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

(21) **PI0520450-0 A2**

BRPI0520450A2

(22) Data de Depósito: 26/07/2005
(43) Data da Publicação: 29/03/2011
(RPI 2099)

(51) Int.Cl.:
H04B 1/707
H04L 7/04
G06F 17/15
G06F 13/40

(54) Título: **CORRELACIONADOR PARA BUSCA DE CÉLULAS PRIMÁRIAS USANDO ARQUITETURA DE MEMÓRIA**

(73) Titular(es): THOMSON LICENSING

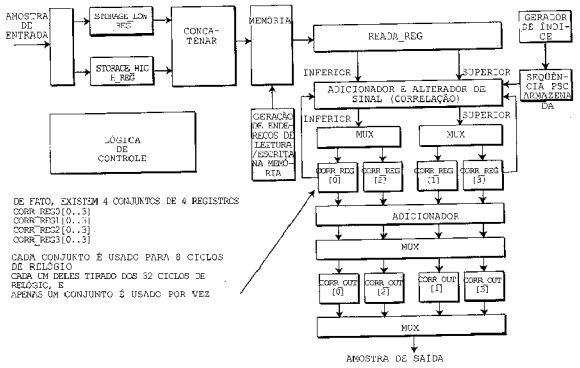
(72) Inventor(es): LOUIS ROBERT LITWIN

(74) Procurador(es): ALEXANDRE FERREIRA

(86) Pedido Internacional: PCT US2005026453 de 26/07/2005

(87) Publicação Internacional: WO 2007/018497de 15/02/2007

(57) Resumo: CORRELACIONADOR PARA BUSCA DE CELULAS PRIMÁRIAS USANDO ARQUITETURA DE MEMÓRIA Um aparelho incluindo um correlacionador de segun-do estágio, para receber dados de entrada de um correlacionador de primeiro estágio, em que o dito correlacionador de segundo estágio usa uma arquitetura de memória, é descrito. Um método para conduzir uma correlação de segundo estágio nos dados, incluindo restauração de um ponteiro de leitura e de um ponteiro de escrita, alternativamente, multiplexação dos dados de entrada em um de um par de registros de armazenamento, concatenação do conteúdo do par de registros de armazenamento, escrita do conteúdo concatenado em uma memóriade acordo com o ponteiro de escrita, transferência do conteúdo concatenado da memória para um registro de leitura, de acordo com o ponteiro de leitura, atualização do ponteiro de endereço de leitura e atualização do ponteiro de endereço de escrita, é também descrito.



"CORRELACIONADOR PARA BUSCA DE CÉLULAS PRIMÁRIAS USANDO ARQUITETURA DE MEMÓRIA"

CAMPO DA INVENÇÃO

A presente invenção se refere a terminais móveis e, em particular, a correlacionadores usados na busca de células primárias.

ANTECEDENTES DA INVENÇÃO

A unidade básica de tempo em sinais de rádio UMTS é um quadro de rádio de 10 milissegundos (ms), que é dividido em 15 espaços de 2.560 chips cada. Os sinais de rádio UMTS de uma célula (ou estação de base) para um receptor UMTS são "sinais de ligação inferior", enquanto que os sinais de rádio na direção reversa são denominados "sinais de ligação superior".

A camada física do padrão de acesso múltiplo por divisão de código de banda larga (WCDMA) do sistema de telecomunicação móvel universal (UMTS) usa a modulação de propagação de espectros de seqüência direta (DSSS), com uma taxa de chip de 3,84 Mcps. O modo duplex de divisão de freqüência (FDD) conduz os canais de ligação superior e os de ligação inferior em bandas de freqüência separadas de 5 MHz cada. Esse modo é tipicamente usado para grandes células externas, porque pode suportar um maior número de usuários do que o modo duplex de divisão de tempo (TDD). No modo TDD, as transmissões partilham os mesmos canais de ligação superior e de ligação inferior, durante os diferentes espaços de tempo. O modo TDD não suporta tanto usuários quanto o modo FDD, e, por conseguinte, o modo TDD é mais adequado para células

menores. O modo TDD é também mais adequado para conduzir tráfego assimétrico, comparado com o modo FDD.

Um procedimento importante conduzido por um receptor dentro de uma rede UMTS, por exemplo, um receptor móvel CDMA, é o operação de busca de células. A busca de células é tipicamente feita por um sistema de busca de células, que é incorporado como parte do receptor. O sistema de busca de células é ativado após o receptor ser energizado, para determinar as informações de sincronização relativas à célula, na qual o receptor é localizado. A operação de busca de células é um processo de três estágios. Isto é, o sistema de busca de células realiza sincronização de espaço (sincronização primária), sincronização de quadros e determinação de grupo de códigos de embaralhamento (sincronização secundária), e determinação de códigos de embaralhamento.

Após energização, o terminal móvel (MT) tem que executar várias operações, antes que as comunicações de voz / dados possam ser iniciadas. Primeiro, o receptor precisa implementar o controle de ganho automático (AGG), para representar em escala a potência do sinal recebido e impedir limitação no conversor de analógico em digital. O processo pode ser primeiro conduzido no canal de sincronização (SCH) e depois o canal piloto comum desembaralhado (CPICH) pode ser usado, desde que o código de embaralhamento das células seja obtido.

A seguir, o receptor precisa obter a sincronização de temporização. A sincronização de temporização pode ser obtida do canal SCH. O MT busca o sinal SCH mais forte que

ele possa achar e esse sinal determina com que célula o MT ai iniciar as comunicações. Uma vez que o canal SCH é periódico, o receptor pode se correlacionar com o SCH primário, para derivar um erro de temporização. Com base nesse canal, o receptor pode obter a sincronização de chips, símbolos e espaço.

O SHC primário conduz o mesmo sinal para todas as células no sistema. O SCH secundário é diferente para cada célula e conduz um modelo de códigos de sincronização secundária (SSCs), que repetem cada um dos quadros. Uma vez que o MT recebe essa seqüência, vai ter uma sincronização de quadros.

Na condução da busca de células, o sistema de busca de células acessa um canal de sincronização (SCH) e um canal piloto comum (CPICH) do sinal sem fio recebido. O SCH é um canal composto formado de um SCH primário e um SCH secundário. Dentro de cada espaço, o SCH primário especifica um código de sincronização primária (PSC). O SCH primário, no entanto, apenas contém dados durante os primeiros 256 chips de cada 2.560 espaços de chips. Como é conhecido, "chip" ou "taxa de chip" se refere à taxa do código de espalhamento dentro de um sistema de comunicação CDMA.

