

(12) FASCÍCULO DE PATENTE DE INVENÇÃO

(22) Data de pedido: 2009.06.03	(73) Titular(es): LG ELECTRONICS INC.	
(30) Prioridade(s): 2009.02.18 US 153310 P	20, YEOUIDO-DONG, YEONGDEUNGPO-GU	
(43) Data de publicação do pedido: 2010.08.25	SEOUL 150-721	KR
(45) Data e BPI da concessão: 2012.05.30 134/2012	(72) Inventor(es): WOO SUK KO SANG CHUL MOON	KR KR
	(74) Mandatário: LUÍS MANUEL DE ALMADA DA SILVA CARVALHO RUA VÍCTOR CORDON, 14 1249-103 LISBOA	PT

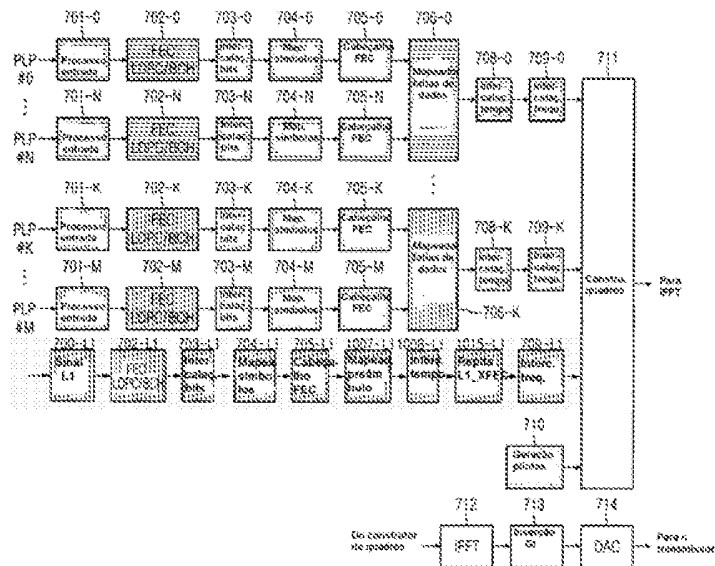
(54) Epígrafe: **DISPOSITIVO E PROCESSO PARA A TRANSMISSÃO E RECEPÇÃO DE UM SINAL DE RADIODIFUSÃO**

(57) Resumo:

A PRESENTE INVENÇÃO REFERE-SE A UM PROCESSO PARA A TRANSMISSÃO E RECEPÇÃO DE SINAIS E A UM DISPOSITIVO CORRESPONDENTE. UM ASPECTO DA PRESENTE INVENÇÃO REFERE-SE A UM PROCESSO DE PROCESSAMENTO EFICIENTE DE CAMADA 1 (L1) PARA UM TRANSMISSOR E RECEPTOR UTILIZANDO FATIAS DE DADOS.

RESUMO**"DISPOSITIVO E PROCESSO PARA A TRANSMISSÃO E RECEPÇÃO DE UM SINAL DE RADIODIFUSÃO"**

A presente invenção refere-se a um processo para a transmissão e recepção de sinais e a um dispositivo correspondente. Um aspecto da presente invenção refere-se a um processo de processamento eficiente de camada 1 (L1) para um transmissor e receptor utilizando fatias de dados.



DESCRIÇÃO

"DISPOSITIVO E PROCESSO PARA A TRANSMISSÃO E RECEPÇÃO DE UM SINAL DE RADIODIFUSÃO"

Antecedentes da invenção

Campo da invenção

A presente invenção refere-se a um processo para a transmissão e recepção de um sinal e a um dispositivo para a transmissão e recepção de um sinal, e mais particularmente, a um processo para a transmissão e recepção de um sinal e a um dispositivo para a transmissão e recepção de um sinal, que são passíveis de melhorar a eficiência da transmissão de dados.

Descrição da técnica relacionada

Com o desenvolvimento da tecnologia de transmissão digital, os utilizadores passaram a receber uma imagem em movimento de alta definição (*HD*). Com o desenvolvimento contínuo de um algoritmo de compressão e alto desempenho de hardware, será proporcionado aos utilizadores um melhor ambiente no futuro. Um sistema de televisão digital (*DTV*) pode receber um sinal de radiodifusão digital e fornecer uma variedade de serviços adicionais aos utilizadores, bem como um sinal de vídeo e

um sinal de áudio. A menção na Internet da *ETSI* sobre a rádio difusão vídeo digital (*DVB*) "*Frame structure channel coding and modulation for a second generation digital terrestrial television broadcasting system (DVB-T2)*", Outubro 2008, descreve a codificação e modulação do canal para serviços de televisão digital.

A radiodifusão de vídeo digital (*Digital Video Broadcasting (DVB) - C2*) é a terceira especificação a unir-se à família *DVB* de sistemas de transmissão de segunda geração. Desenvolvida em 1994, a *DVB-C* actual encontra-se implantada em mais de 50 milhões de sintonizadores de cabo em todo o mundo. Em consonância com os outros sistemas de segunda geração *DVB*, o *DVB-C2* utiliza uma combinação de códigos de baixa densidade com controlo de paridade (*Low-density parity-check - LDPC*) e *BCH*. Esta correcção poderosa de erros de reencaminhamento (*Forward Error correction - FEC*) proporciona cerca de 5dB de melhoria do rácio portadora-ruído em relação ao *DVB-C*. Esquemas adequados de intercalamento de bits optimizam a robustez geral do sistema *FEC*. Prolongado por um cabeçalho, esses quadros são denominados de condutas de camada física (*Physical Layer Pipes - PLP*). Um ou mais dessas *PLPs* são multiplexadas transformando-se numa fatia de dados. Dois intercalamentos dimensionais (nos domínios do tempo e da frequência) são aplicados a cada parcela permitindo ao receptor eliminar o impacto das deficiências das rajadas e da interferência selectiva da frequência tal como entrada de uma única frequência.

Com o desenvolvimento dessas tecnologias de radiodifusão digital, aumentou a exigência por um serviço, tal como um sinal de vídeo e um sinal de áudio e a dimensão dos dados desejados pelos utilizadores ou o número de canais de transmissão aumentou gradualmente.

Sumário da invenção

Sendo assim, a presente invenção refere-se a um processo para a transmissão e recepção de um sinal e a um dispositivo para a transmissão e recepção de um sinal que substancialmente evita um ou mais problemas devido às limitações e desvantagens da técnica relacionada.

Um objectivo da presente invenção é o de proporcionar um processo para a transmissão e recepção de um sinal e um dispositivo para a transmissão e recepção de um sinal, que são passíveis de melhorar a eficiência da transmissão de dados.

Um outro objectivo da presente invenção é o de proporcionar um processo para a transmissão e recepção de um sinal e um dispositivo para a transmissão e recepção de um sinal, que são passíveis de melhorar a capacidade de correcção de erro de bits que configuram um serviço.

As vantagens adicionais, objectos e características da invenção serão apresentados em parte na descrição que se segue e em parte se tornarão evidentes

para o técnico após análise do seguinte. Os objectivos e outras vantagens da invenção podem ser realizados e alcançados pela estrutura especialmente assinalada na descrição e reivindicações da mesma bem como nos desenhos anexos.

Para alcançar estes objectivos, a presente invenção proporciona um transmissor para transmitir dados de radiodifusão para um receptor, compreendendo o transmissor: um codificador *FEC* (*Forward Error Correction*) configurado para codificar *FEC* dados de sinalização de camada 1; um intercalador de bits configurado para bit intercalar os dados de sinalização de camada 1 codificados em *FEC*; um mapeador *QAM* configurado para demultiplexar os dados de sinalização da camada 1 bit intercalados em palavras de células e para mapear as palavras de células para valores de constelação que correspondem aos dados de sinalização da camada 1; um intercalador no tempo configurado para intercalar no tempo os valores de constelação mapeados que correspondem aos dados de sinalização da camada 1; um insersor configurado para inserir um cabeçalho de camada 1 nos valores de constelação intercalados no tempo que correspondem aos dados de sinalização da camada 1; um meio de repetição configurado para repetir os valores de constelação que correspondem aos dados de sinalização da camada 1 e ao cabeçalho da camada 1; e um intercalador de frequência configurado para intercalar na frequência os valores de constelação repetidos que correspondem aos dados de sinalização da

camada 1 e cabeçalho da camada 1.

Uma outra forma de realização da presente invenção proporciona um receptor para processar dados de transmissão, compreendendo o receptor: um desintercalador de frequência configurado para desintercaladar na frequência valores de constelação que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1; um extrator configurado para extrair os valores de constelação que correspondem aos dados de sinalização da camada 1 dos valores de constelação desintercalados na frequência que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1; um desintercalador no tempo configurado para desintercaladar no tempo os valores de constelação extraídos que correspondem aos dados de sinalização da camada 1; um desmapeador *QAM* configurado para desmapear valores de constelação que correspondem aos dados de sinalização da camada 1 em dados de sinalização da camada 1; um desintercalador de bits configurado para bit deintercaladar os dados de sinalização da camada 1 desmapeado; e um decodificador *FEC* (*Forward Error Correction*) configurado para decodificar *FEC* os dados de sinalização da camada 1.

Ainda uma outra forma de realização da presente invenção proporciona um processo para receber dados de transmissão, compreendendo o processo: desintercaladar na frequência valores de constelação que correspondem a dados de sinalização da camada 1 e a um cabeçalho de camada 1;

extrair os valores da constelação que correspondem aos dados de sinalização da camada 1 dos valores constelação desintercalados na frequência que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1; desintercalar no tempo os valores de constelação extraídos que correspondem aos dados de sinalização da camada 1; desmapear os valores de constelação desintercalados no tempo que correspondem aos de dados de sinalização da camada 1 para os dados de sinalização da camada 1; bit desintercalar os dados de sinalização desmapeados da camada 1; e descodificação *FEC* dos dados de sinalização da camada 1.

Ainda uma outra forma de realização da presente invenção proporciona um processo para transmitir dados de emissão para um receptor, compreendendo o processo: Dados de sinalização de camada 1 da codificação *FEC*; intercalar bits nos dados de sinalização de camada 1 codificados em *FEC*; desmultiplexar os dados de sinalização bit intercalados da camada 1 em palavras de células; mapear as palavras de células em valores que correspondem aos dados de sinalização da camada 1; intercalar no tempo valores de constelação mapeados que correspondem aos dados de sinalização da camada 1; inserir um cabeçalho de camada 1 nos valores de constelação intercalados no tempo que correspondem aos dados de sinalização da camada 1; repetir os valores de constelação que correspondem aos dados de sinalização de camada 1 e cabeçalho da camada 1; e intercalar na frequência os valores das constelações

repetidos que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1.

Descrição das formas de realização preferidas

Os desenhos anexos, que se encontram incluídos para proporcionar uma maior compreensão da invenção, constituindo uma parte deste pedido, ilustram formas de realização da invenção e juntamente com a descrição servem para explicar o princípio da invenção. As figuras representam:

Figura 1 exemplo de sistema de transmissão digital.

Figura 2 exemplo de um processador de entrada.

Figura 3 informação que pode ser incluída na banda base (BB).

Figura 4 exemplo de módulo *BICM*.

Figura 5 exemplo de codificador encurtado/puncionado.

Figura 6 exemplo de aplicação de várias constelações.

Figura 7 outro exemplo de casos onde é considerada a compatibilidade entre os sistemas convencionais.

Figura 8 estrutura de quadro que compreende o preâmbulo para a sinalização *L1* e símbolo de dados para dados *PLP*.

Figura 9 exemplo de construtor de quadros.

Figura 10 exemplo de módulo de inserção de piloto 404 tal como apresentado na figura 4.

Figura 11 estrutura de *SP*.

Figura 12 nova estrutura *SP* ou padrão piloto (*PP5'*).

Figura 13 estrutura PP5' sugerida.

Figura 14 relacionamento entre símbolo de dados e preâmbulo.

Figura 15 outro relacionamento entre símbolo de dados e preâmbulo.

Figura 16 exemplo de perfil de atrasos de canal por cabo.

Figura 17 estrutura piloto dispersa que utiliza $z=56$ e $z=112$.

Figura 18 exemplo de modulador baseado em *OFDM*.

Figura 19 exemplo de estrutura de preâmbulo.

Figura 20 exemplo de descodificação do preâmbulo.

Figura 21 processo para conceber preâmbulo mais otimizado.

Figura 22 outro exemplo de estrutura de preâmbulo

Figura 23 outro exemplo de descodificação de preâmbulo.

Figura 24 exemplo de estrutura de preâmbulo.

Figura 25 exemplo de descodificação *L1*.

Figura 26 exemplo de processador analógico.

Figura 27 exemplo de sistema de recepção digital.

Figura 28 exemplo de processador analógico utilizado no receptor.

Figura 29 exemplo de desmodulador.

Figura 30 exemplo de analisador sintáctico de quadros.

Figura 31 exemplo de desmodulador *BICM*.

Figura 32 exemplo de descodificação *LDPC* utilizando encurtamento / punção.

Figura 33 exemplo de processador de saída.

Figura 34 exemplo de taxa de repetição de bloco *L1* de 8 MHz.

Figura 35 exemplo de taxa de repetição de bloco *L1* de 8

MHz.

Figura 36 nova taxa de repetição de bloco *L1* de 7.61 MHz.

Figura 37 exemplo de sinalização *L1* que é transmitida no cabeçalho do quadro.

Figura 38 resultado do preâmbulo e simulação da estrutura *L1*.

Figura 39 exemplo de intercalador de símbolos.

Figura 40 exemplo de uma transmissão de bloco *L1*.

Figura 41 outro exemplo de sinalização *L1* transmitida dentro de um cabeçalho de quadro.

Figura 42 um exemplo de intercalamento/desintercalamento de frequência ou tempo.

Figura 43 quadro analisando a informação complementar da sinalização *L1*, que é transmitida no cabeçalho *FECFRAME* no módulo de inserção de cabeçalho *ModCod* 307 no trajecto de dados do módulo *BICM* apresentado na figura 3.

Figura 44 estrutura para cabeçalho *FECFRAME* para minimizar a informação complementar.

Figura 45 desempenho da taxa de erro de bit (*BER*) da protecção *L1* acima mencionada.

Figura 46 exemplos de um quadro de transmissão e estrutura de quadro *FEC*.

Figura 47 exemplo de sinalização *L1*.

Figura 48 exemplo de pré-sinalização *L1*.

Figura 49 estrutura do bloco de sinalização *L1*.

Figura 50 intercalação no tempo *L1*.

Figura 51 exemplo de extracção de informação de modulação e código.

Figura 52 outro exemplo de pré-sinalização *L1*.

Figura 53 exemplo de agendamento de bloco de sinalização *L1* que é transmitido no preâmbulo.

Figura 54 exemplo de pré-sinalização *L1* onde é considerado um reforço da potência.

Figura 55 exemplo de sinalização *L1*.

Figura 56 outro exemplo de extracção de informação de modulação e código.

Figura 57 outro exemplo de extracção de informação de modulação e código.

Figura 58 exemplo de pré-sincronização *L1*.

Figura 59 exemplo de pré-sinalização *L1*.

Figura 60 exemplo de sinalização *L1*.

Figura 61 exemplo de trajecto de sinalização *L1*.

Figura 62 outro exemplo de sinalização *L1* transmitida dentro de um cabeçalho de quadro.

Figura 63 outro exemplo de sinalização *L1* transmitida dentro de um cabeçalho de quadro.

Figura 64 outro exemplo de sinalização *L1* transmitida dentro de um cabeçalho de quadro.

Figura 65 exemplo de sinalização *L1*.

Figura 66 exemplo de intercalador de símbolos.

Figura 67 desempenho de intercalação do intercalador no tempo da figura 66.

Figura 68 exemplo de intercalador de símbolos.

Figura 69 desempenho de intercalação do intercalador no tempo da figura 68.

Figura 70 exemplo de desintercalador de símbolo.

Figura 71 outro exemplo de intercalador no tempo.

Figura 72 resultado de intercalação utilizando o processo apresentado na figura 71.

Figura 73 exemplo de processo de endereçamento da figura 72.

Figura 74 outro exemplo de intercalação no tempo *L1*.

Figura 75 exemplo de desintercalador de símbolo.

Figura 76 outro exemplo de desintercalador.

Figura 77 exemplo de desintercalador de símbolo.

Figura 78 exemplo de endereços de linha e coluna para desintercalação no tempo.

Figura 79 exemplo de intercalação geral em bloco num domínio de símbolo de dados onde os pilotos não são utilizados.

Figura 80 exemplo de um transmissor *OFDM* que utiliza fatia de dados.

Figura 81 exemplo de um receptor *OFDM* que utiliza fatia de dados.

Figura 82 exemplo de intercalador no tempo e um exemplo de desintercalador no tempo.

Figura 83 exemplo de símbolos que formam *OFDM*.

Figura 84 exemplo de intercalador no tempo (*Time Interleaver - TI*).

Figura 85 exemplo de intercalador no tempo (*Time Interleaver - TI*).

Figura 86 exemplo de uma estrutura de preâmbulo num transmissor e um exemplo de um processo num receptor.

Figura 87 exemplo de um processo num receptor para obter *L1_XFEC_FRAME* do preâmbulo.

Figura 88 exemplo de uma estrutura de preâmbulo num transmissor e um exemplo de um processo num receptor.

Figura 89 exemplo de um intercalador no tempo (*Time Interleaver - TI*).

Figura 90 exemplo de um transmissor *OFDM* que utiliza fatia de dados.

Figura 91 exemplo de um receptor *OFDM* que utiliza fatias de dados.

Figura 92 exemplo de intercalador no tempo (*Time Interleaver - TI*).

Figura 93 exemplo de um desintercalador no tempo (*Time De-Interleaver - TDI*).

Figura 94 exemplo de intercalador no tempo (*Time Interleaver - TI*).

Figura 95 exemplo de fluxo de intercalação e desintercalação no tempo de preâmbulos.

Figura 96 parâmetro de profundidade de intercalação no tempo na sinalização *L1* de cabeçalho.

Figura 97 exemplo de uma sinalização *L1* de cabeçalho, estrutura *L1*, e um processo de zona de preenchimento.

Figura 98 exemplo de sinalização *L1*.

Figura 99 exemplo de *dslice_ti_depth*.

Figura 100 exemplo de *dslice_type*.

Figura 101 exemplo de *plp_type*.

Figura 102 exemplo de *Plp_payload_type*.

Figura 103 exemplo de *Plp_ModCod*.

Figura 104 exemplo de *GI*.

- Figura 105 exemplo de *PAPR*.
- Figura 106 exemplo de sinalização *L1*.
- Figura 107 exemplo de *plp_type*.
- Figura 108 exemplo de sinalização *L1*.
- Figura 109 exemplo de uma sinalização *L1* de cabeçalho, estrutura *L1*, e um processo de zona de preenchimento.
- Figura 110 exemplo de sinalização *L1*.
- Figura 111 exemplo de campos de sinalização *L1*.
- Figura 112 exemplo de sinalização *L1*.
- Figura 113 exemplo de *plp_type*.
- Figura 114 exemplo de sinalização *L1* e sinalização *L2* para tipos de *PLP* normal e integrado.
- Figura 115 exemplo de fluxo de acção de descodificação *L1* e *L2* de um receptor *DVB-C2* convencional com sintonizador individual de 8MHz.
- Figura 116 exemplo de fluxo de acção de descodificação *L1* e *L2* de um receptor *DVB-C2* muito apreciado com vários sintonizadores ou um único sintonizador de banda larga.
- Figura 117 exemplo de uma sinalização *L2* para *C2*.
- Figura 118 exemplo de duração do símbolo *OFDM* activo.
- Figura 119 exemplo de valores de intervalo de guarda.
- Figura 120 exemplo de sinalização *L1*
- Figura 121 exemplo de intercalador no tempo de bloco *L1*.
- Figura 122 exemplo de um transmissor *OFDM* que utiliza fatia de dados.
- Figura 123 exemplo de um receptor *OFDM* que utiliza

fatia de dados.

Figura 124 exemplo de um fluxo de processamento de dados *L1* de um transmissor.

Figura 125 exemplo de um fluxo de processamento de dados *L1* de um receptor.

Figura 126 exemplo de um processo de intercalação no tempo *L1* de um transmissor.

Figura 127 exemplo de um processo de desintercalação no tempo *L1* de um receptor.

Descrição das formas de realização preferidas

Será feita agora referência pormenorizada às formas de realização preferidas da presente invenção, exemplos esses que se encontram ilustrados nos desenhos anexos. Sempre que possível, serão utilizados os mesmos números de referência em todos os desenhos para designar as mesmas ou partes semelhantes.

Na descrição a seguir, o termo "serviço" é indicativo de qualquer conteúdo difundido que pode ser transmitido/recebido pelo dispositivo de transmissão/recepção de sinal.

A figura 1 mostra um exemplo de sistema de transmissão digital de acordo com uma forma de realização da presente invenção. As entradas podem incluir um número de fluxos *MPEG-TS* ou fluxos *GSE* (*General Stream Encapsulation*). Um processador de entrada 101 pode

adicionar parâmetros de transmissão ao fluxo de entrada e executar agendamentos para um módulo *BICM* 102. O módulo *BICM* 102 pode adicionar redundância e intercalar dados para correcção de erros de canal de transmissão. Um construtor de quadros 103 pode construir quadros adicionando a informação de sinalização da camada física e pilotos. Um modulador 104 pode executar a modulação nos símbolos de entrada em processos eficientes. Um processador analógico 105 pode executar vários processos para converter sinais de entrada digitais em sinais de saída analógicos.

A figura 2 mostra um exemplo de um processador de entrada. O fluxo de entrada *MPEG-TS* ou *GSE* pode ser transformado pelo pré-processador de entrada num total de n fluxos que serão processados independentemente. Cada um destes fluxos pode ser um quadro *TS* completo que inclui componentes de vários serviços ou um quadro *TS* mínimo que inclui componente de serviço (isto é vídeo ou áudio). Além disso, cada um destes fluxos pode ser um fluxo *GSE* que transmite vários serviços ou um único serviço.

A interface de entrada 202-1 pode atribuir vários bits de entrada iguais à capacidade máxima do campo de dados de um quadro de banda base (*BB*). Uma zona de preenchimento pode ser inserida para completar a capacidade do bloco de código *LDPC/BCH*. O sincronizador de fluxo de entrada 203-1 pode proporcionar um mecanismo para regenerar, no receptor, o relógio do fluxo de transporte (ou fluxo genérico em pacotes), a fim de garantir taxas e

atrasos de bits constantes extremidade-a-extremidade.

De modo a permitir a recombinação do fluxo de transporte sem exigir memória adicional no receptor, os fluxos de transporte de entrada são atrasados por compensadores de atraso 204-1~n considerando os parâmetros do intercalamento dos *PLPs* de dados num grupo e o *PLP* comum correspondente. O módulo de apagar 205-1~n de pacote de tamanho zero pode aumentar a eficiência de transmissão através da remoção de pacotes de tamanho zero inseridos em caso de serviço *VBR* (*variable bit rate* - taxa de bits variável). Os módulos codificadores de verificação cíclica de redundância (*Cyclic Redundancy Check* - *CRC*) 206-1 podem adicionar paridade *CRC* para aumentar a confiabilidade da transmissão de quadro *BB*. Os módulos de inserção de cabeçalho (207-1~n) *BB* podem adicionar o cabeçalho do quadro *BB* numa parte inicial do quadro *BB*. A informação que pode ser incluída no cabeçalho *BB* encontra-se apresentada na figura 3.

Um módulo de fusão/corte 208 pode executar o corte em fatias do quadro *BB* de cada *PLP*, fundindo quadros *BB* de vários *PLPs*, e agendando cada quadro *BB* dentro de um quadro de transmissão. Por isso, o módulo de fusão/corte 208 pode emitir informação de sinalização *L1* que se refere à atribuição de *PLP* no quadro. Por último, um módulo misturador *BB* 209 pode randomizar fluxos de bits de entrada para minimizar a correlação entre os bits dentro de fluxos de bits. Os módulos a sombreado na figura 2 são módulos

usados quando o sistema de transmissão utiliza um único *PLP*, sendo os outros módulos na figura 2 módulos usados quando o dispositivo de transmissão utiliza vários *PLPs*.

A figura 4 mostra uma forma de realização do módulo *BICM* de acordo com a presente invenção. A figura 4a mostra um *BICM* para um trajecto de dados e A figura 4b mostra um *BICM* para um trajecto de sinalização *L1*.

Em relação agora à figura 4a, um codificador externo 301 e um codificador interno 303 podem adicionar redundância a fluxos de bits de entrada para a correcção de erros. Um intercalador externo 302 e um intercalador interno 304 podem intercalar bits para evitar erro de rajada. O intercalador externo 302 pode ser omitido se o *BICM* for especificamente para *DVB-C2*. Um desmultiplexador de bits 305 pode controlar a fiabilidade de cada bit emitido pelo intercalador interno 304. Um mapeador de símbolo 306 pode mapear os fluxos de bits de entrada para fluxos de símbolos. Nesse momento, é possível utilizar qualquer de uma *QAM* convencional, uma *MQAM* que utiliza o *BRGC* acima descrito para a melhoria do desempenho, uma *NU-QAM* que utiliza modulação não-uniforme, ou uma *NU-MQAM* que utiliza *BRGC* aplicado de modulação não-uniforme para melhoria do desempenho. Para construir um sistema que é mais robusto contra o ruído, podem ser consideradas as combinações de modulações que utilizam *MQAM* e/ou *NU-MQAM* dependendo da taxa de código do código de correcção de erros e da capacidade de constelação. Neste momento, o

mapeador de símbolo 306 pode utilizar uma constelação adequada de acordo com a taxa de código e capacidade de constelação. A figura 6 mostra um exemplo de tais combinações.

O caso 1 mostra um exemplo de utilização exclusiva de *NU-MQAM* à taxa de código baixa para implantação simplificada do sistema. O caso 2 mostra um exemplo de utilização de constelação otimizada a cada taxa de código. O transmissor pode enviar informação sobre a taxa de código do código de correcção de erros e sobre a capacidade da constelação para o receptor de tal forma que o receptor pode utilizar uma constelação adequada. A figura 7 mostra um outro exemplo de casos em que a compatibilidade entre os sistemas convencionais é considerada. Adicionalmente aos exemplos são possíveis combinações adicionais para otimizar o sistema.

O módulo de inserção 307 do cabeçalho *ModCod* apresentado na figura 4 pode obter informação de retorno de codificação e modulação adaptativa (*Adaptive coding and modulation - ACM*) / de codificação e modulação variável (*Variable coding and modulation - VCM*) e adicionar informação de parâmetro utilizada na codificação e modulação para um bloco *FEC* como cabeçalho. O cabeçalho do tipo de modulação/taxa de código (*ModCod*) pode incluir a seguinte informação:

* Tipo de *FEC* (1 bit) - *LDPC* longo ou curto

- * Taxa de código (3 bits)
- * Modulação (3 bits) - até 64K QAM
- * Identificador *PLP* (8 bits)

O intercalador de símbolos 308 pode realizar o intercalamento no domínio dos símbolos para obter efeitos adicionais de intercalamento. Processos semelhantes realizados no trajecto dos dados podem ser realizados no trajecto de sinalização *L1* mas possivelmente com parâmetros diferentes 301-1 - 308-1. Neste ponto pode ser utilizado um codificador encurtado/puncionado 303-1 para código interno.

A figura 5 mostra um exemplo de codificação *LDPC* utilizando encurtamento / puncionagem. O processo de encurtamento pode ser realizado em blocos de entrada que apresentam menos bits do que um número necessário de bits para a codificação *LDPC* dado que tantos zeros bits necessários para a codificação *LDPC* podem ser preenchidos pelo módulo de zona de preenchimento zero 301c. Os fluxos de bits de entrada de zonas de preenchimento zero podem apresentar bits de paridade através do codificador *LDPC* 302c. Nesse momento, para fluxos de bits que correspondem a fluxos de bits originais, os zeros podem ser removidos (303c) e para fluxos de bits de paridade, o puncionamento pode ser executado de acordo com as taxas de código pelo módulo de puncionagem de paridade 304c. Estes fluxos processados de bits de informação e fluxos de bits de paridade podem ser multiplexados em sequências originais e emitidos pelo Mux 305c.

A figura 8 mostra uma estrutura de quadro que compreende o preâmbulo para a sinalização *L1* e símbolo de dados para dados *PLP*. Pode-se observar que o preâmbulo e os símbolos de dados são gerados ciclicamente, utilizando um quadro como uma unidade. Os símbolos de dados incluem *PLP* tipo 0 que é transmitido utilizando uma modulação/codificação fixa e *PLP* tipo 1 que é transmitido utilizando uma modulação/codificação variável. Para *PLP* tipo 0, a informação tal como a modulação, tipo de *FEC*, e taxa de código *FEC* são transmitidos no preâmbulo (veja figura 9 para módulo de inserção de cabeçalho do quadro 401). Para *PLP* tipo 1, a informação correspondente pode ser transmitida num cabeçalho de bloco *FEC* de um símbolo de dados (veja figura 3 para módulo de inserção de cabeçalho *ModCod* 307). Pela separação de tipos *PLP*, a informação complementar *ModCod* pode ser reduzida em 3~4% de uma taxa de transmissão total, para *PLP* tipo 0 que é transmitido a uma taxa de bits fixa. Num receptor, para a modulação fixa/codificação *PLP* do *PLP* tipo 0, o removedor de cabeçalho do quadro r401 apresentado na figura 30 pode extrair informação sobre a modulação e taxa do código *FEC* e fornecer a informação extraída a um módulo de descodificação *BICM*. Para modulação/codificação variável *PLP* de *PLP* tipo 1, o extractor *ModCod*, r307, r307-1 apresentado na figura 31 pode extrair e fornecer os parâmetros necessários para a descodificação *BICM*.

A figura 9 mostra um exemplo de um construtor de quadros. Um módulo de inserção de cabeçalho de quadro 401

pode formar um quadro de fluxos de símbolos de entrada e pode adicionar o cabeçalho do quadro na frente de cada quadro transmitido. O cabeçalho do quadro pode incluir a seguinte informação:

- * Número de canais ligados (4 bits)
- * Intervalo de guarda (2 bits)
- * *PAPR* (2 bits)
- * Padrão piloto (2 bits)
- * Identificação do sistema digital (16 bits)
- * Identificação do quadro (16 bits)
- * Comprimento do quadro (16 bits) - número de símbolos de multiplexação ortogonal por divisão de frequências (*OFDM*) por quadro
- * Comprimento do superquadro (16 bits) - número de quadros por superquadro
- * Número de *PLPs* (8 bits)
- * Para cada *PLP*
 - PLP identification (8 bits)*
 - Channel bonding id (4 bits)*
 - PLP start (9 bits)*
 - PLP type (2 bits) - common PLP or others*
 - PLP payload type (5 bits)*
 - MC type (1 bit) - fixed/variable modulation & coding*
 - if MC type == fixed modulation & coding*
 - FEC type (1 bits) - long or short LDPC*
 - Coderate (3 bits)*
 - Modulation (3 bits) - up-to 64K QAM*

end if;

```
Number of notch channels (2 bits)
for each notch
    Notch start (9 bits)
    Notch width (9 bits)
end for;
PLP width (9 bits) - max number of FEC blocks of
PLP
PLP time interleaving type (2 bits)
end for;
* CRC-32 (32 bits)
```

O ambiente de ligação de canal é assumido para informação *L1* transmitida no cabeçalho do quadro, sendo os dados que correspondem a cada fatia de dados definidos como *PLP*. Por isso, a informação tal como o identificador *PLP*, identificador de ligação de canal, e endereço de início de *PLP* é necessária para cada canal utilizado na ligação. Uma forma de realização da presente invenção sugere transmitir o campo *ModCod* no cabeçalho do quadro *FEC* se o tipo *PLP* suportar modulação variável/codificação e transmitir campo *ModCod* no cabeçalho do quadro se o tipo *PLP* suportar modulação fixa/codificação para reduzir a informação complementar da sinalização. Adicionalmente, se existir uma banda de entalhe para cada *PLP*, ao transmitir o endereço de início do entalhe e a sua largura, podem tornar-se desnecessárias as portadoras correspondentes de descodificação no receptor.

