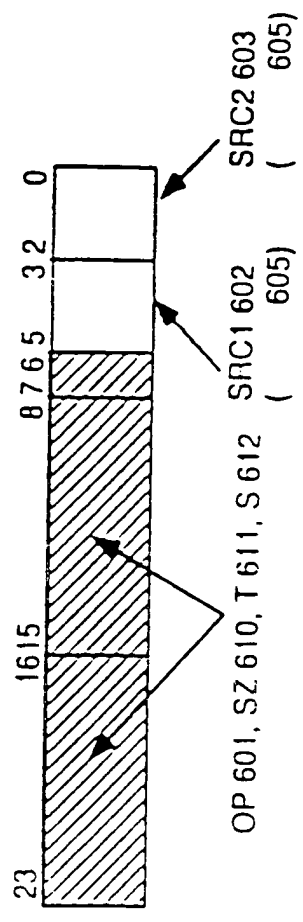


Fig 6b

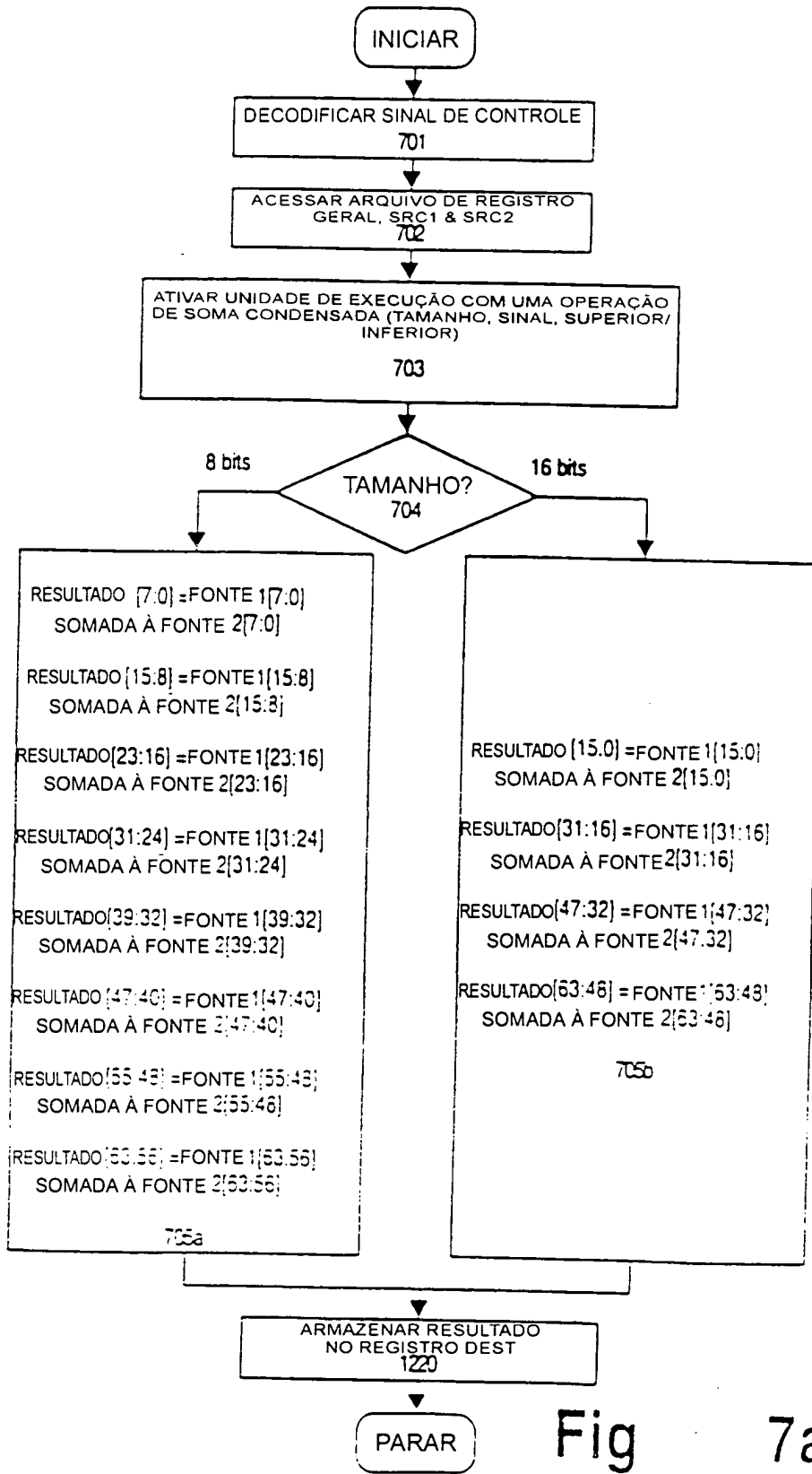


Fig 7a

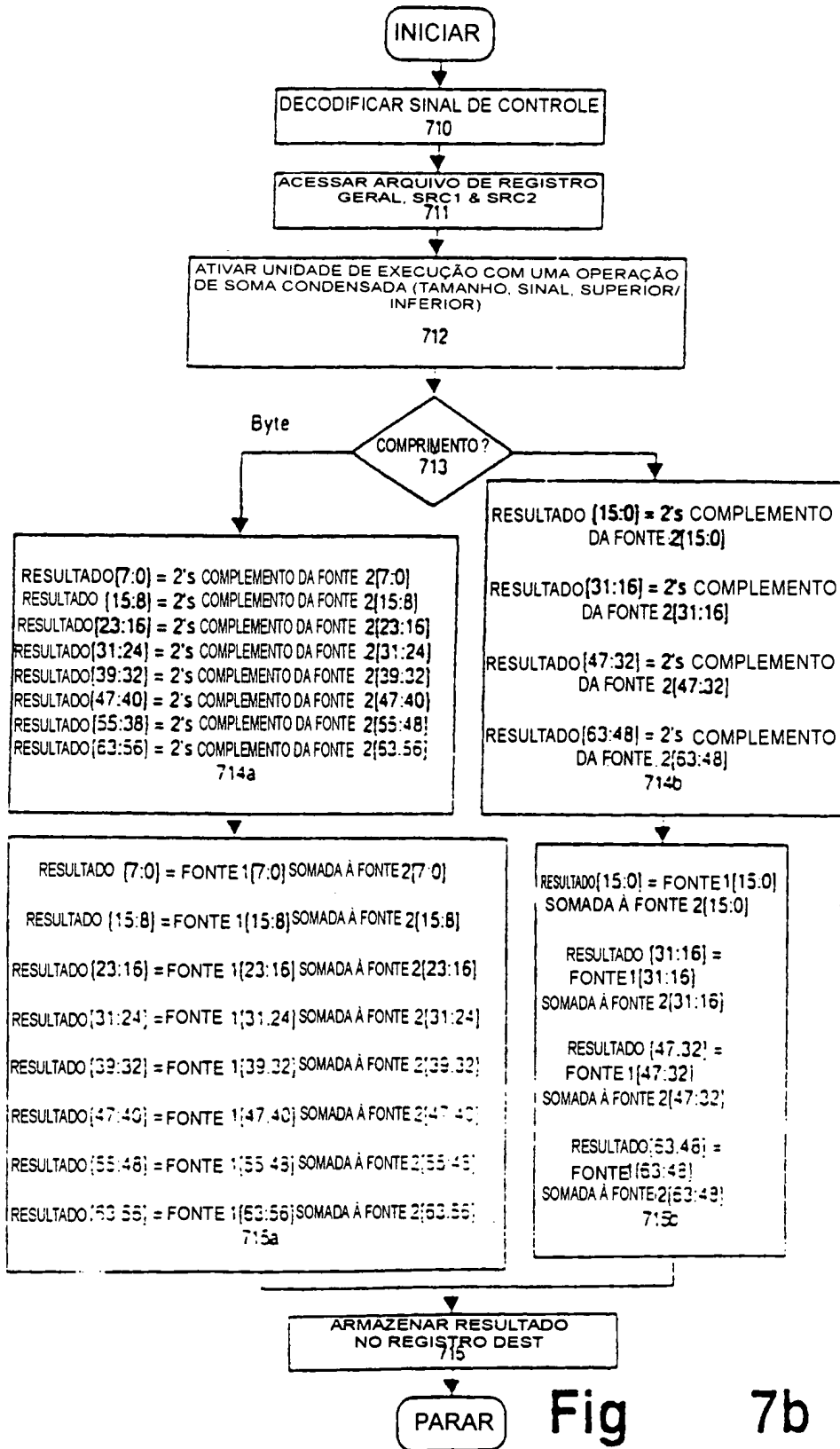


Fig 7b

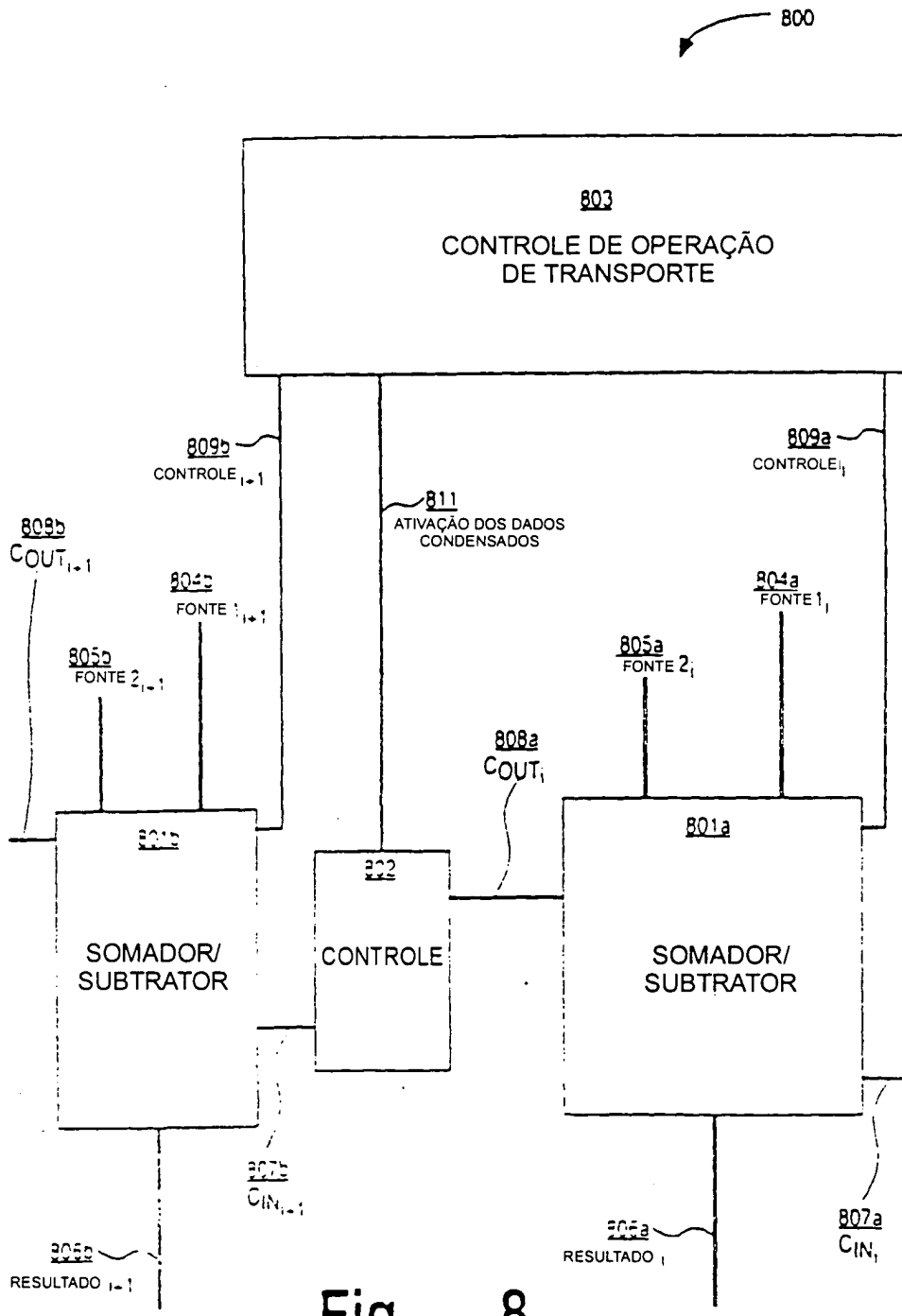


Fig 8