Além disso, o modelo identifica a que grupo de códigos de embaralhamento pertence o presente código de embaralhamento das células. Há 64 grupos de códigos de embaralhamento, e cada grupo contém oito códigos de embaralhamento. Uma vez que o MT determinou presente grupo de códigos de embaralhamento de células, a busca para o código de embara-

lhamento de células atual é estreitada para os oito códigos nesse grupo.

O processo de aquisição para um receptor baseado em portadora é o seguinte:

1. busca das células primárias;

2. busca das células secundárias;

3. determinação do código de embaralhamento;

4. busca por rotas múltiplas;

5. atribuição de derivação;

6. bloqueio de rastreamento de código e de circuitos fechados de controle de freqüência automática (AFC);

7. combinação de razões máximas (MRC) de saída de derivação; e

8. bloqueio de receptor é obtido e dados podem ser enviados para as camadas superiores.

Esse processo de aquisição é longo e envolvente e pode levar da ordem de vários segundos para se completar.

O problema abordado é como implementar um bloco de correlação eficiente por área para o segundo estágio no processamento de busca de células primárias, em um receptor WCDMA 3G. O primeiro estágio do processamento de busca de células primárias envolve a correlação 16 de amostras sucessivas em uma linha e a geração de uma saída de correlação a cada 16 chips. Desse modo, os requisitos de armazenamento para o correlacionador do primeiro estágio são que ele apenas precisa armazenar 16 chips a um tempo para uma dada correlação, o que é relativamente simples de fazer. Mesmo para um receptor que está usando 4 amostras por chip, os requisi-

tos de armazenamento são ainda apenas de 256 amostras e são amostras sucessivas. Isso significa que o correlacionado do primeiro estágio processa um grupo contíguo de amostras, na medida em que chegam.

Cada correlação no segundo estágio de processamento também requer 16 chips. No entanto, por causa da natureza dos códigos hierárquicos de Golay usados no padrão WCDMA 3G, cada um desses 16 chips é localizado a uma separação de 16 chip entre si. Desse modo, para um receptor que usa 4 amostras por chip, 256 chips ainda precisam ser processados, mas não são localizados contiguamente. Em vez disso, uma dada correlação precisa de 256 chips localizados 16 * 4 = 64 amostras entre si. Para armazenar todas as amostras necessárias para uma dada correlação de segundo estágio, o receptor vai requerer uma linha de retardo com derivação, com 1.024 locais (16 chips localizados 16 chips entre si são 256 chips, e 4 amostras por chip são 1.024 amostras). A técnica anterior usou um projeto baseado em registro, para implementar a correlação de segundo estágio. Esse número de registros (por exemplo, 1.024) não é prático em um projeto ASIC, porque consome uma grande quantidade de espaço morto no ASIC. Desse modo, uma abordagem mais eficiente em área seria vantajosa.

RESUMO DA INVENÇÃO

A presente invenção é uma arquitetura para o segundo estágio hierárquico de correlacionadores usados no processamento de busca de células primárias de um receptor WCDMA 3G. A arquitetura usada é baseada em memória e propi-

cia que o projeto seja eficiente em área, em termos do espaço morto disponível em um ASIC.

A presente invenção usa uma abordagem baseada em memória, porque, para um dado número de locais, uma memória é mais eficiente do que registros. No entanto, a natureza de um bloco de memória RAM de porta dupla significa que o número de leituras / escritas de memória, que pode ser feito em um dado ciclo de relógio, é limitado a uma leitura e a uma escrita por ciclo. Isso apresentou alguns desafios no projeto do bloco, uma vez que isso não permitiu suficientes leituras e escritas, para propiciar que o processamento integral fosse feito dentro da limitação dos ciclos de 32 relógios do receptor por chip. Vários itens foram adicionados à arquitetura, para usar uma única leitura e uma única escrita por ciclo de relógio, para executar o processamento desejado dentro dos ciclos de 32 relógios por chip.

Um aparelho incluindo um correlacionador de segundo estágio, para receber dados de entrada de um correlacionador de primeiro estágio, em que o dito correlacionador de segundo estágio usa uma arquitetura de memória, é descrito. Um método para conduzir uma correlação de segundo estágio nos dados, incluindo restauração de um ponteiro de leitura e de um ponteiro de escrita, alternativamente, multiplexação (MUX) dos dados de entrada em um de um par de registros de armazenamento, concatenação do conteúdo do par de registros de armazenamento, escrita do conteúdo concatenado em uma memória de acordo com o ponteiro de escrita, transferência do conteúdo concatenado da memória para um registro de leitura,

de acordo com o ponteiro de leitura, atualização do ponteiro de endereço de leitura e atualização do ponteiro de endereço de escrita, é também descrito.

BREVE DESCRIÇÃO DOS DESENHOS

A presente invenção é melhor entendida da descrição detalhada apresentada a seguir, quando lida em conjunto com os desenhos em anexo. Os desenhos incluem as figuras apresentadas a seguir, descritas sucintamente abaixo, nas quais os números similares nas figuras representam elementos similares.

A Figura 1 é um diagrama de blocos de nível de topo de processamento de busca de células.

A Figura 2 é um diagrama de blocos da arquitetura da presente invenção.

A Figura 3 é uma modalidade do uso do ponteiro de leitura / escrita para memória, de acordo com os princípios da presente invenção.

A Figura 4 é um fluxograma de acordo com os princípios da presente invenção.

DESCRIÇÃO DETALHADA DAS MODALIDADES PREFERIDAS

As buscas de células são feitas em terminais móveis. Com referência agora à Figura 1, que é um diagrama de blocos do nível de topo de processamento de busca de células, a presente invenção envolve os correlacionadores 125, 130, usados no segundo estágio da busca de células primárias, que recebem entradas reais 115 e imaginárias 120 dos correlacionadores do primeiro estágio 105, 110 da busca de células primárias. A saída dos correlacionadores de busca de

células primárias de primeiro estágio 105, 110 é a entrada para os correlacionadores do segundo estágio 125, 130. A saída dos correlacionadores do segundo estágio 125, 130 é a saída para o combinador não coerente 135, que proporciona a entrada para o armazenamento temporário de quadros 140. O armazenamento temporário de quadros 140 proporciona os resultados da busca de células.