A figura 10 mostra um exemplo de padrão piloto

(PP5) aplicado num ambiente de ligação de canal. Tal como apresentado, se as posições *SP* forem coincidentes com as posições do preâmbulo piloto, pode surgir uma estrutura piloto irregular.

A figura 10a mostra um exemplo de módulo de inserção de piloto 404 tal como apresentado na figura 9. Tal como representado na figura 10a, se for utilizada uma banda de uma só frequência (por exemplo 8 MHz), a largura de banda disponível é de 7,61 MHz, mas se se encontrarem ligadas várias bandas de frequência, as bandas de guarda pode ser removidas, sendo que, deste modo, a eficiência da frequência pode aumentar consideravelmente. A figura 10b é um exemplo do módulo de inserção do preâmbulo 504 tal como apresentado na figura 18 que é transmitido na parte frontal do quadro e mesmo com ligação de canal, o preâmbulo apresenta uma taxa de repetição de 7,61 MHz, que é a largura de banda do bloco *L1*. Esta é uma estrutura considerando a largura de banda de um sintonizador que executa o varrimento inicial do canal.

Existem padrões piloto para ambos, o preâmbulo e os símbolos de dados. Para o símbolo de dados, podem ser usados padrões piloto difundidos (*scattered pilot - SP*). Os padrões piloto (PP5) e padrões piloto (PP7) de T2 podem ser bons candidatos para a interpolação somente de frequência. PP5 tem $x=12$, $y=4$, $z=48$ para $GI=1/64$ e PP7 tem $x=24$, $y=4$, $z=96$ para $GI=1/128$. É também possível tempo de interpolação adicional para uma melhor estimativa de canal. Os padrões

piloto para preâmbulo podem cobrir todas as posições piloto possíveis para a aquisição inicial do canal. Adicionalmente, as posições do preâmbulo piloto devem ser coincidentes com as posições *SP*, sendo desejado um padrão piloto único para ambos, o preâmbulo e o *SP*. Os preâmbulos piloto podem também ser utilizados para a interpolação no tempo e todos os preâmbulos podem apresentar um padrão piloto idêntico. Estes requisitos são importantes para a detecção de C2 no varrimento e necessários para a estimativa *offset* da frequência com correlação da sequência de codificação. Num ambiente de ligação de canal, a coincidência nas posições piloto deve também ser mantida para a ligação do canal porque a estrutura piloto irregular pode prejudicar o desempenho da interpolação.

Em pormenor, se uma distância z entre pilotos difundidos (*SPs*) num símbolo *OFDM* for de 48 e se uma distância y entre *SPs* que correspondem a uma portadora *SP* específica ao longo do eixo do tempo for de 4, uma distância efetiva x após a interpolação no tempo torna-se 12. Isto é quando a fracção do intervalo de guarda (*GI*) é de $1/64$. Se a fracção *GI* for de $1/128$, pode ser utilizado $x=24$, $y=4$, e $z=96$. Se for utilizada a ligação de canal, as posições *SP* podem ser realizadas coincidentes com as posições do preâmbulo piloto gerando pontos não-contínuos na estrutura piloto difundida.

Nesse momento, as posições do preâmbulo piloto podem ser coincidentes com todas as posições de *SP* do

símbolo de dados. Quando é utilizada a ligação do canal, a fatia de dados onde um serviço é transmitido, pode ser determinada independentemente da granularidade da largura de banda de 8 MHz. Contudo, para reduzir a informação complementar para o endereçamento da fatia de dados, pode ser escolhida a transmissão que inicia na posição *SP* e que termina na posição *SP*.

Quando um receptor recebe tais *SPs*, se necessário, a estimativa de canal (r501) apresentada na figura 29 pode executar a interpolação no tempo para obter os pilotos apresentados em linhas a ponteados na figura 10 e executar a interpolação da frequência. Nesse momento, para os pontos não-contínuos dos quais os intervalos se encontram assinalados como '32' na figura 10a, pode ser implementada a execução de interpolações à esquerda e direita separadamente ou executar interpolações em apenas um lado, executando depois a interpolação no outro lado utilizando as posições piloto já interpoladas, em que pode ser implementado o intervalo de 12 como ponto de referência. Nesse momento, a largura da fatia de dados pode variar dentro de 7,61 MHz, sendo que deste modo pode minimizar o consumo de energia executando a estimativa de canal e decodificando apenas as subportadoras necessárias.

A figura 11 mostra um outro exemplo de PP5 aplicado no ambiente de ligação de canal ou uma estrutura de *SP* para manter a distância efectiva x como 12 para evitar a estrutura *SP* irregular apresentada na figura 10

quando é utilizada a ligação de canal. Tal como apresentado, se a distância SP for mantida consistente em caso de ligação de canal, não haverá qualquer problema na interpolação da frequência mas as posições piloto entre os símbolos de dados e preâmbulo podem não ser coincidentes. Por outras palavras, esta estrutura não necessita de estimativa adicional de canal para uma estrutura SP irregular, no entanto, as posições SP utilizadas na ligação de canal e posições do preâmbulo piloto tornam-se diferentes para cada canal.

A figura 12 mostra uma nova estrutura SP ou $PP5'$ para proporcionar uma solução aos dois problemas acima mencionados no ambiente de ligação de canal. Especificamente, uma distância piloto de $x=16$ pode solucionar aqueles problemas. Para preservar a densidade piloto ou para manter a mesma informação complementar, um $PP5'$ pode apresentar $x=16$, $y=3$, $z=48$ para $GI=1/64$ e um $PP7'$ pode apresentar $x=16$, $y=6$, $z=96$ para $GI=1/128$. A capacidade de interpolação somente da frequência pode ainda ser mantida. As posições piloto encontram-se representadas na figura 12 para comparação com a estrutura $PP5$.

A figura 13 mostra um exemplo de um novo padrão SP ou estrutura $PP5'$ em ambiente de ligação de canal. Tal como apresentado na figura 46, seja utilizando um canal simples ou ligação de canal, pode ser proporcionada uma distância piloto $x=16$ eficaz. Adicionalmente, porque as posições SP podem ser realizadas coincidentes com as

posições do preâmbulo piloto, pode ser evitada a deterioração da estimativa do canal provocada pela irregularidade *SP* ou posições *SP* não coincidentes. Por outras palavras, não existe qualquer posição *SP* irregular para o interpolador de frequência, encontrando-se proporcionada a coincidência entre o preâmbulo e as posições *SP*.

Consequentemente, os novos padrões *SP* propostos podem ser vantajosos em que um único padrão *SP* pode ser utilizado para ambos o canal único e canal ligado; não pode ser provocada qualquer estrutura piloto irregular, sendo deste modo possível uma boa estimativa de canal; ambas as posições do preâmbulo e as posições piloto *SP* podem ser mantidas coincidentes, a densidade dos pilotos pode ser mantida a mesma que para PP5 e PP7 respectivamente, podendo também ser preservada a capacidade de interpolação somente da frequência.

Adicionalmente, a estrutura do preâmbulo pode corresponder aos requisitos de modo que as posições do preâmbulo piloto devem cobrir todas as posições *SP* possíveis para aquisição inicial do canal; o número máximo de portadoras deve ser de 3409 (7,61 MHz) para o varrimento inicial; exactamente os mesmos padrões piloto e sequência de codificação devem ser utilizados para detecção de C2, não sendo necessário nenhum preâmbulo específico de detecção tal como P1 e T2.

Em termos de relação com a estrutura do quadro, a granularidade da posição da fatia de dados pode ser alterada para 16 portadoras em vez de 12, sendo que deste modo pode surgir menos informação complementar para endereçamento de posição, não se esperando qualquer outro problema no que se refere à condição da fatia de dados, podendo ser esperada condição de fenda zero, etc.

Por isso, no módulo de estimativa de canal r501 da figura 62, os pilotos em todos os preâmbulos podem ser utilizados quando é executada a interpolação no tempo *SP* dos símbolos de dados. Por isso, a aquisição de canal e estimativa de canal nos limites do quadro podem ser melhoradas.

Agora, no que se refere aos requisitos relacionados com o preâmbulo e a estrutura piloto, existe consenso em que as posições dos preâmbulos piloto e SPs devem coincidir independentemente da ligação do canal; o número total de portadoras no bloco *L1* deve ser divisível pela distância piloto para evitar a estrutura irregular na extremidade da banda; os blocos *L1* devem ser repetidos no domínio da frequência; e os blocos *L1* devem ser sempre descodificáveis na posição da janela de sintonia arbitrária. Os requisitos adicionais deveriam ser que as posições piloto e padrões devem ser repetidos por períodos de 8 MHz; o desvio correcto da frequência portadora deve ser estimado sem o conhecimento da ligação de canal; e a descodificação *L1* (reordenamento) impossível antes de o

desvio da frequência ser compensado.

A figura 14 mostra um relacionamento entre o símbolo de dados e preâmbulo quando são utilizadas as estruturas do preâmbulo tal como apresentado na figura 19 e figura 20. O bloco *L1* pode ser repetido por períodos de 6 MHz. Para a descodificação *L1*, devem ser encontrados tanto o desvio da frequência como o padrão da deslocação do preâmbulo. A descodificação *L1* não é possível na posição arbitrária do sintonizador sem informação sobre a ligação de canal e um receptor não consegue diferenciar entre o valor de deslocação do preâmbulo e desvio da frequência.

Deste modo tem que ser obtido um receptor especificamente para o removedor (r401) do cabeçalho do quadro apresentado na figura 30 para executar a descodificação de sinal *L1*, estrutura de ligação de canal. Dado que é conhecida a quantidade esperada de deslocação do preâmbulo em duas regiões sombreadas verticalmente na figura 30, o sincronizador de tempo/frequência r505 na figura 29 pode estimar o desvio da frequência portadora. Com base na estimativa, o trajecto de sinalização *L1* r308-1~r301-1 na figura 31 pode descodificar o bloco *L1*.

A figura 15 mostra um relacionamento entre o símbolo de dados e o preâmbulo quando é utilizada a estrutura preâmbulo tal como apresentada na figura 22. O bloco *L1* pode ser repetido por períodos de 8 MHz. Para a descodificação *L1* é necessário ser encontrado somente o

desvio da frequência, sendo que pode não ser necessário o conhecimento da ligação do canal. O desvio da frequência pode ser facilmente estimado utilizando uma sequência binária pseudo-aleatória (*Pseudo Random Binary Sequence - PRBS*). Tal como apresentado na figura 48, o preâmbulo e símbolos de dados encontram-se alinhados, sendo que deste modo a procura da sincronização adicional pode tornar-se desnecessária. Por isso, para um receptor, especificamente para o removedor r401 de cabeçalho do quadro apresentado na figura 63, é possível que tenha que ser obtido somente o pico de correlação com sequência de codificação do piloto para executar a descodificação do sinal *L1*. O sincronizador de tempo/frequência r505 na figura 29 pode estimar o desvio da frequência portadora da posição pico.

A figura 16 mostra um exemplo de perfil de atraso de canal por cabo.

Do ponto de vista da concepção do piloto, o *GI* actual já sobreprotege a dispersão dos tempos de propagação do canal de cabo. No pior dos casos, redesenhar o modelo de canal pode ser uma opção. Para repetir o padrão exactamente a cada 8 MHz, a distância piloto deve ser um divisor de 3584 portadoras ($z=32$ ou 56). Uma densidade piloto de $z=32$ pode aumentar a informação complementar piloto, deste modo pode ser escolhido $z=56$. Uma cobertura do tempo de propagação ligeiramente menor pode não ser importante no canal de cabo. Por exemplo, pode ser $8\text{ }\mu\text{s}$ para PP5' e $4\text{ }\mu\text{s}$ para PP7' comparado com $9.3\text{ }\mu\text{s}$ (PP5) e $4.7\text{ }\mu\text{s}$ (PP7).

Atrasos significativos podem ser cobertos por ambos os padrões piloto mesmo no pior dos casos. Para a posição do preâmbulo piloto, não são necessários mais do que todas as posições *SP* no símbolo de dados.

Se o trajecto de atraso de -40 dB puder ser ignorado, a distribuição actual dos tempos de propagação pode tornar-se 2,5 us, $1/64 \text{ GI} = 7 \text{ } \mu\text{s}$, ou $1/128 \text{ GI} = 3,5$ [micro] s. Isso mostra que o parâmetro da distância piloto, $z=56$ pode ser um valor suficientemente bom. Adicionalmente, $z=56$ pode ser um valor conveniente para estruturar o padrão piloto que permite a estrutura preâmbulo apresentada na figura 48.

A figura 17 mostra uma estrutura piloto dispersa que utiliza $z=56$ e $Z=112$ que é construída no módulo de inserção de piloto 404 na figura 42. São propostos PP5' ($x=14$, $y=4$, $z=56$) e PP7' ($x=28$, $y=4$, $z=112$). Podem ser inseridas portadoras de extremidade para o fecho da extremidade.

Tal como apresentado na figura 50, os pilotos encontram-se alinhados a 8 MHz de cada extremidade da banda, podendo cada posição piloto e estrutura piloto ser repetida a cada 8 MHz. Deste modo, esta estrutura pode suportar a estrutura do preâmbulo apresentada na figura 48. Adicionalmente, pode ser utilizada uma estrutura piloto comum entre o preâmbulo e símbolos de dados. Por isso, o módulo de estimativa de canal r501 na figura 29 pode

executar a estimativa de canal utilizando a interpolação no preâmbulo e símbolos de dados porque não pode surgir qualquer padrão piloto irregular, independentemente da posição da janela que é decidida pelos locais das fatias de dados. Nesse momento, utilizar somente a interpolação na frequência pode ser suficiente para compensar a distorção de canal da dispersão dos tempos de propagação. Se a interpolação no tempo for adicionalmente executada, pode ser realizada uma estimativa de canal mais precisa.

Por conseguinte, no novo padrão piloto proposto, a posição piloto e padrão podem ser repetidos com base num período de 8 MHz. Pode ser utilizado um único padrão piloto para ambos, o preâmbulo e os símbolos de dados. A descodificação *L1* pode sempre ser possível sem o conhecimento da ligação do canal. Adicionalmente, o padrão-piloto proposto pode não afectar vulgarmente com T2 porque pode ser utilizada a mesma estratégia piloto de padrão piloto difundido; T2 já utiliza 8 padrões piloto diferentes, sendo que nenhuma complexidade significativa do receptor pode ser aumentada por padrões piloto modificados. Para uma sequência de codificação piloto, o período de *PRBS* pode ser 2047 (sequência-m); a geração de *PRBS* pode ser reinicializada a cada 8 MHz, em que o período é de 3584; a taxa de repetição piloto de 56 pode ser também co-primo com 2047; não sendo esperado qualquer problema *PAPR*.

A figura 18 mostra um exemplo de um modulador baseado em *OFDM*. Os fluxos de símbolos de entrada podem ser

transformados em domínio do tempo pelo módulo *IFFT 501*. Se necessário, a relação potência de pico/potência média (*peak-to-average power ratio - PAPR*) pode ser reduzida no módulo 502 redutor de *PAPR*. Para os processos *PAPR*, podem ser utilizadas a extensão da constelação activa (*ACE*) ou reserva de tom. O módulo insersor de *GI 503* pode copiar uma última parte do símbolo *OFDM* efectivo para preencher o intervalo de guarda numa forma de prefixo cíclico.

O módulo insersor de preâmbulo 504 pode inserir preâmbulo na frente de cada quadro transmitido de modo que um receptor pode detectar o sinal digital, quadro e adquirir a aquisição do desvio de tempo/frequência. Nesse momento, o sinal do preâmbulo pode executar a sinalização da camada física tal como dimensão FFT (3 bits) e dimensão de intervalo de guarda (3 bits). O módulo de inserção do preâmbulo 504 pode ser omitido se o modulador for especificamente para *DVB-C2*.

A figura 19 mostra um exemplo de uma estrutura de preâmbulo para ligação de canal, gerado no módulo de inserção do preâmbulo 504 na figura 51. Um bloco *L1* completo deve ser "sempre decodificável" em qualquer posição arbitrária de janela de sintonia 7,61 MHz e não deve ter lugar qualquer perda da sinalização *L1* independentemente da posição da janela do sintonizador. Tal como apresentado, os blocos *L1* podem ser repetidos no domínio da frequência por períodos de 6 MHz. O símbolo de dados pode ser ligado por canal a cada canal de 8 MHz. Se,

para a descodificação *L1*, um receptor utilizar um sintonizador tal como o sintonizador r603 representado na figura 28 que utiliza uma largura de banda de 7,61 MHz, o removedor de cabeçalho r401 de quadro na figura 30 precisa de reorganizar o bloco recebido *L1* deslocado ciclicamente (figura 20) para a sua forma original. Este rearranjo é possível porque o bloco *L1* é repetido para cada bloco de 6MHz.

A figura 21 mostra um processo para conceber um preâmbulo mais optimizado. A estrutura de preâmbulo da figura 19 utiliza apenas 6MHz de largura de banda total de 7,61 MHz do sintonizador para descodificação *L1*. Em termos de eficiência de espectro, a largura de banda de 7,61 MHz não é totalmente utilizada. Portanto, pode haver uma maior optimização na eficiência espectral.

A figura 22 mostra um outro exemplo da estrutura preâmbulo ou estrutura preâmbulo dos símbolos para a eficiência completa do espectro, gerada no módulo 401 de inserção do cabeçalho no quadro na figura 42. Tal como o símbolo de dados, os blocos *L1* podem ser repetidos no domínio da frequência por períodos de 8 MHz. Um bloco *L1* completo encontra-se ainda "sempre descodificável" em qualquer posição arbitrária da janela de sintonia de 7,61 MHz. Após a sintonia, os dados a 7,61 MHz podem ser vistos como um código virtualmente puncionado. Tendo exactamente a mesma largura de banda para ambos o preâmbulo e os símbolos de dados e exactamente a mesma estrutura piloto para ambos

o preâmbulo e símbolos de dados, pode maximizar a eficiência do espectro. Outras características tais como a propriedade deslocada ciclicamente e não enviar o bloco *L1* em caso de não haver fatias de dados, podem ser mantidas inalteradas. Por outras palavras, a largura de banda dos símbolos preâmbulo pode ser idêntica à largura de banda dos símbolos de dados ou, tal como apresentado na figura 57, a largura de banda dos símbolos preâmbulo pode ser a largura de banda do sintonizador (aqui é de 7,61 MHz). A largura de banda do sintonizador pode ser definida como uma largura de banda que corresponde a um número de portadoras activas totais quando é utilizado um único canal. Quer dizer, a largura de banda do símbolo preâmbulo pode corresponder ao número de portadoras activas totais (aqui é de 7,61 MHz).

A figura 23 mostra um código virtualmente punccionado. Os dados de 7,61 MHz entre o bloco *L1* de 8 MHz pode ser considerado como código punccionado. Quando um sintonizador r603 apresentado na figura 28 utiliza uma largura de banda de 7,61 MHz para a descodificação *L1*, o removedor de cabeçalho r401 de quadro na figura 30 precisa de reorganizar o bloco *L1* recebido, deslocado ciclicamente para a sua forma original tal como apresentado na figura 56. Nesse momento, é executada a descodificação *L1* utilizando toda a largura de banda do sintonizador. Assim que o bloco *L1* estiver rearranjado, um espectro do bloco *L1* reorganizado pode apresentar uma região em branco dentro do espectro tal como apresentado na parte superior direita da figura 23 porque um tamanho original do bloco *L1* apresenta

8 MHz de largura de banda.

Assim que a região vazia se encontrar preenchida com zeros, após o desintercalamento no domínio dos símbolos pelo desintercalador de frequência r403 na figura 30 ou pelo desintercalador de símbolos r308-1 na figura 31 ou após o desintercalamento no domínio de bits pelo desmapeador de símbolos r306-1, multiplexador de bits r305-1, e desintercalador interno r304-1 na figura 31, o bloco pode apresentar uma forma que parece ser puncionada tal como apresentado no lado direito inferior da figura 23.

Este bloco *L1* pode ser descodificado no módulo de descodificação puncionado/encurtado r303-1 na figura 31. Ao utilizar esta estrutura preâmbulo, pode ser utilizada toda a largura de banda do sintonizador, podendo deste modo ser aumentados a eficiência do espectro e o ganho da codificação. Adicionalmente pode ser utilizada uma largura de banda idêntica e estrutura piloto comum para o preâmbulo e símbolos de dados.

Adicionalmente, se a largura de banda do preâmbulo ou a largura de banda dos símbolos do preâmbulo se encontrar definida como uma largura de banda de sintonizador como apresentado na figura 25, (é de 7,61 MHz no exemplo), pode ser obtido um bloco *L1* completo após a reorganização mesmo sem o puncionamento. Por outras palavras, para um quadro que apresenta símbolos de preâmbulo, onde os símbolos do preâmbulo apresentam pelo

menos um bloco de camada 1 ($L1$), pode-se dizer que o bloco $L1$ tem 3408 subportadoras activas e as 3408 subportadoras activas correspondem a 7,61 MHz de 8MHz de banda de frequência de rádio (RF).

Deste modo, a eficiência do espectro e desempenho da descodificação $L1$ pode ser maximizada. Por outras palavras, num receptor, a descodificação pode ser realizada no bloco do módulo descodificador punccionado/encurtado r303-1 na figura 31, após a realização de apenas desintercalamento no domínio dos símbolos.

Consequentemente, a nova estrutura preâmbulo proposta pode ser vantajosa por ser totalmente compatível com o preâmbulo utilizado anteriormente excepto que a largura de banda é diferente; os blocos $L1$ são repetidos por períodos de 8 MHz; o bloco $L1$ pode ser sempre descodificável independentemente da posição da janela do sintonizador; pode ser utilizada para descodificação $L1$ a largura de banda completa do sintonizador; a eficiência máxima do espectro pode garantir mais ganho de código; o bloco $L1$ incompleto pode ser considerado como codificado punccionado; pode ser utilizada uma estrutura piloto simples e a mesma para ambos, o preâmbulo e os dados; e largura de banda idêntica pode ser utilizada para ambos o preâmbulo e dados.

A figura 26 mostra um exemplo de um processador analógico. Um DAC (601) pode converter uma entrada de sinal

digital em sinal analógico. Após a largura de banda da frequência de transmissão ter sido convertida para cima no conversor elevador 602 e filtrada no filtro analógico 603 o sinal pode ser transmitido.

A figura 27 mostra um exemplo de um sistema de recepção digital de acordo com uma forma de realização da presente invenção. O sinal recebido é convertido em sinal digital num processador analógico r105. Um desmodulador r104 pode converter o sinal em dados no domínio da frequência. Um analisador de quadros r103 pode remover pilotos e cabeçalhos e activar a selecção de informação de serviço que necessita de ser descodificada. Um desmodulador *BICM* r102 pode corrigir erros no canal de transmissão. Um processador de saída r101 pode restaurar o fluxo de serviço transmitido originalmente e informação de temporização.

A figura 28 mostra um exemplo de processador analógico utilizado no receptor. Um módulo sintonizador/AGC (*Auto gain controller*) r603 pode seleccionar a largura de banda de frequência desejada a partir do sinal recebido. Um conversor para baixo r602 pode restaurar a banda de base. Um *ADC* r601 pode converter sinal analógico em sinal digital.

A figura 29 mostra um exemplo de desmodulador. Um detector de quadro r506 pode detectar o preâmbulo, verificar se existe um sinal digital correspondente, e detectar o início de um quadro. Um sincronizador de

tempo/frequência r505 pode executar a sincronização nos domínios do tempo e frequência. Nesse momento, para a sincronização no domínio do tempo, pode ser utilizada uma correlação de intervalo de guarda. Para a sincronização no domínio da frequência, pode ser utilizada a correlação ou o desvio pode ser estimado a partir da informação da fase de uma subportadora que é transmitida no domínio da frequência. Um removedor de preâmbulo r504 pode remover o preâmbulo da frente do quadro detectado. Um removedor de GI, r503, pode remover o intervalo de guarda. Um módulo FFT, r501, pode transformar o sinal no domínio do tempo em sinal no domínio da frequência. Um módulo de estimativa/igualização de canal r501 pode compensar erros estimando a distorção no canal de transmissão utilizando o símbolo piloto. O removedor de preâmbulo r504 pode ser omitido se o desmodulador for especificamente para *DVB-C2*.

A figura 30 mostra um exemplo de analisador de quadros. Um removedor de pilotos (r404) pode remover o símbolo piloto. Um desintercalador de frequência r403 pode executar o desintercalamento no domínio da frequência. Um concentrador de símbolos *OFDM* r402 pode restaurar o quadro de dados de fluxos de símbolos transmitidos em símbolos *OFDM*. Um removedor de cabeçalho de quadro r401 pode extrair sinalização de camada física do cabeçalho de cada quadro transmitido e remover o cabeçalho. A informação extraída pode ser utilizada como parâmetros para os seguintes processos no receptor.

A figura 31 mostra um exemplo de um desmodulador *BICM*. A figura 31a mostra um trajecto de dados e A figura 31b mostra um trajecto de sinalização *L1*. Um desintercalador de símbolos r308 pode executar o desintercalamento no domínio dos símbolos. Um extractor *ModCod* r307 pode extrair parâmetros *ModCod* da frente de cada quadro *BB* e tornar os parâmetros disponíveis para a seguinte desmodulação adaptativa/variável e processos de descodificação. Um desmapeador r306 de símbolo pode desmapear fluxos símbolos de entrada transformando em fluxos de bits *Log-Likelihood Ratio (LLR)*. Os fluxos *LLR* de bits de saída podem ser calculados utilizando uma constelação utilizada num mapeador de símbolos 306 do transmissor como ponto de referência. Neste ponto, quando é utilizado o anteriormente referido *MQAM* ou *NU-MQAM*, ao calcular ambos o eixo I e eixo Q quando se calcula o bit mais próximo de *MSB* e ao calcular o eixo I ou eixo Q quando se calcula os bits residuais, pode ser implementado um desmapeador de símbolos eficiente. Este processo pode ser aplicado a, por exemplo, *LLR* aproximado, *LLR* exacto, ou decisão difícil.

Quando é utilizada uma constelação otimizada de acordo com a capacidade da constelação e taxa de código do código de correcção de erro no mapeador de símbolos 306 do transmissor, o desmapeador de símbolos r306 do receptor pode obter uma constelação utilizando a taxa de código e informação sobre a capacidade da constelação transmitida do transmissor. O multiplexador de bits r305 do receptor pode

executar uma função inversa do desmultiplexador de bits 305 do transmissor. O desintercalador interno r304 e desintercalador externo r302 do receptor pode executar funções inversas do intercalador interno (304) e intercalador externo 302 do transmissor, para obter respectivamente a corrente de bits na sua sequência original. O desintercalador externo r302 pode ser omitido se o desmodulador *BICM* for especificamente para *DVB-C2*.

O decodificador interno r303 e o decodificador externo r301 do receptor podem executar os processos de decodificação correspondentes para o codificador interno 303 e codificador externo 301 do transmissor, respectivamente, para corrigir erros no canal de transmissão. Processos semelhantes realizados no trajecto dos dados podem ser realizados no trajecto de sinalização *L1*, mas com parâmetros diferentes r308-1~r301-1. Neste ponto, tal como explicado na parte do preâmbulo, pode ser utilizado um módulo de código encurtado/puncionado r303-1 para decodificação do sinal *L1*.

A figura 32 mostra um exemplo de decodificação *LDPC* utilizando o módulo de encurtamento/puncionagem r303-1. Um desmultiplexador r301 pode emitir separadamente parte de informação e parte de paridade de código sistemático de fluxos de bits de entrada. Para a parte de informação, um módulo de zona de preenchimento com zeros r302a pode executar o preenchimento da zona com zeros de acordo com uma série de fluxos de bits de entrada do decodificador

LDPC, e para a parte de paridade, fluxos de bits de entrada para o decodificador *LDPC* podem ser gerados por despuncionamento da parte puncionada no módulo de despuncionamento de paridade r303a. A descodificação *LDPC* pelo módulo r304a pode ser executada em fluxos de bits gerados, podendo ser removidos os zeros na parte da informação pelo removedor de zeros r305a e emitidos.

A figura 33 mostra um exemplo de processador de saída. Um decodificador *BB*, r209, pode restaurar fluxos de bits codificados no transmissor. Um divisor r208 pode restaurar quadros *BB* que correspondem a vários *PLP*'s que são multiplexados e transmitidos a partir do transmissor de acordo com o trajecto *PLP*. Para cada trajecto *PLP*, um removedor de cabeçalho *BB* r207-1~n pode remover o cabeçalho que é transmitido na frente do quadro *BB*. Um decodificador *CRC* r206-1~n pode executar a descodificação *CRC* e tornar quadros *BB* fiáveis, disponíveis para selecção. Um módulo de inserção de pacote zero r205-1~n pode restaurar pacotes zero que foram removidos para uma maior eficiência de transmissão no seu local original. Um módulo de recuperação de atraso r204-1~n pode restaurar um tempo de propagação que existe entre cada trajecto *PLP*.

Um módulo de recuperação de relógio de saída r203-1~n pode restaurar a temporização original do fluxo de serviço a partir da informação de temporização transmitida do sincronizador 203-1~n do fluxo de entrada.. Um módulo de interface de saída r202-1~n pode restaurar dados no pacote

TS/GS de fluxos de bits de entrada que se encontram divididos em fatias no quadro *BB*. Um pós-processador de saída *r201-1~n* pode restaurar vários fluxos *TS/GS* convertendo-os num fluxo *TS/GS* completo, se necessário. Os blocos sombreados apresentados na figura 33 representam módulos que podem ser usados quando um único *PLP* é processado num período e o resto dos blocos representam os módulos que podem ser utilizados quando vários *PLPs* são simultaneamente processados.

Os padrões do preâmbulo piloto foram cuidadosamente projectados para evitar o aumento *PAPR*, sendo que, deste modo, se a taxa de repetição *L1* aumentar é necessário ser considerado o *PAPR*. O número de bits de informação *L1* varia dinamicamente de acordo com a ligação de canal, o número de *PLPs*, etc. Em pormenor, é necessário considerar coisas tais como o tamanho fixo do bloco *L1* pode introduzir informação complementar desnecessária; a sinalização *L1* deve ser protegida mais fortemente do que os símbolos de dados; e o intercalamento no tempo do bloco *L1* pode melhorar a robustez em relação a danos no canal tal como a necessidade de ruído impulsivo.

Para uma taxa de repetição de bloco *L1* de 8 MHz, tal como apresentado na figura 34, é exibida a eficiência do espectro completo (26,8% de aumento de *BW*) com funcionamento virtual mas o *PAPR* pode ser ampliado dado que a largura de banda *L1* é a mesma que a dos símbolos de dados. Para a taxa de repetição de 8 MHz, pode ser

utilizado o intercalamento de frequência 4K-FFT *DVB-T2* para a uniformização, podendo o mesmo padrão repetir-se a um período de 8 MHz após o intercalamento.