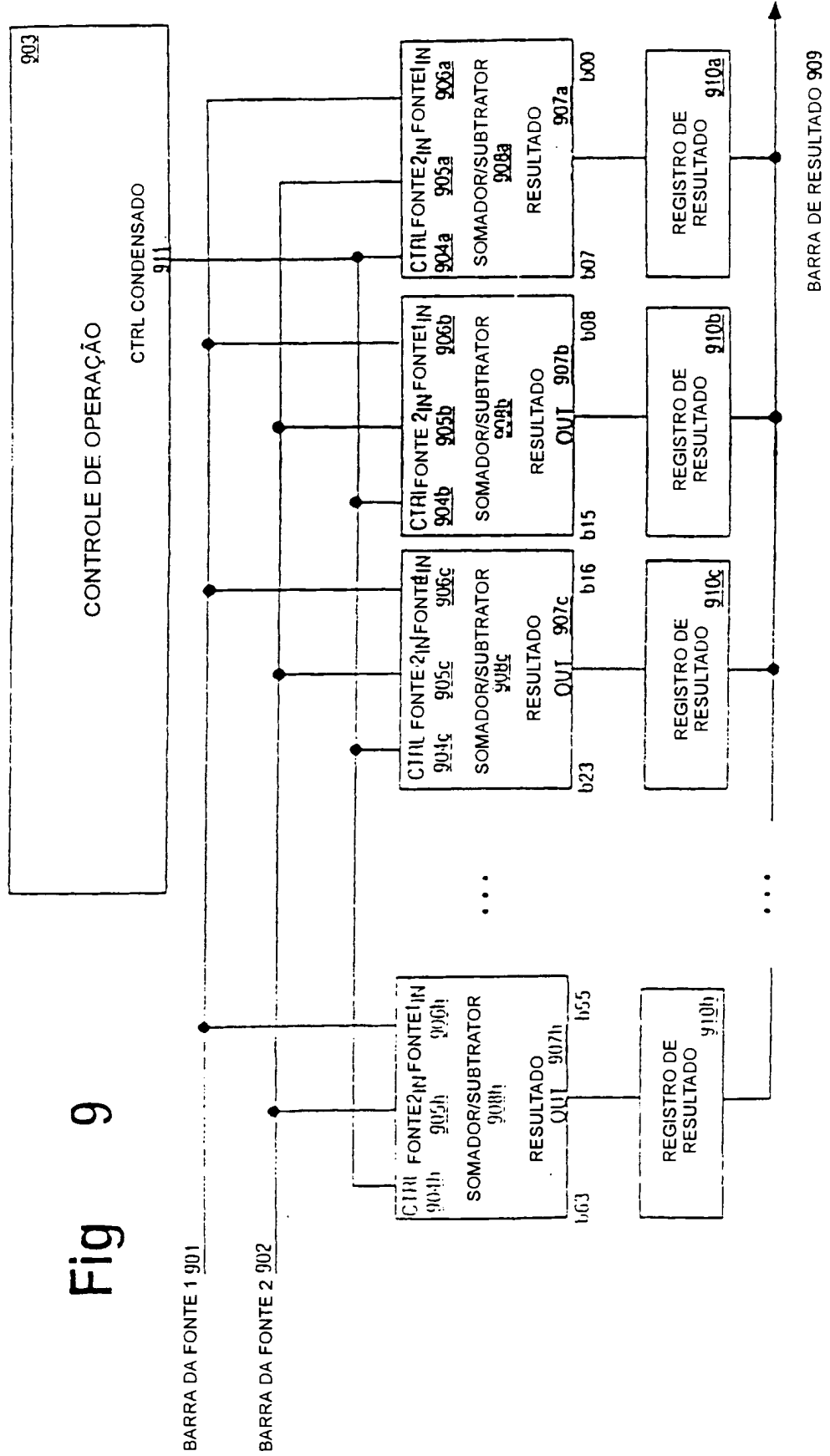


Fig 9

BARRA DA FONTE 1 901

BARRA DA FONTE 2 902

BARRA DE RESULTADO 909

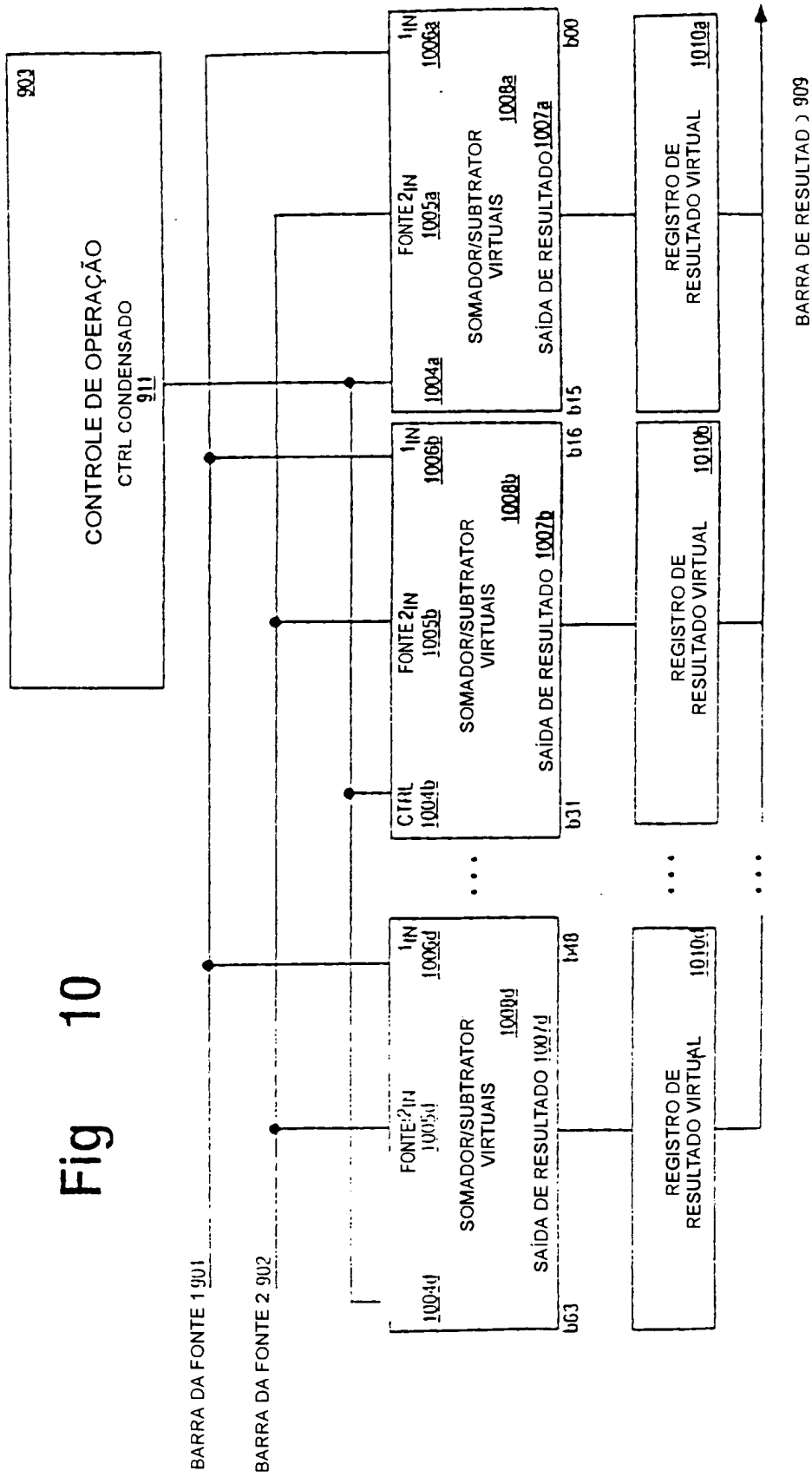
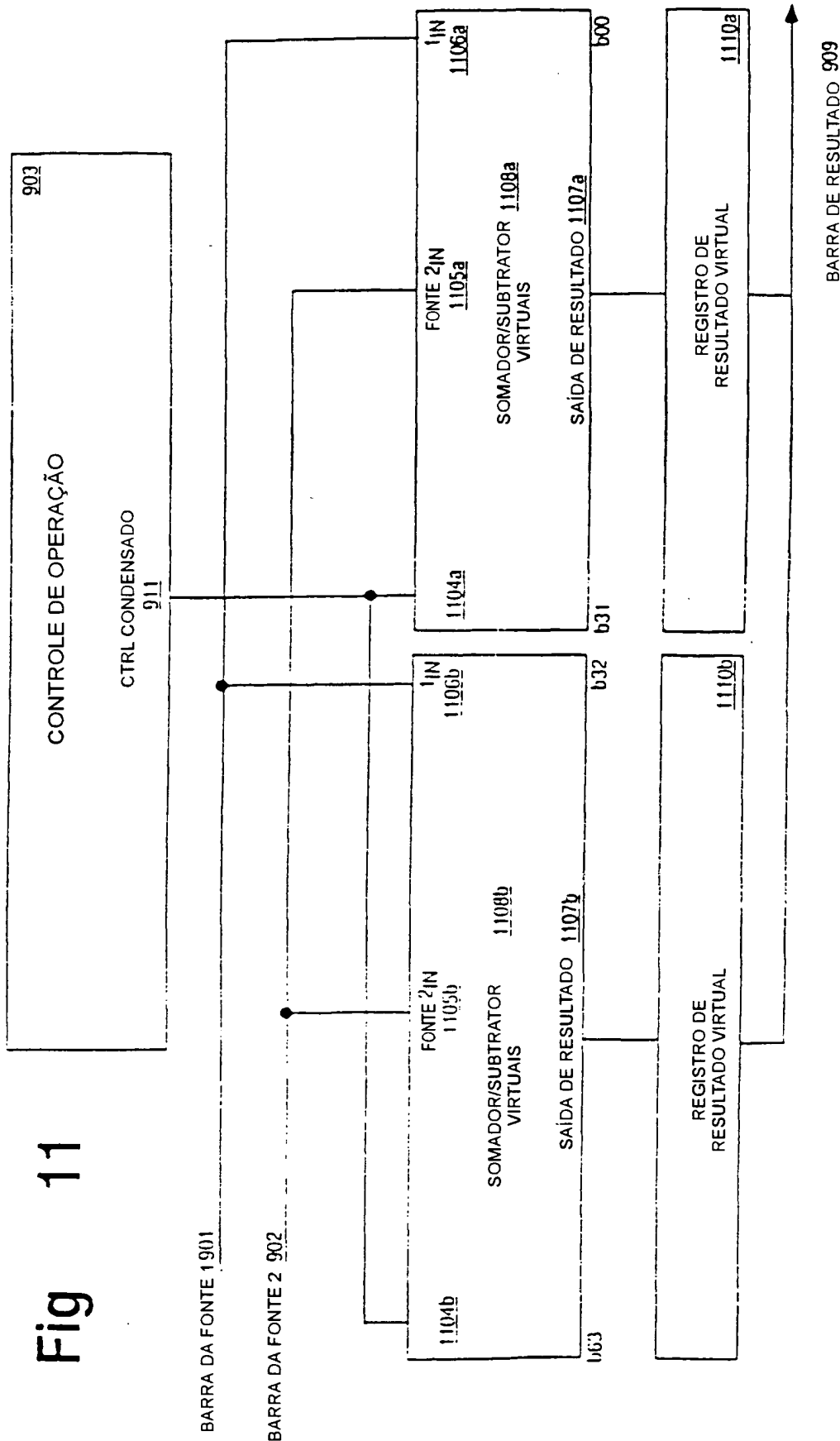