A Figura 2 é um diagrama de blocos da arquitetura da presente invenção. Em particular, a Figura 2 é a arquitetura dos correlacionadores da presente invenção, usados para o segundo estágio da busca de células primárias. Os correlacionadores da presente invenção usam uma arquitetura de memória, que tem a vantagem de ser eficiente em área, em termos do espaço morto no ASIC. O bloco Geração de Endereço de Leitura / Escrita de Memória 235, na Figura 2, gera os valores do ponteiro de leitura / escrita (também mostrados na Figura 3). O correlacionador do segundo estágio 123 da presente invenção é de fato um par de correlacionadores do segundo estágio 125, 130, que são funcionalmente idênticos / equivalentes. A diferença entre o par de correlacionadores do segundo estágio 125, 130 é os dados de entrada (valores reais versus valores imaginários), recebidos dos correlacionadores do segundo estágio.

As saídas de correlação (reais e imaginárias) dos correlacionadores do primeiro estágio (mostradas na Figura 1) chegam no multiplexador 205 da Figura 2. Essas amostras chegam 4 vezes por chip e são multiplexadas alternadamente nos registros de armazenamento, primeiro no storage_low_reg

210, depois no storage_high_reg 215, e depois continuam alternadamente. Com base na lógica, que vai ser descrita em mais detalhes abaixo, os valores de registro de armazenamento baixos e altos (cada um com 16 bits de largura) são concatenados no bloco 225, para formar um único valor de bit 32, que é depois escrito na memória 230, em um ciclo de relógio predeterminado. Essa abordagem é usada por causa da limitação de apenas uma escrita de memória por ciclo de relógio - por armazenamento de duas amostras como um valor, esse projeto permite que duas amostras sejam armazenadas na memória 230 para cada dado ciclo de relógio. O uso da memória 230 nesse ponto salva espaço morto do chip. As implementações da técnica anterior usam um banco de registros em vez de memória.

Os valores são depois lidos da memória de locais predeterminados e armazenados no read_reg 240. Desse ponto, os bits são analisados gramaticamente de novo nos seus valores superiores e inferiores correspondentes, e processados como duas amostras separadas. O gerador de índices 245 gera o índice / seqüência PSC. A correlação é feita no bloco 255, sem os multiplicadores intensivos em área, tomando-se a amostra do read_reg 240 e adicionando ou subtraindo dela a amostra no corr_reg, com base no sinal do bloco de seqüência PSC armazenado 250 (isto é, se a seqüência PSC for +1, o valor é adicionado, se a seqüência for -1, o valor é subtraído). Notar que há 16 registros corr_reg: corr_reg0[0] a corr_reg0[3] 270a, corr_reg1[0] a corr_reg1[3] 270h, corr_reg2[0] a corr_reg2[3] 270c, e corr_reg3[0] a

corr_reg3[3] 270d. É possível armazenar e processar 4 correlações simultâneas computadas em 4 blocos paralelos cada. Cada conjunto de registros é usado para os 8 ciclos de relógio dos 32 ciclos de relógio disponíveis, com apenas um conjunto sendo usado por vez. A saída do bloco 255 é multiplexada pelos multiplexadores 260, 265, para correlacionar os registros 270a - 270d.

Após todos os 16 valores para uma dada correlação serem acumulados no bloco adicionador 275, os valores armazenados em corr_reg são transferidos para um dos 4 registros corr_out correspondentes 285a - 285d, por um multiplexador 280. Isto é, corr_out = corr_reg0[0] ÷ corr_out1[0] ÷ corr_reg2[0] ÷ corr_reg3[0] 285a.

A saída dos registros corr_out é multiplexada ao combinador não coerente 135 da Figura 1. É também necessário tomar o valor absoluto (abs) do conteúdo dos registros corr_out. Esse bloco não é mostrado na Figura 2, mas a função é conduzida nos registros corr_out ou como um bloco adicional, após o multiplexador 290.

O pseudocódigo mostrado na Tabela 1 apresenta mais detalhes de como funciona a arquitetura. O bloco de controle 220 da Figura 2 coordena e controla as funções e os componentes do correlacionador da presente invenção. Os números à esquerda indicam o ciclo de relógio. A arquitetura da presente invenção é baseada em uma estrutura de ciclo de relógio com 32 relógios por amostra.

Pseudocódigo

Código de restauração

```
rp = 3              //ponteiro de leitura - número de 9 bits
wp = 1              // ponteiro de escrita - número de 9 bits
corr_reg[0..3] = 0
corr_out[0..3] = 0
Código de saída de correlação
 0
        corr_out[0] = corr_reg0[0] + corr_reg1[0] + corr_reg2[0] + corr_reg3[0]
        corr_out[1] = corr_reg0[1] + corr_reg1[1] + corr_reg2[1] + corr_reg3[1]

 1
        corr_out[2] = corr_reg0[2] + corr_reg1[2] + corr_reg2[2] + corr_reg3[2]
        corr_out[3] = corr_reg0[3] + corr_reg1[3] + corr_reg2[3] + corr_reg3[3]

Código de saída de amostra
 7
        samp_out = abs(corr_out[0])

 15
        samp_out = abs(corr_out[1])
 23
        samp_out = abs(corr_out[2])
 31
        samp_out = abs(corr_out[3])

Código de entrada / saída de memória
  0,16
         storage_low_reg = samp_in

  8,24
         storage_high_reg = samp_in
         memory write address = wp
         memory data in = storage_high_reg concatenated with storage_low_reg
         wp--
  every clock
         read_reg = data_out from memory

Código de interfaceamento de correlação e memória
```

```
0
        update corr_reg3[0] and corr_reg3[1] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp
1
        update corr_reg3[2] and corr_reg3[3] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp-1
        rp = rp + 32
2,4,6,8
        update corr_reg0[0] and corr_reg0[1] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp
3,5,7,9
        update corr_reg0[2] and corr_reg0[3] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp-1
        rp = rp + 32
10,12.14,16
        update corr_reg1[0] and corr_reg1[1] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp
11,13,15,17
        update corr_reg1[2] and corr_reg1[3] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp-1
        rp = rp + 32
18,20,22,24
        update corr_reg2[0] and corr_reg2[1] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp
19,21,23,25
        update corr_reg2[2] and corr_reg2[3] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp-1
        rp = rp + 32
26,28,30
        update corr_reg3 [0] and corr_reg3[1] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp
27,29,31
        update corr_reg3[2] and corr_reg3[3] with samples in read_reg (upper and lower)
        write "read" address to memory for two clock cycles ahead - read address is rp-1
        if not clock cycle = 31
                rp = rp + 32
        if clock cycle = 31
                rp = rp - 482
```

O código de restauração do pseudocódigo inicializa o ponteiro de leitura (rp) e o ponteiro de escrita (wp), que são ambos números de 9 bits, antes que qualquer outro processamento comece. Os registros de correlação (corr_reg) e os registros de saída de correlação (corr_out) são também inicializados.