Para uma taxa de repetição de bloco *L1* de 6 MHz, tal como apresentado na figura 35, pode ser exibida uma eficiência reduzida de espectro sem punçãoamento virtual. Pode ocorrer um problema semelhante de *PAPR* como para o caso de 8MHz dado que as larguras de banda de *L1* e dos símbolos de dados partilham $LCM=24$ MHz. Para a taxa de repetição de 6 MHz, pode ser utilizado o intercalamento de frequência 4K-FFT *DVB-T2* para a uniformização, podendo o mesmo padrão repetir-se a um período de 24 MHz após o intercalamento.

A figura 36 mostra uma nova taxa de repetição de bloco *L1* de 7.61 MHz ou uma largura de banda completa de sintonizador. Uma eficiência de espectro largo (aumento de 26.8% de BW) pode ser obtida sem punçãoamento virtual. Não pode haver qualquer problema com o *PAPR* dado que *L1* e larguras de banda de símbolo de dados partilham 1704 MHz. Para a taxa de repetição de 7,61 MHz, pode ser utilizado o intercalamento de frequência 4K-FFT *DVB-T2* para a uniformização, podendo o mesmo padrão repetir-se a um período de cerca de 1704 MHz após o intercalamento.

A figura 37 é um exemplo de sinalização *L1* que é transmitida no cabeçalho do quadro. Cada informação na sinalização *L1* pode ser transmitida para o receptor e pode

ser utilizada como um parâmetro de descodificação. Em especial, a informação pode ser utilizada no trajecto de sinal *L1* apresentado na figura 31, podendo os *PLPs* ser transmitidos em cada fatia de dados. Pode ser obtido um aumento da robustez para cada *PLP*.

A figura 39 é um exemplo de um intercalador de símbolos 308-1 tal como apresentado no trajecto de sinalização *L1* na figura 4 e também pode ser um exemplo do seu desintercalador de símbolos r308-1 correspondente tal como apresentado no trajecto de sinalização *L1* na figura 31. Os blocos com linhas inclinadas representam blocos *L1* e os blocos sólidos representam portadoras de dados. Os blocos *L1* podem ser transmitidos não somente dentro de um único preâmbulo, mas também podem ser transmitidos dentro de vários blocos *OFDM*. Dependendo do tamanho do bloco *L1*, o tamanho do bloco de intercalamento pode variar. Por outras palavras, *num_L1_sym* e amplitude *L1* podem ser diferentes umas das outras. Para minimizar a informação complementar desnecessária, podem ser transmitidos dados no resto das portadoras dos símbolos *OFDM* onde o bloco *L1* é transmitido. Neste ponto, pode ser garantida a eficiência completa do espectro porque o ciclo de repetição do bloco *L1* é ainda um sintonizador de largura de banda completa. Na figura 39, os números em blocos com linhas inclinadas representam a ordem dos bits dentro de um único bloco *LDPC*.

Consequentemente, quando os bits são escritos numa memória de intercalamento na direcção da fila de

acordo com um índice de símbolo tal como apresentado na figura 72 e lidos na direcção da coluna de acordo com um índice de portadora, pode ser obtido um efeito de intercalamento de bloco. Por outras palavras, um bloco *LDPC* pode ser intercalado no domínio do tempo e no domínio da frequência e depois ser transmitido. Num *L1_sym* pode ser um valor predeterminado, por exemplo um número entre 2-4 pode ser definido como uma série de símbolos *OFDM*. Neste ponto, para aumentar a granularidade da dimensão do bloco *L1*, pode ser utilizado para protecção do *L1* um código *LDPC* punccionado/encurtado que apresenta um comprimento mínimo da palavra-chave.

A figura 40 é um exemplo de uma transmissão de bloco *L1*. A figura 40 ilustra a figura 39 no domínio do quadro. Como apresentado no lado esquerdo da figura 40, os blocos *L1* podem ser gerados em largura de banda completa de sintonizador ou tal como apresentado na figura 40, os blocos *L1* podem ser parcialmente gerados e o resto das portadoras pode ser utilizado para o transporte de dados. Em ambos os casos, pode ser verificado que a taxa de repetição do bloco *L1* pode ser idêntico a uma largura de banda completa do sintonizador. Adicionalmente, para os símbolos *OFDM* que utilizam a sinalização *L1* incluindo o preâmbulo, só pode ser executado o intercalamento de símbolos enquanto não se permite a transmissão de dados nesses símbolos *OFDM*. Por conseguinte, para o símbolo *OFDM* utilizado para a sinalização *L1*, um receptor pode decodificar *L1* realizando o desintercalamento sem a

descodificação de dados. Neste ponto, o bloco *L1* pode transmitir a sinalização *L1* do quadro actual ou sinalização *L1* de um quadro subsequente. No lado do receptor, os parâmetros *L1* descodificados, do trajecto de descodificação da sinalização *L1* apresentado na figura 31, podem ser utilizados para o processo de descodificação para o trajecto de dados do analisador de quadros do quadro subsequente.

Em resumo, num transmissor, os blocos de intercalamento da região *L1* podem ser executados por blocos de escrita numa memória numa direcção em linha e leitura dos blocos escritos da memória no sentido da coluna. Num receptor, os blocos de desintercalamento da região *L1* podem ser executados escrevendo blocos numa memória numa direcção de coluna e a leitura dos blocos escritos da memória na direcção da fila. As indicações de leitura e escrita do transmissor e receptor podem ser intercambiadas.

Quando a simulação é realizada com suposições tais como $CR=1/2$ para protecção *L1* e para a uniformização T2; o mapeamento de símbolo 16-QAM; densidade piloto de 6 no preâmbulo; o número de LDPC curto implica que seja efectuada a quantidade necessária de punção/encurtamento, sendo que resultados ou conclusões tais como somente preâmbulo para a transmissão *L1* podem não ser suficientes; o número de símbolos OFDM depende da quantidade da dimensão do bloco *L1*; podendo ser a palavra-chave LDPC mais curta (por exemplo 192 bits de informação)

entre o código encurtado/puncionado pode ser utilizada para flexibilidade e granularidade fina; podendo ser adicionada a zona de preenchimento, se necessário, com informações complementares negligenciáveis. O resultado encontra-se resumido na figura 38.

Consequentemente, para uma taxa de repetição de bloco *L1*, a largura de banda completa do sintonizador sem puncionamento virtual pode ser uma boa solução e ainda não surgir qualquer problema do *PAPR* com a eficiência total do espectro. Para a sinalização *L1*, uma estrutura de sinalização eficiente pode permitir a configuração máxima num ambiente de 8 ligações de canal, 32 entalhes, 256 fatias de dados, e 256 *PLPs*. Para a estrutura de bloco *L1*, pode ser implementada a sinalização *L1* flexível de acordo com a dimensão do bloco *L1*. O intercalamento no tempo pode ser executado para melhor robustez para a uniformização T2. Menos informação complementar pode permitir a transmissão de dados no preâmbulo.

Para uma melhor robustez pode ser executado o intercalamento de blocos do bloco *L1*. O intercalamento pode ser realizado com um número pré-definido fixo de símbolos *L1* (*num_L1_sym*) e um número de portadoras, gerados por *L1* como um parâmetro (*L1_span*). A mesma técnica é utilizada para o intercalamento do preâmbulo P2 em *DVB-T2*.

Pode ser utilizado um bloco *L1* de dimensão variável. O tamanho pode ser adaptável à quantidade de bits

de sinalização *L1*, resultando numa informação complementar reduzida. A eficiência do espectro completo pode ser obtida sem nenhum problema *PAPR*. Uma repetição de menos do que 7,61 MHz pode significar que mais redundância pode ser enviada mas sem ter sido utilizada. Não pode surgir qualquer problema *PAPR* devido à taxa de repetição de 7,61 MHz para o bloco *L1*.

A figura 41 é um outro exemplo de sinalização *L1* transmitida dentro de um cabeçalho de quadro. A figura 41 é diferente da figura 37 em que o campo *L1_span* que apresenta 12 bits se encontra dividido em dois campos. Por outras palavras, o campo *L1_span* encontra-se dividido numa *L1_column* que apresenta 9 bits e uma *L1_row* que apresenta 3 bits. A *L1_column* representa o índice da portadora que *L1* gera. Dado que a fatia de dados inicia e termina a cada 12 portadoras, que é a densidade piloto, os 12 bits da informação complementar podem ser reduzidos em 3 bits para chegar a 9 bits.

L1_row representa o número de símbolos *OFDM* em que *L1* é gerado quando o intercalamento no tempo é aplicado. Consequentemente, o intercalamento no tempo pode ser realizado dentro de uma área de *L1_columns* multiplicada por *L1_rows*. Alternativamente, um tamanho total de blocos *L1* pode ser transmitido de tal forma que *L1_span* apresentado na figura 37 pode ser utilizado quando o intercalamento no tempo não é realizado. Para esse caso, o tamanho do bloco *L1* é de 11.776 x 2 bits no exemplo, sendo

deste modo 15 bits o suficiente. Consequentemente, o campo *L1_span* pode ser composto por 15 bits.

A figura 42 é um exemplo de intercalamento/desintercalamento de frequência ou tempo. A figura 42 mostra uma parte de um quadro completo de transmissão. A figura 42 mostra também a ligação de várias larguras de banda de 8 MHz. Um quadro pode ser composto por um preâmbulo que transmite blocos *L1* e um símbolo de dados que transmite dados. Os diferentes tipos de símbolos de dados representam fatias de dados para serviços diferentes. Tal como apresentado na figura 42, o preâmbulo transmite blocos *L1* por cada 7,61 MHz.

Para o preâmbulo, o intercalamento na frequência ou tempo é realizado dentro de blocos *L1* e não realizado entre blocos *L1*. Ou seja, para o preâmbulo, pode-se dizer que o intercalamento é realizado ao nível do bloco *L1*. Isto permite a descodificação dos blocos *L1* transmitindo blocos *L1* dentro de uma largura de banda de janela de sintonizador mesmo quando a janela do sintonizador se tiver movimentado para um local aleatório dentro de um sistema de ligação de canal.

Para a descodificação de símbolos de dados numa largura de banda aleatória de janela de sintonizador, não deve ter lugar o intercalamento entre fatias de dados. Quer dizer, para fatias de dados, pode-se dizer que o intercalamento é realizado ao nível da fatia de dados.

Consequentemente, o intercalamento na frequência e intercalamento no tempo devem ser realizados dentro de uma fatia de dados. Por isso, um intercalador de símbolos 308 num trajecto de dados de um módulo *BICM* do transmissor tal como apresentado na figura 4 pode realizar o intercalamento de símbolos para cada fatia de dados. Um intercalador de símbolos 308-1 num trajecto de sinal *L1* pode executar o intercalamento de símbolos para cada bloco *L1*.

Um intercalador de frequência 403 apresentado na figura 9 precisa de realizar separadamente o intercalamento no preâmbulo e símbolos de dados. Especificamente, para o preâmbulo, o intercalamento na frequência pode ser realizado para cada bloco *L1* e para o símbolo de dados, o intercalamento na frequência pode ser realizada para cada fatia de dados. Neste ponto, o intercalamento no tempo no trajecto de dados ou trajecto de sinal *L1* pode não ser realizado considerando o modo de baixa latência.

A figura 43 é um quadro que analisa a informação complementar da sinalização *L1* que é transmitida no cabeçalho *FECFRAME* no módulo de inserção de cabeçalho *ModCod* (307) no trajecto de dados do módulo *BICM* apresentado na figura 37. Tal como apresentado na figura 76, para o bloco *LDPC* curto (tamanho=16.200), pode ocorrer uma informação complementar máxima de 3,3% que pode não ser insignificante. Na análise são assumidos 45 símbolos para protecção *FECFRAME*, sendo o preâmbulo uma sinalização *L1* específica do quadro *C2* e o cabeçalho *FECFRAME* uma

sinalização *L1* específica de *FECFRAME*, isto é identificador *Mod*, *Cod*, e *PLP*.

Para reduzir a informação complementar *L1*, podem ser consideradas abordagens de acordo com dois tipos de fatias de dados. Para o tipo *ACM/VCM* e vários casos *PLP*, o quadro pode ser mantido o mesmo que para o cabeçalho *FECFRAME*. Para o tipo *ACM/VCM* e casos *PLP* individuais, o identificador *PLP* pode ser removido do cabeçalho *FECFRAME*, resultando numa redução da informação complementar de até 1,8%. Para o tipo *CCM* e vários casos *PLP*, o campo *Mod/Cod* pode ser removido do cabeçalho *FECFRAME*, resultando numa redução da informação complementar de até 1,5%. Para o tipo *CCM* e casos *PLP* individuais, não é necessário qualquer cabeçalho *FECFRAME*, podendo ser obtido deste modo até 3,3% de redução.

Numa sinalização *L1* encurtada, pode ser transmitido identificador *Mod/Cod* (7 bits) ou *PLP* (8 bits), mas pode ser demasiado curto para obter novamente qualquer codificação. No entanto é possível não requerer sincronização porque os *PLPs* podem estar alinhados com o quadro de transmissão *C2*; todos os *ModCod* de cada *PLP* podem ser conhecidos a partir do preâmbulo; e um cálculo simples pode permitir a sincronização com o *FECFRAME* específico.

A figura 44 mostra uma estrutura para o cabeçalho *FECFRAME* para minimizar a informação complementar. Na figura 44, os blocos com linhas inclinadas e o construtor

FECFRAME representam um diagrama de blocos pormenorizado do módulo de inserção do cabeçalho *ModCod* 307 no trajecto de dados do módulo *BICM* tal como apresentado na figura 4. Os blocos sólidos representam um exemplo do módulo de codificação interno 303, intercalador interno 304, desmultiplicador de bits 305, e mapeador de símbolos 306 no trajecto de dados do módulo *BICM* tal como apresentado na figura 4. Neste ponto, pode ser executada a sinalização *L1* encurtada porque *CCM* não requer um campo *Mod/Cod* e *PLP* único não requer um identificador de *PLP*. Neste sinal *L1* com um número reduzido de bits, o sinal *LI* pode ser repetido três vezes no preâmbulo e modulação *BPSK* pode ser realizada, sendo assim possível uma sinalização muito robusta. Finalmente, o módulo de inserção de cabeçalho *ModCod* 307 pode inserir o cabeçalho gerado em cada quadro *FEC*. A figura 51 mostra um exemplo do extractor *ModCod* r307 no trajecto de dados do módulo de demod *BICM* apresentado na figura 31.

Tal como apresentado na figura 51, o cabeçalho *FECFRAME* pode ser analisado no analisador sintáctico r301b, sendo que depois símbolos que transmitem informações idênticas em símbolos repetidos podem ser atrasados, alinhados, e depois combinados no módulo de combinação *Rake* r302b. Finalmente, quando é realizada a desmodulação *BPSK* no módulo r303b, o campo de sinal *L1* recebido pode ser restaurado, sendo que este campo de sinal *L1* restaurado pode ser enviado para o controlador de sistema para ser utilizado como parâmetros para decodificação. O *FECFRAME*

analisado sintacticamente pode ser enviado para o desmapeador de símbolos.

A figura 45 mostra o desempenho de uma taxa de erro de bit (*BER*) desempenho da protecção *L1* acima mencionada. Pode-se observar que cerca de 4,8 dB de ganho de *SNR* é obtido através de uma repetição por três vezes. A *SNR* necessária é de 8,7 dB a *BER* = 1E-11.

A figura 46 apresenta exemplos de um quadro de transmissão e estruturas de quadro *FEC*. As estruturas de quadro *FEC* apresentadas no lado superior direito da figura 46 representam o cabeçalho *FECFRAME* inserido pelo módulo de inserção 307 de cabeçalho *ModCod* apresentado na figura 4. Pode ser visto que, dependendo de várias combinações de condições, ou seja, tipo *CCM* ou *ACM/VCM* e *PLP* simples ou múltiplo, podem ser inseridas diferentes dimensões de cabeçalhos. Ou, nenhum cabeçalho pode ser inserido. Os quadros de transmissão formados de acordo com tipos de fatia de dados e apresentado na parte inferior do lado esquerdo da figura 46 pode ser formado pelo módulo de inserção 401 de cabeçalho de quadro do construtor de quadros tal como apresentado na figura 9 e o fusor/cortador 208 do processador de entrada apresentado na figura 2. Neste ponto, o *FECFRAME* pode ser transmitido de acordo com diferentes tipos de fatias de dados. Utilizando este processo, pode ser reduzida um máximo de 3,3% de informação complementar. No lado superior direito da figura 79 são apresentados quatro diferentes tipos de estruturas, mas um

técnico irá compreender que estes são apenas exemplos, e qualquer um destes tipos ou as suas combinações podem ser utilizados para a fatia de dados.

No lado do receptor, o removedor de cabeçalho de quadro r401 do módulo analisador sintáctico de quadros tal como apresentado na figura 30 e o extractor *ModCod* r307 do módulo demod *BICM* apresentado na figura 31 pode extrair um parâmetro de campo *ModCod* que é necessário para descodificação. Neste ponto, de acordo com o tipo de fatia de dados de propulsão, podem ser extraídos parâmetros de quadro de transmissão. Por exemplo, para o tipo *CCM*, os parâmetros podem ser extraídos da sinalização *L1* que é transmitida no preâmbulo e para o tipo *ACM/VCM*, os parâmetros podem ser extraídas do cabeçalho *FECFRAME*.

Tal como apresentado no lado superior direito da figura 79, a estrutura *FECFRAME* pode ser dividida em dois grupos, em que o primeiro grupo são as estruturas dos três quadros superiores com cabeçalho e o segundo grupo é a última estrutura de quadro sem cabeçalho.

A figura 47 apresenta um exemplo de sinalização *L1* que pode ser transmitida dentro do preâmbulo pelo módulo de inserção 401 do cabeçalho do quadro do módulo de construção de quadro apresentado na figura 42. Esta sinalização *L1* é diferente da sinalização anterior *L1* em que o tamanho do bloco *L1* podem ser transmitido em bits (*L1_size*, 14 bits); é possível ligar/desligar o

intercalamento no tempo na fatia de dados (*dslice_time_intrlv*, 1 bit); sendo que ao definir o tipo de fatia de dados (*dslice_type*, 1 bit), é reduzida a informação complementar da sinalização. Neste ponto, quando o tipo de fatia de dados é *CCM*, o campo *Mod/Cod* pode ser transmitidos dentro do preâmbulo em vez de dentro do cabeçalho *FECFRAME* (*PLP_mod* (3 bits), *PLP_FEC_type* (1 bit), *PLP_cod* (3 bits)).

No lado do receptor, o módulo de descodificação interna R303-1 encurtado/perfurado do *BICM* demod tal como apresentado na figura 31 pode obter o primeiro bloco *LDPC*, que tem um tamanho de bloco *L1* fixo, transmitido dentro do preâmbulo, através da descodificação. Podem também ser obtidos os números e a dimensão do resto dos blocos *LDPC*.

A intercalação do tempo pode ser utilizada quando vários símbolos *OFDM* são necessários para a transmissão de *L1* ou quando existe uma fatia de dados intercalada no tempo. Um ligar/desligar flexível da intercalação do tempo é possível com uma marca de intercalação. Para intercalação do tempo do preâmbulo, pode ser necessária uma marca de intercalação do tempo (1 bit) e podem ser necessários vários símbolos *OFDM* intercalados (3 bits), deste modo, um total de 4 bits podem ser protegidos de um modo semelhante a um *FECFRAME* encurtado.

A figura 48 mostra um exemplo de pré-sinalização *L1* que pode ser executada no módulo de inserção de

cabeçalho *ModCod* 307-1 no trajecto de dados do módulo *BICM* apresentado na figura 4. Os blocos com as linhas inclinadas e construtor de preâmbulo são exemplos do módulo de inserção de cabeçalho *ModCod* 307-1 na sinalização *L1* do módulo *BICM* apresentado na figura 4. Os blocos sólidos são exemplos do módulo de inserção 401 de cabeçalho de quadro do construtor de quadros tal como apresentado na figura 42.

Deste modo, os blocos sólidos podem ser exemplos do módulo de codificação interno 303-1 encurtado/puncionado, intercalador interno 304-1, e mapeador de símbolos 306-1 no trajecto de sinalização *L1* do módulo *BICM* apresentado na figura 4.

Tal como pode ser visto na figura 48, o sinal *L1* que é transmitido no preâmbulo pode ser protegido utilizando codificação *LDPC* encurtada/puncionada. Os parâmetros relacionados podem ser inseridos no cabeçalho numa forma de uma pré-sinalização *L1*. Neste ponto, somente parâmetros de intercalação do tempo podem ser transmitidos no cabeçalho do preâmbulo. Para garantir maior robustez, pode ser executada uma repetição de quatro vezes. No lado do receptor, para ser passível de descodificar o sinal *L1* que é transmitido no preâmbulo, o extractor *ModCod* r307-1 no trajecto de sinalização *L1* do demod *BICM* tal como apresentado na figura 31 precisa de utilizar o módulo de descodificação apresentado na figura 18. Neste ponto, porque existe uma repetição de quatro vezes ao contrário do anterior cabeçalho *FECFRAME* de descodificação, é necessário

um processo de recebimento de Rake que sincroniza os símbolos repetidos quatro vezes e adicionando os símbolos.

A figura 49 apresenta uma estrutura do bloco de sinalização *L1* que é transmitido do módulo de inserção 401 do cabeçalho do quadro do módulo de construção de quadro tal como apresentado na figura 42. É apresentado um caso onde não é utilizada qualquer intercalação no tempo num preâmbulo. Tal como apresentado na figura 49, podem ser transmitidos diferentes géneros de blocos *LDPC* na ordem das portadoras. Assim que estiver formado e transmitido um símbolo *OFDM* é formado e transmitido um símbolo *OFDM* seguinte. Para que o último símbolo *OFDM* seja transmitido, se sobrar alguma portadora, estas portadoras podem ser utilizadas para a transmissão de dados ou podem ser colocadas em zonas de preenchimento fictícias. O exemplo na figura 49 mostra um preâmbulo que compreende três símbolos *OFDM*. No lado do receptor, para este caso de não intercalação, o desintercalador de símbolos *r308-1* no trajecto de sinalização *L1* do demod *BICM* tal como apresentado na figura 31 pode ser ignorado.

A figura 50 mostra um caso onde é executada a intercalação no tempo *L1*. Tal como apresentado na figura 50, intercalação em blocos pode ser executada de modo a formar um símbolo *OFDM* para índices de portadora idênticos, formando depois um símbolo *OFDM* para os próximos índices de portadora. Tal como no caso onde não é executada qualquer intercalação, se sobrar alguma portadora, estas portadoras

podem ser utilizadas para a transmissão de dados ou podem ser colocadas em zonas de preenchimento fictícias. No lado do receptor, para este caso de não intercalação, o desintercalador de símbolos r308-1 no trajecto de sinalização *L1* do demod *BICM* apresentado na figura 31 pode executar a desintercalação em blocos lendo blocos *LDPC* em ordem crescente de números de blocos *LDPC*.

Adicionalmente, podem existir pelo menos dois tipos de fatias de dados. A fatia de dados do tipo 1 tem *dslice_type* = 0 nos campos de sinalização *L1*. Este tipo de fatia de dados não tem nenhum cabeçalho de *XFECFrame* e tem os seus valores *Mod/Cod* em campos de sinalização *L1*. A fatia de dados do tipo 2 tem *dslice_type* = 1 nos campos de sinalização *L1*. Este tipo de fatia de dados tem cabeçalho de *XFECFrame* e tem os seus valores *Mod/Cod* em cabeçalho de *XFECFrame*.

XFECFrame significa *XFEC(complex Forward Error Correction)Frame* e *Mod/Cod* significa tipo de modulação /taxa de código (*modulation type/coderate*).

Num receptor, um analisador sintáctico de quadro pode formar um quadro de sinais desmodulados. O quadro apresenta símbolos de dados e os símbolos de dados podem ter um primeiro tipo de fatias de dados que apresenta uma *XFECFrame* e um cabeçalho de *XFECFrame* e um segundo tipo de fatia de dados que apresenta *XFECFrame* sem cabeçalho de *XFECFrame*. Além disso, um receptor pode extrair um campo

para indicar se é para executar o tempo desintercalação nos símbolos do preâmbulo ou não para executar desintercalação no tempo nos símbolos do preâmbulo, a partir do *L1* dos símbolos preâmbulo.

Num transmissor, um construtor de quadro pode construir um quadro. Os símbolos de dados compreendem um primeiro tipo de fatias de dados que apresentam uma *XFECFrame* e um cabeçalho de *XFECFrame* e um segundo tipo de fatias de dados que apresenta *XFECFrame* sem cabeçalho de *XFECFrame*. Adicionalmente, um campo para indicar se deve ou não executar a intercalação no tempo em símbolos preâmbulo para executar a intercalação no tempo em símbolos de preâmbulo pode ser inserido no *L1* dos símbolos de preâmbulo.

Por fim, para código encurtado/puncionado para o módulo de inserção 401 do cabeçalho do quadro do construtor de quadros apresentado na figura 9, pode ser determinada uma dimensão mínima de palavra-chave que pode obter ganho de codificação e pode ser transmitida num primeiro bloco *LDPC*. Desta forma, para o resto das dimensões de bloco *LDPC* podem ser obtidas a partir daquela dimensão de bloco *L1* transmitida.

A figura 52 mostra um outro exemplo de pré-sinalização *L1* que pode ser transmitida do módulo de inserção de cabeçalho *ModCod* 307-1 no trajecto de sinalização do módulo *BICM* apresentado na figura 4. A

figura 52 é diferente da figura 48 em que o mecanismo de protecção da parte do cabeçalho foi modificado. Tal como pode ser visto na figura 52, a informação sobre a dimensão do bloco $L1$, $L1_size$ (14 bits), não é transmitida no bloco $L1$, mas transmitida no cabeçalho. No cabeçalho, a informação sobre a intercalação no tempo de 4 bits pode também ser transmitida. Para um total de 18 bits de entrada, o código BCH (45, 18) que gera 45 bits são utilizados e copiados para os dois trajectos e, finalmente, a $QPSK$ é mapeada. Para o trajecto Q , pode ser executada uma deslocação cíclica de 1 bit para ganho de diversidade, podendo ser executada a modulação $PRBS$ de acordo com a palavra de sincronização. Um total de 45 símbolos $QPSK$ podem ser emitidos destas entradas de trajecto I/Q . Neste ponto, se a profundidade da intercalação no tempo estiver definida como um número de preâmbulos que é necessário para transmitir o bloco $L1$, pode não ser necessário transmitir $L1_span$ (3bits) que indica a profundidade da intercalação no tempo. Por outras palavras, só pode ser transmitida a marca (1 bit) de ligar/desligar intercalação no tempo. Num lado do receptor, ao verificar apenas um número de preâmbulos transmitidos, sem utilizar $L1_span$, pode ser obtida a profundidade da desintercalação no tempo.

A figura 53 apresenta um exemplo de agendamento de bloco de sinalização $L1$ que é transmitido em preâmbulo. Se uma dimensão de informação $L1$ que pode ser transmitida num preâmbulo for N_{max} , quando a dimensão de $L1$ é menor do que N_{max} , um preâmbulo pode transmitir a informação. No

entanto, quando a dimensão $L1$ é maior do que N_{max} , a informação $L1$ pode ser igualmente dividida de modo que o bloco secundário $L1$ dividido é menor do que N_{max} , então o bloco secundário $L1$ dividido pode ser transmitido num preâmbulo. Neste ponto, para uma portadora que não é utilizada porque a informação $L1$ é menor do que N_{max} , não são transmitidos quaisquer dados.

Em vez disso, tal como apresentado na figura 55, a potência das portadoras onde os blocos $L1$ são transmitidos, pode ser aumentada para manter uma potência do sinal total de preâmbulo idêntica à potência do símbolo de dados. O factor de reforço de potência pode variar dependendo da dimensão $L1$ transmitida e um transmissor e um receptor pode ter um valor definido deste factor de reforço de potência. Por exemplo, se forem utilizadas somente metade das portadoras totais, o factor de reforço de potência pode ser dois.

A figura 54 apresenta um exemplo de pré-sinalização $L1$ onde o reforço da potência é considerado. Quando comparado com a figura 52, pode-se ver que a potência do símbolo $QPSK$ pode ser reforçada e enviada para o construtor de preâmbulo.

A figura 56 apresenta outro exemplo do extractor *ModCod* r307-1 no trajecto de sinalização do módulo demod *BICM* apresentado na figura 31. Do símbolo de preâmbulo de entrada, a sinalização $L1$ *FECFRAME* pode ser emitida para

dentro do desmapeador de símbolos e somente a parte do cabeçalho pode ser decodificada.

Para o símbolo do cabeçalho de entrada, o desmapeamento *QPSK* pode ser executado e pode ser obtido o valor do relatório do rácio de probabilidade (*Log-Likelihood Ratio* - *LLR*). Para o trajecto *Q*, a desmodulação *PRBS* de acordo com a palavra de sincronização pode ser executada, podendo ser realizado para restauração um processo inverso da deslocação cíclica de 1-bit.

Estes dois valores alinhados de trajecto *I/Q* podem ser combinados, podendo ser obtido o ganho *SNR*. A saída de decisão difícil pode ser entrada para o decodificador *BCH*. O decodificador *BCH* pode restaurar 18 bits de *L1* pré a partir da entrada de 45 bits.

A figura 57 mostra uma parte contrária, extractor *ModCod* de um receptor. Quando comparado com a figura 56, o controlo da potência podem ser executado nos símbolos de entrada *QPSK* do desmapeador para restaurar do nível de potência reforçado pelo transmissor para o seu valor original. Neste ponto, o controlo de potência pode ser executado considerando um número de portadoras utilizadas para a sinalização *L1* num preâmbulo e tomando o inverso do factor de reforço de potência obtido de um transmissor. O factor de reforço de potência configura a potência do preâmbulo e potência dos símbolos de dados idênticos uns aos outros.

A figura 58 apresenta um exemplo da pré-sincronização *L1* que pode ser executada no extractor *ModCod* r307-1 no trajecto de sinalização *L1* do módulo de desmodulação *BICM* apresentado na figura 31. Este é um processo de sincronização para obter uma posição inicial de cabeçalho num preâmbulo. Os símbolos de entrada podem ser *QPSK* desmapeados do que para a saída trajecto *Q*, um inverso de uma deslocação cíclica de 1 bit pode ser executada, podendo ser realizado o alinhamento. Dois valores de trajectos *I/Q* podem ser valores multiplicados e modulados por pré-sinalização *L1* podem ser desmodulados. Assim, a saída do multiplicador pode expressar apenas *PRBS* que é uma palavra de sincronização. Quando a saída é correlacionada com uma sequência conhecida *PRBS*, pode ser obtido um pico de correlação no cabeçalho. Deste modo, pode ser obtida uma posição inicial de cabeçalho num preâmbulo. Se necessário, o controlo de potência que é executado para restaurar o nível de potência original, como na fig. 57, pode ser executado na entrada do desmapeador *QPSK*.

A figura 59 mostra um outro exemplo de campo de cabeçalho de bloco *L1* que é enviado para o módulo de inserção de cabeçalho *ModCod* 307-1 no trajecto de sinalização *L1* do módulo *BICM* tal como apresentado na figura 4. A figura 59 é diferente da figura 52 em que *L1_span* que representa a profundidade do intercalamento no tempo é reduzida a 2 bits e os bits reservados são aumentados em 1 bit. Um receptor pode obter parâmetro de intercalação no tempo do bloco *L1* do *L1_span* transmitido.