Fig 10

BARRA DE RESULTADO 909

Fig 11



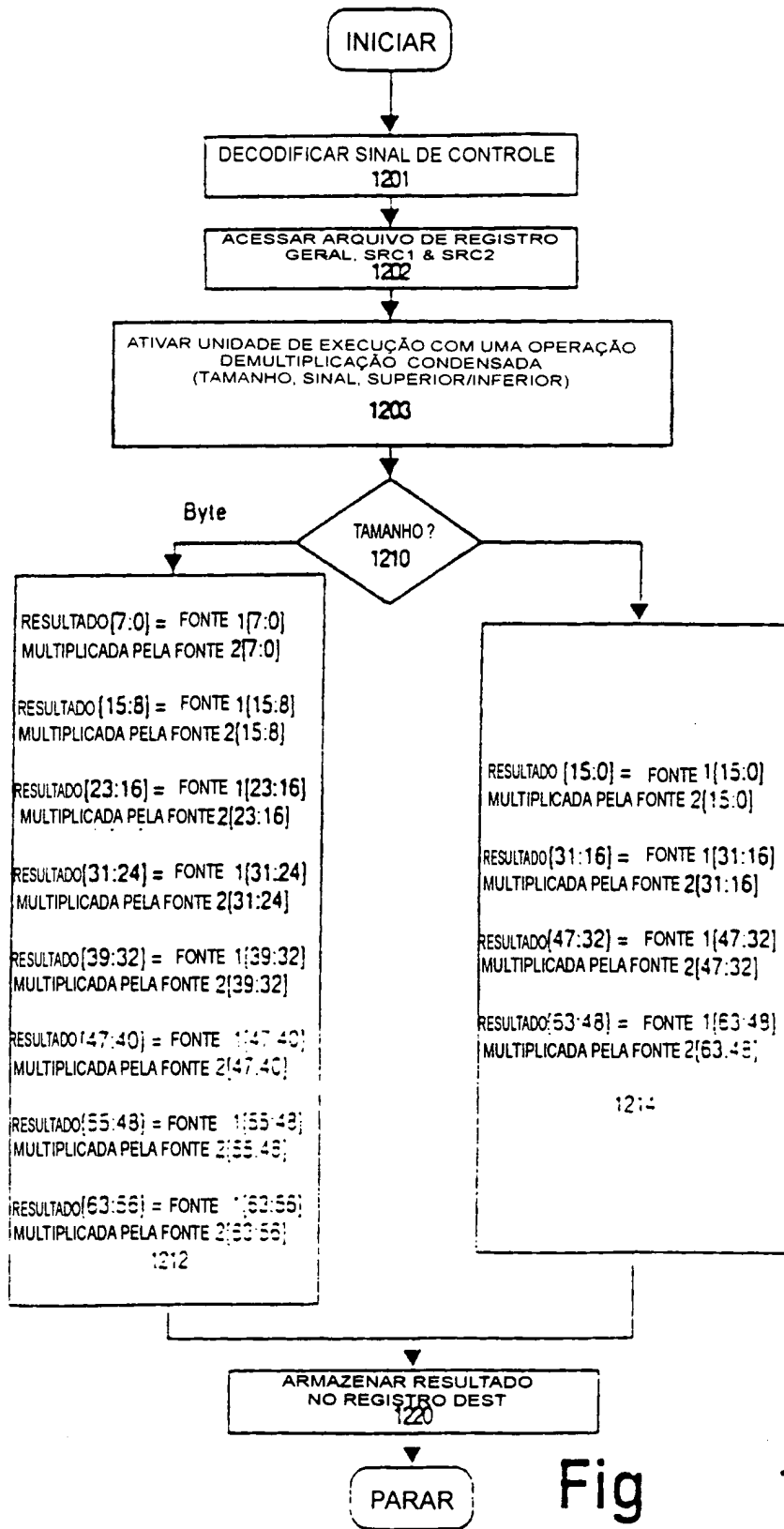


Fig 12

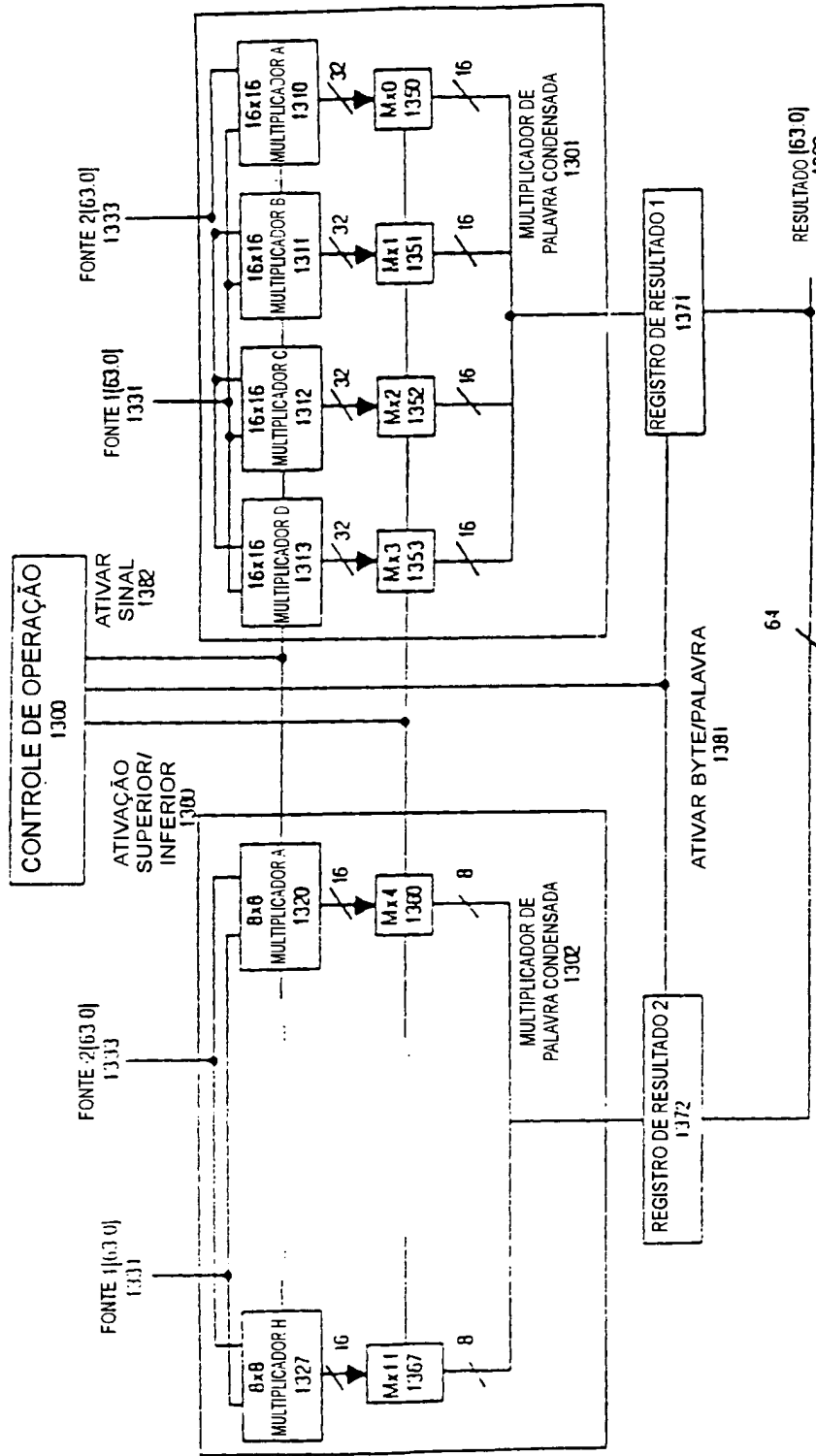


Fig 13