O código de saída de correlação do pseudocódigo ajusta os registros corr_out [0] e [1] ao conteúdo dos registros corr_reg no ciclo de relógio 0 e os registros corr_out [2] e [3] ao conteúdo dos registros corr_reg no ciclo de relógio 1.

O código de saída da amostra do pseudocódigo proporciona a amostra de saída (samp_out) do valor absoluto (abs) do registro corr_out[0] no ciclo de relógio 7. O código de saída de amostra do pseudocódigo proporciona a amostra de saída (samp_out) do valor absoluto (abs) do registro corr_out[1] no ciclo de relógio 15. O código de saída de amostra do pseudocódigo proporciona a amostra de saída (samp_out) do valor absoluto do registro corr_out[2] no ciclo de relógio 23. O código de saída de amostra do pseudocódigo proporciona a amostra de saída (samp_out) do valor absoluto (abs) do registro corr_out[3] no ciclo de relógio 31.

Nos ciclos de relógio 0 e 16, o código de entrada / saída de memória do pseudocódigo ajusta o storage_low_reg a uma amostra de entrada (samp_in). Nos ciclos de relógio 8 e 24, o código de entrada / saída de memória do pseudocódigo ajusta o storage_high_reg a uma amostra de entrada (samp_in). Adicionalmente, nos ciclos de relógio 8 e 24 o endereço de escrita n a memória é ajustado para o ponteiro de escrita (wp), os dados de memória no endereço são ajustados ao storage_high_reg concatenado com o storage_low_reg, e o ponteiro de escrita é depois decrementado. Em cada ciclo de relógio, o read_reg é ajustado ao data_out da memória, de acordo com os endereços de leitura gerados pelo bloco de ge-

ração de endereços de leitura / escrita da memória 235.

O código de correlação e de interfaceamento de memória do pseudocódigo funciona da seguinte maneira:

No ciclo de relógio 0, corr_reg3[0] e corr_reg3[1] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.

No ciclo de relógio 1, corr_reg3[2] e corr_reg3[3] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp. O ponteiro de leitura é então incrementado por 32.

Nos ciclos de relógio 2, 4, 6 e 8, corr_reg0[0] e corr_reg0[1] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio e o endereço "read" é igual a rp.

Nos ciclo de relógios 3, 5, 7 e 9, corr_reg0[2] e corr_reg0[3] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.1. O ponteiro de leitura é então incrementado por 32.

Nos ciclos de relógio 10, 12, 14 e 16, corr_reg1[0] e corr_reg1[1] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.

Nos ciclos de relógio 11, 13, 15 e 17, corr_reg1[2] e corr_reg1[3] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.1. O ponteiro de leitura é então incrementado por 32.

Nos ciclos de relógio 18, 20, 22 e 24, corr_reg2[0] e corr_reg2[1] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.

Nos ciclos de relógio 19, 21, 23 e 25, corr_reg2[2] e corr_reg2[3] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.1. O ponteiro de leitura é então incrementado por 32.

Nos ciclos de relógio 26, 28 e 30, corr_reg3[0] e corr_reg3[1] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.

Nos ciclos de relógio 27, 29 e 31, corr_reg3[2] e corr_reg3[3] são atualizados com as amostras superior e inferior em read_reg. O endereço "read" é escrito na memória para dois ciclos de relógio à frente e o endereço "read" é igual a rp.1. Se esse não for o ciclo de relógio 31, então incrementar o ponteiro de leitura por 32. Se esse for o ci-

clo de relógio 31, então decrementar o ponteiro de leitura por 482.

Com respeito à Figura 3, o ponteiro de escrita (wp) é inicializado a um valor de 1, e é decrementado duas vezes dentro de cada período de ciclo de relógio 32 (módulo 512). O ponteiro de leitura (rp) é inicializado a um valor de 3, incrementado por 32 por 15 vezes dentro de cada período de ciclo de relógio 32 e decrementado por 482 (512 - 30), uma vez a cada período de ciclo de relógio 32. A memória de porta dupla e o seu uso na presente invenção é similar a uma janela deslizante ou armazenamento temporário, em que os ponteiros de leitura e escrita estão endereçando a mesma memória em tempos diferentes. Isto é, não há qualquer sobreposição dos locais de memória que são lidos e os locais de memória que são escritos. Isso é porque há apenas uma leitura e uma escrita por ciclo de relógio. Os índices dos ponteiros de leitura e escrita e os valores do incremento e do decremento vão variar, se o número de amostras/chips aumentar ou diminuir. Especificamente, com referência à Figura 3, que ilustra a memória de porta dupla tendo nesse exemplo 512 locais, cada local sendo de 32 bits, na restauração o ponteiro de escrita (wp) foi inicializado em 1 e o ponteiro de leitura (rp) foi inicializado em 3. Após os primeiros 32 ciclos de relógio, o ponteiro de escrita (wp) é 511 e o ponteiro de leitura (rp) é 1.

Com referência agora à Figura 4, que é um fluxograma das ações do correlacionador de segundo estágio da presente invenção. Na etapa 405, as amostras são multiplica-

das alternativamente nos registros storage_reg_low e storage_reg_high. Na etapa 410, os conteúdos dos registros storage_reg_low e storage_reg_high são concatenados e escritos como um único valor na memória, de acordo com o ponteiro de escrita (wp) especificado pelo bloco de geração de endereço de leitura / escrita da memória 235. Na etapa 415, em cada ciclo de relógio, uma amostra da memória 230 é transferida para o registro read_reg 240, de acordo com o ponteiro de leitura (rp) especificado pelo bloco de geração de endereço de leitura / escrita da memória 235. A correlação é conduzida na etapa 420 por adição (±) dos valores 240 de read_reg para os valores 270a - 270d de corr_reg, com base no sinal dos índice / seqüência armazenados no bloco 250, gerado pelo bloco 245. Na etapa 425, após dezesseis acúmulos, os valores corr_reg são armazenados nos correspondentes registros corr_out 285a - 285d, por meio do adicionador 275 e do multiplexador 280, desse modo, completando efetivamente quatro correlações paralelas. O valor absoluto (abs) dos valores nos registros corr_out 285a - 285d é tirado dos registros 285a - 285d de corr_out, ou os registros 285e - 285d de corr_out são multiplicados a um bloco de valor absoluto (não mostrado), antes de transferir os valores de correlação na etapa 430.