A figura 60 apresenta processos de divisão igual de um bloco *L1* em várias partes quanto um número de preâmbulos, inserindo depois um cabeçalho em cada um dos blocos *L1* divididos e em seguida atribuindo os blocos *L1* com cabeçalho inserido num preâmbulo. Isto pode ser executado quando uma intercalação do tempo é executada com vários preâmbulos onde o número de preâmbulos é maior do que um número mínimo de preâmbulos que é necessário para a transmissão de blocos *L1*. Isto pode ser executado no bloco *L1*, no trajecto de sinalização *L1* do módulo *BICM* tal como apresentado na figura 37. O resto das portadoras, após a transmissão de blocos *L1* podem ter padrões de repetição cíclica ao invés de ser munidas com zonas de preenchimento de zeros.

A figura 61 apresenta um exemplo do desmapeador de símbolos r306-1 do módulo de desmodulação *BICM*, tal como apresentado na figura 31. Para um caso onde os blocos *L1 FEC* são repetidos tal como apresentado na figura 60, cada ponto de partida de blocos *L1 FEC* pode ser alinhado, combinado no módulo r301f e depois desmapeado em *QAM* no desmapeador de *QAM* r302f para obter ganho de diversidade e ganho de *SNR*. Neste ponto, o combinador pode incluir processos de alinhamento e adicionar cada bloco *L1 FEC* e dividir o bloco *L1 FEC* adicionado. Para o caso onde somente parte do último bloco *FEC* é repetido tal como apresentado na figura 60, apenas a parte repetida pode ser dividida em até um número de cabeçalho de bloco *FEC* e a outra parte pode ser dividida por um valor que é um a menos do que um

número de cabeçalho de bloco *FEC*. Por outras palavras, o número divisor corresponde a um número de portadoras que é adicionado a cada portadora.

A figura 65 apresenta um outro exemplo de agendamento de bloco *L1*. A figura 65 é diferente da figura 60 em que, em vez de executar a zona de preenchimento zeros ou repetição quando os blocos *L1* não enchem um símbolo *OFDM*, o símbolo *OFDM* pode ser preenchido com redundância de paridade executando menos puncionagem em código encurtado/puncionado no transmissor. Por outras palavras, quando o módulo de puncionagem de paridade 304c é executado na figura 5, a taxa de código efectiva pode ser determinada de acordo com o rácio de puncionagem, deste modo, puncionando dado que menos bits têm que ter zonas de preenchimento zeros, a taxa de código efectiva pode ser reduzida podendo ser obtido um melhor ganho de codificação. O módulo de despuncionagem r303a de um receptor tal como apresentado na figura 32 pode executar a despuncionagem considerando a redundância de paridade menos puncionada. Neste ponto, porque um receptor e um transmissor podem ter informações da dimensão total do bloco *L1*, o rácio de puncionagem pode ser calculado.

A figura 62 apresenta um outro exemplo de campo de sinalização *L1*. A figura 62 é diferente da figura 41 em que, para um caso em que o tipo de fatia de dados é *CCM*, pode ser transmitido um endereço inicial (21 bits) da *PLP*. Isso pode possibilitar que o *FECFRAME* de cada *PLP* forme um

quadro de transmissão, sem o *FECFRAME* estar alinhado com uma posição inicial de um quadro de transmissão. Deste modo, a informação complementar da zona de preenchimento, que pode ocorrer quando a largura de uma fatia de dados é estreita, pode ser eliminada. Um receptor, quando um tipo de fatia de dados é *CCM*, pode obter informação *ModCod* do preâmbulo no trajecto de sinalização *L1* do módulo desmodulador *BICM* tal como apresentado na figura 31, em vez de obter a mesma do cabeçalho do *FECFRAME*. Além disso, mesmo quando ocorre uma passagem com velocidade num local aleatório do quadro de transmissão, a sincronização *FECFRAME* pode ser realizada sem atraso porque o endereço de início da *PLP* já pode ser obtido a partir do preâmbulo.

A figura 63 apresenta um outro exemplo de campos de sinalização *L1* que podem reduzir a informação complementar do endereçamento *PLP*.

A figura 64 apresenta os números de símbolos *QAM* que correspondem a um *FECFRAME* dependendo dos tipos de modulação. Neste ponto, um divisor comum maior do símbolo *QAM* é 135, sendo que deste modo pode ser reduzida uma informação complementar de $\log_2(135) \approx 7$ bits. Deste modo a figura 63 é diferente da figura 62 em que um número de bits de campo *PLP_start* pode ser reduzido de 21 bits para 14 bits. Este é um resultado de considerar 135 símbolos como um único grupo e endereçar o grupo. Um receptor pode obter um índice de portadora *OFDM* onde a *PLP* começa num quadro de transmissão após a obtenção do valor do campo *PLP_start* e

multiplicá-lo por 135.

A figura 66 e figura 68 mostram exemplos de intercalador de símbolos 308 o qual pode intercalar no tempo símbolos de dados que são enviados do módulo de inserção de cabeçalho *ModCod* 307 no trajecto de dados do módulo *BICM* tal como apresentado na figura 4.

A figura 66 é um exemplo do intercalador de blocos para intercalar no tempo, que pode operar numa base de fatias de dados. O valor da linha quer dizer um número de células de carga útil em quatro dos símbolos *OFDM* dentro de uma fatia de dados. A intercalação com base no símbolo *OFDM* pode não ser possível porque o número de células pode variar entre células contíguas *OFDM*. A coluna valor *K* significa uma profundidade de intercalação no tempo, que pode ser de 1, 2, 4, 8 ou 16... A sinalização de *K* para cada fatia de dados pode ser executada dentro da sinalização *L1*. A intercalação de frequência 403 tal como apresentado na figura 9 pode ser executada antes da intercalação da hora 308 tal como apresentado na figura 4.

A figura 67 mostra um desempenho de intercalação do intercalador no tempo tal como apresentado na figura 66. Supõe-se que um valor de coluna é 2, um valor de linha é 8, uma largura de fatia de dados é 12 células de dados, e que não se encontram na fatia de dados pilotos contínuos. A figura de topo na figura 67 é uma estrutura de símbolo *OFDM* quando a intercalação no tempo não é executada e a figura

inferior é uma estrutura de símbolo *OFDM* quando a intercalação do tempo é executada. As células negras representam um piloto disperso e as células não-negras representam as células de dados. O mesmo tipo de células de dados representa um símbolo *OFDM*. Na figura 100 as células de dados que correspondem a um único símbolo *OFDM* são intercaladas em dois símbolos. É utilizada uma memória de intercalação que corresponde a oito símbolos *OFDM* mas a profundidade da intercalação corresponde a apenas dois símbolos *OFDM*, sendo que desse modo não é obtida a profundidade completa de intercalação.

A figura 68 é sugerida para alcançar uma profundidade de intercalação total. Na figura 68 as células negras representam pilotos dispersos e as células não-negras representam as células de dados. O intercalador no tempo tal como apresentado na figura 68 pode ser implementado numa forma de intercalador de bloco e pode intercalar fatias de dados. Na figura 68, um número de coluna, K representa uma largura de fatia de dados, um número de linha, N representa a profundidade da intercalação no tempo e o valor, K pode ser valores aleatórios ou seja, $K=1,2,3,\dots$. O processo de intercalação inclui escrever célula de dados de um modo de retorção da coluna e leitura na direcção da coluna, excluindo as posições piloto. Quer dizer, pode-se dizer que a intercalação é executada de um modo torcido de linha-coluna.

Além disso, num transmissor, as células que são lidas de um modo torcido em coluna da memória de intercalação correspondem a um único símbolo *OFDM*, podendo as posições-piloto dos símbolos *OFDM* ser mantidas enquanto se intercala as células.

Deste modo, num receptor, as células que são lidas de um modo torcido em coluna da memória de desintercalação correspondem a um único símbolo *OFDM*, podendo as posições-piloto dos símbolos *OFDM* ser mantidas enquanto se desintercala as células.

A figura 69 mostra o desempenho de intercalação no tempo da figura 68. Para comparação com a figura 66, supõe-se que um número de coluna é 8, uma largura de fatia de dados é 12 células de dados, e que não se encontram na fatia de dados pilotos contínuos. Na figura 69 as células de dados que correspondem a um único símbolo *OFDM* são intercaladas em oito símbolos *OFDM*. Tal como apresentado na figura 102, é utilizada uma memória de intercalação que corresponde a oito símbolos *OFDM* e a profundidade da intercalação corresponde a apenas oito símbolos *OFDM*, sendo que desde modo é obtida a profundidade completa de intercalação.

O intercalador no tempo tal como apresentado na figura 68 pode ser vantajoso em que a profundidade total de intercalação pode ser obtida utilizando memória idêntica; a profundidade da intercalação pode ser flexível, ao

contrário da figura 66; conseqüentemente, um comprimento de quadro de transmissão pode também ser flexível, ou seja, as linhas não precisam ser múltiplos de quatro. Além disso, o intercalador no tempo utilizado para a fatia de dados pode ser idêntico ao processo de intercalação utilizado para o preâmbulo e também pode ter afinidade com um sistema de transmissão digital que usa *OFDM* geral. Especificamente, o intercalador no tempo 308 tal como apresentado na figura 4 pode ser utilizado antes do intercalador de frequência 403 tal como apresentado na figura 9 é utilizada. A respeito da complexidade de um receptor, nenhuma memória adicional pode ser necessário a não ser a lógica de endereço adicional que pode exigir uma complexidade muito pequena.

A figura 70 mostra um desintercalador de símbolos correspondente (r308) num receptor. Pode executar desintercalação depois de receber a saída do removedor de cabeçalho de quadro r401. Nos processos de desintercalação comparados com a figura 66, os processos de escrita e leitura da intercalação de blocos encontram-se invertidos. Ao utilizar informação de posição-piloto, o desintercalador no tempo pode executar a desintercalação virtual ao não escrever para ou ler a partir de uma posição piloto na memória do intercalador e ao escrever para ou ler a partir de uma posição de célula de dados na memória do intercalador. A informação desintercalada pode ser emitida para o extractor *ModCod* r307.

A figura 71 mostra um outro exemplo de intercalador no tempo. Pode ser executada a escrita no

sentido diagonal e a leitura linha a linha. Tal como na figura 68 a intercalação é realizada tendo em conta as posições piloto. A leitura e escrita não é realizada para as posições piloto, mas a memória de intercalação é acessada considerando apenas as posições da célula de dados.

A figura 72 mostra um resultado de intercalação utilizando o processo apresentado na figura 71. Quando comparado com a figura 69, as células com os mesmos padrões encontram-se dispersas, não somente no domínio do tempo, mas também no domínio da frequência. Por outras palavras, a profundidade completa da intercalação pode ser obtida em ambos os domínios do tempo e frequência.

A figura 75 mostra um desintercalador de símbolos r308 de um receptor correspondente. A saída do removedor do cabeçalho do quadro r401 pode ser desintercalada. Quando comparada com a figura 66, a desintercalação mudou a ordem de leitura e escrita. A desintercalação no tempo pode utilizar informação da posição-piloto para realizar desintercalação virtual tal como não ser executada qualquer leitura ou escrita em posições-piloto mas de forma que a leitura ou a escrita possa ser realizada apenas em posições de células de dados. Os dados desintercalados podem ser emitidos para o extractor *ModCod* r307.

A figura 73 mostra um exemplo do processo de endereçamento da figura 72. *NT* significa profundidade da

intercalação no tempo e ND significa largura da fatia de dados. Supõe-se que um valor de coluna, N é 8, uma largura de fatia de dados é 12 células de dados, e que não se encontram na fatia de dados pilotos contínuos. A figura 73 representa um processo de geração de endereços para escrever dados numa memória de intercalação no tempo, quando um transmissor realiza a intercalação no tempo. O endereçamento começa a partir de um primeiro endereço com endereço de linha (*Row Address - RA*) = 0 e endereço de coluna (*Column Address - CA*) = 0. A cada ocorrência de endereçamento, RA e CA são aumentados. Para RA , pode ser executada uma operação de módulo com os símbolos *OFDM* utilizados no intercalador no tempo. Para CA , pode ser executada uma operação de módulo com um número de transportadoras que corresponde a uma largura de fatia de dados. RA pode ser incrementado em 1 quando portadoras que correspondem a uma fatia de dados são gravadas numa memória. Escrever numa memória pode ser executada somente quando uma localização do endereço actual não é uma localização de um piloto. Se a localização do endereço actual é uma localização de um piloto, somente o valor do endereço pode ser aumentado.

Na figura 73, um número de coluna, K representa a largura de fatia de dados, um número de linha, N representa a profundidade da intercalação no tempo e o valor, K podem ser valores aleatórios ou seja, $K=1,2,3,\dots$. O processo de intercalação pode incluir escrever células de dados de um modo de retorção da coluna e ler na direcção da coluna,

excluindo as posições piloto. Por outras palavras, a memória de intercalação virtual pode incluir posições piloto mas posições-piloto podem ser excluídas na intercalação actual.

A figura 76 mostra a desintercalação, um processo inverso da intercalação no tempo tal como apresentado na figura 71. Escrever linha a linha e ler na direcção diagonal pode restaurar células nas suas sequências originais.

O processo de endereçamento utilizado num transmissor pode ser utilizado num receptor. O receptor pode escrever dados recebidos numa memória de desintercalação no tempo linha a linha e pode ler os dados escritos utilizando valores de endereço gerados e informação de localização piloto que podem ser gerados de forma semelhante àquele de um transmissor. Como uma forma alternativa, os valores de endereço gerados e informação piloto que foram utilizados para a escrita podem ser utilizados para a leitura linha a linha.

Estes processos podem ser aplicados num preâmbulo que transmite *L1*. Dado que cada símbolo *OFDM* que compreende preâmbulo pode ter pilotos em locais idênticos, pode ser executada a intercalação referente a valores de endereço tendo em conta as localizações piloto ou intercalação referentes a valores de endereço sem ter em conta as localizações piloto. Para o caso de se referir aos valores

de endereço sem levar em conta as localizações piloto, o transmissor armazena de cada vez dados numa memória de intercalação no tempo. Para caso deste género, um tamanho de memória necessário para executar a intercalação/desintercalação de preâmbulos num receptor ou num transmissor torna-se idêntico a uma série de células de carga útil existentes nos símbolos *OFDM* utilizados para a intercalação no tempo.

A figura 74 é um outro exemplo de intercalação no tempo $L1$. Neste exemplo, a intercalação no tempo pode colocar portadoras em todos os símbolos *OFDM* enquanto que as portadoras estariam todas localizadas num único símbolo *OFDM* único se não tiver sido executada qualquer intercalação no tempo. Por exemplo, para dados localizados num primeiro símbolo *OFDM*, a primeira portadora do primeiro símbolo *OFDM* estará localizada na sua localização original. A segunda portadora do primeiro símbolo *OFDM* estará localizada num índice de portadora do segundo símbolo *OFDM*. Por outras palavras, a i -ésima portadora de dados que se encontra localizada no n -ésimo símbolo *OFDM* estará localizada num i -ésimo índice portador do $(i+n) \bmod N$ -ésimo símbolo *OFDM* símbolo, em que $i = 0, 1, 2, \dots$ número de portadora -1 , $n=0, 1, 2, \dots, N-1$, e N é um número de símbolos *OFDM* utilizados na intercalação no tempo $L1$. Neste processo de intercalação no tempo $L1$, pode-se dizer que a intercalação para todos os símbolos *OFDM* é executada de um modo distorcido tal como apresentado na figura 107. Apesar de as posições piloto não estarem ilustrados na figura 107,

tal como mencionado acima, a intercalação pode ser aplicada a todos os símbolos *OFDM* incluindo aos símbolos piloto. Ou seja, pode-se dizer que a intercalação pode ser realizada para todos os símbolos *OFDM* sem considerar posições piloto ou independentemente de os símbolos *OFDM* serem símbolos piloto ou não.

Se um tamanho de um bloco *LDPC* utilizado em *L1* for menor do que um tamanho de um único símbolo *OFDM*, as portadoras restantes podem ter cópias de peças do bloco *LDPC* ou podem estar preenchidas com zeros. Neste ponto, pode ser realizada uma intercalação simultânea como acima. De modo semelhante, na figura 74, um receptor pode executar a desintercalação armazenando todos os blocos utilizados na intercalação no tempo *L1* interleaving numa memória e lendo os blocos na ordem em que foram intercalados, ou seja, na ordem de números escritos em blocos apresentados na figura 74.

Quando é utilizado um intercalador de blocos tal como apresentado na figura 73 são utilizadas duas memórias tampão. Especificamente, enquanto uma memória tampão se encontra a armazenar símbolos de entrada, símbolos de entrada anteriores podem ser lidos da outra memória tampão. Uma vez que estes processos tenham sido realizados por um bloco de intercalação de um símbolo, a desintercalação pode ser realizada comutando a ordem de leitura e escrita, para evitar conflitos de acesso à memória. Esta desintercalação do estilo "pingue-pongue" pode ter uma simples lógica de

geração de endereço. No entanto, a complexidade do hardware pode ser aumentada quando se utilizam duas memórias tampão de intercalação de símbolos.

A figura 77 mostra um exemplo de um desintercalador de símbolos r308 ou r308-1 tal como apresentado na figura 31. Esta forma de realização da invenção proposta pode utilizar somente uma única memória tampão para realizar a desintercalação. Uma vez que um valor de endereço seja gerado pela lógica de geração de endereço, o valor do endereço pode ser emitido da memória tampão, podendo ser executada a operação de reposicionamento pode ser realizada através do armazenamento de um símbolo que é introduzido no mesmo endereço. Por estes processos, pode ser evitado um conflito de acesso à memória durante a leitura e escrita. Além disso, a desintercalação do símbolo pode ser realizada utilizando apenas uma única memória tampão. Podem ser definidos parâmetros para explicar esta regra de geração de endereço. Tal como apresentado na figura 73, um número de linhas de uma memória de desintercalação pode ser definido como a profundidade da intercalação no tempo, D e um número de colunas da memória de desintercalação pode ser definido como largura da fatia de dados, W . Em seguida o gerador de endereços pode gerar os seguintes endereços.

i-th sample ON j-th block, including pilot

i=0, 1, 2, ..., N-1;

*N=D*W;*

$$Ci, j = i \bmod W;$$

$$Tw = ((Ci, j \bmod D) * j) \bmod D;$$

$$Ri, j = ((i \operatorname{div} W) + Tw) \bmod D;$$

$$Li, j(1) = Ri, j * W + Ci, j;$$

Or

$$Li, j(2) = Ci, j * D + Ri, j;$$

Os endereços incluem posições piloto, sendo que deste modo assume-se que os símbolos de entrada incluem posições piloto. Se os símbolos de entrada que incluem apenas símbolos de dados precisam ser processados, podem ser necessária a lógica de controlo adicional que ignora os endereços correspondentes. Neste ponto, i representa um índice de símbolo de entrada, j representa um índice de bloco de intercalação de entrada, e $N = D * W$ representa um comprimento de bloco de interligação. Operação Mod representa operação do módulo que emite o restante após a divisão. A operação Div representa a operação de divisão que emite o quociente após a divisão. Ri, j e Ci, j representam o endereço da linha e endereço da coluna da i -ésima entrada de símbolo do j -ésimo bloco de intercalação, respectivamente. Tw representa valor de retorção da coluna para endereços onde os símbolos estão localizados. Por outras palavras, cada coluna pode ser considerada como uma memória tampão onde a retorção independente é realizada de acordo com valores Tw . Li, j representa um endereço quando a memória tampão individual é implementada numa memória sequencial de uma dimensão sequencial, e não de duas

dimensões. L_i, j podem ter valores de 0 a $(N-1)$. São possíveis dois processos diferentes. L_i, j (1) é utilizado quando a matriz da memória se encontra ligada linha a linha e L_i, j (2) é utilizado quando a matriz da memória se encontra ligada em coluna a coluna.

A figura 78 mostra um exemplo de endereços de linha e coluna para a desintercalação no tempo quando D é 8 e W é 12. J tem início a partir de $j=0$ e para cada valor j , uma primeira linha pode representar o endereço da linha e uma segunda linha pode representar o endereço da coluna. A figura 78 mostra apenas endereços dos primeiros 24 símbolos. Cada índice de coluna pode ser idêntica ao índice i dos símbolos de entrada.

A figura 80 mostra um exemplo de um transmissor *OFDM* que utiliza fatias de dados. Tal como apresentado na figura 80, o transmissor pode incluir um trajecto *PLP* de dados, um trajecto de sinalização *L1*, um construtor de quadro, e uma parte de modulação *OFDM*. O trajecto *PLP* de dados é indicado por blocos com linhas horizontais e linhas verticais. O trajecto de sinalização *L1* é indicado por blocos com linhas inclinadas. Os módulos de processamento de entrada 701-0, 701-N, 701-K, e 701-M podem incluir blocos e sequências de módulo de interface de entrada 202-1, sincronizador de fluxo de entrada 203-1, compensador de atraso 204-1, módulo de eliminação de pacotes nulos 205-1, codificador *CRC* 206-1, módulo de inserção de cabeçalho *BB* 207-1, e codificador *BB* 209 executado para cada *PLP* tal

como apresentado na figura 2 . Os módulos *FEC* 702-0, 702-N, 702-K, e 702-M podem incluir blocos e sequências de módulo de codificação externa 301 e módulo de codificação interna 303 tal como apresentado na figura 4. Um módulo *FEC* 702-L1 utilizado no trajecto *L1* pode incluir blocos e sequências do módulo de codificação externo 301-1 e módulo de codificação interno 303-1 encurtado/puncionado tal como apresentado na figura 4. O módulo de sinal *L1* 700-L1 *L1* pode gerar informação *L1* necessária para compor um quadro.

Os módulos de intercalação de bits 703-0, 703-N, 703-K, e 703-M podem incluir blocos e sequências de intercalador interno 304 e desmultiplicador de bits 305 tal como apresentado na figura 37. O módulo de intercalação de bits 703-L1 utilizado no trajecto *L1* pode incluir blocos e sequências de intercalador interno 304-1 e desmultiplicador de bits 305-1 tal como apresentado na figura 4. Módulos mapeadores de símbolos 704-0, 704-N, 704-K, e 704-M podem executar funções idênticas às funções do mapeador de símbolos 306 apresentado na figura 4. O módulo mapeador de símbolos 704-L1 utilizado no trajecto *L1* pode executar funções idênticas às funções do mapeador de símbolos 306-1 apresentado na figura 4. Os módulos de cabeçalho *FEC* 705-0, 705-N, 705-K, e 705-M podem executar funções idênticas às funções do módulo de inserção de cabeçalho *ModCod* 307 apresentado na figura 4. O módulo de cabeçalho *FEC* 705-L1 para o trajecto *L1* pode executar funções idênticas às funções do módulo de inserção de cabeçalho *ModCod* 307-1 apresentado na figura 4.

Os módulos mapeadores de fatias de dados 706-0 e 706-K podem agendar blocos *FEC* a fatias de dados correspondentes e podem transmitir os blocos agendados *FEC*, onde os blocos *FEC* correspondem a *PLPs* que se encontram atribuídos a cada fatia de dados. O mapeador de preâmbulo 707-*L1* pode agendar blocos *FEC* de sinalização *L1* para preâmbulos. Blocos *FEC* de sinalização *L1* são transmitidos em preâmbulos. Os módulos de intercalação no tempo 708-0 e 708-K podem executar funções idênticas às funções do intercalador de símbolos 308 apresentado na figura 4 que pode intercalar fatias de dados. O intercalador no tempo 708-*L1* utilizado no trajecto *L1* pode executar funções idênticas às funções do intercalador de símbolos 308-1 apresentado na figura 4.

Alternativamente, o intercalador no tempo 708-*L1* utilizado no trajecto *L1* pode executar funções idênticas ao intercalador de símbolos 308-1 apresentado na figura 3, mas apenas nos símbolos de preâmbulo.

Os blocos de intercalador de frequência 709-0 e 709-K podem executar a intercalação de frequência em fatias de dados. O intercalador de frequência 709-*L1* utilizado no trajecto *L1* pode executar a intercalação de frequência de acordo com a largura de banda do preâmbulo.

O módulo gerador piloto 710 pode gerar pilotos que são adequados para piloto contínuo (CP), o piloto disperso (SP), bordo de fatia de dados, e preâmbulo. Um quadro pode ser construído a partir de agendamento da fatia

de dados, preâmbulo, e piloto no módulo 711. O módulo IFFT 712 e módulo de inserção GI 713 pode executar funções idênticas às funções do módulo *IFFT 501*, e blocos de módulo de inserção *GI 503* apresentados na figura 18, respectivamente. Por último, *DAC 714* pode converter sinais digitais em sinais analógicos e os sinais convertidos podem ser transmitidos.

A figura 81 mostra um exemplo de um receptor *OFDM* que utiliza fatia de dados. Na figura 81, o sintonizador r700 pode executar as funções do sintonizador/AGC r603 e as funções do conversor descendente r602 apresentado na figura 61. O ADC r701 pode converter sinais analógicos recebidos em sinais digitais. O sincronizador de tempo/frequência r702 pode executar funções idênticas às funções do sincronizador de tempo/frequência r505 apresentado na figura 62. O detector de quadro r703 pode executar funções idênticas às funções do detector de quadro r506 apresentado na figura 62.

Neste ponto, após a sincronização do tempo/frequência ter sido realizada, a sincronização pode ser melhorada utilizando preâmbulo em cada quadro que é enviado a partir do detector r703 durante o processo de localização.

O removedor de GI r704 e módulo FFT r705 podem executar funções idênticas às funções do removedor GI r503 e o módulo *FFT r502* apresentado na figura 62, respectivamente.

O avaliador de canal r706 e canal EQ r707 pode executar uma parte da avaliação de canal e uma parte de equalização de canal, do canal *Est/Eq* r501 tal como apresentado na figura 62. Analisador sintáctico de quadros r708 pode produzir uma fatia de dados e preâmbulo onde os serviços seleccionados por um utilizador são transmitidos. Os blocos assinalados por linhas inclinadas processam um preâmbulo. Os blocos assinalados por linhas horizontais que podem incluir *PLP* comum, processam fatias de dados. O desintercalador de frequência r709-*L1* utilizado no trajecto *L1* pode executar a desintercalação de frequência dentro da largura de banda do preâmbulo. O desintercalador de frequência r709 utilizado no trajecto da fatia de dados pode executar a desintercalação de frequência dentro da fatia de dados. O módulo de descodificação de cabeçalho *FEC* r712-*L1*, o desintercalador no tempo r710-*L1*, e desmapeador de símbolo r713 *L1* utilizado no trajecto *L1* pode executar funções idênticas às funções do extractor *ModCod* r307-1, desintercalador de símbolos r308-1, e desmapeador de símbolo r306-1 apresentado na figura 31.

O desintercalador de bits r714-*L1* pode compreender módulos e sequências de demux de bits r305-1 e desintercalador interno 304-1 tal como apresentado na figura 31. O módulo de descodificação r702-*L1* pode compreender módulos e sequências de módulo de codificação interno encurtado/puncionado r301-1 e módulo de descodificação externo r303-1 apresentado na figura 31. Neste ponto, a saída do trajecto *L1* pode ser informação de

sinalização *L1* e pode ser enviada para um controlador de sistema para restaurar dados *PLP* que são transmitidos em fatias de dados.

O desintercalador no tempo *r710* utilizado no trajecto da fatia de dados pode executar funções idênticas às funções do desintercalador de símbolos *r308* apresentado na figura 31. O analisador de fatia de dados *r711* pode emitir *PLP* seleccionado pelo utilizador das fatias de dados e, se necessário, *PLP* comum associado à *PLP* seleccionada pelo utilizador. O módulo de descodificação de cabeçalho *FEC* *r712-C* e *r712-K* pode executar funções idênticas às funções do extractor *ModCod* *r307* apresentado na figura 31. Os desmapeadores de símbolos *r713-C* e *r713-K* podem executar funções idênticas às funções do desmapeador de símbolos *r306* apresentado na figura 31.

O desintercalador de bits *r714-C* e *r714-K* pode compreender blocos e sequências de bit demux *r305* e desintercalador interno *r304* tal como apresentado na figura 31. Os módulos de descodificação *FEC* *r715-C* e *r715-K* podem compreender blocos e sequências de módulo de descodificação interna *r303* e módulo de descodificação externa *r301* tal como apresentado na figura 31. Por último, o processador de saída *r716C* e *r716-K* pode compreender blocos e sequências de descodificador *BB* *r209*, removedor de cabeçalho *r207-1* *BB*, descodificador *CRC* *r206-1*, módulo de inserção de pacotes nulos *r205-1*, recuperação de atraso *r204-1*, módulo de recuperação de relógio de saída *r203-1*, e módulo de

interface de saída r202-1 que são executados para cada *PLP* na figura 2. Se um *PLP* comum for utilizado, à *PLP* comum e *PLP* de dados associado à *PLP* comum podem ser transmitidos para um recombinação *TS* e podem ser transformados num *PLP* seleccionado por utilizador.

Deverá ser notado da figura 81, que num receptor, os blocos no trajecto *L1* não se encontram simetricamente sequenciados para um transmissor ao contrário ao trajecto de dados onde os blocos se encontram simetricamente posicionados ou na sequência oposta de um transmissor. Por outras palavras, para o trajecto de dados, encontram-se posicionados o desintercalador de frequência r709, desintercalador no tempo r710, analisador sintáctico de fatia de dados r711, e módulo de descodificação de cabeçalho *FEC* r712-C e r712-K. Contudo, para o trajecto *L1*, encontram-se posicionados o desintercalador de frequência r709-*L1*, módulo de descodificação de cabeçalho *FEC* r712-*L1*, e desintercalador no tempo r710-*L1*.

A figura 79 mostra um exemplo de intercalação geral de blocos num domínio de símbolo de dados onde os pilotos não são utilizados. Tal como pode ser visto na figura à esquerda, a memória de intercalação pode ser preenchida sem pilotos negros. Para formar uma memória rectangular, se necessário podem ser utilizadas células de preenchimento. Na figura à esquerda, as células de preenchimento encontram-se indicadas como células com linhas inclinadas. No exemplo, porque um piloto contínuo

pode se sobrepor a um tipo de padrão piloto disperso, é necessário um total de três células de preenchimento durante quatro durações de símbolo *OFDM*. Finalmente, é apresentado na figura do meio, o conteúdo da memória intercalada.

Tal como na figura à esquerda da figura 79, seja escrevendo linha a linha e realizando a torção da coluna, ou escrevendo de uma forma distorcida desde o início, pode ser executado. A saída do intercalador pode incluir a leitura linha a linha da memória. Os dados de saída que foram lidos podem ser colocados tal como apresentado na figura à direita, quando é considerada a transmissão *OFDM*. Neste momento, por simplicidade, o intercalamento da frequência pode ser ignorado. Como pode ser visto na figura, a diversidade da frequência não é tão elevada como na figura 73, mas é mantida num nível semelhante. Acima de tudo, pode ser vantajoso na medida em que a memória necessária para executar a intercalação e desintercalação pode ser optimizada. No exemplo, o tamanho da memória pode ser reduzido de $W * D$ para $(W-1) * D$. Como a largura da fatia de dados se torna maior, o tamanho da memória pode ser ainda mais reduzida.

Para entradas de desintercalador no tempo, um receptor deve restaurar o conteúdo da memória tampão da memória numa forma da figura do meio, considerando simultaneamente as células de zona de preenchimento. Basicamente, os símbolos *OFDM* podem ser lidos símbolo a

símbolo e podem ser guardados linha a linha. Pode ser executada a destorção correspondente à torção da coluna. A saída do desintercalador pode ser saída numa forma de leitura linha a linha a partir da memória da figura da esquerda. Desta forma, quando comparado com o processo apresentado na figura 73, a informação complementar piloto pode ser minimizada e, conseqüentemente, a memória de intercalação/desintercalação pode ser minimizada.