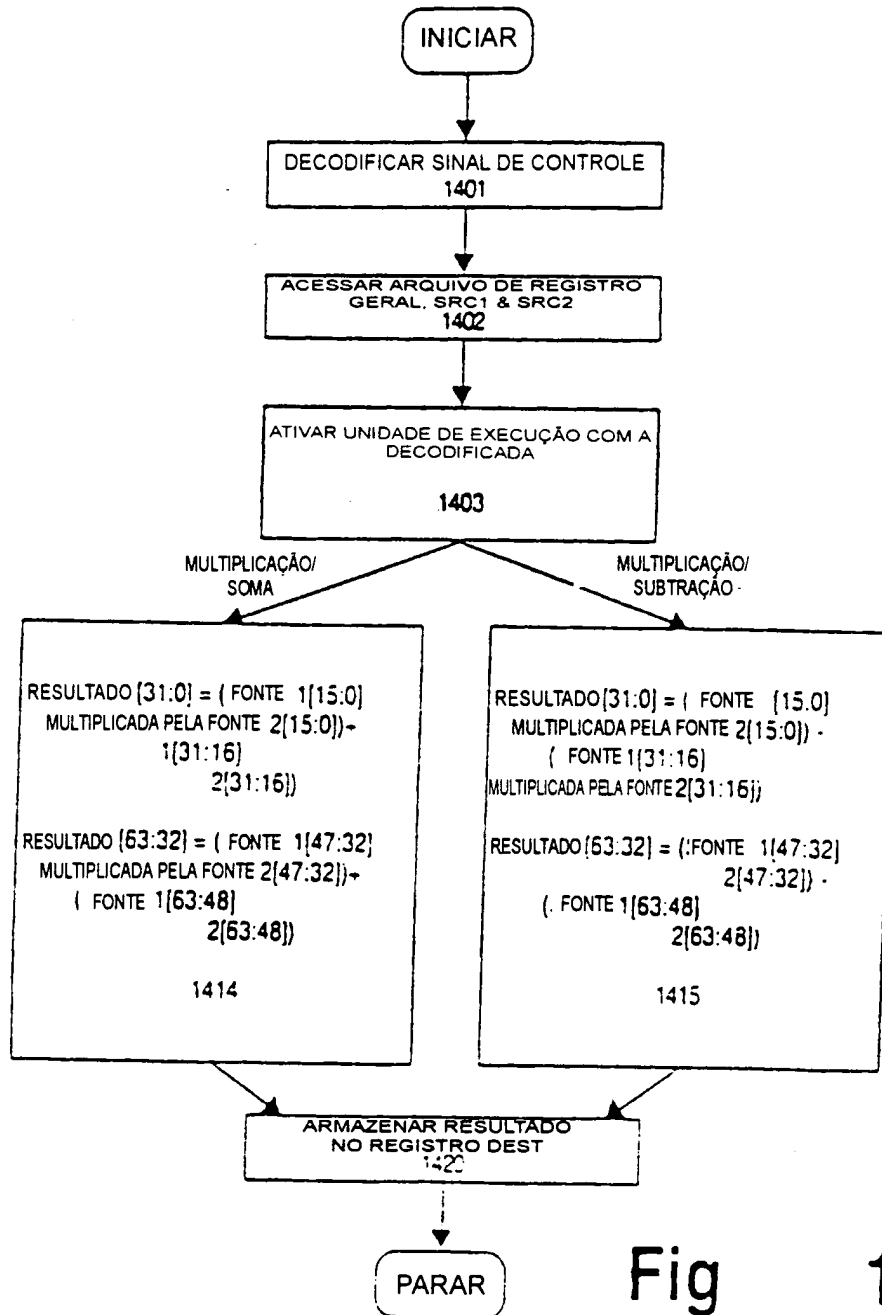


Fig 14

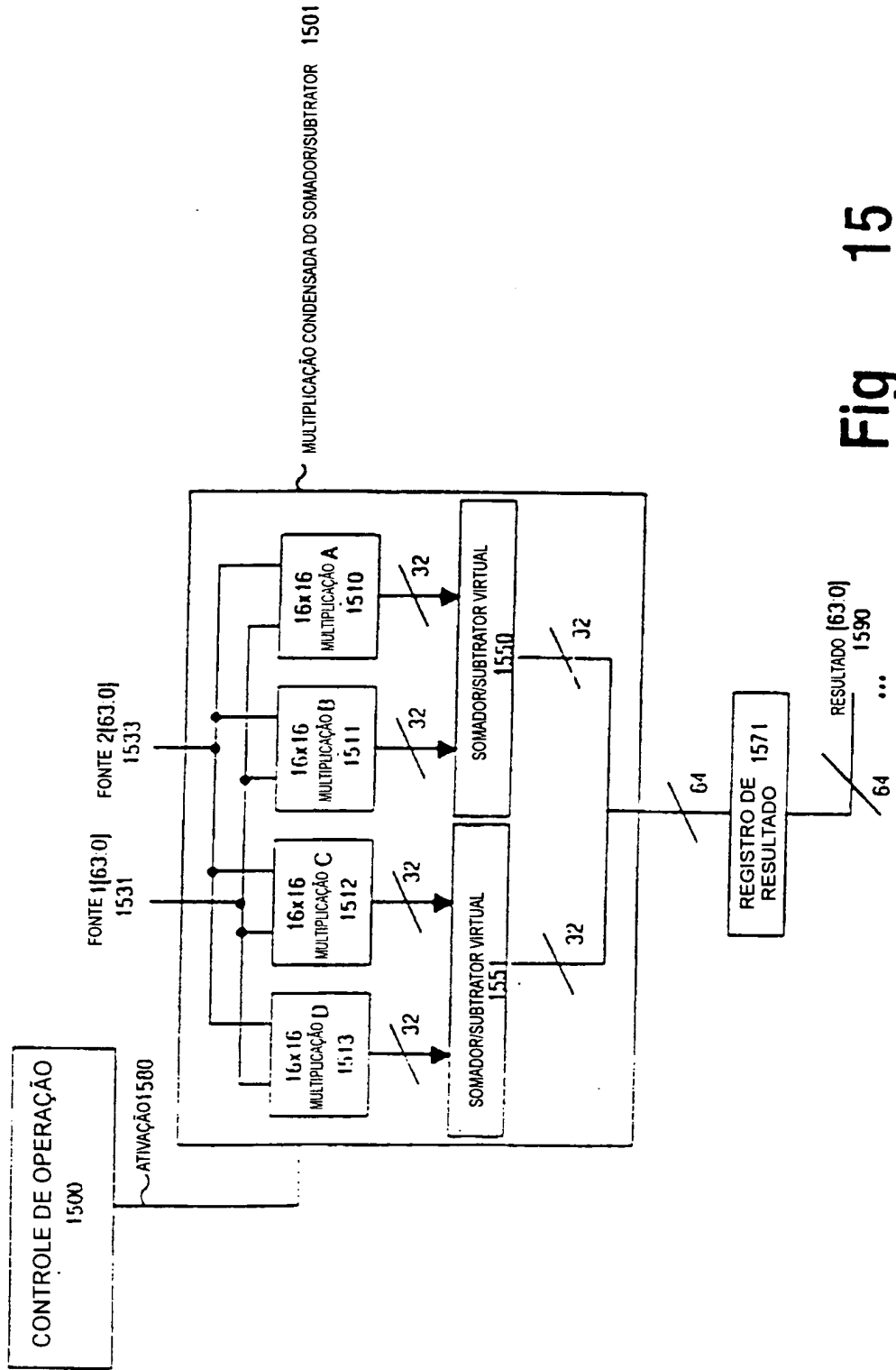


Fig 15

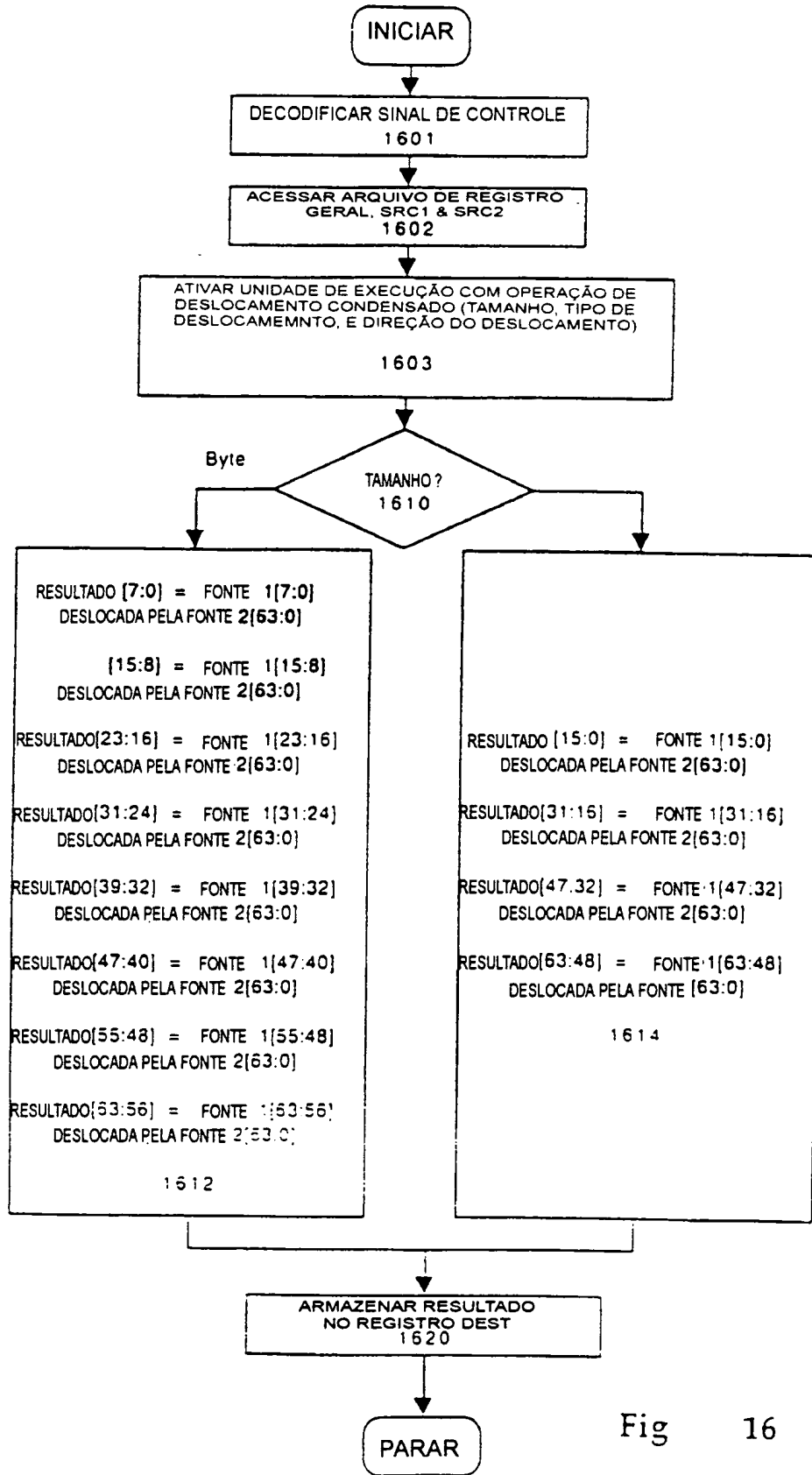
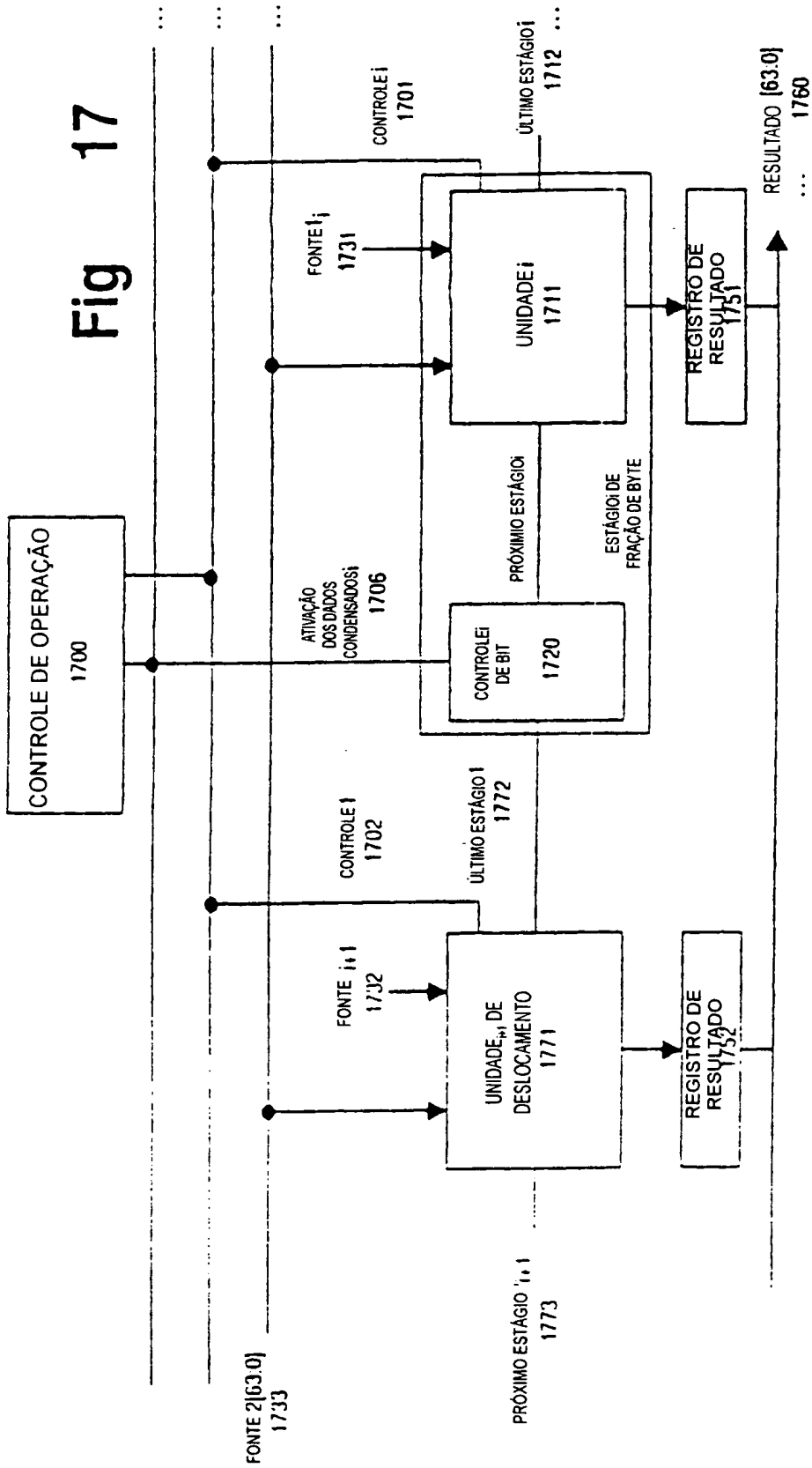