Deve-se entender que a presente invenção pode ser implementada em várias formas de hardware, software, programação em hardware, processadores multipropósito, ou uma combinação deles, por exemplo, dentro de um terminal móvel, ponto de acesso ou uma rede celular. De preferência, a pre-

sente invenção é implementada como uma combinação de hardware e software. Além do mais, o software é preferivelmente implementado como um programa de aplicação tangivelmente representado em um dispositivo de armazenamento de programas. O programa de aplicação pode ser transferido para, e executado por, uma máquina compreendendo qualquer arquitetura adequada. De preferência, a máquina é implementada em uma plataforma de computador tendo hardware, tal como uma ou mais unidades de processamento central (CPU), uma memória de acesso aleatório (RAM) e interface(s) de entrada / saída (I/O). A plataforma de computador inclui também um sistema operacional e um código de microinstrução. Os vários processos e funções aqui descritos podem ser parte do código de microinstrução ou parte do programa de aplicação (ou uma combinação deles), que é executado pelo sistema operacional. Além disso, vários outros dispositivos periféricos podem ser conectados à plataforma do computador, tal como um dispositivo de armazenamento de dados e um dispositivo de impressão adicionais.

Deve-se entender que, porque alguns dos componentes do sistema e das etapas do método constituintes, ilustrados nas figuras em anexo, serem preferivelmente implementados em software, as conexões efetivas entre os componentes do sistema (ou as etapas do processo) podem diferir, dependendo da maneira na qual a presente invenção é programada. Em vista dos ensinamentos aqui apresentados, uma pessoa versada na técnica vai ser capaz de considerar essas implementações ou configurações ou similares da presente invenção.

REIVINDICAÇÕES

1. Aparelho, **CARACTERIZADO** pelo fato de que compreende um correlacionador de segundo estágio, para receber dados de entrada de um correlacionador de primeiro estágio, em que o dito correlacionador de segundo estágio inclui uma memória.

2. Aparelho, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que o dito correlacionador de segundo estágio compreende um par de correlacionadores de segundo estágio, em que ainda um primeiro de um par de correlacionadores de segundo estágio recebe e processa dados de entrada de valores reais, e o dito segundo do dito par de correlacionadores de segundo estágio recebe e processa dados de entrada de valores imaginários.

3. Aparelho, de acordo com a reivindicação 2, **CARACTERIZADO** pelo fato de que ambos do dito par dos ditos correlacionadores de segundo estágio são funcionalmente equivalentes.

4. Aparelho, de acordo com a reivindicação 2, **CARACTERIZADO** pelo fato de que ambos do dito par dos ditos correlacionadores de segundo estágio compreendem ainda:

um primeiro multiplexador para receber dados de entrada;

um primeiro registro de armazenamento para receber e armazenar uma primeira unidade dos ditos dados de entrada;

um segundo registro de armazenamento para receber e armazenar uma segunda unidade dos ditos dados de entrada;

um concatenador para concatenar a dita primeira

unidade de dados de entrada e a dita segunda unidade de dados de entrada;

uma memória para receber e armazenar os ditos dados de entrada concatenados;

uma unidade de geração de endereço de leitura / escrita para gerar valores de ponteiros de leitura / escrita para a dita memória; e

um registro de leitura para recuperar e armazenar os ditos dados de entrada concatenados.

5. Aparelho, de acordo com a reivindicação 4, **CARACTERIZADO** pelo fato de que compreende ainda:

um meio para analisar gramaticalmente os ditos dados de entrada concatenados em duas unidades separadas de dados;

uma pluralidade de registros de correlação;

um adicionador e um alterador de sinal para executar uma correlação por um de adicionar os ditos dados analisados gramaticalmente aos dados em um da dita pluralidade de registros de correlação e subtrair os ditos dados analisados gramaticalmente em uma da dita pluralidade de registros de correlação;

um segundo multiplexador para multiplexar a saída dos dito adicionador e alterador de sinal com a dita pluralidade de registros de correlação;

um terceiro multiplexador para multiplexar a saída dos dito adicionador e alterador de sinal com a dita pluralidade de registros de correlação;

um adicionador para acumular os valores correla-

cionados armazenados na dita pluralidade de registros de correlação;

uma pluralidade de registros de saída de correlação;

um quarto multiplexador para multiplexar os ditos valores correlacionados acumulados em uma da dita pluralidade de registros de saída de correlação;

um quinto multiplexador para transferir os ditos valores correlacionados acumulados dos ditos registros de saída de correlação;

um gerador de índices para gerar um índice de código de sincronização primária;

uma unidade de armazenamento de sincronização primária para armazenar uma seqüência de sincronização primária; e

uma unidade de controle para controlar um processo de correlação.

6. Aparelho, de acordo com a reivindicação 5, **CARACTERIZADO** pelo fato de que a dita unidade de sincronização primária disponibiliza a dita seqüência de sincronização primária para o dito adicionador e alterador de sinal, para determinar se os ditos dados analisados gramaticalmente são adicionados ou subtraídos dos ditos dados em um da dita pluralidade de registros de correlação.

7. Aparelho, de acordo com a reivindicação 4, **CARACTERIZADO** pelo fato de que a dita memória é uma memória de porta dupla, que é escrita por uso de um ponteiro de escrita e lida por uso de um ponteiro de leitura.

8. Aparelho, de acordo com a reivindicação 5, **CARACTERIZADO** pelo fato de que os ditos processos de registros de correlação se correlacionam simultaneamente.

9. Aparelho, de acordo com a reivindicação 5, **CARACTERIZADO** pelo fato de que compreende ainda um meio para executar uma função de valor absoluto nos ditos valores correlacionados acumulados.

10. Aparelho, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que é um dispositivo móvel.

11. Método para conduzir uma correlação de segundo estágio em dados, **CARACTERIZADO** pelo fato de que compreende:

restaurar um ponteiro de leitura e um ponteiro de escrita;

multiplexar, alternativamente, os dados de entrada em um de um par de registros de armazenamento;

concatenar o conteúdo do dito par de registros de armazenamento;

escrever o dito conteúdo concatenado em uma memória, de acordo com o dito ponteiro de escrita;

transferir o dito conteúdo concatenado da dita memória para um registro de leitura, de acordo com o dito ponteiro de leitura;

atualizar o dito ponteiro de endereço de leitura; e

atualizar o dito ponteiro de endereço de escrita.