A figura 82 mostra um exemplo de um intercalador no tempo 708-L1 para trajecto L1 da figura 80. Tal como apresentado na figura 82, a intercalação no tempo para o preâmbulo onde L1 é transmitida, pode incluir células de dados L1 de intercalação, excluindo os pilotos que são geralmente transmitidos no preâmbulo. O processo de intercalação pode incluir a escrita de dados de entrada numa direcção diagonal (linhas sólidas) e leitura dos dados linha a linha (linhas pontilhadas), utilizando processos idênticos aos que são apresentados em referência à figura 73.

A figura 82 mostra um exemplo de um desintercalador no tempo r712-L1 no trajecto L1 tal como apresentado na figura 81. Tal como apresentado na figura 82, para um preâmbulo onde L1 é transmitido, pode ser realizada a desintercalação da célula de dados L1, excluindo os pilotos que são regularmente transmitidos no preâmbulo. O processo de desintercalação pode ser idêntico ao processo apresentado na figura 76, onde os dados de

entrada são escritos linha a linha (linhas sólidas) e lidos na diagonal (linha pontilhada). Os dados de entrada não incluem qualquer piloto, conseqüentemente, os dados de saída apresentam células de dados $L1$ que também não incluem piloto. Quando um receptor utiliza uma única memória tampão num desintercalador no tempo para o preâmbulo, a estrutura do gerador de endereços que apresenta uma memória desintercaladora tal como apresentado na figura 77 pode ser utilizada.

A desintercalação $r712-L1$ pode ser realizada utilizando operações de endereço tal como se segue:

i-th sample ON j-th block, including pilot

$i=0, 1, 2, \dots, N-1;$

$N=D*W;$

$C_{i,j}=i \bmod W;$

$T_w=((C_{i,j} \bmod D)*j) \bmod D;$

$R_{i,j}=(i \div W + T_w) \bmod D;$

$L_{i,j(1)}=R_{i,j}*W+C_{i,j};$

Or

$L_{i,j(2)}=C_{i,j}*D+R_{i,j};$

Nas operações acima, um comprimento de uma linha, W é um comprimento de uma linha de uma memória de intercalação tal como apresentado na figura 82. O comprimento da coluna D é uma profundidade de intercalação no tempo de preâmbulo, que é um número de símbolos *OFDM* que

são necessários para a transmissão de preâmbulos.

A figura 83 mostra um exemplo de formação de símbolos *OFDM* agendando pilotos e preâmbulos de entrada do construtor de quadros 711 tal como apresentado na figura 80. As células em branco formam um cabeçalho *L1* que é um sinal de saída do cabeçalho *FEC 705-L1* no trajecto *L1*, tal como apresentado na figura 80. As células cinzentas representam os pilotos contínuos para o preâmbulo que são gerados pelo módulo de geração de piloto 710 tal como apresentado na figura 80. As células com padrões representam as células de sinalização *L1* que são um sinal de saída do mapeador de preâmbulos 707-*L1* tal como apresentado na figura 80. A figura da esquerda representa símbolos *OFDM* quando a intercalação no tempo se encontra desligada e a figura da direita representa símbolos *OFDM* quando a intercalação no tempo se encontra ligada. O cabeçalho *L1* pode ser excluído da intercalação no tempo porque o cabeçalho *L1* transmite um comprimento de campo de sinalização *L1* e uma informação de marca de ligar/desligar a intercalação no tempo. É porque o cabeçalho *L1* é adicionado antes da intercalação no tempo. Tal como referido acima, a intercalação no tempo é realizada excluindo células piloto. O restante das células de dados *L1* pode ser intercalado tal como apresentado na figura 82, podendo ser atribuído a subportadoras *OFDM*.

A figura 84 mostra um exemplo de intercaladores no tempo 708-0~708-K que podem intercalar símbolos de dados

que são enviados dos mapeadores de fatias de dados 706-0~706-K no trajecto de dados de um transmissor *OFDM* utilizando fatia de dados apresentado na figura 80. O intercalamento no tempo pode ser realizado para cada fatia de dados. Símbolos intercalados no tempo podem ser emitidos em intercaladores de frequência 709-0~709-K.

A figura 84 mostra também um exemplo de um intercalador no tempo simples que utiliza uma única memória tampão. A figura 84a mostra uma estrutura de símbolos *OFDM* antes da intercalação no tempo. Os blocos com os mesmos padrões representam o mesmo tipo de símbolos *OFDM*. As figuras 84b e 84c mostram uma estrutura de símbolos *OFDM* após a intercalação no tempo. O processo de intercalação no tempo pode ser dividido em tipo 1 e tipo 2. Cada tipo pode ser executado alternativamente para símbolos pares e símbolos ímpares. Um receptor pode executar a desintercalação em conformidade. Uma das razões para utilizar alternativamente o tipo 1 e tipo 2 é o reduzir a memória necessária num receptor utilizando uma única memória tampão durante a desintercalação no tempo.

A figura 84b mostra uma intercalação no tempo utilizando a intercalação tipo 1. Os símbolos de entrada podem ser escritos na direcção diagonal descendente e pode ser lidos num sentido de fila. A figura 84c mostra uma intercalação no tempo utilizando a intercalação tipo 2. Os símbolos de entrada podem ser escritos na direcção diagonal ascendente e podem ser lidos num sentido de fila. A

diferença entre tipo 1 e tipo 2 é se uma direcção de escrita de símbolo de entrada é para cima ou para baixo. Os dois processos são diferentes num modo de escrever símbolos, contudo os dois processos são idênticos em termos de exibir profundidade de intercalação em tempo integral e diversidade de frequência integral. No entanto utilizar estes processos pode provocar um problema durante a sincronização num receptor devido à utilização de dois esquemas de intercalação.

Podem existir duas soluções possíveis. A primeira solução pode ser sinalizar 1 bit de um tipo de intercalação de um primeiro bloco de intercalação que surge em primeiro lugar após cada preâmbulo, através de sinalização *L1* de preâmbulo. Este processo é a execução de uma intercalação correcta através de sinalização. A segunda solução pode ser a formação de um quadro para apresentar um comprimento de um número par de blocos de intercalação. Utilizando este processo, um primeiro bloco de intercalação de cada quadro pode ter um tipo idêntico, portanto, o problema de sincronização do bloco de intercalação pode ser resolvido. Por exemplo, o problema de sincronização pode ser resolvido aplicando a intercalação do tipo 1 a um primeiro bloco de intercalação e aplicando sequencialmente aos próximos blocos de intercalação dentro de cada quadro, terminando depois um último bloco de intercalação de cada quadro com a intercalação tipo 2. Este processo requer que um quadro seja composto por dois blocos de intercalação mas pode ser vantajoso em que não é necessária qualquer sinalização

adicional tal como no primeiro processo.

A figura 89 mostra uma estrutura de um desintercalador no tempo $r710$ de um receptor apresentado na figura 81. A desintercalação no tempo pode ser realizada na saída do desintercalador de frequência $r709$. O desintercalador no tempo da figura 89 representa um esquema de desintercalamento que é um processo inverso de um intercalamento no tempo apresentado figura 84. A desintercalação, em comparação com a figura 84, terá uma maneira oposta de leitura e escrita. Por outras palavras, o desintercalador de tipo 1 pode escrever símbolos de entrada numa direcção de linha e pode ler os símbolos escritos no sentido diagonal descendente. O desintercalador de tipo 2 pode escrever símbolos de entrada no sentido diagonal descendente e pode ler os símbolos escritos no sentido da fila. Estes processos podem permitir a escrita dos símbolos recebidos onde os símbolos são previamente lidos ao realizar um sentido de escrita dos símbolos do desintercalador do tipo 2 idêntico a um sentido de símbolos de leitura de desintercalador do tipo 1. Deste modo, um receptor pode executar a desintercalação utilizando uma única memória tampão. Além disso, pode ser realizada uma implementação simples devido ao facto dos processos de desintercalação do tipo 1 e tipo 2 serem realizados por símbolos de escrita e leitura num sentido diagonal ou num sentido em linha.

Contudo utilizar estes processos pode provocar um

problema na sincronização num receptor por causa da utilização de dois esquemas de intercalação. Por exemplo, desintercalando símbolos intercalados do tipo 1 de um modo do tipo 2 pode provocar a deterioração no desempenho. Podem existir duas soluções possíveis. A primeira solução pode ser determinar um tipo de um bloco de intercalação que surge após um preâmbulo, utilizando 1 bit de um tipo de intercalação de uma parte de sinalização *L1* transmitida. A segunda solução pode ser a realização da desintercalação utilizando um tipo de acordo com um primeiro bloco de intercalação dentro de um quadro, se um número de blocos de intercalação dentro de um quadro for um número par. O símbolo desintercalado pode ser emitido para dentro do analisador sintáctico r711.

A figura 85 mostra uma lógica de geração de endereço que é idêntica a uma lógica de geração de endereço de uma única memória tampão, quando um intercalador de blocos utiliza duas memórias tampão de memória tal como na figura 73. A lógica da geração de endereço pode executar funções idênticas às funções apresentadas na figura 73. Ao definir um tempo uma profundidade *D* de intercalação no tempo como um número de linhas de uma memória de desintercalação e definindo uma largura *W* de fatia de dados como um número de coluna, os endereços apresentados na figura 85 podem ser gerados por um gerador de endereços. Os endereços podem incluir posições piloto. Para intercalar no tempo os símbolos de entrada que incluem apenas símbolos de dados, pode ser necessária uma lógica de controlo que

ignora os endereços. Os endereços utilizados nos preâmbulos de intercalação podem não exigir posições piloto e a intercalação pode ser realizada utilizando blocos $L1$. O i representa um índice de um símbolo de entrada, $N=D*W$ representa um comprimento de bloco de interligação. R_i e C_i representa um endereço de linha e um endereço de coluna de um i -ésimo de símbolo de entrada, respectivamente. T_w representa um valor de retorção de coluna ou parâmetro de retorção de um endereço onde os símbolos se encontram localizados. L_i representa endereços quando uma memória dimensional com uma única memória tampão é implementada. Valores de L_i podem ser de 0 a $(N-1)$. Nesta memória de uma só dimensão, são possíveis pelo menos dois processos. $L_i(1)$ encontra-se a acoplar uma matriz de memória linha a linha e $L_i(2)$ encontra-se a acoplar uma matriz de memória coluna a coluna. Um receptor pode utilizar a lógica de geração de endereços na leitura de símbolos durante uma desintercalação.

A figura 86 mostra um outro exemplo de um preâmbulo. Para um caso em que é utilizado um símbolo OFDM que apresenta um tamanho de 4K-FFT na largura de banda de 7,61MHz e uma sexta portadora dentro de um símbolo OFDM e portadoras em ambas as extremidades são utilizadas como pilotos, pode-se assumir que o número de portadoras que podem ser utilizadas na sinalização $L1$ sejam de 2840. Quando vários canais se encontram ligados, podem existir várias larguras de banda de preâmbulo. O número de portadoras pode mudar dependendo de um tipo de pilotos a

ser utilizado, um tamanho de FFT, um número de canais ligados, e outros factores. Se um tamanho de um *L1_XFEC_FRAME* que inclui *L1_Header* (H) que deve ser atribuído a um único símbolo *OFDM* e bloco *L1 FEC* (*L1_FEC1*) for menor do que um único símbolo *OFDM* ($5w-a-1$), *L1_XFEC_FRAME* incluindo *L1_Header* pode ser repetido para preencher uma parte restante do único símbolo *OFDM* ($5w-a-2$). Isto é semelhante à estrutura do preâmbulo da figura 60. Para que um receptor receba uma fatia de dados que se encontra localizada numa determinada largura de banda de canais ligados, uma janela de sintonizador do receptor pode estar localizada numa determinada largura de banda.

Se uma janela de sintonizador de um receptor se encontrar localizada como $5w-a-3$ da figura 86, pode ocorrer um resultado incorrecto durante a fusão de *L1_XFEC_FRAMES* repetidos. O caso 1 da figura 86 pode ser um tal exemplo. Um receptor encontra *L1_Header* (H) para localizar uma posição inicial de um *L1_Header* (H) dentro de uma janela de sintonizador, mas o *L1_Header* encontrado pode ser um cabeçalho de um *L1_XFEC_FRAME* ($5w-a-4$) incompleto. A informação de sinalização *L1* pode não ser obtida correctamente se for obtido um comprimento de *L1_XFEC_FRAME* com base naquele *L1_Header* e o resto da parte ($5w-a-5$) for adicionado a uma posição inicial daquela da *L1_Header*. Para evitar um caso do género, um receptor pode precisar de operações adicionais para encontrar um cabeçalho de um *L1_XFEC_FRAME* completo. A figura 87 mostra essas operações. No exemplo, para encontrar um cabeçalho de um *L1_XFEC_FRAME*

completo, se existir um *L1_XFEC_FRAME* incompleto num preâmbulo, um receptor pode utilizar pelo menos dois *L1-Headers* para encontrar um local de início de *L1_Header* para fusão de *L1_XFEC_FRAME*. Em primeiro lugar, um receptor pode encontrar *L1_Header* de um símbolo *OFDM* de preâmbulo (5w-b-1). Depois utilizando um comprimento de um *L1_XFEC_FRAME* dentro do *L1_Header* encontrado, o receptor pode verificar se cada *L1_XFEC_FRAME* dentro de um símbolo *OFDM* actual é um bloco completo (5w-b-2). Se não for, o receptor pode encontrar outro *L1_Header* do símbolo de preâmbulo actual (5w-b-3). De uma distância calculada entre *L1_Header* encontrado novo e um *L1_Header* anterior, pode ser determinado se um certo *L1_XFEC_FRAME* é um bloco completo (5w-b-4). Depois pode ser utilizado um *L1_Header* de um *L1_XFEC_FRAME* completo como um ponto de exposição para fusão. Usando o ponto de exposição, *L1_XFEC_FRAME* pode ser integrado (5w-b-5). Utilizando estes processos, o caso 2 ou a fusão correcta apresentada na figura 86 podem ser esperados num receptor. Estes processos podem ser realizados no descodificador de cabeçalho *FEC r712-L1* no trajecto do sinal *L1* da figura 81.

A figura 88 é um exemplo de uma estrutura de preâmbulo que pode eliminar as operações adicionais acima descritas num receptor. Em oposição à estrutura de preâmbulo anterior, quando uma parte restante de um símbolo *OFDM* é preenchido, apenas *L1_FEC1* de um *L1_XFEC_FRAME*, excluindo *L1_Header* (H) pode ser repetidamente cheio (5w-c-2). Deste modo, quando um receptor encontra uma posição de

início de um *L1_Header* (H) para fundir *L1_XFEC_FRAME*, só pode ser encontrado *L1_Header* de somente *L1_XFEC_FRAME* (5w-c-4), deste modo, sem operações adicionais, *L1_XFEC_FRAME* pode ser fundido utilizando o *L1_Header* encontrado. Por isso, os processos tais como (5W-b-2), (5w-b-3), e (5w-b-4) apresentados na figura 87 podem ser eliminados num receptor. Estes processos e processos da parte contrária dos processos podem ser executados no decodificador de cabeçalho *FEC 712-L1* no trajecto do sinal *L1* de um receptor da figura 81 e no cabeçalho *FEC 705 L1-L1* no trajecto do sinal *L1* de um transmissor da figura 80.

O desintercalador no tempo *r712-L1* no trajecto *L1* de um receptor da figura 81 pode desintercalas células de bloco *L1* ou células com padrões, excluindo outras células tais como cabeçalho de preâmbulo e células piloto. As células de bloco *L1* são representadas por células com padrões tal como apresentado na figura 83. A figura 90 mostra um outro exemplo de um transmissor *OFDM* que utiliza fatias de dados. Este transmissor pode ter estrutura idêntica e pode realizar função idêntica ao transmissor da figura 80, excepto os blocos adicionados e modificados. O mapeador de preâmbulo *1007-L1* pode mapear blocos *L1* e cabeçalhos de bloco *L1* que são saídas de cabeçalho de *FEC 705-L1* em símbolos de preâmbulo utilizados num quadro de transmissão. Especificamente, o cabeçalho de bloco *L1* pode ser repetido para cada preâmbulo e o bloco *L1* pode ser dividido tanto quanto um número de preâmbulos utilizados. O intercalador no tempo *1008-L1* pode intercalar blocos *L1* que

são divididos em preâmbulos. Neste ponto, o cabeçalho de bloco *L1* pode ser incluído na intercalação ou não incluído na intercalação. Se o cabeçalho de bloco *L1* se encontra incluído ou não, pode não alterar mudar uma estrutura de sinal de um cabeçalho de bloco *L1*, mas pode alterar uma ordem de intercalação e transmissão de blocos *L1*. O repetidor *L1_XFEC* 1015-*L1* pode repetir os blocos *L1_XFEC* intercalados no tempo dentro de uma largura de banda de preâmbulo. Neste ponto, o cabeçalho de bloco *L1* pode ser repetido dentro de um preâmbulo ou não repetido dentro de um preâmbulo.

A figura 91 mostra um outro exemplo de um receptor *OFDM* que utiliza fatias de dados. Este receptor apresenta estrutura idêntica e pode realizar função idêntica ao receptor da figura 81, excepto os blocos adicionados e modificados. O módulo de descodificação de cabeçalho *FEC* r1012-*L1* pode sincronizar cabeçalhos *L1* dentro de um preâmbulo. Se os cabeçalhos *L1* forem repetidos, os cabeçalhos *L1* podem ser combinados para obter um ganho *SNR*. Então, o módulo de descodificação de cabeçalho *FEC* r712-*L1* da figura 81 pode executar uma descodificação *FEC*. O processo de sincronização pode dar uma localização de um cabeçalho correlacionando a palavra de sincronização de um cabeçalho e preâmbulos. Para deslocamentos de frequência de múltiplos de um inteiro, pode ser determinada uma gama de correlação a partir de endereçamento circular.

O módulo de combinação *L1_XFEC* r1017-*L1* pode combinar blocos *L1_XFEC* para obter um ganho *SRN*, quando blocos *L1* divididos são recebidos dentro de um preâmbulo. O desintercalador no tempo r1010-*L1* pode desintercaladar no tempo blocos *L1* dentro de um preâmbulo. Dependendo se cabeçalhos de bloco *L1* são intercalados no tempo num transmissor ou não, os cabeçalhos de bloco *L1* podem ser desintercalados de modo correspondente num receptor. Uma ordem de desintercalação dos blocos *L1* pode ser alterada dependendo se os cabeçalhos de bloco *L1* são intercalados no tempo num transmissor ou não. Por exemplo, quando a intercalação no tempo se encontra ligada (*ON*) tal como na figura 83, uma localização da célula número 33 que é uma primeira célula de bloco *L1* dentro de um primeiro preâmbulo, pode mudar. Por outras palavras, quando os cabeçalhos de bloco *L1* não são incluídos numa intercalação, o sinal intercalado que tem as localizações das células tal como apresentado na figura 83 será recebido. Se os cabeçalhos de bloco *L1* forem incluídos numa intercalação, uma localização da célula número 33 precisa de ser alterada para desintercaladar células que se encontram intercalados na diagonal, utilizando uma primeira célula de um primeiro cabeçalho de bloco *L1* dentro de um primeiro preâmbulo considerando como referência. O misturador *L1_FEC* r1018-*L1* pode misturar blocos *L1* que se encontram divididos em muitos preâmbulos num único bloco *L1* para descodificação *FEC*.

Com um 1 bit adicional, o campo *PLP_type L1* de

campos de sinalização *L1* que são transmitidos num preâmbulo podem ter os seguintes valores.

PLP_type = 00 (*PLP* comum)

PLP_type = 01 (*PLP* de dados normais)

PLP_type = 10 (*PLP* de dados desmultiplexados)

PLP_type = 11 (reservado)

Dados normais *PLP* representam dados *PLP* quando um único serviço é transmitido numa única fatia de dados. Dados *PLP* desmultiplexados representam um *PLP* de dados quando um único serviço é desmultiplexado em várias fatias de dados. Quando um utilizador muda de serviço, se a sinalização *L1* e sinalização *L2* estiverem armazenados num receptor, pode ser eliminada a espera de uma informação de sinalização *L1* dentro de um próximo quadro. Por isso, um receptor pode alterar os serviços de forma eficiente e um utilizador pode beneficiar de menos atraso durante uma troca de serviço. A figura 95 mostra as estruturas do sinal do bloco *L1* que é transmitido num preâmbulo, para o fluxo de intercalação no tempo e fluxo de desintercalação. Tal como pode ser visto na figura 95, a intercalação e desintercalação pode ser realizada não numa largura de banda completa de preâmbulo, mas num bloco *L1* dividido.

A figura 96 é um exemplo de um campo de intercalação no tempo *L1* de campos de sinalização *L1*, processado pelo cabeçalho *FEC 705-L1* no trajecto *L1* apresentado na figura 90. Tal como apresentado na figura 96, um bit ou dois bits podem ser utilizados para o

parâmetro de intercalação no tempo. Se for utilizado um bit, a intercalação não é executada quando o valor de bit é 0, sendo que a intercalação tendo uma profundidade de símbolos *OFDM* utilizados nos símbolos do preâmbulo pode ser realizada quando o valor do bit é 1. Se forem utilizados dois bits, a intercalação com uma profundidade de intercalação de 0 ou não intercalação é executada quando o valor de bit é 00, podendo a intercalação tendo uma profundidade de símbolos *OFDM* utilizados em símbolos de preâmbulo ser executada quando o valor do bit é 01. A intercalação tendo uma profundidade de quatro símbolos *OFDM* pode ser executada quando o valor do bit é 10. A intercalação que apresenta uma profundidade de oito símbolos *OFDM* pode ser executada quando o valor de bits é 11.

Um receptor, especificamente, decodificador de cabeçalho *FEC r1012-L1* no trajecto *L1* apresentado na figura 91 pode extrair parâmetros de intercalação no tempo (*time interleaving - TI*) apresentados na figura 96. Utilizando os parâmetros, o desintercalador no tempo *r1010-L1* pode realizar a desintercalação de acordo com a profundidade de intercalação. Os parâmetros que são transmitidos no cabeçalho *L1* são o tamanho da informação *L1* (15bits), o parâmetro da intercalação no tempo (máximo de 2 bits), e *CRC* (max. 2 bits). Se for utilizado o código *Reed-Muller RM* (16, 32) para codificação do campo de sinalização do cabeçalho *L1*, porque os bits que podem ser transmitidos são 16 bits, não existe um número suficiente de bits. A figura

97 mostra um exemplo de campo de sinalização *L1* que pode ser utilizado para um tal caso.

A figura 97 mostra um processamento realizado no cabeçalho *FEC 705-L1* no trajecto *L1* da figura 90. Na figura 97a, *L1*() na coluna dos campos de sinalização representa o tamanho *L1* e *TI*() representa o tamanho para os parâmetros da intercalação no tempo. Para o primeiro caso ou quando o tamanho *L1* (15 bits) e *TI* (1 bit) são transmitidos, o preenchimento adicional pode não ser necessário, podendo ser obtido um desempenho de descodificação substancial do cabeçalho *L1*, sendo que porque é transmitida a informação se se deseja realizar uma intercalação no tempo ou não, para um bloco *L1* curto não pode ser obtido o efeito de intercalação.

Para o segundo caso ou quando o tamanho *L1* é reduzido a 1/8 do tamanho original, torna-se possível a transmissão de informação com números de bits como *L1* (12 bits), *TI* (2 bits) e *CRC* (2 bits). Deste modo, para o segundo caso, pode ser esperado um melhor desempenho da descodificação *L1* e efeito de intercalação no tempo. No entanto, o segundo caso requer processo de preenchimento adicional para fazer do tamanho *L1* um múltiplo de oito se o tamanho *L1* não for um múltiplo de oito. A figura 97b representa o processo de preenchimento que pode ser realizado no sinal *L1 700-L1* da Figura 90. Mostra que o preenchimento se encontra localizado após o bloco *L1* e coberto com codificação *CRC*. Consequentemente, num

receptor, o decodificador *FEC BCH/LDPC* r715-L1 no trajecto *L1* da figura 91 pode executar a descodificação *FEC*, porque se não existir erro quando o campo *CRC* é verificado, pode ser executada a análise sintáctica de bit de acordo com o campo de sinalização *L1*, sendo então necessário um processo que define o resto dos bits como preenchimento ou *CRC32* e excluindo o resto de bits a partir de parâmetros.

Para o terceiro caso ou quando o tamanho *L1* é expresso como um número de células mapeadas *QAM*, não um número de bits, o número de bits pode ser reduzido. Para o quarto caso, o tamanho *L1* é expresso não como um tamanho de um bloco *L1* completo, mas como um tamanho *L1* por cada símbolo *OFDM*. Assim, para que um receptor obtenha um tamanho de um bloco *L1* completo é necessário multiplicar o tamanho do bloco *L1* num símbolo *OFDM* por uma série de símbolos *OFDM* utilizados no preâmbulo. Neste caso, o tamanho *L1* actual tem que excluir o preenchimento.

Para o quinto caso, ao expressar o bloco *L1* não como um número de bits mas como um número de células mapeadas *QAM*, é possível maior redução de bits. Para o terceiro até ao quinto caso são apresentados os parâmetros *TI*, *CRC*, e um número de bits de preenchimento necessários. Para um caso em que o tamanho do bloco *L1* é expresso como um número de células, para que um receptor obtenha o tamanho *L1* em bits, o receptor precisa de multiplicar um número de bits onde apenas as células são transmitidas por um tamanho *L1* recebido. Adicionalmente, tem que ser

excluído um número de bits de preenchimento.

O último caso mostra um número total crescente de bits para 32 bits utilizando dois blocos de código *RM* no cabeçalho. Campos *CRC* totais transformam-se em quatro bits porque cada bloco de código *RM* precisa de dois bits de campo *CRC*. Um receptor ou decodificador de cabeçalho *FEC* r1012-*L1* no trajecto *L1* da figura 91, precisa de obter os parâmetros necessários ao executar a decodificação *FEC* num total de dois blocos *FEC*. Utilizando os parâmetros obtidos, um receptor, especificamente desintercalador no tempo r1010-*L1* no trajecto *L1* da figura 91, pode determinar se é para realizar a desintercalação ou não, e pode obter uma profundidade de desintercalação, se for determinado que deve ser executada a desintercalação. Além disso, o decodificar *FEC BCH/LDPC* r715-*L1* pode obter o comprimento de bloco *LDPC* necessário para realizar a decodificação *FEC* e parâmetros de encurtamento/puncionagem. Os campos de preenchimento desnecessários, exigidos para enviar o sinal *L1* para um controlador de sistema, podem ser removidos.

A figura 92 mostra um exemplo de um intercalador no tempo (*TI*) de fatias de dados. O processo de *TI* assume que todas as posições-piloto são conhecidas. O *TI* pode produzir apenas células de dados, excluindo os pilotos. Saber as posições piloto habilita o número correcto de células de saída para cada símbolo *OFDM*. Além disso, o *TI* pode ser implementado por uma única memória tampão num receptor.

A figura 93 mostra um exemplo de uma implementação eficiente da intercalação no tempo num receptor. A figura 93a mostra quatro esquemas diferentes de desintercalação de acordo com uma forma de realização da presente invenção. A figura 93b mostra uma única memória tampão que realiza a desintercalação. A figura 93c mostra um esquema exemplar para endereçar os blocos *L1* numa matriz 2D ou uma sequência 1D.

Tal como apresentado na figura 93a-c, utilizar um algoritmo de memória tampão individual pode ser uma implementação mais eficiente do intercalador no tempo. O algoritmo pode ser caracterizado primeiro pela leitura das células de saída da memória, e depois escrever células de entrada onde as células de saída são lidas. O endereçamento diagonal pode ser considerado como um endereçamento circular em cada coluna.

Mais especificamente, em relação à figura 93a, estes quatro processos de leitura e escrita aplicam-se sequencialmente aos quadros C2 que são recebidos num receptor. O primeiro quadro recebido num receptor é escrito na memória do desintercalador na figura 93b no caminho para o 0-ésimo bloco na figura 93a e lido no caminho para o 1º bloco. O segundo quadro recebido é escrito na memória do desintercalador na figura 93b no caminho para o 1º bloco e lido para o 2º bloco. O terceiro quadro recebido é escrito na memória do desintercalador na figura 93b no caminho para o 2º bloco e lido no caminho para o 3º bloco. O quarto

quadro recebido é escrito na memória do desintercalador na figura 93b no caminho para o 3º bloco e lido no caminho para o 0º bloco, etc. Quer dizer, os processos de escrita e leitura na figura 93a podem ser sequencialmente e ciclicamente aplicados aos quadros C2 que são recebidos sequencialmente.

O processo de intercalação no tempo (*TI*) pode ser executado em preâmbulos tal como apresentado na figura 94. As posições piloto são periódica e facilmente removidas, não sendo necessária qualquer intercalação para o cabeçalho de bloco *L1*. É porque o cabeçalho do preâmbulo carrega parâmetros *TI* e tanto a intercalação como a não-intercalação têm os mesmos resultados devido a repetição. Deste modo, somente as células de sinalização *L1* são intercaladas. Pode ser aplicada a única memória tampão utilizada na fatia de dados *TI*.

A figura 95 mostra o fluxo do preâmbulo da intercalação/desintercalação no tempo. A intercalação pode ser realizada dentro de um bloco *L1*, em vez de um preâmbulo completo. Num transmissor, tal como apresentado na figura 128a, o bloco *L1* pode ser codificado - então pode ser realizada uma intercalação dentro do bloco *L1* -, e o bloco *L1* intercalado pode ser repetido dentro de um preâmbulo. Num receptor, tal como apresentado na figura 128b, de um preâmbulo recebido -, o bloco *L1* pode ser combinado ou sincronizado, podendo ser obtido um único período de bloco *L1* -, podendo o bloco *L1* combinado ser desintercalado -.

A figura 96 mostra parâmetros de profundidade de intercalação no tempo em sinalização de cabeçalho *L1*. Para a estrutura de cabeçalho *L1*, *RM* (16, 32) tem uma capacidade de 16 bits. Um máximo de 2 bits de *CRC* pode melhorar o desempenho *RM BER*. Os campos de sinalização necessários do cabeçalho *L1* são *L1_info_size* (15 bits) que podem exigir um máximo de 5 símbolos *OFDM* e *TI_depth* (2 bits ou 1 bit). No entanto, um total de 18 ou 19 bits excede a capacidade do cabeçalho *L1*.

A figura 97 mostra um exemplo de sinalização *L1* de cabeçalho e uma estrutura e um processo de zona de preenchimento.

A figura 98 mostra um exemplo de uma sinalização *L1* transmitida num cabeçalho de quadro. A informação de sinalização *L1* pode ser utilizada como parâmetros de descodificação num receptor. Em especial, os módulos no trajecto de sinal *L1* da figura 91 pode realizar a descodificação de sinalização *L1* e os módulos no trajecto *PLP* da figura 91 podem utilizar parâmetros, sendo que assim os serviços podem ser descodificados. Um receptor pode obter parâmetros da sinalização *L1* a partir de sinais do trajecto *L1* que são descodificados de acordo com uma ordem de cada campo e comprimento de campo. O seguinte explica o significado de cada campo e a sua utilização. Um nome de cada campo, um número de bits para cada campo, ou um exemplo de cada campo podem ser modificados.