Fig 16

Fig 17



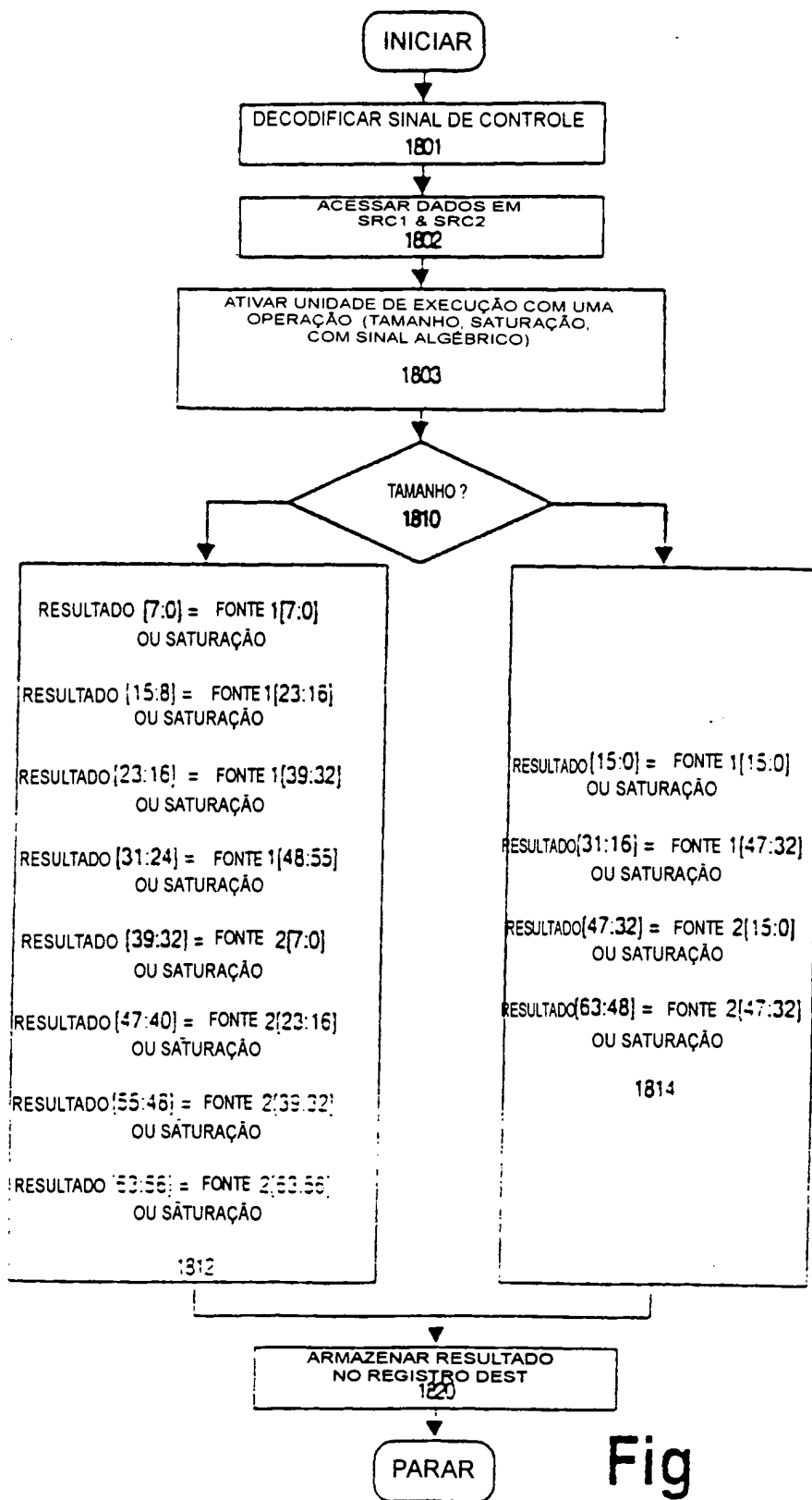


Fig 18

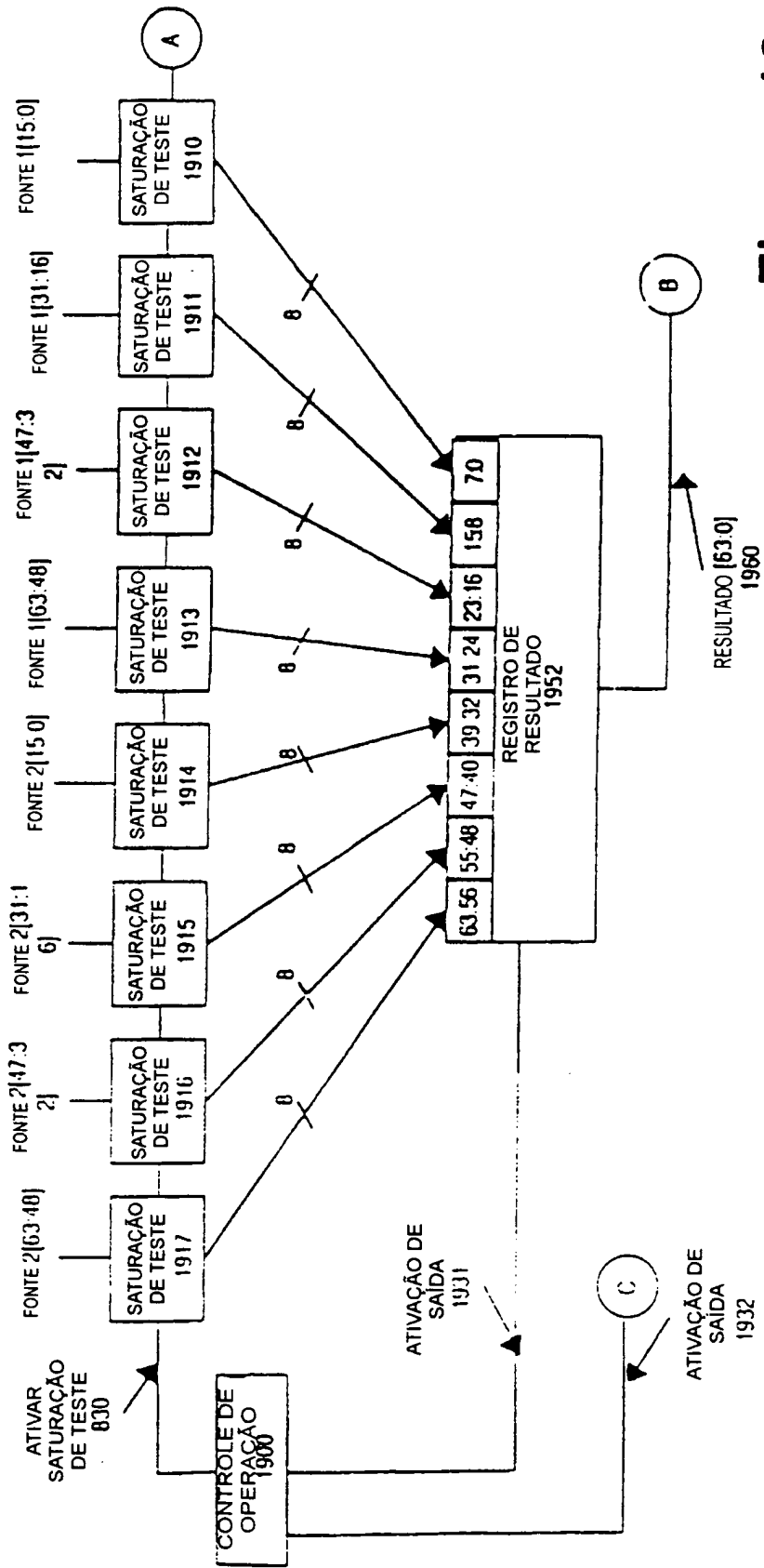


Fig 19a

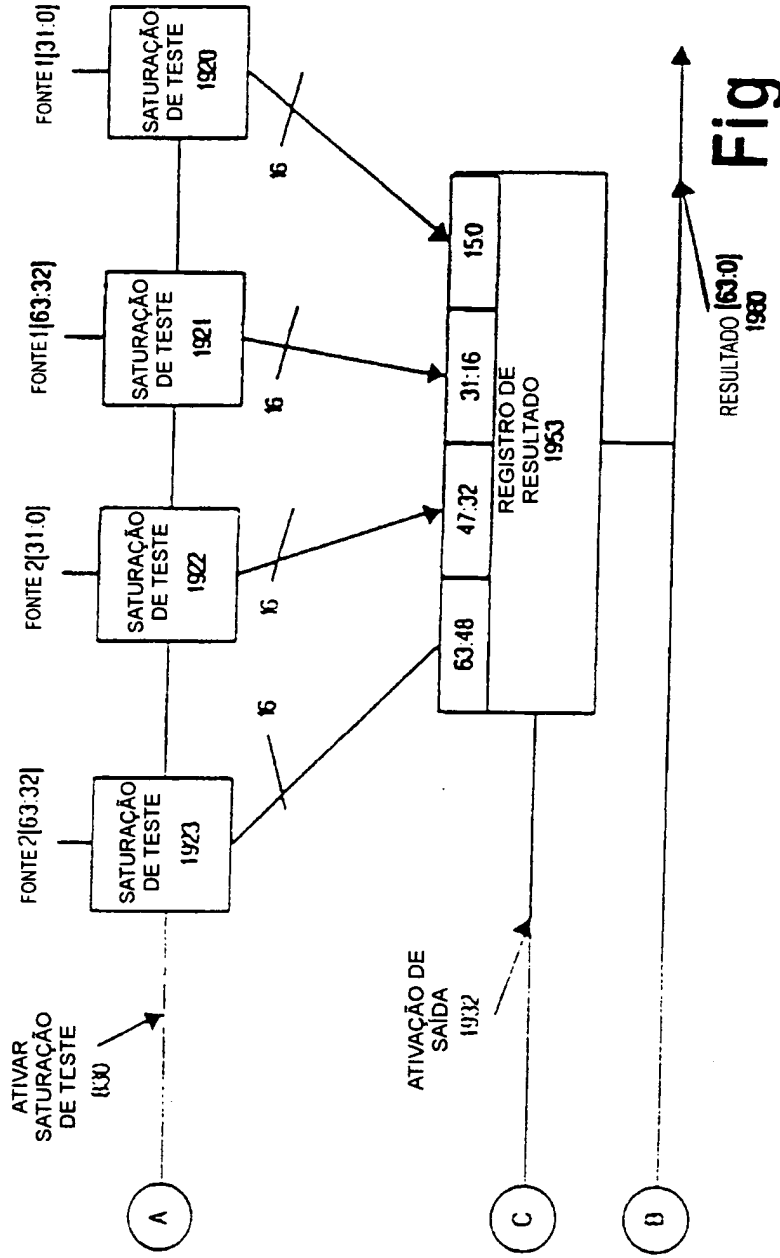