12. Método, de acordo com a reivindicação 11, **CARACTERIZADO** pelo fato de que compreende ainda:

limpar uma pluralidade de registros de correlação;

atualizar a dita pluralidade de registros de correlação com dados no dito registro de leitura;

armazenar os valores correlacionados acumulados em uma pluralidade de registros de saída de correlação; e

executar uma função de valor absoluto nos ditos valores correlacionados acumulados armazenados na dita pluralidade de registros de saída de correlação.

13. Método, de acordo com a reivindicação 12, **CARACTERIZADO** pelo fato de que a dita atualização da dita pluralidade de registros de correlação com os dados, em um registro de leitura, é conduzida com base em um sinal de uma seqüência de códigos de sincronização primária.

14. Método, de acordo com a reivindicação 13, **CARACTERIZADO** pelo fato de que a dita atualização da dita pluralidade de registros de correlação com os dados, em um registro de leitura, executa correlações paralelas.

15. Método, de acordo com a reivindicação 12, **CARACTERIZADO** pelo fato de que a dita atualização da dita pluralidade de registros de correlação com os dados, em um registro de leitura, executa correlações por uma de adição e subtração de dados no dito registro de leitura com os dados armazenados na dita pluralidade de registros de correlação.

16. Método para executar uma busca de células primária, **CARACTERIZADO** pelo fato de que compreende:

executar sincronização de temporização;

executar sincronização de quadros; e

determinar um grupo de códigos de embaralhamento e um código de embaralhamento dentro do dito grupo de códigos

de embaralhamento, em que o dito ato de determinação compreende ainda:

restaurar um ponteiro de leitura e um ponteiro de escrita;

multiplexar, alternativamente, os dados de entrada em um de um par de registros de armazenamento;

concatenar o conteúdo do dito par de registros de armazenamento;

escrever o dito conteúdo concatenado em uma memória, de acordo com o dito ponteiro de escrita;

transferir o dito conteúdo concatenado da dita memória para um registro de leitura, de acordo com o dito ponteiro de leitura;

atualizar o dito ponteiro de endereço de leitura; e

atualizar o dito ponteiro de endereço de escrita.

17. Método, de acordo com a reivindicação 16, **CARACTERIZADO** pelo fato de que compreende ainda:

limpar uma pluralidade de registros de correlação;

atualizar a dita pluralidade de registros de correlação com dados no dito registro de leitura;

armazenar os valores correlacionados acumulados em uma pluralidade de registros de saída de correlação; e

executar uma função de valor absoluto nos ditos valores correlacionados acumulados armazenados na dita pluralidade de registros de saída de correlação.

18. Método, de acordo com a reivindicação 16, **CARACTERIZADO** pelo fato de que a dita atualização da dita

pluralidade de registros de correlação com os dados, em um registro de leitura, é conduzida com base em um sinal de uma seqüência de códigos de sincronização primária.

19. Método, de acordo com a reivindicação 18, **CARACTERIZADO** pelo fato de que a dita atualização da dita pluralidade de registros de correlação com os dados, em um registro de leitura, executa correlações paralelas.

20. Método, de acordo com a reivindicação 17, **CARACTERIZADO** pelo fato de que a dita atualização da dita pluralidade de registros de correlação com os dados, em um registro de leitura, executa correlações por uma de adição e subtração de dados no dito registro de leitura com os dados armazenados na dita pluralidade de registros de correlação.

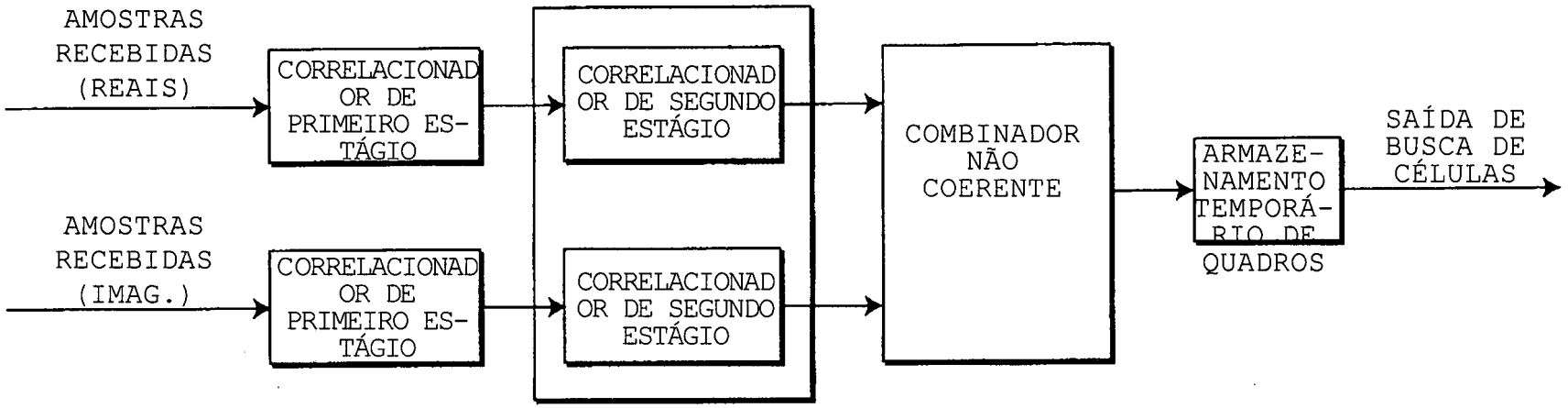

AMOSTRAS RECEBIDAS (REAIS)
AMOSTRAS RECEBIDAS (IMAG.)
CORRELACIONADOR DE PRIMEIRO ESTÁGIO
CORRELACIONADOR DE PRIMEIRO ESTÁGIO
CORRELACIONADOR DE SEGUNDO ESTÁGIO
CORRELACIONADOR DE SEGUNDO ESTÁGIO
COMBINADOR NÃO COERENTE
ARMAZENAMENTO TEMPORÁRIO DE QUADROS
SAÍDA DE BUSCA DE CÉLULAS