Num_chbon: Este campo indica um número de canais utilizados numa ligação de canal. Utilizando este campo, um receptor pode obter uma largura de banda total de canais utilizados. O canal pode ter 6MHz, 7MHz, 8MHz, ou outros valores de largura de banda.

Num_dslice: Este campo indica um número de fatias de dados existentes num canal ligado. Depois da descodificação da sinalização *L1*, um receptor acessa a um circuito fechado onde a informação das fatias de dados se encontra contida, para obter a informação da fatia de dados. Utilizando este campo, um receptor pode obter um tamanho do circuito fechado para descodificação.

Num_notch: Este campo indica um número de bandas de entalhe existentes num canal ligado. Depois da descodificação da sinalização *L1*, um receptor acessa a um circuito fechado onde a informação da banda de entalhe se encontra contida, para obter a informação da banda de entalhe. Utilizando este campo, um receptor pode obter um tamanho do circuito fechado para descodificação.

Para cada fatia de dados, podem ser transmitidos *dslice_id*, *dslice_start*, *dslice_width*, *dslice_TI_depth*, *dslice_type*, *dslice_pwr_allocation* e informações *PLP* num preâmbulo de um cabeçalho de quadro. A fatia de dados pode ser considerada como uma largura de banda específica que contém um ou mais *PLPs*. Os serviços podem ser transmitidos nos *PLPs*. Um receptor precisa de aceder às fatias de dados

que contêm um *PLP* específico, para descodificar um serviço.

Dslice_id: Este campo pode ser utilizado para identificação de fatia de dados. Cada fatia de dados num canal ligado pode ter um valor único. Quando um receptor acessa uma das *PLPs* para descodificar serviços, este campo pode ser utilizado para o receptor diferenciar uma fatia de dados onde a *PLP* se encontra localizada, de outras fatias de dados.

Dslice_start: Este campo indica um local de início de uma fatia de dados dentro de um canal ligado. Utilizando este campo, um receptor pode obter uma frequência onde a fatia de dados começa. Adicionalmente, a sintonização para aceder a uma fatia de dados pode ser executada utilizando este campo.

Dslice_width: Este campo indica uma largura de banda de uma fatia de dados. Utilizando este campo, um receptor pode obter um tamanho de uma fatia de dados. Em especial, este campo pode ser utilizado na desintercalação no tempo para possibilitar a descodificação. Juntamente com o campo *dslice_start*, um receptor pode determinar que frequência descodificar dos sinais *RF* recebidos. Este processo pode ser realizado no sintonizador r700 da figura 91. As informações tais como *dslice_start* e *dslice_width* podem ser utilizadas como sinal de controlo do sintonizador (r700).

Dslice_ti_depth: Este campo indica uma profundidade de intercalador no tempo utilizada na intercalação no tempo de fatias de dados. Juntamente com *dslice_width*, um receptor pode obter uma largura e uma profundidade de um desintercalador no tempo e pode realizar a desintercalação no tempo. A figura 99 mostra um exemplo de um *Dslice_TI_depth*. No exemplo, 1, 4, 8 ou 16 de símbolos *OFDM* são utilizados na intercalação no tempo. Esta é executada no desintercalador no tempo r710 da figura 91. *Dslice_width* e *Dslice_TI_depth* podem ser utilizados como sinal de controlo.

Dslice_type: Este campo indica um tipo de fatia de dados. Fatia de dados do tipo 1 tem um único *PLP* dentro do mesmo e o *PLP* é uma *CCM* (*constant coding and modulation* - codificação e modulação constantes) aplicada. A fatia de dados do tipo 2 representa todos os outros géneros de fatias de dados. Utilizando este campo, um receptor pode realizar a descodificação de acordo com *PLP*. A *PLP* do tipo 1 não tem cabeçalho *FECFRAME*, sendo que assim um receptor não procura pelo cabeçalho *FECFRAME*. Para o tipo 2, um receptor procura pelo cabeçalho *FECFRAME* da *PLP* para obter informação *MODCOD*. A figura 100 mostra um exemplo de *dslice_type*. Utilizando este campo, o analisador sintáctico de fatias de dados r711 da figura 91 pode controlar o descodificador do cabeçalho *FEC* r712-c, k.

Dslice_pwr_allocation: Este campo indica uma potência de uma fatia de dados. Cada fatia de dados pode

ter uma potência diferente de outras fatias de dados. É para adaptação da ligação no sistema de cabo. Um receptor pode utilizar este campo para controlar a potência da fatia de dados recebida. O sintonizador r700 da figura 91 pode ajustar o ganho do sinal utilizando este campo.

Num_plp: Este campo indica um número de *PLPs* numa fatia de dados. Após a descodificação da sinalização *L1*, um receptor acessa a um circuito fechado que inclui informação *PLP*. Utilizando este campo, um receptor pode obter um tamanho do circuito fechado e descodificar *PLPs*.

Para cada *PLP*, *plp_id*, *plp_type*, reproprocessamento *PSI/SI*, *plp_payload_type*, *plp_ModCod* e *plp_start_addr* podem ser transmitidos num cabeçalho de quadro (preâmbulo). Cada *PLP* pode transmitir um ou mais fluxos ou pacotes tais como *TS* e *GSE*. Um receptor pode obter serviços descodificando *PLPs* onde os serviços são transmitidos.

Plp_id: Este campo é um identificador de *PLP* e tem um valor único para cada *PLP* num canal ligado. Utilizando este campo, um receptor pode aceder à *PLP* onde existe um serviço para descodificar. Este campo pode servir a um propósito idêntico à *plp_id* transmitida num cabeçalho *FECFRAME*. O descodificador de cabeçalho *FEC* r712-c, k da figura 91 pode aceder à *PLP* necessária utilizando este campo.

Plp_type: Este campo indica se um tipo de *PLP* é

uma *PLP* comum ou uma *PLP* de dados. Utilizando este campo, um receptor pode encontrar *PLP* comum e pode obter informação necessária para descodificar um pacote *TS* da *PLP* comum. Além disso, o receptor pode descodificar um pacote *TS* dentro de uma *PLP* de dados. A figura 101 mostra um exemplo de *PLP_type*.

Reprocessamento *PSI/SI*: Este campo indica se um *PSI/SI* de um sinal recebido é reprocessado ou não. Utilizando este campo, um receptor pode determinar referir *PSI/SI* de um serviço específico de um serviço transmitido. Se um receptor não pode referir *PSI/SI* de um serviço específico de um serviço transmitido, *PSI/SI* que pode ser referido por um serviço específico pode ser transmitido através *PLP* comum, por exemplo. Utilizando esta informação, um receptor pode descodificar serviços. Além disso, este campo pode ser um campo de 1 bit. Este campo pode ser utilizado por um receptor para reconhecer se pode contar com as partes *PSI/SI* relacionadas. Quando o reprocessamento *PSI/SI* é realizado, este campo deve ser configurado para '1', caso contrário ele deve ser configurado para '0'.

***Plp_payload_type*:** Este campo indica o tipo de dados de carga útil que a *PLP* transmite. Um receptor pode utilizar este campo antes de descodificar dados dentro de *PLPs*. Se um receptor não pode descodificar um tipo específico de dados, pode ser impedida a descodificação de uma *PLP* que contém este tipo específico de dados. A figura 102 mostra um exemplo de *PLP_payload_type*. Se uma fatia de

dados tem uma única *PLP* e uma *CCM* é aplicada à fatia de dados fatia, isto é, fatia de dados tipo 1, os campos tais como *PLP_ModCod* e *PLP_start_addr* podem ser adicionalmente transmitidos.

Plp_ModCod: Este campo indica um tipo de modulação e taxa de código *FEC* utilizado na *PLP*. Utilizando este campo, um receptor pode realizar a desmodulação *QAM* e decodificação *FEC*. A figura 103 mostra um exemplo de *PLP_ModCod*. Os valores apresentados na figura podem ser utilizados em *ModCod* que é transmitido num cabeçalho de um *FECFRAME*. O desmapeador de símbolos r713-c, k e decodificador *FEC BCH/LDPC* r715-c, k da figura 91 pode utilizar este campo para a decodificação.

Plp_start_addr: Este campo indica onde um primeiro *FECFRAME* de uma *PLP* aparece num quadro de transmissão. Utilizando este campo, um receptor pode obter uma localização inicial de *FECFRAME* e realizar a decodificação *FEC*. Utilizando este campo, o analisador sintáctico de fatias de dados r711 da figura 91 pode sincronizar *FECFRAMEs* para *PLPs* do tipo 1. Para cada banda de entalhe, as informações tais como *notch_start* e *notch_width* podem ser transmitidas num cabeçalho de quadro (preâmbulo).

Notch_start: Este campo indica um local de início de uma banda de entalhe. *Notch_width*: Este campo indica uma largura de uma banda de entalhe. Utilizando *notch_start*

e *notch_width*, um receptor pode obter um local e um tamanho de uma banda de entalhe dentro de um canal ligado. Além disso, pode ser obtido um local de sintonia para uma descodificação de serviço correcta e pode ser verificada uma existência de um serviço dentro de uma determinada largura de banda. O sintonizador r700 da figura 91 pode executar a sintonia utilizando esta informação.

GI: Este campo indica a informação do intervalo de guarda utilizado num sistema. Um receptor pode obter informação de intervalo de guarda utilizando este campo. O sincronizador de tempo/frequência r505 e removedor GI r704 da figura 91 pode utilizar este campo. A figura 104 mostra um exemplo.

Num_data_symbols: Este campo indica um número de símbolos de dados *OFDM*, excepto preâmbulo, utilizados num quadro. Um comprimento de quadro de transmissão pode ser definido por este campo. Utilizando este campo, um receptor pode prever a localização de um preâmbulo a seguir, assim, este campo pode ser utilizado para descodificar a sinalização *L1*. O analisador sintáctico de quadros r708 da figura 91 pode utilizar este campo e prever símbolos *OFDM* que são preâmbulo e enviar sinal para o trajecto de descodificação de preâmbulo.

Num_c2_frames: Este campo indica um número de quadros existentes num canal ligado. Utilizando este campo, um receptor pode obter um limite de um super quadro e pode

prever informação repetida por cada super quadro.

Frame_idx: Este campo é um índice de quadros e é reinicializado para cada super quadro. Utilizando este campo, um receptor pode obter um número actual de quadros e encontrar uma localização do quadro actual dentro de um super quadro. Utilizando este campo, o analisador sintáctico de quadros r708 da figura 91 pode descobrir quantos quadros estão à frente de um quadro actual num super quadro. Juntamente com *Num_c2_frames*, pode ser prevista a mudança que ocorre numa sinalização *L1* e a descodificação pode ser controlada.

PAPR: Este campo indica se é utilizada ou não uma reserva de tom para reduzir uma *PAPR*. Utilizando este campo, um receptor pode processar em conformidade. A figura 105 mostra um exemplo. Por exemplo, se for utilizada uma reserva de tom, um receptor pode excluir portadoras utilizadas numa reserva de tom, a partir da descodificação. Especificamente, o analisador sintáctico de fatias de dados r711 da figura 91 pode utilizar este campo para excluir portadoras da descodificação.

Reservado: Este campo é para bits adicionais reservados para utilização futura.

A figura 106 mostra um outro exemplo de sinalização *L1* transmitida num cabeçalho de quadro.

Na figura 106, a informação acrescentada

adicionalmente à figura 98 pode tornar a descodificação de serviço por um receptor mais eficiente. Os campos a seguir explicam apenas a informação adicional. Os outros campos são os mesmos que a figura 98.

Network_id: Este campo indica uma rede onde o sinal transmitido pertence. Utilizando este campo, um receptor pode descobrir uma rede actual. Quando um receptor sintoniza para outra rede para encontrar um serviço na rede, o receptor pode processar mais rápido porque utilizando apenas a descodificação *L1* é o suficiente para tomar a decisão se a rede sintonizada é uma rede desejada ou não.

C2_system_id: Este campo identifica um sistema onde o sinal transmitido pertence. Utilizando este campo, um receptor pode descobrir o sistema actual. Quando um receptor sintoniza para outra rede para encontrar um serviço no sistema, o receptor pode processar mais rápido porque utilizar apenas a descodificação *L1* é o suficiente para tomar a decisão se a rede sintonizada é um sistema desejado ou não.

C2_signal_start_frequency: Este campo indica uma frequência inicial de canais ligados.
C2_signal_stop_frequency: Este campo indica uma frequência final de canais ligados. Utilizando *c2_signal_start_frequeny* e *c2_signal_stop_frequency*, as larguras de banda *RF* de todas as fatias de dados podem ser encontradas por descodificação de *L1* de determinada largura

de banda dentro de canais ligados. Além disso, este campo pode ser utilizado para obter um valor de deslocamento de frequência exigido na sincronização de *L1_XFEC_FRAMES*. *L1_XFEC Combiner r1017-L1* da figura 91 pode utilizar este campo. Além disso, quando um receptor recebe fatias de dados localizadas em ambas as extremidades de um canal ligado, este campo pode ser utilizado para sintonizar para uma frequência adequada. O sintonizador *r700* da figura 91 pode utilizar esta informação.

Plp_type: Este campo indica se um tipo de *PLP* é uma *PLP* comum, uma *PLP* de dados normal, ou *PLP* de dados agrupada. Utilizando este campo, um receptor pode identificar *PLP* comum e pode obter informação necessária para decodificar um pacote *TS* da *PLP* comum, depois pode decodificar o embalagem *TS* dentro de uma *PLP* de dados agrupada. Aqui, a *PLP* comum pode ser uma *PLP* que contém dados compartilhados por várias *PLPs*. A figura 107 mostra um exemplo deste campo. A *PLP* de dados normais é uma *PLP* de dados que não tem *PLP* comum. Neste caso, um receptor não precisa de encontrar uma *PLP* comum. A *PLP* comum ou *PLP* agrupada pode transmitir informação tal como *PLP_group_id*. Para os outros tipos de *PLP* é possível uma transmissão mais eficiente porque não é necessário transmitir nenhuma informação adicional.

Plp_group_id: Este campo indica um grupo onde uma *PLP* actual pertence. *PLP* de dados agrupados pode transmitir parâmetros *TS* comuns utilizando *PLP* comum. Utilizando este

campo, se uma *PLP* actualmente descodificada é uma *PLP* agrupada, um receptor pode encontrar uma *PLP* necessária comum, obter parâmetros necessários para pacote *TS* de *PLP* agrupada, e formar um pacote *TS* completo.

Reserved_1/reserved_2/reserved_3: Estes campos são bits adicionais reservados respectivamente para utilização futura para um circuito fechado de fatia de dados, um circuito fechado *PLP*, e um quadro de transmissão.

A figura 108 mostra um outro exemplo de sinalização *L1* transmitida num cabeçalho de quadro. Quando comparado com a figura 106, mais informações optimizadas podem ser transmitidas, assim, pode ocorrer menos sinalização de informação complementar. Correspondentemente, um receptor pode descodificar serviços de forma eficiente. Em especial, os módulos no trajecto de sinal *L1* da figura 91 podem realizar a descodificação de sinalização *L1* e os módulos no trajecto *PLP* da figura 91 podem utilizar parâmetros, sendo que assim os serviços podem ser descodificados. Um receptor pode obter parâmetros da sinalização *L1* a partir de sinais do trajecto *L1* que são descodificados de acordo com uma ordem de cada campo e comprimento de campo. Um nome de cada campo, um número de bits para cada campo, ou um exemplo de cada campo podem ser modificados. As descrições dos campos, excepto *dslice_width* são idênticas às descrições acima descritas dos campos. Uma função de *dslice_width* de acordo com um exemplo é a seguinte.

Dslice_width: Este campo indica uma largura de banda de uma fatia de dados Utilizando este campo, um receptor pode obter um tamanho de uma fatia de dados. Em especial, este campo pode ser utilizado em desintercalação no tempo permitir a descodificação. Juntamente com o campo *dslice_start*, um receptor pode determinar que frequência descodificar dos sinais *RF* recebidos. Este processo pode ser realizado no sintonizador r700 da figura 91. As informações tais como *dslice_start* e *dslice_width* podem ser utilizadas como sinal de controlo do sintonizador r700. Neste ponto, a largura de uma fatia de dados pode-se estender até 64MHz usando 12 bits para este campo *dslice_width*. Usando este campo, um receptor pode determinar se um sintonizador actualmente disponível pode descodificar a fatia de dados actual. Se a largura de uma fatia de dados for maior do que uma largura de banda de um sintonizador herdado de um receptor, para descodificar uma tal uma fatia de dados, um receptor pode usar, pelo menos, dois sintonizadores herdados ou um sintonizador com uma largura de banda suficientemente grande. No exemplo, uma granularidade de valores utilizada no *dslice_start*, *dslice_width*, *notch_start* e *notch_width* pode ser 12 portadoras *OFDM* (células). Por outras palavras, um receptor pode encontrar um local de uma célula *OFDM* actual multiplicando os valores transmitidos por 12. No exemplo, para uma granularidade de *PLP_start_addr*, pode ser utilizada uma portadora *OFDM* (célula). Por outras palavras, um receptor pode descobrir quantos símbolos *OFDM* e células *OFDM* estão à frente de um local de início de uma *PLP* dentro

de um símbolo *OFDM*. *Dslice_start* e *dslice_width* podem ser usadas para esta finalidade. O analisador sintáctico de fatias de dados *r711* da figura 91 pode realizar um tal processo.

A figura 109 mostra um exemplo de processamento no cabeçalho *FEC 705-L1* no trajecto *L1* da figura 90. Um total de 16 bits podem ser transmitidos no cabeçalho *FEC* de um trajecto *L1*. Catorze bits podem ser atribuídos para *L1_info_size*. Se *L1_info_size* tiver um valor que é metade do comprimento do bloco *L1* transmitido actualmente, um receptor pode multiplicar *L1_info_size* recebido por dois e obter o comprimento actual do bloco *L1* e começar a descodificar *L1*. Este comprimento obtido do bloco *L1* é um comprimento que inclui zona de preenchimento.

Para bloco *L1* que se encontra determinado em não apresentar qualquer erro apesar da verificação *CRC*, um receptor pode considerar o resto de bits após a descodificação *L1* como zona de preenchimento. Os dois últimos bits, semelhantes tal como nos processos anteriores, podem ser utilizados para indicar a profundidade da intercalação no tempo de preâmbulos. O mapeador de preâmbulo *1007-L1* da figura 90 pode determinar os símbolos *OFDM* necessários para transmitir blocos *L1*. Após isto, o intercalador *1008-L1* da figura 90 pode realizar a intercalação no tempo. Usando a informação de profundidade de intercalação no tempo e *L1_info_size*, um receptor pode descobrir que tamanho do bloco *L1* é

transmitido em quantos símbolos *OFDM*. Combinar, fundir, e desintercalar no tempo blocos *L1* pode ser realizado no combinador *L1 XFEC* 12417-*L1*, unificador *L1_FEC* 12418-*L1*, e desintercalador no tempo 12410-*L1* da figura 91, respectivamente.

Num receptor na figura 91, um comprimento de um bloco *L1 XFEC* dentro de um símbolo *OFDM* pode ser obtido dividindo um comprimento total de bloco *L1* por um número de símbolos *OFDM* utilizado num preâmbulo. O número de símbolos *OFDM* pode ser obtido a partir de um valor definido em *TI_depth*. O combinador *L1 XFEC* 12417-*L1* de um receptor pode obter o bloco *L1 XFEC*. Então, a desintercalação no tempo 12410-*L1* pode ser realizada utilizando *TI_depth*. Finalmente, os blocos *L1 XFEC* podem ser fundidos para obter um bloco *L1_FEC*. Após a fusão *L1_FEC* 12418-*L1*, desintercalação de bits *r714-L1*, e descodificação *LDPC/BCH* *r715-L1*, pode ser obtido o bloco *L1*. *L1_info_size* pode ser multiplicado por dois, o bloco *L1* pode ser verificado com *CRC*, e *L1* pode ser descodificado. Zona de preenchimento desnecessário pode ser desconsiderado.

A figura 110 mostra um outro exemplo de sinalização *L1* transmitida num cabeçalho de quadro. Quando comparado com a figura 108, o número de bits para alguns campos encontram-se modificados e alguns campos encontram-se adicionados para melhorar a eficiência da descodificação de serviço por um receptor. Em especial, os módulos no trajecto de sinal *L1* da figura 91 podem realizar a

descodificação de sinalização *L1* e os módulos no trajecto *PLP* da figura 91 podem utilizar parâmetros, sendo que assim os serviços podem ser descodificados. Um receptor pode obter parâmetros da sinalização *L1* a partir de sinais do trajecto *L1* que são descodificados de acordo com uma ordem de cada campo e comprimento de campo. Um nome de cada campo, um número de bits para cada campo, ou um exemplo de cada campo podem ser modificados. Excepto os campos modificados da figura anterior, as descrições dos campos são idênticas às descrições acima descritas dos campos. *RESERVED_1*, *RESERVED_2*, *RESERVED_3* e *RESERVED_4* são campos reservados para uso futuro. No exemplo, *PLP_START* pode indicar informações idênticas com *PLP_start_addr* acima mencionados.

L1_PART2_CHANGE_COUNTER indica um número de quadros a partir de primeiro quadro para um quadro que apresenta uma alteração em qualquer das informações de sinalização *L1*, excluindo a mudança em *PLP_START*, de quadros anteriores. Quer dizer, este campo indica o número de quadros à frente onde a configuração mudará. Usando este campo, um receptor pode pular a descodificação *L1* para cada quadro para obter informação *L1*. Por outras palavras, ao utilizar o valor de *L1_PART2_CHANGE_COUNTER*, um receptor pode determinar qual o quadro é que tem uma mudança na informação *L1* dos quadros anteriores, portanto, nenhuma descodificação *L1* é executada para quadros antes que tenha lugar um quadro com mudança em *L1*, pois a descodificação *L1* pode ser executada para o quadro que apresenta alteração em

L1. Deste modo pode ser puladas operações desnecessárias. Usando este campo, um receptor pode evitar a operação de descodificação *L1* redundante. Este valor pode também ser calculado por um receptor com informação *L1* já descodificada.

Se *L1_PART2_CHANGE_COUNTER* for 0, isso significa que houve uma mudança em *L1* em pelo menos 256 (2^8 , 8 é um número de bits utilizados para *L1_PART2_CHANGE_COUNTER*) quadros. Neste um dos melhores casos, um receptor precisa descodificar *L1* apenas a cada 51 segundos. Este processo pode ser realizado no analisador sintáctico de quadros r708 da figura 91. O analisador sintáctico de quadros pode determinar se preâmbulo actual tem uma mudança em *L1* e pode controlar os processos subsequentes no trajecto do sinal *L1*. Um receptor pode calcular *PLP_START* para quadro específico de *PLP_START* já obtido e *PLP_MODCOD*, sem realizar a descodificação *L1* para obter *PLP_START*.

A figura 111 mostra exemplos de campos apresentados na figura 110. Os blocos de um receptor podem executar processos de acordo com os valores indicados pelos campos nos exemplos.

A figura 112 mostra um outro exemplo de sinalização *L1* transmitida num cabeçalho de quadro. Quando comparado com a figura 110, alguns campos encontram-se modificados e alguns campos encontram-se adicionados para melhorar a eficiência da descodificação de serviço por um

receptor. Em especial, os módulos no trajecto de sinal *L1* da figura 91 podem realizar a descodificação de sinalização *L1* e os módulos no trajecto *PLP* da figura 91 podem utilizar parâmetros, sendo que assim os serviços podem ser descodificados. Um receptor pode obter parâmetros da sinalização *L1* a partir de sinais do trajecto *L1* que são descodificados de acordo com uma ordem de cada campo e comprimento de campo. Um nome de cada campo, um número de bits para cada campo, ou um exemplo de cada campo podem ser modificados. Excepto os campos modificados da figura anterior, as descrições dos campos são idênticas às descrições acima descritas dos campos.

As descrições de *DSLICE_START*, *DSLICE_WIDTH*, *NOTCH_START* e *NOTCH_WIDTH* são idênticas às descrições anteriores. Contudo, a informação complementar de sinalização pode ser minimizada sinalizando os campos com um número mínimo de bits de acordo com o modo GI. Assim, pode-se dizer que a sinalização de *DSLICE_START*, *DSLICE_WIDTH*, *NOTCH_START* e *NOTCH_WIDTH* é baseada no modo GI. A informação *L1* pode ser obtida no trajecto do sinal *L1* de um receptor da figura 91. Um controlador de sistema pode determinar um número de bits utilizados para cada campo de acordo com o valor GI obtido e pode ler os campos de modo correspondente. O valor GI precisa de ser transmitido antes de outros valores.

Em vez de *DSLIC_START* e *DSLICE_WIDTH*, podem ser transmitidos 12 bits de posição de sintonização que indica

uma localização otimizada para obter fatia de dados e 11 bits de valor de desvio de uma posição de sintonização para indicar uma largura de uma fatia de dados. Em especial, ao utilizar 11 bits de valor de desvio, as fatias de dados que ocupam um máximo de 8 canais ligados podem ser sinalizadas e um receptor que pode receber tais fatias de dados pode operar de forma adequada. Um sintonizador r700 de um receptor da figura 91 pode determinar a largura de banda de *RF* usando uma posição de sintonização e pode obter uma largura de uma fatia de dados usando o valor de desvio, para servir o mesmo propósito do *DSLICE_WIDTH* acima mencionado.

DSLICE_CONST_FLAG é um campo para indicar se uma configuração de uma fatia específica de dados é mantida como uma constante. Utilizando este campo obtido a partir de uma *L1* de uma determinada largura de banda, um receptor pode determinar se uma fatia de dados específica tem uma configuração constante, pois o receptor pode receber *PLPs* da fatia de dados específica, sem descodificação *L1* adicional. Esse tipo de processo pode ser útil para a recepção de fatia de dados que se encontra localizado numa largura de banda onde a descodificação *L1* não se encontra disponível.

DSLICE_NOTCH_FLAG é um campo ou uma marca para indicar bandas de entalhe em ambas os bordos de uma fatia de dados específica. O bit mais significativo (*MSB*) pode ser utilizado como um indicador da proximidade da banda de

entalhe a uma largura de banda baixa, sendo que o bit menos significativo (*LSB*) pode ser utilizado como um indicador para a proximidade da banda de entalhe a uma largura de banda elevada. Usando o campo, quando um receptor descodifica uma fatia de dados específica, o receptor pode ter em conta uma banda de entalhe descobrindo mudanças em portadoras activas provocadas por pilotos contínuos afins em ambas as extremidades de uma banda de entalhe. Esta informação também pode ser obtida a partir de informação de fenda transmitida em notch *NOTCH_START* e *NOTCH_WIDTH*. O desintercaldor no tempo *r710* de um receptor da figura 91 pode usar a informação para encontrar a localização de portadoras activas e enviar apenas dados que correspondem às portadoras activas, para um analisador sintáctico de fatia de dados.

Para *PLP_TYPE*, é adicionado um bit adicional à figura 110. A figura 113 mostra um exemplo de *PLP_TYPE* da figura 112. Pode ser transmitido um valor que indica dados integrados *PLP*. Um fluxo *TS* grande que possui uma alta taxa de dados pode ser multiplexado em várias *PLPs*. Dados *PLP* integrados podem ser utilizados para indicar *PLPs* onde fluxos multiplexados são transmitidos. Para um receptor herdado que não é passível de descodificar uma *PLP* específica, este campo pode evitar que o receptor acesse a *PLP*, podendo assim ser impedido um possível mau funcionamento.

Ainda como um processo alternativo, se o

dslice_width acima descrito for utilizado juntamente com o campo *dslice_start* e informação de fenda, um receptor pode determinar que frequência descodificar dos sinais *RF* recebidos. Este processo pode ser realizado no sintonizador (r700) da figura 91. As informações tais como *dslice_start*, *dslice_width*, *notch_start*, e *notch_width* podem ser utilizadas como sinal de controlo do sintonizador r700. Assim, a obtenção de uma fatia de dados e simultaneamente a sintonização para uma banda de *RF* onde não existem quaisquer problemas de descodificação *L1* pode-se tornar possível, evitando o entalhe.

No que se refere à sinalização *L1* da figura 112, figura 114 mostra uma relação entre sinalização *L1* e sinalização *L2* quando uma *PLP* é do tipo integrado. Além disso, a figura 114 mostra também uma acção que pode ser tomada por um receptor para um tal caso. O TS1 pode ser mapeado em PLP37 através c2dsd de *L2*. Este TS1 corresponde a uma *PLP* normal de *L1*, deste modo, a *PLP* pode ser descodificada por um receptor normal (sintonizador individual de 8 MHz) e um receptor melhorado (sintonizador múltiplo ou sintonizador de banda larga (>8MHz)). TS2 e TS3 são mapeados em PLP39 e PLP44 respectivamente, através de c2dsd. Estes correspondem a *PLP* integrada de *L1*, deste modo, estas *PLPs* podem ser descodificadas por um receptor melhorado (sintonizador múltiplo ou sintonizador de banda larga (>8MHz)) mas não por um receptor normal (sintonizador individual de 8 MHz). Consequentemente, de acordo com informação *L1*, um receptor pode verificar se *TS*

correspondente é recebido ou não.

A figura 115 e figura 116 são fluxogramas descrevendo acções de descodificação *L1* e descodificação *L2* para tipo de *PLP* integrada e tipo de *PLP* normal num receptor normal e num receptor melhorado, respectivamente. A figura 117 mostra um exemplo de estrutura de *c2_delivery_system_descriptor* e sintaxe para sinalização *L2* tendo em conta a figura 112. Este descritor pode mapear *TS_id* em *PLP_id* tal como apresentado na figura 114. A informação do feixe pode ser processada em *L1*, assim, não precisa ser sinalizada em *L2*. As variáveis apresentadas na figura 117 são descritas a seguir.

PLP_id: Este campo de 8 bits identifica exclusivamente uma *PLP* de dados dentro de um sistema C2.

C2_system_id: Este campo de 16 bits identifica exclusivamente um sistema C2. A parte restante desse descritor, imediatamente após o campo *C2_system_id* está presente apenas uma vez por sistema C2, porque os parâmetros são exclusivamente aplicáveis a todas as fatias de dados transportadas por um sistema C2 particular. A presença ou ausência desta parte pode ser derivada a partir do campo de comprimento do descritor. Na ausência da parte restante, este comprimento é igual a 0x07, caso contrário é atribuído um valor maior.

C2_System_tuning_frequency: Este campo de 32 bits

indica um valor de frequência. A gama de codificação pode ser de no mínimo 1 Hz (0x00000001) até um máximo de 4, 294, 967, 295 Hz (0xFFFFFFFF). Este campo de dados pode proporcionar uma frequência de sintonia, onde um preâmbulo completo é transmitido dentro da janela da sintonização. Geralmente o *C2_System_tuning_frequency* é a frequência central de um *C2_System*, mas pode desviar-se da frequência central caso existam entalhes nesta área.

Active_OFDM_symbol_duration: Este campo de 3 bits indica a duração do símbolo *OFDM* activo. Um exemplo é apresentado na figura 118.

Guard_interval: Este campo de 3 bits indica um intervalo de guarda. Um exemplo é apresentado na figura 119.