Fig 19b

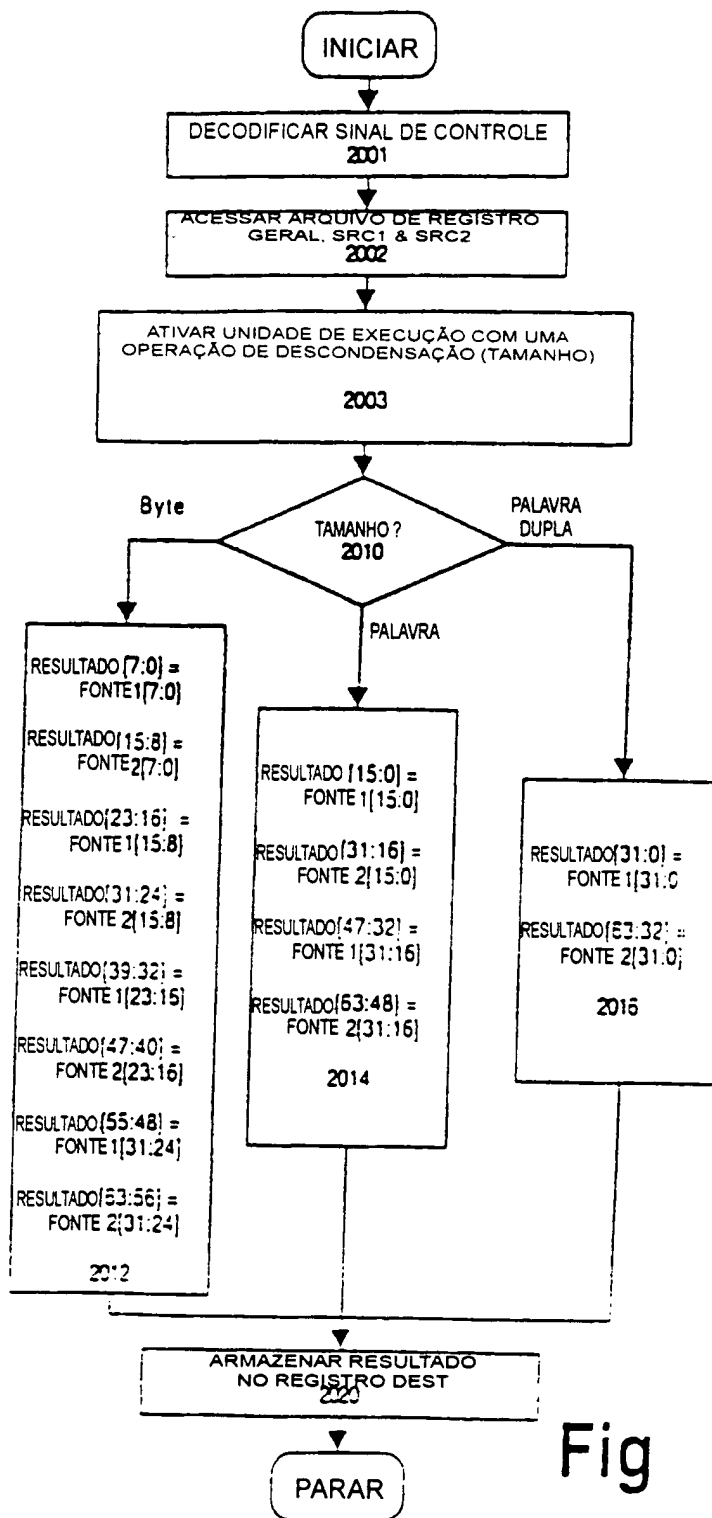
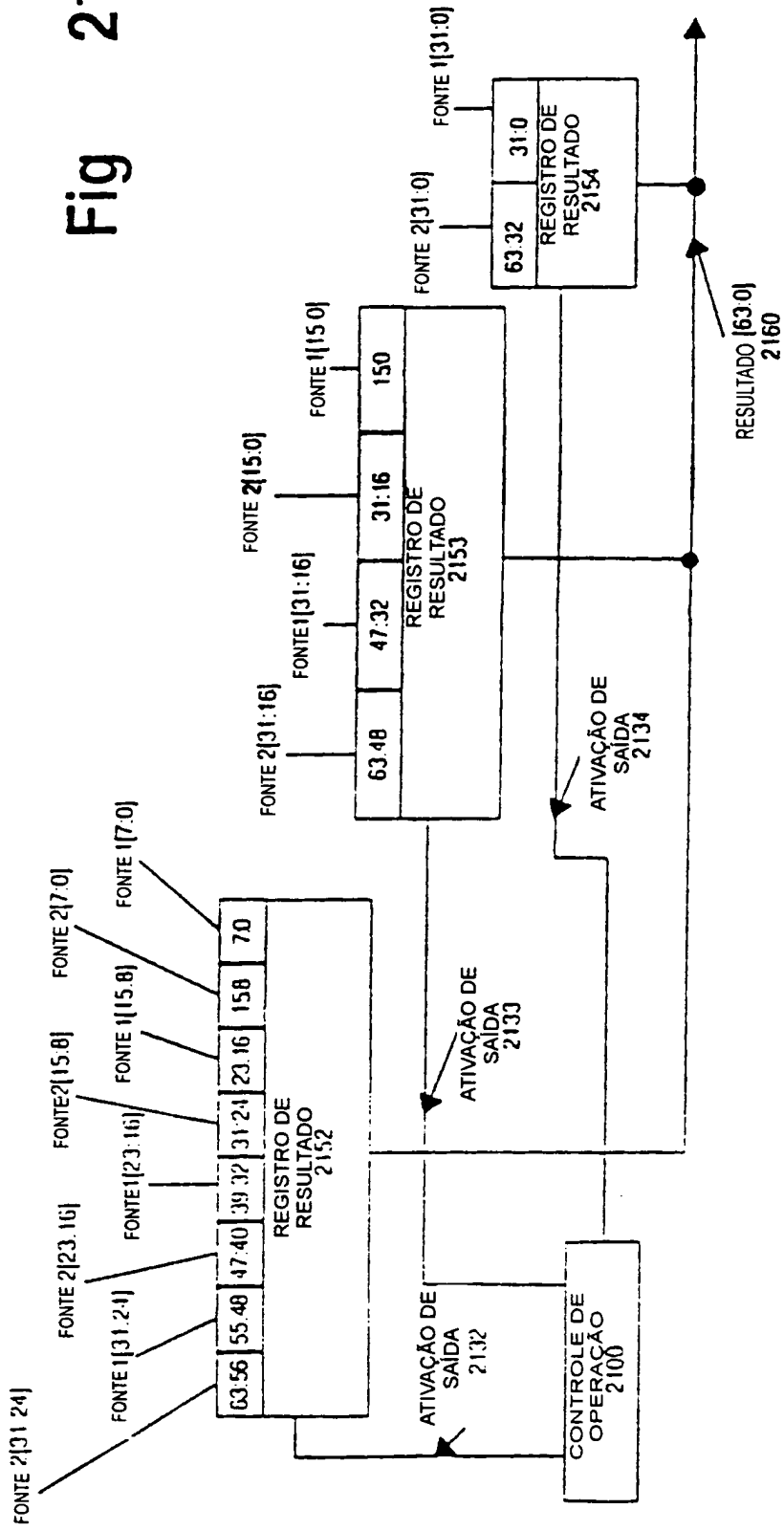
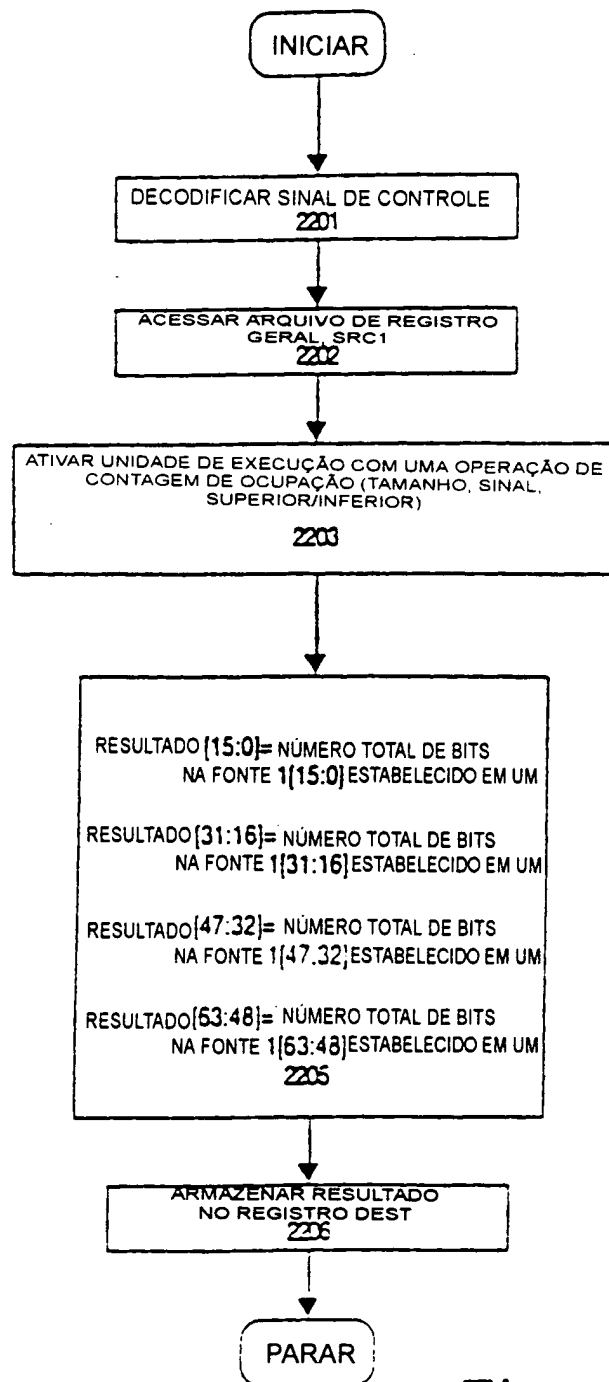