FIG. 1

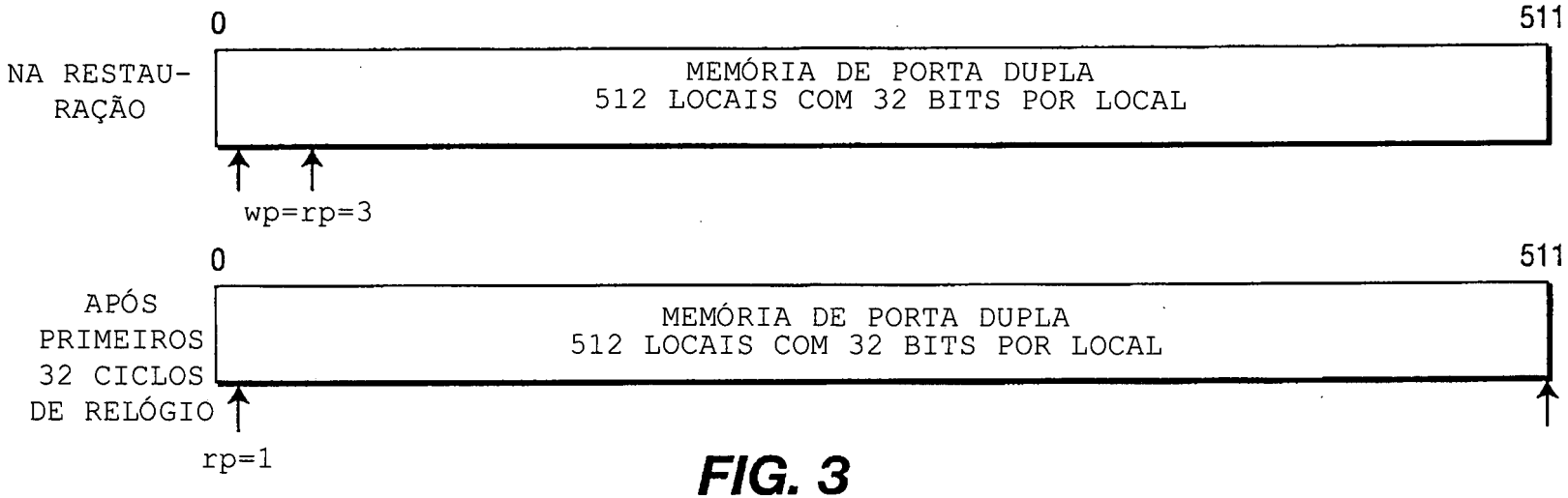

0
511
NA RESTAURAÇÃO
MEMÓRIA DE PORTA DUPLA
512 LOCAIS COM 32 BITS POR LOCAL
wp=rp=3
0
511
APÓS PRIMEIROS 32 CICLOS DE RELÓGIO
MEMÓRIA DE PORTA DUPLA
512 LOCAIS COM 32 BITS POR LOCAL
rp=1