Nos exemplos anteriores de intercalação/desintercalação no tempo *L1*, para os casos quando *TI_DEPTH* é "10" ou "11", o mapeador de preâmbulo 1007-*L1* da figura 90 pode dividir uniformemente o bloco *L1* original em quatro ou oito blocos secundários. No entanto, se o tamanho do bloco secundário for menor do que um tamanho mínimo necessário para executar uma codificação *FEC*, a codificação *FEC* pode não ser realizada de forma adequada. Uma possível solução pode ser a criação de um limite. Se um tamanho de um bloco *L1* for menor que um limiar estabelecido, o bloco *L1* pode ser repetido por quatro ou oito vezes para os casos em que *TI_DEPTH* é "10"

ou "11". Se um tamanho de um bloco *L1* for maior do que um limiar estabelecido, o bloco *L1* pode ser dividido uniformemente em quatro a oito blocos secundários. O limite pode ser definido como quatro ou oito vezes de um tamanho mínimo necessário para executar uma codificação *FEC*.

Além disso, a definição *TI_DEPTH* como "10" ou "11" é para casos quando o efeito de intercalação não é obtido devido ao pequeno tamanho do bloco *L1*. Assim, o limiar pode ser definido como um tamanho de bits de informação que pode ser transmitido por um único símbolo de preâmbulo. Por exemplo, se for assumida uma codificação idêntica *FEC L1* com *DVB-T2*, um limite será 4,772 bits.

Para casos em que *TI_DEPTH* é "10" ou "11", usando informação de tamanho *L1*, profundidade *TI*, e um valor limiar compartilhado entre um transmissor e um receptor, os módulos de um receptor, a partir do descodificador de cabeçalho *FEC r1012-L1* para *L1_FEC_Merger r1018-L1* da figura 91 pode determinar um tamanho de bloco secundário *L1*, combinar e fundir os blocos secundários *L1* que são transmitidos num símbolo *OFDM* de um preâmbulo.

Se um tamanho de *L1* for menor do que um valor limite, *L1_FEC_Merger r1018-L1* da figura 91 não precisa se fundir blocos secundários divididos porque o bloco *L1* original é transmitido repetidamente de acordo com um *TI_DEPTH* em quatro ou oito símbolos *OFDM*. No entanto, se um tamanho *L1* for maior do que um valor limite, porque é

utilizada uma série de símbolos que é mais do que uma série de símbolos *OFDM* necessários para transmitir bloco *L1*, o decodificador de cabeçalho *FEC* r1012-*L1* da figura 91 pode obter um tamanho de um bloco secundário utilizando *TI_DEPTH*. Então, o combinador *L1_FEC* r1017-*L1* pode combinar blocos *FEC L1* e o desintercalador no tempo r1010-*L1* pode executar a desintercalação. Finalmente, o misturador *L1_FEC* r1018-*L1* pode misturar blocos *L1_FEC* para restaurar o bloco *L1* original.

A figura 120 mostra outros dois exemplos de intercalação no tempo que pode ser utilizada no trajecto *L1* da figura 90. Tal como se pode ver na intercalação no tempo *ON* (1), a intercalação só pode ser intercalação de blocos. Em comparação com o processo apresentado na figura 83, o desempenho da intercalação de frequência pode não ser tão bom quanto o processo apresentado na figura 83. No entanto, para os casos em que *TI_DEPTH* é "10" ou "11", sem repetir ou dividir os blocos *L1* de acordo com um limiar, os blocos *L1* podem ser espalhados numa direcção no tempo independentemente do tamanho do bloco *L1* pois pode ser repetido num preâmbulo se houver espaço no preâmbulo, podendo deste modo este processo ser vantajoso em que o controlo pode ser simplificado. A intercalação pode ser realizada escrevendo fluxos dos símbolos de entrada num sentido do tempo e lendo os fluxos dos símbolos numa direcção da frequência. O desintercalador no tempo r1010-*L1* no trajecto *L1* de um receptor da figura 91 pode realizar a desintercalação escrevendo fluxos de símbolos de entrada

num sentido da frequência e lendo os fluxos dos símbolos escritos numa direcção do tempo.

DSLICE_TUNE_POS indica uma posição de sintonização para um receptor obter uma fatia de dados. Dependendo de um modo *GI*, este valor pode ser expresso em 12 ou 11 bits. *DSLICE_OFFSET_RIGHT* e *DSLICE_OFFSET_LEFT* que indicam valor offset de uma posição de sintonia ou uma largura de uma fatia de dados, pode ser expresso em 9 ou 8 bits, dependendo de um modo *GI*. Se o deslocamento pode ter um valor assinalado, isto é, um valor positivo ou um negativo, pode também ser expressa uma posição e uma largura de uma fatia de dados que tem uma banda estreita. O sintonizador r700 de um receptor da figura 91 pode determinar uma banda *RF* que utiliza uma posição de sintonização, utilizando em seguida este valor de deslocamento assinalado, pode ser obtida a largura da fatia de dados. Deste modo, este campo pode servir uma mesma finalidade que o *DSLICE_WIDTH* acima mencionado. Um receptor pode obter largura de bit usando um valor *GI*.

DSLICE_NOTCH_FLAG é uma referência que indica que uma determinada fatia de dados se encontra contígua a uma banda de entalhe. Ela pode servir uma mesma finalidade, tal como nos exemplos acima mencionados, mas aqui é utilizado apenas 1 bit para este campo por cada fatia de dados. Utilizando esta informação de 1 bit, um receptor pode executar a mesma função que os exemplos acima mencionados.

PLP_BUNDLED_FLAG indica que uma *PLP* é uma *PLP* de dados agrupados. Ou seja, *PLP_BUNDLED_FLAG* indica se ou não uma *PLP* se encontra agrupada a outra *PLP* dentro de um sistema de transmissão. Este campo pode servir um mesmo propósito que os dados agrupados acima descritos *PLP* de *PLP_TYPE* da figura 112. *PLP_TYPE* encontra-se apresentado na figura 110.

A figura 121 mostra outros dois exemplos de intercalação no tempo que pode ser utilizada no trajecto *L1* da figura 90. Tal como se pode ver na intercalação no tempo *ON* (1), a intercalação só pode ser intercalação de blocos. Em comparação com o processo apresentado na figura 83, o desempenho da intercalação de frequência pode não ser tão bom quanto o processo apresentado na figura 83. No entanto, para os casos em que *TI_DEPTH* é "10" ou "11", sem repetir ou dividir os blocos *L1* de acordo com um limiar, os blocos *L1* podem ser espalhados numa direcção no tempo independentemente do tamanho do bloco *L1* pois pode ser repetido num preâmbulo se houver espaço no preâmbulo, podendo deste modo este processo ser vantajoso em que um controlo pode ser simplificado. A intercalação pode ser realizada escrevendo fluxos dos símbolos de entrada num sentido do tempo e lendo os fluxos dos símbolos numa direcção da frequência. O desintercalador no tempo *r1010-L1* no trajecto *L1* de um receptor da figura 91 pode realizar a desintercalação escrevendo fluxos de símbolos de entrada num sentido da frequência e lendo os fluxos dos símbolos escritos numa direcção do tempo.

Um segundo exemplo ou a intercalação no tempo *ON* (2) da figura 120 inclui processo adicional para a intercalação no tempo *ON* (1), que é uma deslocação circular na direcção da linha. Por este processo, além das vantagens da intercalação no tempo *ON* (1), pode ser obtido um efeito de propagação num domínio da frequência. O desintercalador no tempo *r1010-L1* no trajecto *L1* de um receptor da figura 91 precisa executar circularmente uma nova deslocação na direcção da linha antes de executar o processo de intercalação no tempo *ON* (1).

A figura 122 é um outro exemplo de um transmissor *OFDM* que utiliza fatia de dados. Difere da figura 90, em blocos no trajecto *L1*. Figura 124 e figura 126 são proporcionadas para descrição pormenorizada dos diferentes blocos. O módulo de sinalização *L1* 700-*L1* pode executar funções idênticas às funções do mesmo bloco na figura 90. O codificador *FEC LDPC/BCH* 1902-*L1* pode realizar o particionamento *L1* e codificação mostrada na figura 124. Usando uma informação *L1* os bits podem ser transmitidos por um único símbolo de preâmbulo *OFDM* como uma referência, se necessário, *L1* pode ser particionada e a *L1* particionada pode ser *FEC* codificada. O intercalador de bits 703-*L1* 704-*L1* e mapeador de símbolos 704-*L1* pode executar funções idênticas às funções dos mesmos blocos na figura 80 ou figura 90. Isto é, o intercalador de bits 703-11 intercala o bloco de sinalização *L1* e o mapeador de símbolo desmultiplexa o bloco de sinalização *L1* bit intercalado em palavras de células e executa o mapeamento das palavras de

células para os valores de constelação que correspondem ao símbolo de informação de sinalização de camada 1. Neste caso, o mapeador de símbolo pode ser um mapeador QAM.

O intercalador no tempo 1908-*L1* pode intercalar no tempo símbolos de preâmbulo com uma profundidade de intercalação no tempo *L1* tal como apresentado na figura 124. Dependendo da profundidade da intercalação no tempo, a intercalação no tempo pode ser realizada tal como na figura 126. Para o caso de não intercalação no tempo (*L1_TI_MODE*="00"), a intercalação no tempo não é realizada. Para o caso de profundidade de intercalação no tempo ser um número mínimo de símbolos *OFDM* necessários para transmitir dados *L1* (*L1_TI_MODE*="01") é executada a intercalação no tempo de acordo com um número de símbolos *OFDM*. Para o caso da profundidade da intercalação no tempo ser mais do que um número mínimo de símbolos *OFDM* necessários para transmitir dados *L1* (*L1_TI_MODE*="10" e profundidade = 4 símbolos *OFDM*), um tamanho de um bloco de intercalação no tempo pode ter um número de linhas, tanto quanto um valor de uma profundidade de intercalação no tempo e um número de colunas tanto quanto um quociente resultante da divisão de um número de símbolos QAM necessários para transmitir dados *L1* pela profundidade de intercalação no tempo. Uma intercalação no tempo pode ser realizada numa memória de matriz de linha-coluna que tem um tamanho do bloco de intercalação no tempo. O módulo de inserção de cabeçalho *L1* 1905-*L1* pode inserir cabeçalho *L1* no bloco *L1* que é intercalado no tempo para cada símbolo *OFDM* dentro de um

preâmbulo, tal como mostrado na figura 124. O mapeador de preâmbulo 1907-L1 pode mapear o cabeçalho L1 e bloco L1 em símbolos OFDM pré-determinados num preâmbulo. Para cada símbolo OFDM, módulo de repetição L1 1915-L1 pode repetir o cabeçalho L1 e bloco L1 para preencher a largura de banda do preâmbulo. Finalmente, o intercalador de frequências 709-L1 pode executar funções idênticas às funções do mesmo bloco da figura 90.

A figura 123 mostra um outro exemplo de um receptor OFDM que utiliza fatia de dados. Difere da figura 91, em blocos no trajecto L1. A figura 125 e figura 127 são proporcionadas para descrição pormenorizada dos diferentes blocos . O desintercalador de frequências r709-L1 pode executar funções idênticas às funções dos mesmos blocos da figura 91. O combinador L1 r1917-L1 pode sincronizar blocos L1 tal como apresentado na figura 125. Além disso, o ganho de SNR pode ser obtido da combinação de cabeçalho L1 e bloco L1 repetido numa largura de banda de preâmbulo. O decodificador de cabeçalho L1 r1912-L1 pode obter ganho SNR adicional ao combinar cabeçalhos L1 que são repetidamente transmitidos ao longo de uma direcção do tempo, fazendo referência à profundidade da intercalação no tempo L1. Além disso, os parâmetros de intercalação no tempo L1 e tamanho dos dados L1 podem ser obtidos da decodificação FEC do cabeçalho L1. O desintercalador no tempo r1910-L1 pode realizar processos apresentados na figura 125 e figura 127 que são processos inversos de processos realizados num transmissor tal como apresentado

na figura 124 e figura 126.

O desmapeador de símbolos *r713-L1* pode calcular o bit *LLR* dos símbolos de entrada e emitir o bit *LLR*. Ao utilizar o comprimento de dados *L1* e profundidade de intercalação no tempo *L1* transmitido no cabeçalho *L1* e tendo em conta uma série de blocos *L1* que particionou dados *L1* e uma série de símbolos *OFDM* onde os blocos *L1* estão espalhados, concentrador *L1* *r1918-L1* pode restaurar blocos *L1* necessários para realizar a descodificação *FEC*. O desintercalador de bits *r714-L1* e descodificação *FEC* *BCH/LDPC* *r715-L1* pode executar funções idênticas às funções dos mesmos blocos da figura 91.

Usando os métodos e dispositivos propostos, entre outras vantagens, é possível implementar um transmissor digital eficiente, receptor e estrutura de sinalização da camada física.

Ao transmitir informação *ModCod* em cada quadro *BB* que é necessário para *ACM/VCM* e transmitir o resto da sinalização da camada física num cabeçalho de quadro, a informação complementar da sinalização pode ser minimizada.

Pode ser implementado o *QAM* modificado para uma transmissão mais eficiente em termos de energia ou um sistema de radiodifusão digital mais resistente ao ruído. O sistema pode incluir o transmissor e receptor para cada exemplo descrito e as suas combinações.

Pode ser implementado um *QAM* melhorado irregular para uma transmissão mais eficiente em termos de energia ou um sistema de radiodifusão digital mais resistente ao ruído. É também descrito um processo de utilização de uma taxa de código de código de correcção de erros de *NU-MQAM* e *MQAM*. O sistema pode incluir o transmissor e receptor para cada exemplo descrito e as suas combinações.

O processo de sinalização *L1* sugerido pode reduzir a informação adicional em 3~4% minimizando a informação complementar de sinalização durante a ligação do canal.

Será visível aos técnicos que podem ser realizadas várias modificações e variações na presente invenção sem fugir do escopo da invenção.

Lisboa, 6 de Julho de 2012

REIVINDICAÇÕES

1. Transmissor para a transmissão de dados de radiodifusão para um receptor, que compreende:

um codificador de erros directo *FEC* (702-L1) configurado para uma codificação *FEC* de dados de sinalização de camada 1;

um intercalador de bits (703-L1) configurado para intercalar bits dos dados de sinalização de camada 1 de codificação *FEC*;

um mapeador *QAM* (704-L1) configurado para desmultiplexar os dados de sinalização da camada 1 de bits intercalados para dentro de palavras de células e para mapear as palavras de células para valores de constelação que correspondem aos dados de sinalização da camada 1;

um intercalador no tempo (1908-L1) configurado para intercalar no tempo os valores de constelação mapeados que correspondem a dados de sinalização de camada 1 com base numa memória de acordo com a *TI* de camada 1, intercalação no tempo, a informação do modo que indica uma profundidade da intercalação no tempo, em que um número de linhas da memória é igual à profundidade da intercalação no tempo, em que um número de colunas da memória é igual a um quociente resultante da divisão de um número de valores de constelação necessários para transmitir dados de sinalização *L1* pela profundidade da intercalação no

tempo, e em que a profundidade da intercalação no tempo é igual ou maior do que um número mínimo de símbolos *OFDM* necessários para transportar dados de sinalização *L1*;

um módulo de inserção (1905-*L1*) configurado para inserir um cabeçalho de camada 1 nos valores de constelação intercalados no tempo que correspondem aos dados de sinalização da camada 1, em que o cabeçalho da camada 1 inclui a informação do modo *TI L1*;

um meio de repetição (1915-*L1*) configurado para repetir os valores de constelação que correspondem aos dados de sinalização da camada 1 e ao cabeçalho da camada 1;

um intercalador de frequência (709-*L1*) configurado para intercalar na frequência os valores de constelação repetidos que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1, e meios (711) para a construção de um quadro de sinal que compreende símbolos de preâmbulo com base nos valores de constelação intercalados na frequência, em que os símbolos de preâmbulo estão divididos em blocos de camada 1, sendo que cada bloco de camada 1 apresenta 3408 subportadoras.

2. Transmissor de acordo com a reivindicação 1, em que o codificador *FEC* compreende:

um codificador BCH configurado para efectuar uma codificação BCH dos dados de sinalização de camada 1;

um codificador LDPC configurado para efectuar uma codificação LDPC dos dados de sinalização de camada 1 de codificação BCH para gerar pelo menos um bit de paridade LDPC; e

um meio de punção configurado para executar a punção no bit de paridade LDPC gerado.

3. Transmissor de acordo com a reivindicação 2, em que o intercalador de bits se encontra adicionalmente configurado para intercalar os dados de sinalização de camada 1 de codificação *FEC* sobre os quais é executada a punção.

4. Transmissor de acordo com qualquer das reivindicações 1 a 3, em que o intercalador no tempo se encontra configurado para não executar um processo de intercalação no tempo nos valores de constelação mapeados que correspondem aos dados de sinalização de camada 1 para o caso de a informação do modo *TI L1* não indicar modo de intercalação no tempo.

5. Transmissor de acordo com qualquer das reivindicações 1 a 4, em que, quando um valor da informação de modo *TI L1* é "01", a profundidade da intercalação no tempo é o número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*.

6. Receptor para processamento de dados de transmissão, que compreende:

meios para a obtenção de um quadro de sinal que compreende símbolos de preâmbulo incluindo dados de sinalização de camada 1 e um cabeçalho de camada 1, em que os símbolos de preâmbulo se encontram divididos em blocos de camada 1 e cada bloco de camada 1 apresenta 3408 subportadoras, e em que o cabeçalho de camada 1 inclui informação do modo *TI L1* que indica uma profundidade de intercalação no tempo;

desintercalador de frequência (r709-*L1*) configurado para desintercaladar na frequência valores de constelação que correspondem aos dados de sinalização da camada 1 e ao cabeçalho da camada 1;

um extrator (r1917-*L1*, r1912-*L1*) configurado para extrair os valores de constelação que correspondem aos dados de sinalização de camada 1 dos valores de constelação desintercalados na frequência que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1;

desintercalador no tempo (r1910-*L1*) configurado para desintercaladar no tempo os valores de constelação extraídos que correspondem aos dados de sinalização de camada 1 com base numa memória de acordo com a informação do modo *TI L1*, em que um número de linhas da memória é igual à profundidade da intercalação no tempo, em que um número de colunas da memória é igual a um quociente resultante da divisão de um número de valores de constelação necessários para transmitir dados de sinalização *L1* pela profundidade

da intercalação no tempo, e em que a profundidade da intercalação no tempo é igual ou maior do que um número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*;

um desmapeador de *QAM* (*r713-L1*) configurado para desmapear valores de constelação que correspondem aos dados de sinalização da camada 1 em dados de sinalização de camada 1;

um desintercalador de bits (*r714-L1*) configurado para desintercaladar bits dos dados de sinalização da camada 1; e

um descodificador de erros directo, *FEC*, (*r715-L1*) configurado para efectuar uma descodificação *FEC* dos dados de sinalização da camada 1 dos bits desintercalados.

7. Receptor de acordo com a reivindicação 6, em que o descodificador *FEC* compreende:

um meio de despuncionagem configurado para executar a despuncionagem em pelo menos um bit de paridade LDPC;

um descodificador LDPC configurado para descodificação LDPC dos dados de sinalização de camada 1 e o bit de paridade LDPC despuncionado; e

um descodificador BCH configurado para descodificação BCH dos dados de sinalização de camada 1 de descodificação LDPC e do bit de paridade LDPC.

8. Receptor de acordo com qualquer das reivindicações 6 ou 7, em que o desintercalador no tempo se encontra configurado para não executar um processo de desintercalação no tempo nos valores de constelação extraídos que correspondem aos dados de sinalização de camada 1 para o caso de a informação do modo *TI L1* indicar nenhum modo de intercalação no tempo.

9. Receptor de acordo com qualquer das reivindicações 6 a 8, em que, quando um valor da informação de modo *TI L1* é "01", a profundidade da intercalação no tempo é o número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*.

10. Processo para receber dados de radiodifusão, que compreende:

obter um quadro de sinal que compreende símbolos de preâmbulo incluindo dados de sinalização de camada 1 e um cabeçalho de camada 1, em que os símbolos de preâmbulo se encontram divididos em blocos de camada 1, apresentando cada bloco de camada 1 3408 subportadoras, e em que o cabeçalho de camada 1 inclui informação do modo *TI L1* que indica uma profundidade de intercalação no tempo;

valores de constelação de desintercalação de frequência que correspondem aos dados de sinalização de camada 1 e ao cabeçalho da camada 1;

extrair os valores de constelação que correspondem aos dados de sinalização da camada 1 dos valores de

constelação desintercalados na frequência que correspondem aos dados de sinalização da camada 1 e cabeçalho da camada 1;

desintercalas no tempo os valores de constelação extraídos que correspondem aos dados de sinalização de camada 1 com base numa memória de acordo com a informação do modo *TI L1*, em que um número de linhas da memória é igual à profundidade da intercalação no tempo, em que um número de colunas da memória é igual a um quociente resultante da divisão de um número de valores de constelação necessários para transmitir dados de sinalização *L1* pela profundidade da intercalação no tempo, e em que a profundidade da intercalação no tempo é igual ou maior do que um número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*;

desmapear valores de constelação desintercalados no tempo que correspondem aos dados de sinalização da camada 1 em dados de sinalização de camada 1;

desintercalas em bits os dados de sinalização de camada 1; e

descodificação *FEC* dos dados de sinalização da camada 1 de bits desintercalados.

11. Processo de acordo com a reivindicação 10, que compreende adicionalmente:

executar a despuncionagem em pelo menos um bit de paridade LDPC;

descodificação LDPC dos dados de sinalização de

camada 1 e do bit de paridade LDPC despuncionado; e descodificação BCH dos dados de sinalização de camada 1 de descodificação LDPC e do bit de paridade LDPC.

12. Processo de acordo com qualquer das reivindicações 10 ou 11, em que não é executado um processo de desintercalação no tempo nos valores de constelação extraídos que correspondem aos dados de sinalização de camada 1 para o caso de a informação do modo *TI L1* indicar nenhum modo de intercalação no tempo.

13. Processo de acordo com qualquer das reivindicações 10 a 12, em que, quando um valor da informação de modo *TI L1* é "01", a profundidade da intercalação no tempo é o número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*.

14. Processo para transmitir dados de radiodifusão para um receptor, que compreende:

dados de sinalização de camada 1 de codificação *FEC*;
intercalar os bits dos dados de sinalização de camada 1 de codificação *FEC*;
desmultiplexar os dados de sinalização de camada 1 de intercalação de bits em palavras de célula;
mapear as palavras de célula em valores de constelação que correspondem aos dados de sinalização de camada 1;
intercalação no tempo dos valores de constelação mapeados que correspondem a dados de sinalização de

camada 1 com base numa memória de acordo com a informação do modo *TI L1* que indica uma profundidade da intercalação no tempo, em que um número de linhas da memória é igual à profundidade da intercalação no tempo, em que um número de colunas da memória é igual a um quociente resultante da divisão de um número de valores de constelação necessários para transmitir dados de sinalização *L1* pela profundidade da intercalação no tempo, e em que a profundidade da intercalação no tempo é igual ou maior do que um número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*;

inserção de um cabeçalho de camada 1 nos valores de constelação intercalados no tempo que correspondem aos dados de sinalização da camada 1, em que o cabeçalho da camada 1 inclui a informação do modo *TI L1*;

repetição dos valores de constelação que correspondem aos dados de sinalização de camada 1 e ao cabeçalho da camada 1;

intercalação na frequência dos valores de constelação que correspondem aos dados de sinalização de camada 1 e ao cabeçalho da camada 1;

construção de um quadro de sinal que compreende símbolos de preâmbulo com base nos valores de constelação intercalados na frequência, em que os símbolos de preâmbulo estão divididos em blocos de camada 1, sendo que cada bloco de camada 1 apresenta 3408 subportadoras.

15. Processo de acordo com a reivindicação 14, que compreende adicionalmente:

codificação BCH dos dados de sinalização de camada 1;

codificação LDPC dos dados de sinalização de camada 1 de codificação BCH para gerar pelo menos um bit de paridade LDPC; e

executar a punção no bit de paridade LDPC gerado.

16. Processo de acordo com a reivindicação 15, caracterizado que compreende adicionalmente os dados de sinalização de camada 1 de codificação *FEC* sobre os quais é executada a punção.

17. Processo de acordo com qualquer das reivindicações 14 a 16, em que o processo de intercalação no tempo não é executado nos valores de constelação mapeados que correspondem aos dados de sinalização de camada 1 caso a informação do modo *TI L1* indique nenhum modo de intercalação no tempo.

18. Processo de acordo com qualquer das reivindicações 14 a 17, em que, quando um valor da informação de modo *TI L1* é "01", a profundidade da

intercalação no tempo é número mínimo de símbolos *OFDM* necessários para transportar os dados de sinalização *L1*.

Lisboa, 6 de Julho de 2012

Fig. 1

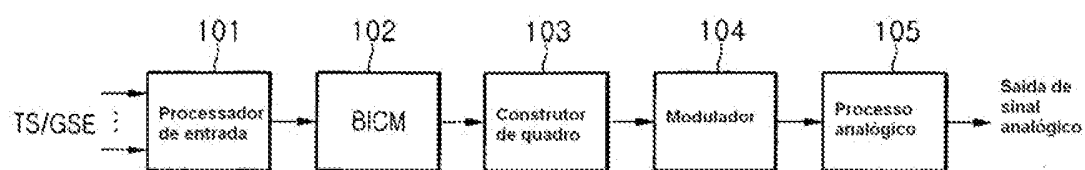


Fig. 2

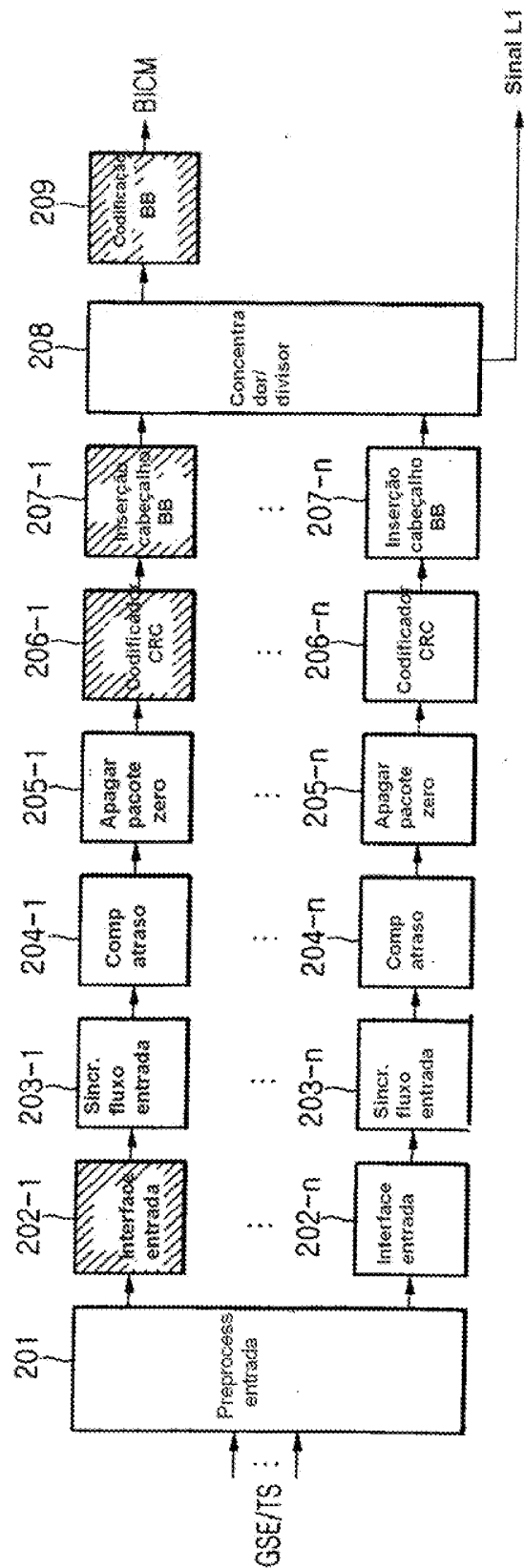


Fig. 3

TS/GS (2 bits)	SIS/MIS (1 bit)	CCM/ACM (1 bit)	ISSYI (1 bit)	NPD (1bit)	EXT (2bits)
00 = GFPS 11 = TS 01 = GCS 10 = GSE	1 = individual 0 = múltiplo	1 = CCM 0 = ACM	1 = activo 0 = não-activo	1 = activo 0 = não-activo	Reservado para utilização futura

Campo	Dimensão (Bytes)	Descrição
MATYPE	2	Tal como descrito acima
UPL	2	Comprimento do pacote do utilizador em bits, na gama [0,65535]
DFL	2	Comprimento do campo de dados, na gama [0,53760]
SYNC	1	Uma cópia do byte de sincronização do pacote de utilizador
SYNCD	2	A distância em bits do início do campo de dados (DATA FIELD) para o primeiro UP completo do campo de dados. SYNCD=0o quer dizer que o primeiro UP se encontra alinhado com o início do campo de dados. SYNCD=65535d quer dizer que nenhum UP tem início no DATA FIELD.
CRC-8 MODE	1	O XOR do campo CRC-8 (1-byte) com o campo MODE (1-byte). CRC-8 é o código de detecção de erro aplicado aos primeiros 9 bytes do BBHEADER. MODE (8 bits) deve ser: * 0o Modo normal * 1o Modo de alta eficiência * Outros valores: reservado para utilização futura