Fig 20

Fig 21





Fig

22

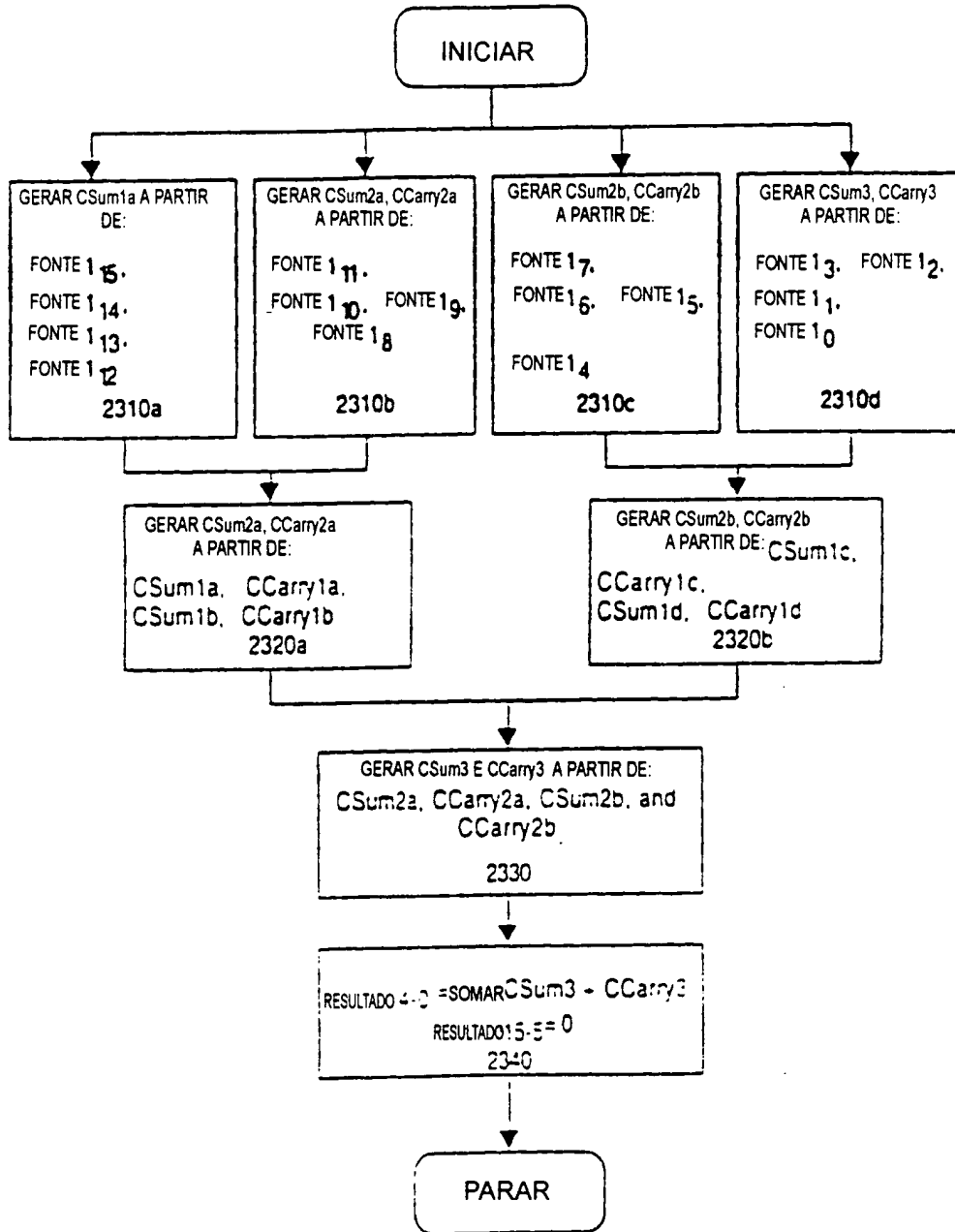


Fig 23

CIRCUITO DE CONTROLE DE OCUPAÇÃO

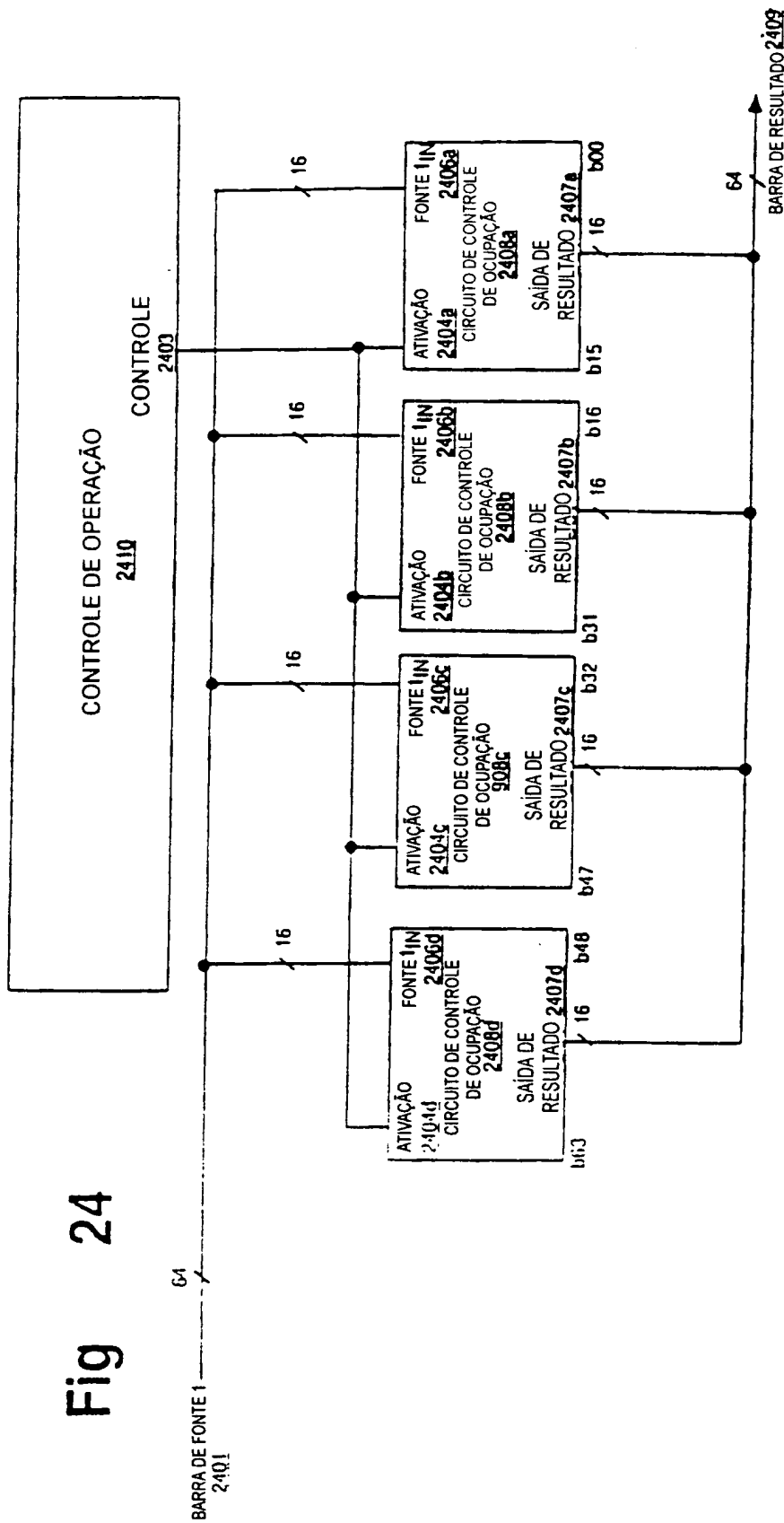


Fig 24

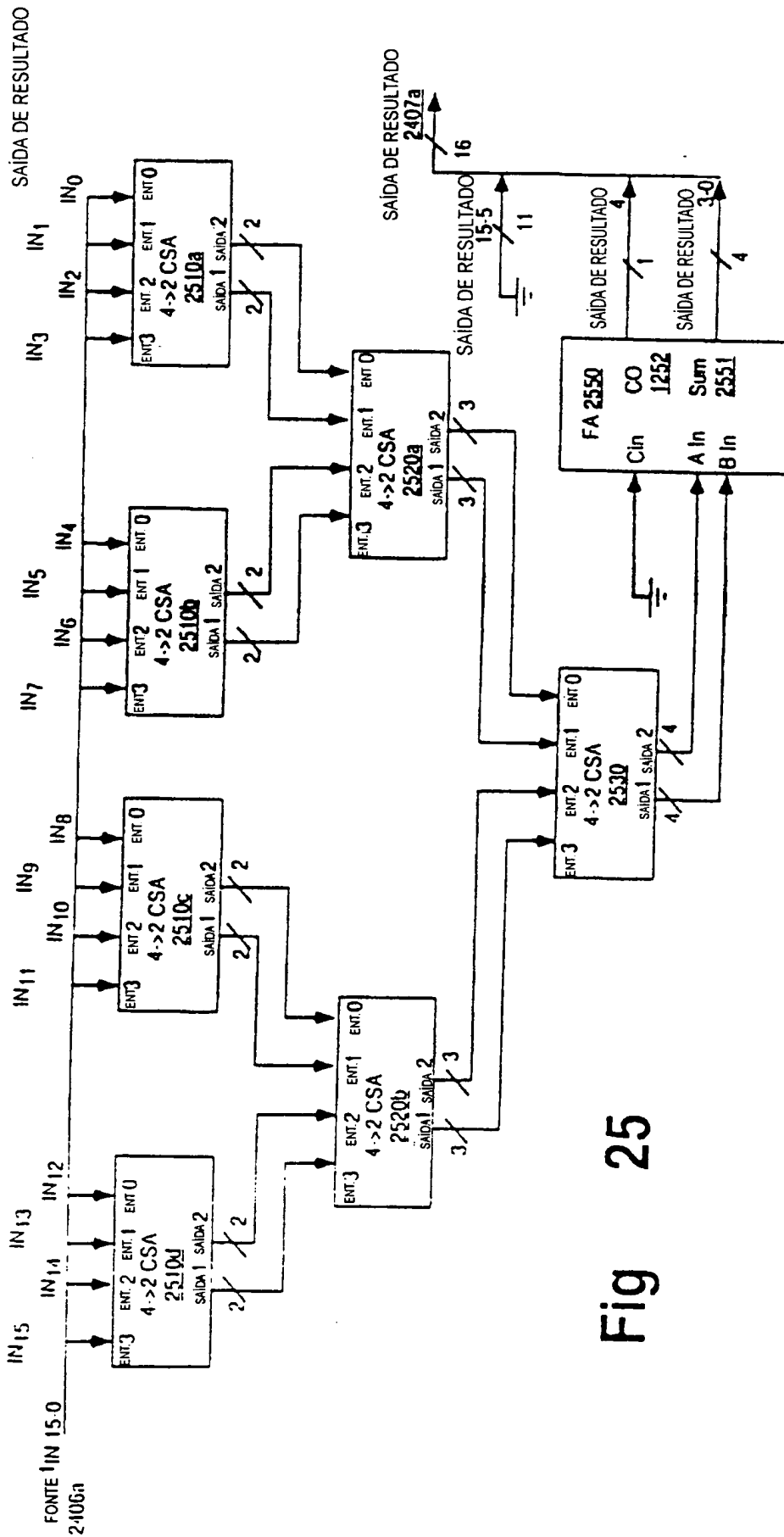


Fig 25

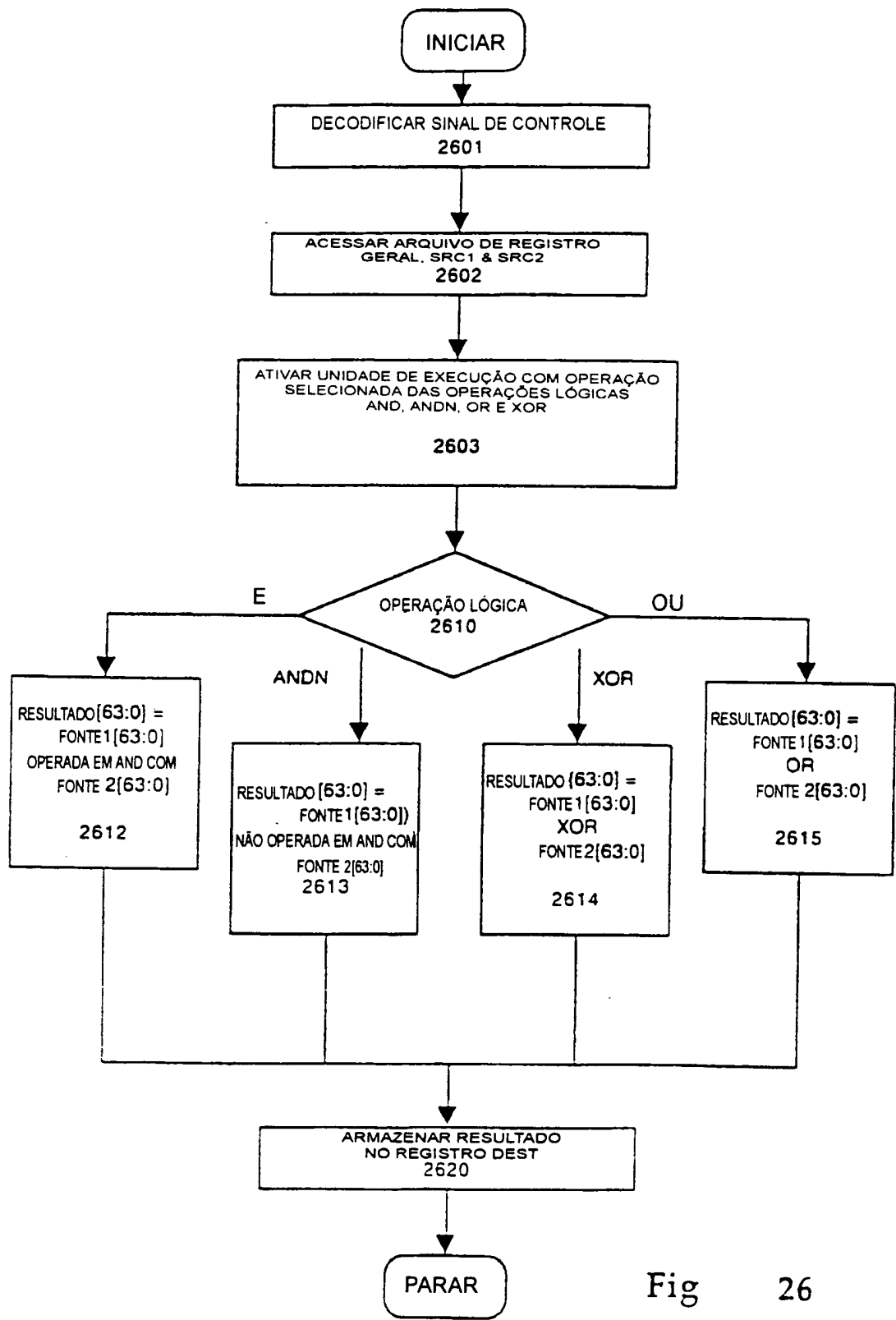


Fig 26

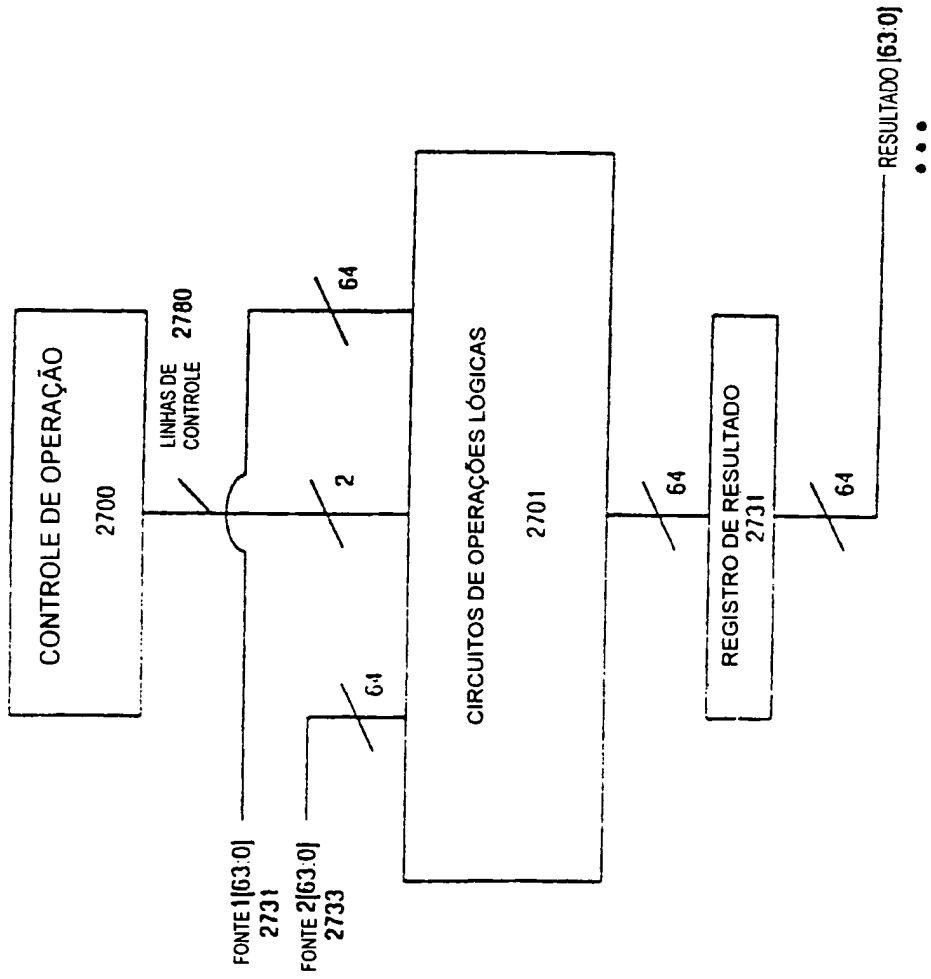


Fig 27

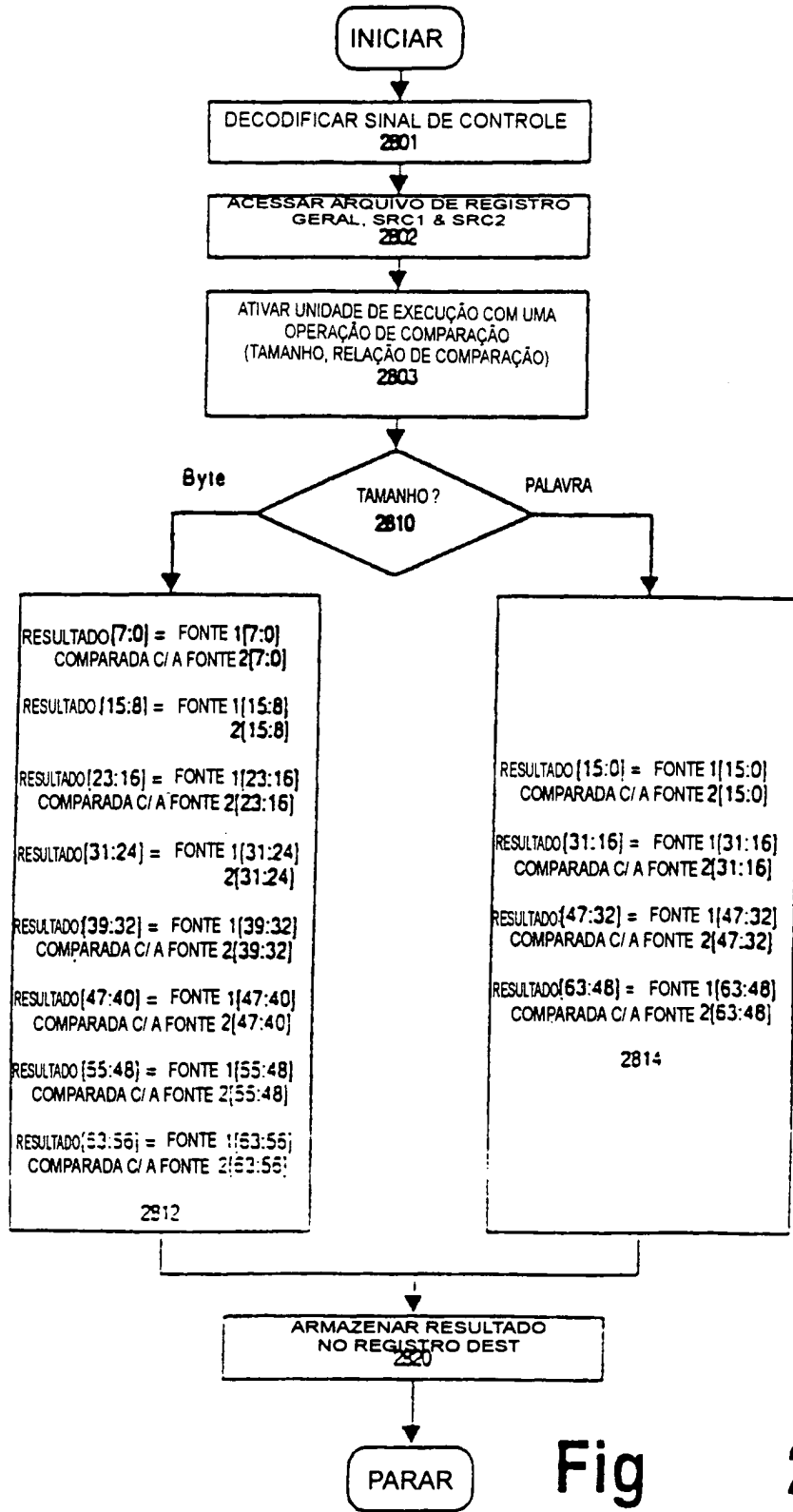
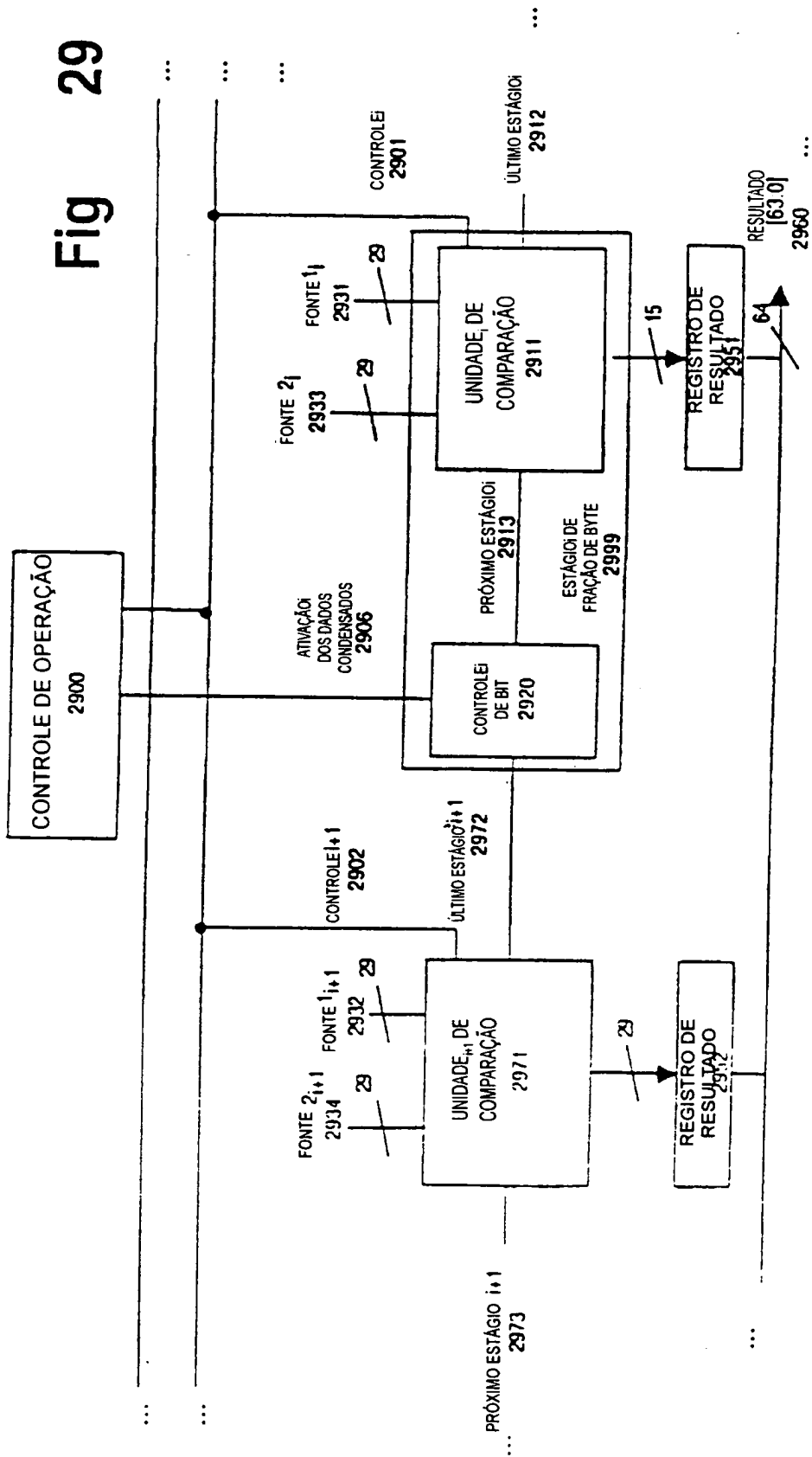


Fig 28

Fig 29



RESUMO

Patente de Invenção: **"APARELHO E MÉTODO PARA REALIZAR OPERAÇÕES MULTIPLICAÇÃO-ADIÇÃO EM DADOS EM PACOTE"**.

5 A presente invenção refere-se a um método e aparelho para incluir em um processador instruções para realizar operações multiplicação-adição em dados em pacote. Em uma modalidade, o processador é ligado a uma memória. A memória possui armazenados nela um primeiro dado em pacote e um segundo dado em pacote. O processador realiza operações em elementos de dados no dito primeiro dado em pacote e no dito segundo
10 dado em pacote para gerar um terceiro dado em pacote em resposta ao recebimento de uma instrução. Pelo menos dois dos elementos de dados nesse terceiro dado em pacote armazenam o resultado da realização de operações multiplicação-adição nos elementos de dados do primeiro dado em pacote e do segundo dado em pacote.