FIG. 3

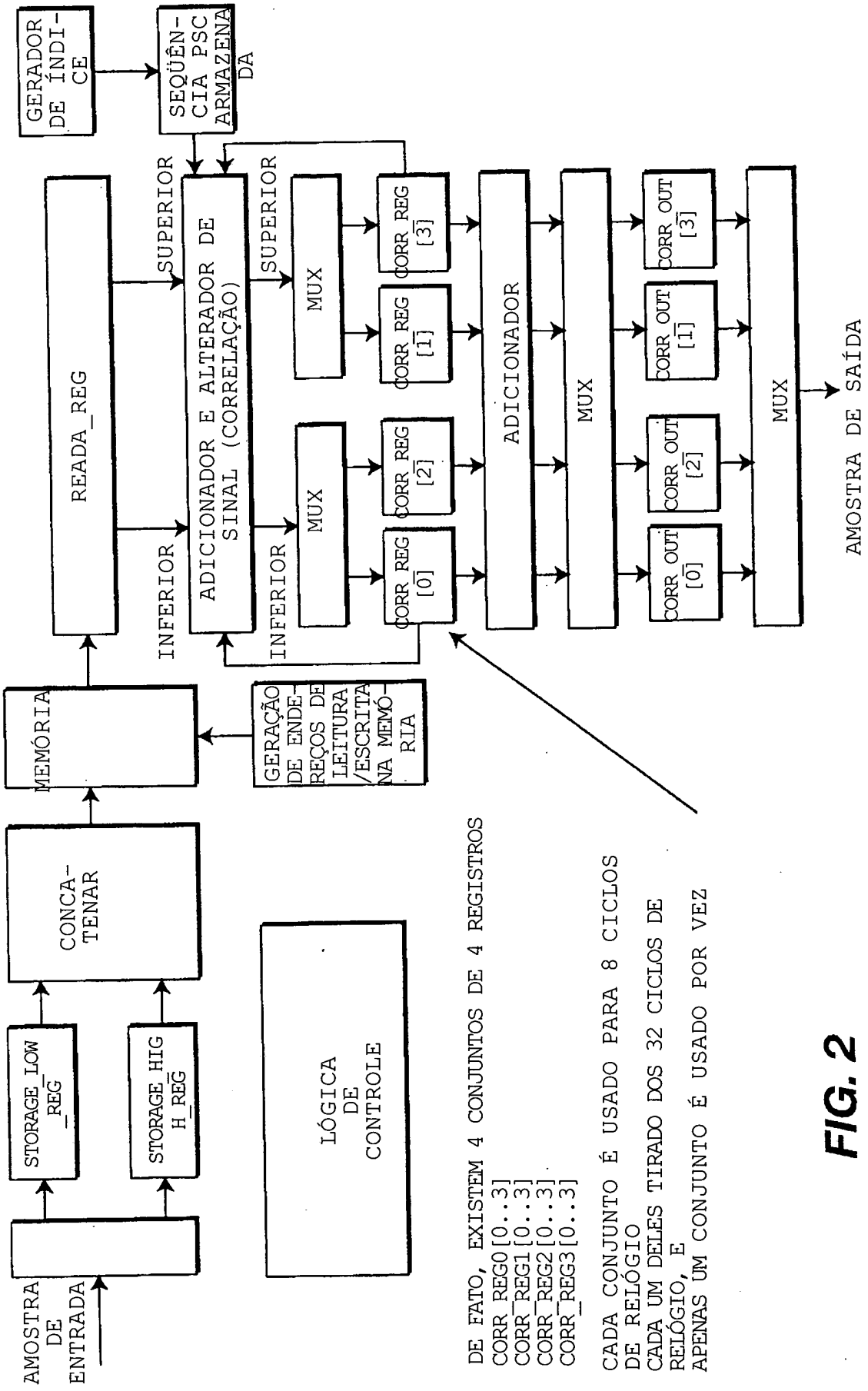

AMOSTRA DE ENTRADA
STORAGE_LOW_REG
STORAGE_HIG H_REG
CONCA-TENAR
MEMÓRIA
GERAÇÃO DE ENDE-REÇOS DE LEITURA /ESCRITA NA MEMÓ-RIA
LÓGICA DE CONTROLE
READA_REG
INFERIOR
SUPERIOR
ADICIONADOR E ALTERADOR DE SINAL (CORRELAÇÃO)
INFERIOR
SUPERIOR
MUX
MUX
CORR REG [0]
CORR REG [2]
CORR REG [1]
CORR REG [3]
ADICIONADOR
MUX
CORR_OUT [0]
CORR_OUT [2]
CORR_OUT [1]
CORR_OUT [3]
MUX
AMOSTRA DE SAÍDA
GERADOR DE ÍNDI-CE
SEQÜÊN-CIA PSC ARMAZENA DA
DE FATO, EXISTEM 4 CONJUNTOS DE 4 REGISTROS
CORR_REG0[0..3]
CORR_REG1[0..3]
CORR_REG2[0..3]
CORR_REG3[0..3]
CADA CONJUNTO É USADO PARA 8 CICLOS DE RELÓGIO
CADA UM DELES TIRADO DOS 32 CICLOS DE RELÓGIO, E
APENAS UM CONJUNTO É USADO POR VEZ

FIG. 2

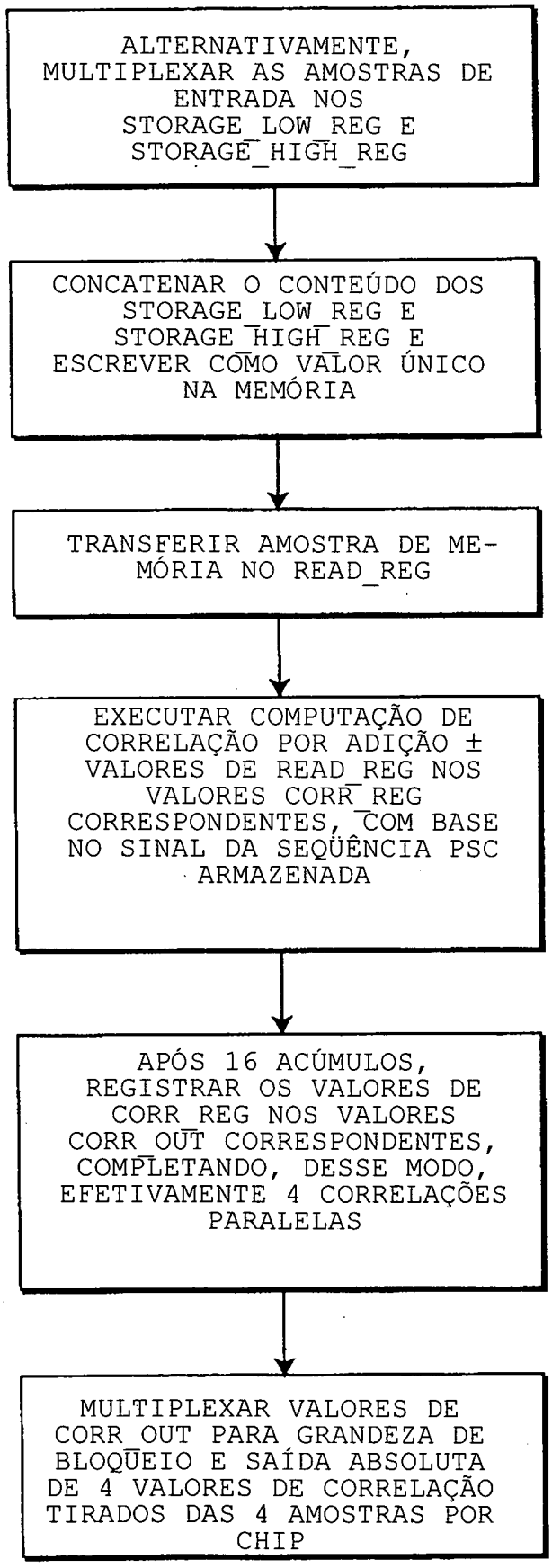

ALTERNATIVAMENTE, MULTIPLEXAR AS AMOSTRAS DE ENTRADA NOS STORAGE_LOW_REG E STORAGE_HIGH_REG
CONCATENAR O CONTEÚDO DOS STORAGE_LOW_REG E STORAGE_HIGH_REG E ESCREVER COMO VALOR ÚNICO NA MEMÓRIA
TRANSFERIR AMOSTRA DE MEMÓRIA NO READ_REG
EXECUTAR COMPUTAÇÃO DE CORRELAÇÃO POR ADIÇÃO ± VALORES DE READ_REG NOS VALORES CORR_REG CORRESPONDENTES, COM BASE NO SINAL DA SEQÜÊNCIA PSC ARMAZENADA
APÓS 16 ACÚMULOS, REGISTRAR OS VALORES DE CORR_REG NOS VALORES CORR_OUT CORRESPONDENTES, COMPLETANDO, DESSE MODO, EFETIVAMENTE 4 CORRELAÇÕES PARALELAS
MULTIPLEXAR VALORES DE CORR_OUT PARA GRANDEZA DE BLOQUEIO E SAÍDA ABSOLUTA DE 4 VALORES DE CORRELAÇÃO TIRADOS DAS 4 AMOSTRAS POR CHIP

FIG. 4

RESUMO

"CORRELACIONADOR PARA BUSCA DE CÉLULAS PRIMÁRIAS USANDO ARQUITETURA DE MEMÓRIA"

Um aparelho incluindo um correlacionador de segundo estágio, para receber dados de entrada de um correlacionador de primeiro estágio, em que o dito correlacionador de segundo estágio usa uma arquitetura de memória, é descrito. Um método para conduzir uma correlação de segundo estágio nos dados, incluindo restauração de um ponteiro de leitura e de um ponteiro de escrita, alternativamente, multiplexação dos dados de entrada em um de um par de registros de armazenamento, concatenação do conteúdo do par de registros de armazenamento, escrita do conteúdo concatenado em uma memória de acordo com o ponteiro de escrita, transferência do conteúdo concatenado da memória para um registro de leitura, de acordo com o ponteiro de leitura, atualização do ponteiro de endereço de leitura e atualização do ponteiro de endereço de escrita, é também descrito.