Fig. 4

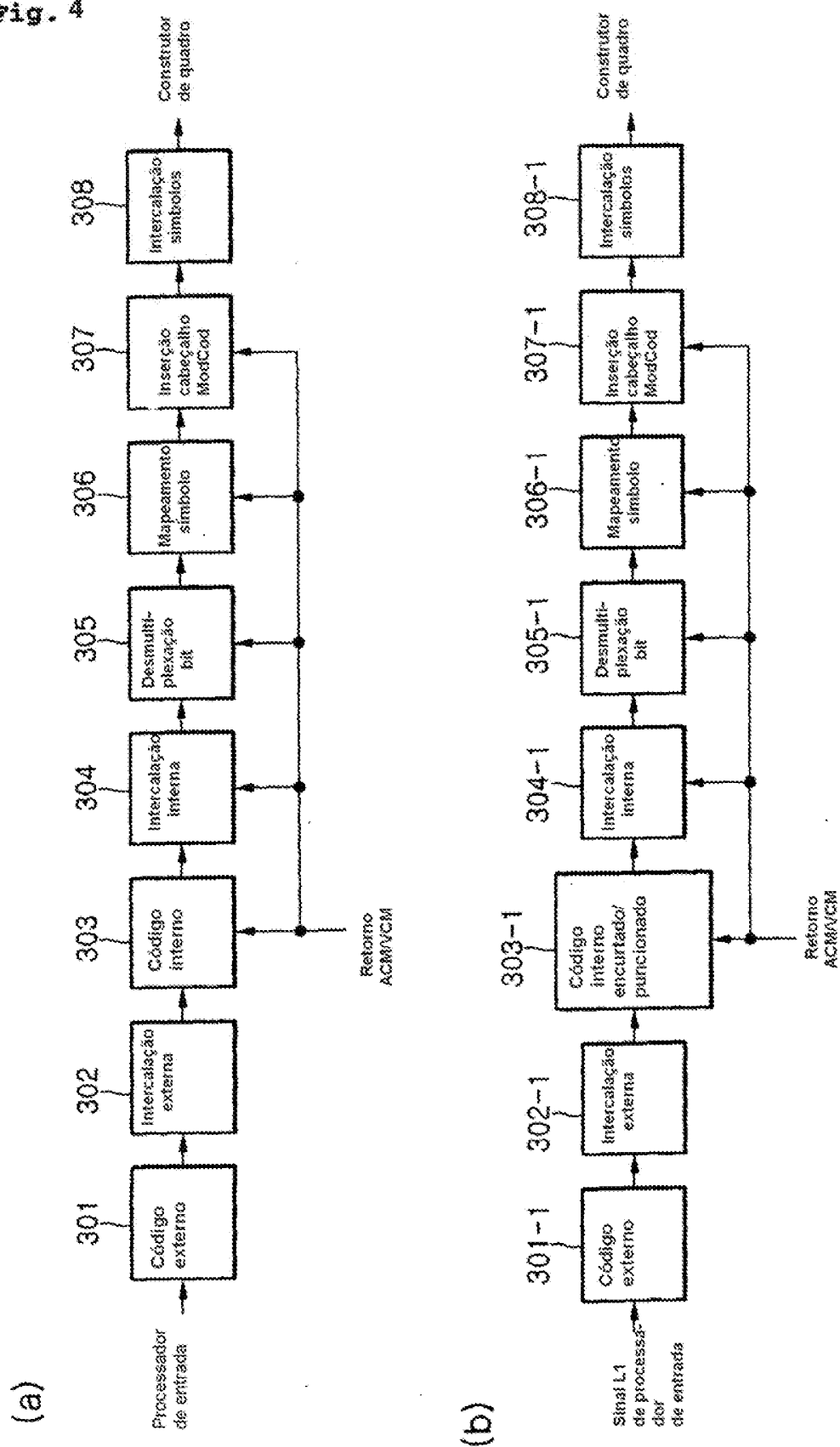


Fig. 5

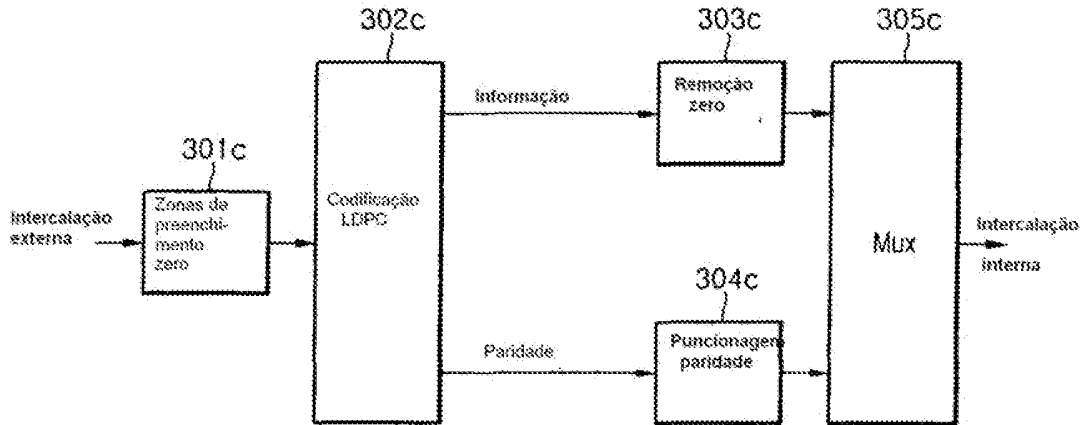


Fig. 6

			Caso 1	Caso 2
Capacidade (bit/s/Hz)			Modulação	Modulação
6	1/2	3.0	NU-MQAM	NU-QAM
	2/3	4.0	NU-MQAM	NU-QAM
	3/4	4.5	NU-MQAM	NU-MQAM
	4/5	4.8	MQAM	MQAM
	5/6	5.0	MQAM	MQAM
	8/9	5.3	MQAM	MQAM
	9/10	5.4	MQAM	MQAM
8	1/2	4.0	NU-MQAM	NU-QAM
	2/3	5.3	NU-MQAM	NU-QAM
	3/4	6.0	NU-MQAM	NU-MQAM
	4/5	6.4	NU-MQAM	NU-MQAM
	5/6	6.7	MQAM	MQAM
	8/9	7.1	MQAM	MQAM
	9/10	7.2	MQAM	MQAM
10	1/2	5.0	NU-MQAM	NU-QAM
	2/3	6.7	NU-MQAM	NU-QAM
	3/4	7.5	NU-MQAM	NU-MQAM
	4/5	8.0	NU-MQAM	NU-MQAM
	5/6	8.3	NU-MQAM	NU-MQAM
	8/9	8.9	MQAM	MQAM
	9/10	9.0	MQAM	MQAM
12	1/2	6.0	NU-MQAM	NU-QAM
	2/3	8.0	NU-MQAM	NU-QAM
	3/4	9.0	NU-MQAM	NU-MQAM
	4/5	9.6	NU-MQAM	NU-MQAM
	5/6	10.0	NU-MQAM	NU-MQAM
	8/9	10.7	MQAM	MQAM
	9/10	10.8	MQAM	MQAM

Fig. 7

			Caso 1	Caso 2	Caso 2
Capacidade (bit/s/Hz)			Modulação	Modulação	Modulação
2	1/2	3.0	QAM	QAM	QAM
	2/3	4.0	QAM	QAM	QAM
	3/4	4.5	QAM	QAM	QAM
	4/5	4.8	QAM	QAM	QAM
	5/6	5.0	QAM	QAM	QAM
	8/9	5.3	QAM	QAM	QAM
	9/10	5.4	QAM	QAM	QAM
4	1/2	3.0	QAM	QAM	QAM
	2/3	4.0	QAM	QAM	QAM
	3/4	4.5	QAM	QAM	QAM
	4/5	4.8	QAM	QAM	QAM
	5/6	5.0	QAM	QAM	QAM
	8/9	5.3	QAM	QAM	QAM
	9/10	5.4	QAM	QAM	QAM
6	1/2	3.0	QAM	QAM	QAM
	2/3	4.0	QAM	QAM	QAM
	3/4	4.5	QAM	QAM	QAM
	4/5	4.8	QAM	QAM	QAM
	5/6	5.0	QAM	QAM	QAM
	8/9	5.3	QAM	QAM	QAM
	9/10	5.4	QAM	QAM	QAM
8	1/2	4.0	QAM	QAM	QAM
	2/3	5.3	QAM	QAM	QAM
	3/4	6.0	QAM	QAM	QAM
	4/5	6.4	QAM	QAM	QAM
	5/6	6.7	QAM	QAM	QAM
	8/9	7.1	QAM	QAM	QAM
	9/10	7.2	QAM	QAM	QAM
10	1/2	5.0	NU-MQAM	NU-QAM	MQAM
	2/3	6.7	NU-MQAM	NU-QAM	MQAM
	3/4	7.5	NU-MQAM	NU-MQAM	MQAM
	4/5	8.0	NU-MQAM	NU-MQAM	MQAM
	5/6	8.3	NU-MQAM	NU-MQAM	MQAM
	8/9	8.9	MQAM	MQAM	MQAM
	9/10	9.0	MQAM	MQAM	MQAM
12	1/2	6.0	NU-MQAM	NU-QAM	MQAM
	2/3	8.0	NU-MQAM	NU-QAM	MQAM
	3/4	9.0	NU-MQAM	NU-MQAM	MQAM
	4/5	9.6	NU-MQAM	NU-MQAM	MQAM
	5/6	10.0	NU-MQAM	NU-MQAM	MQAM
	8/9	10.7	MQAM	MQAM	MQAM
	9/10	10.8	MQAM	MQAM	MQAM

Fig. 8

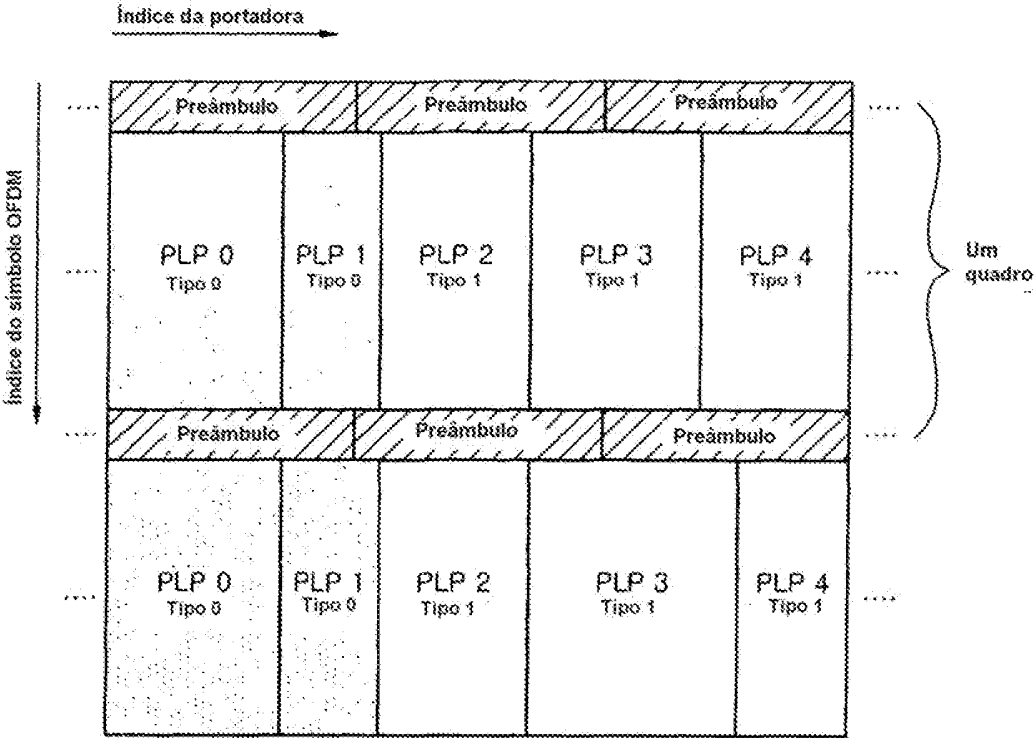


Fig. 9

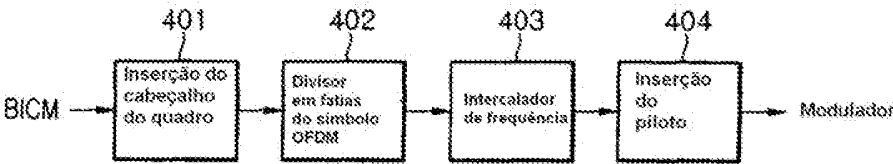


Fig. 10

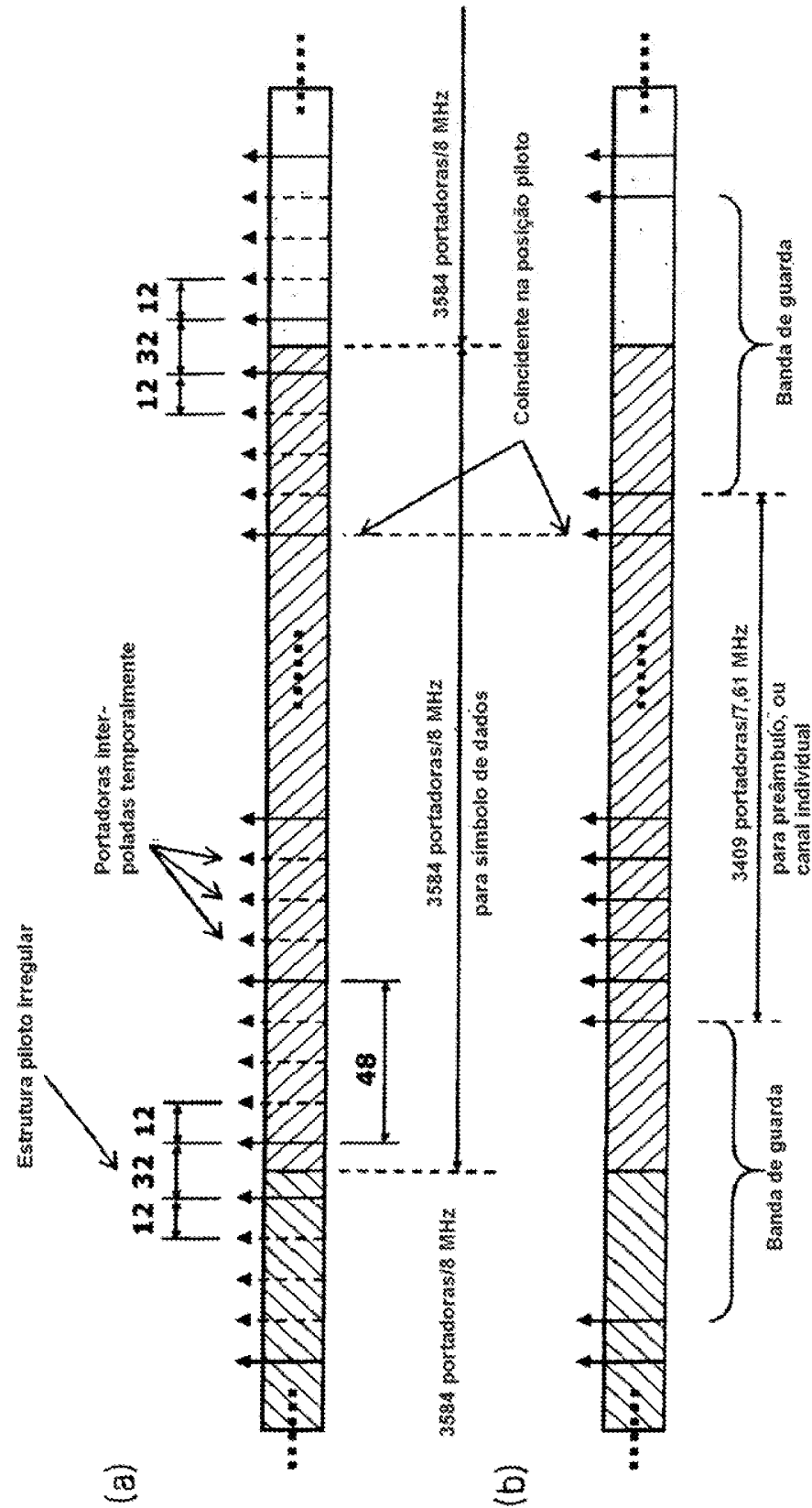


Fig. 11

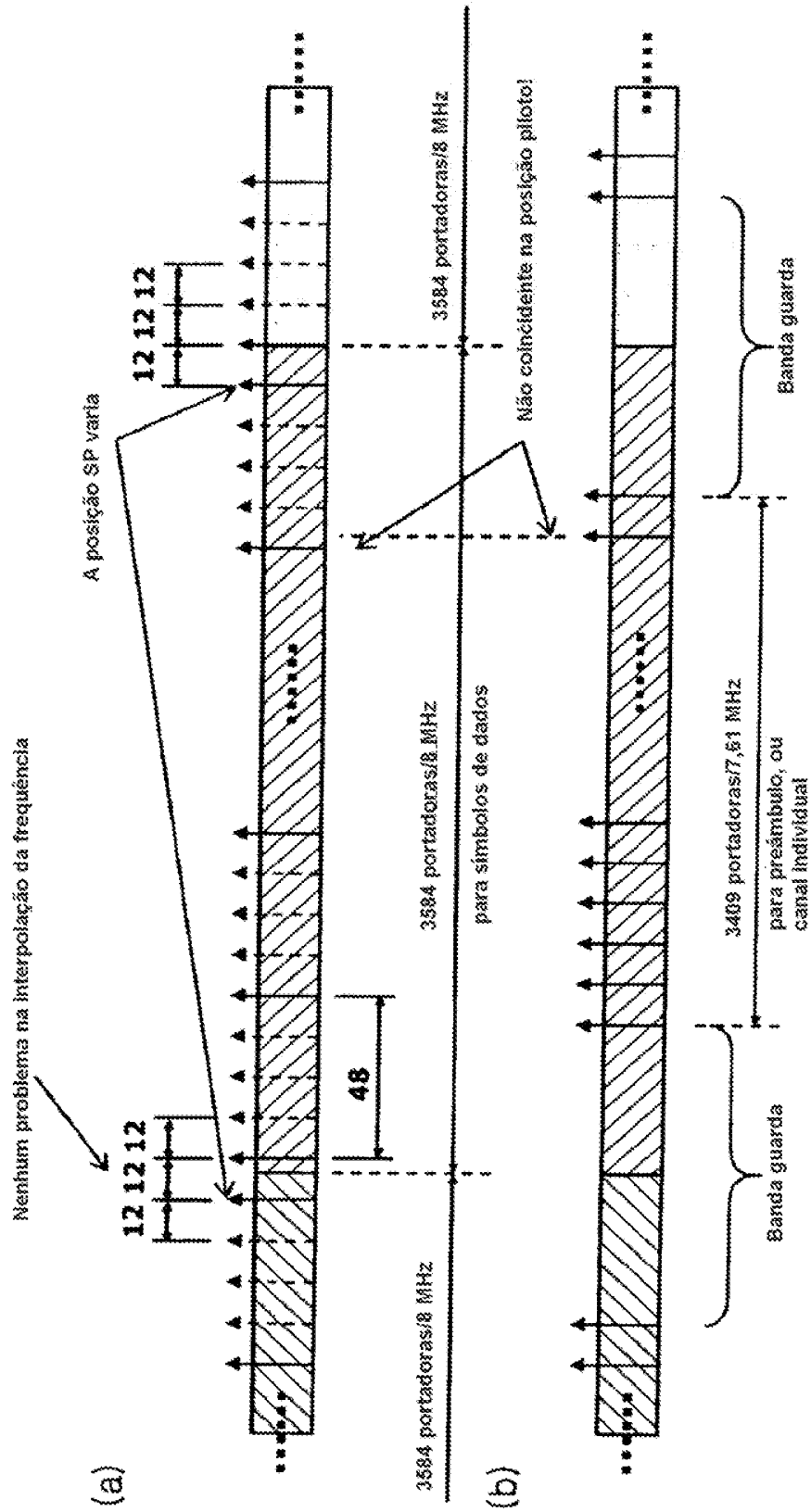


Fig. 12

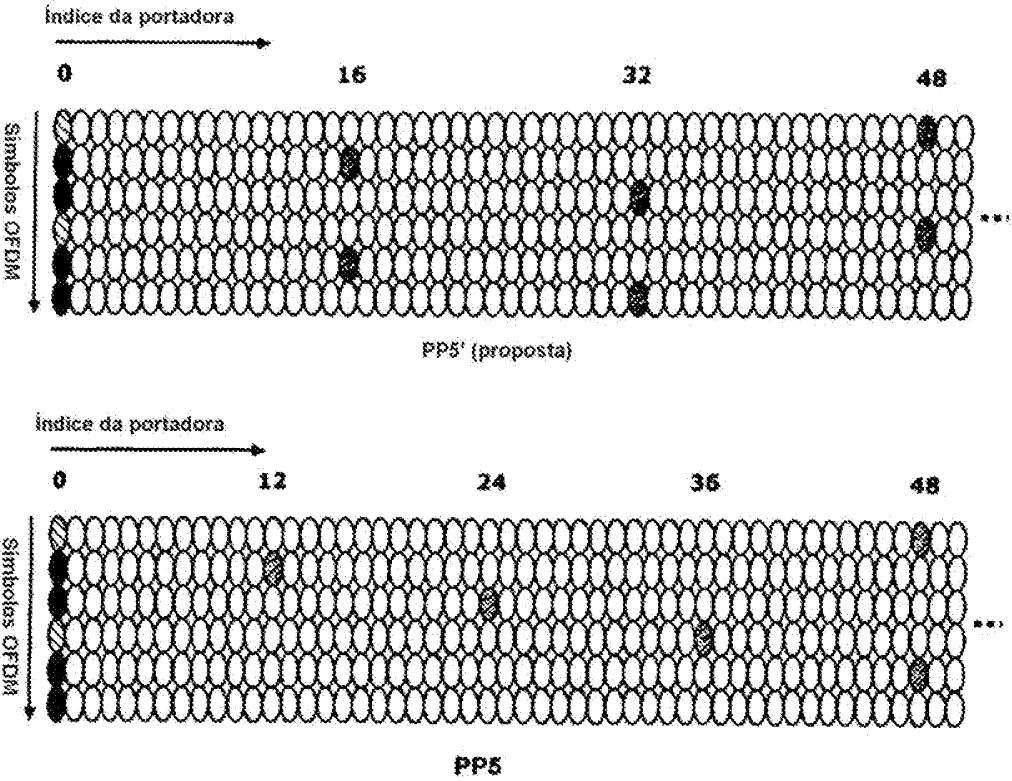


Fig. 13

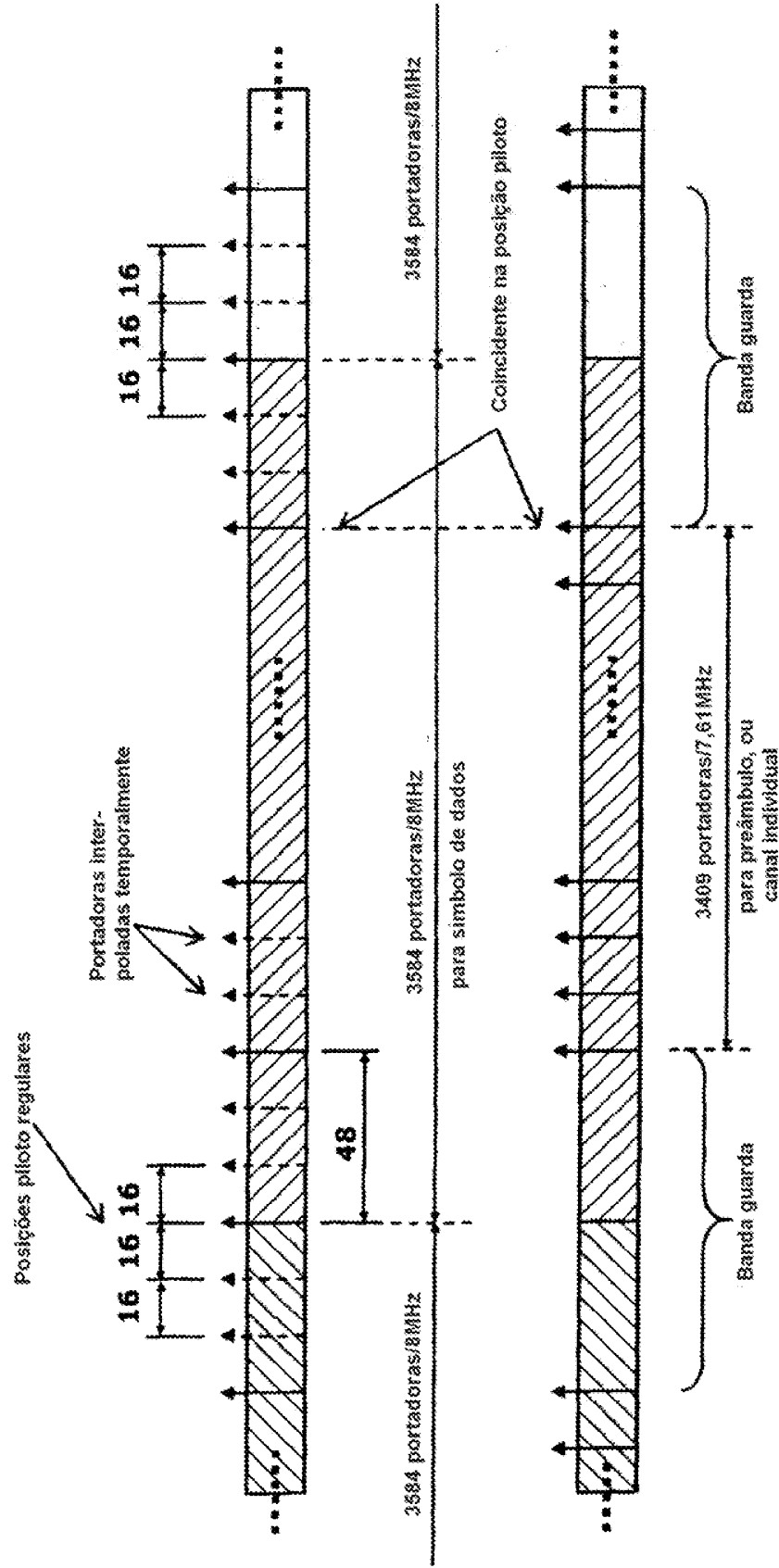


Fig. 17

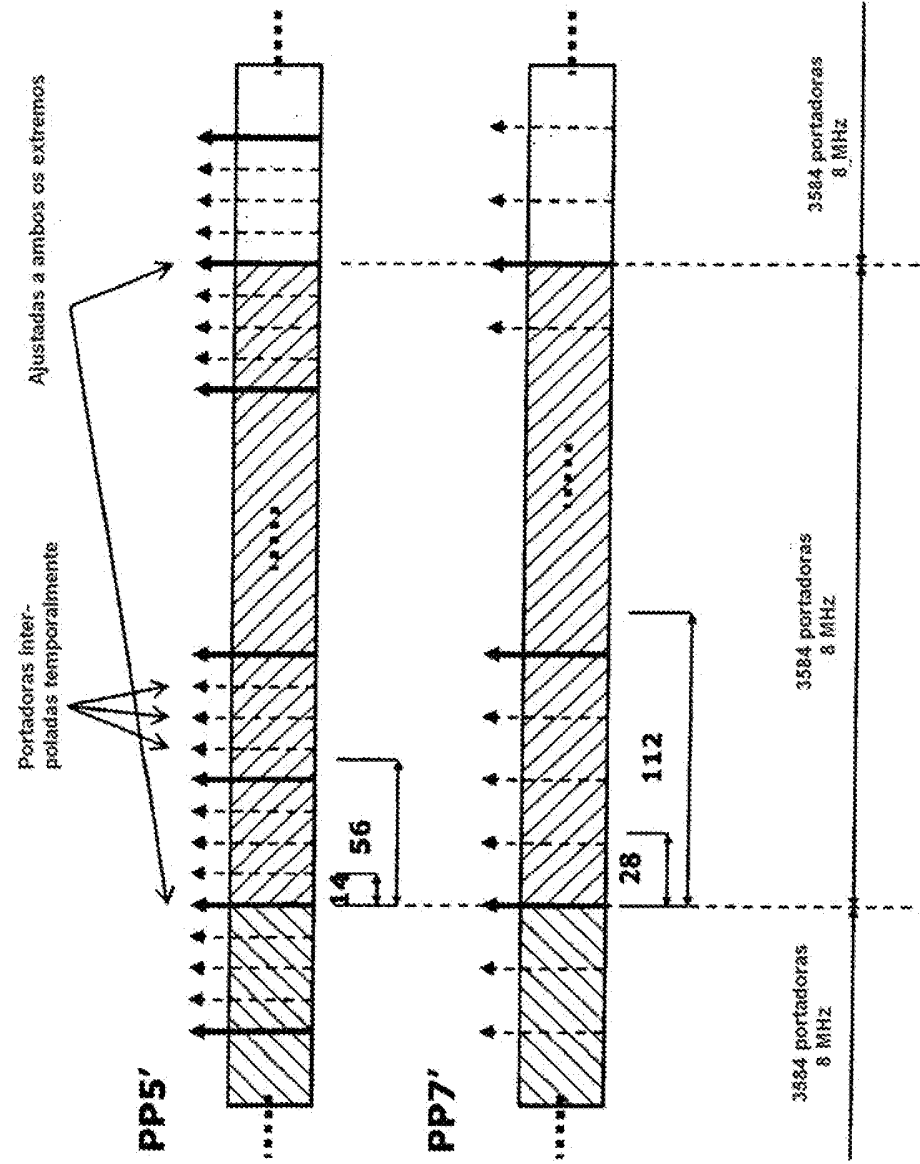


Fig. 18

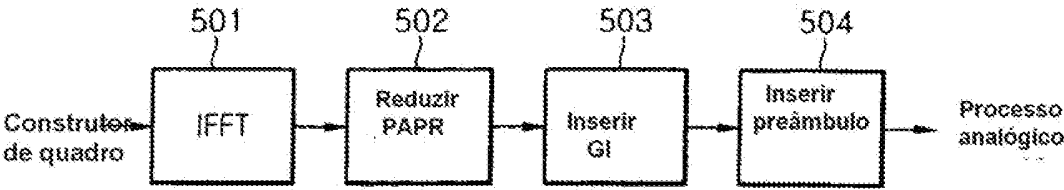


Fig. 19

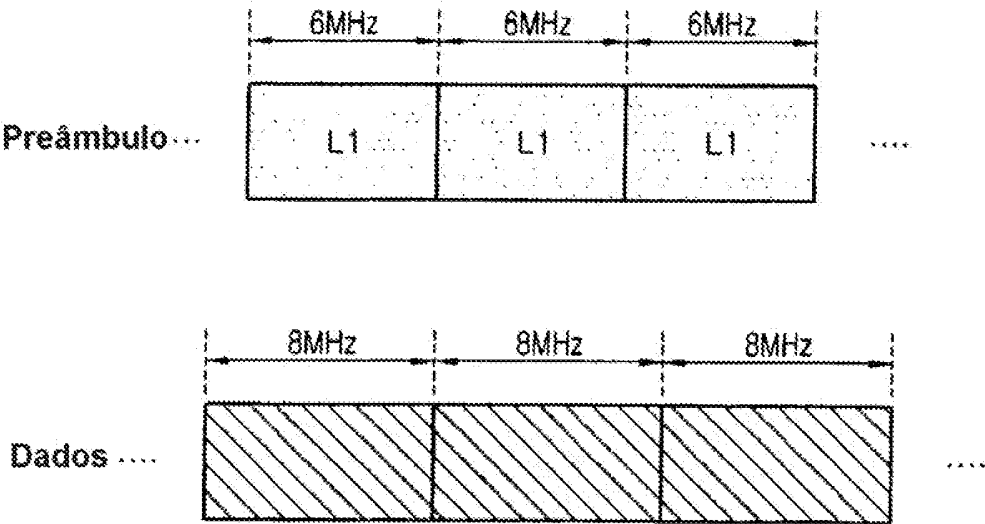


Fig. 20

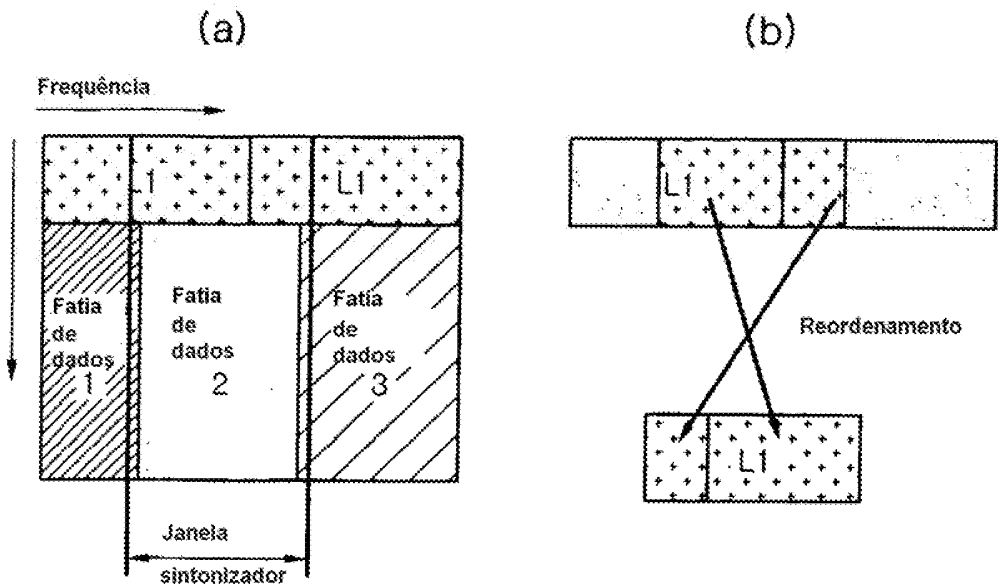


Fig. 21

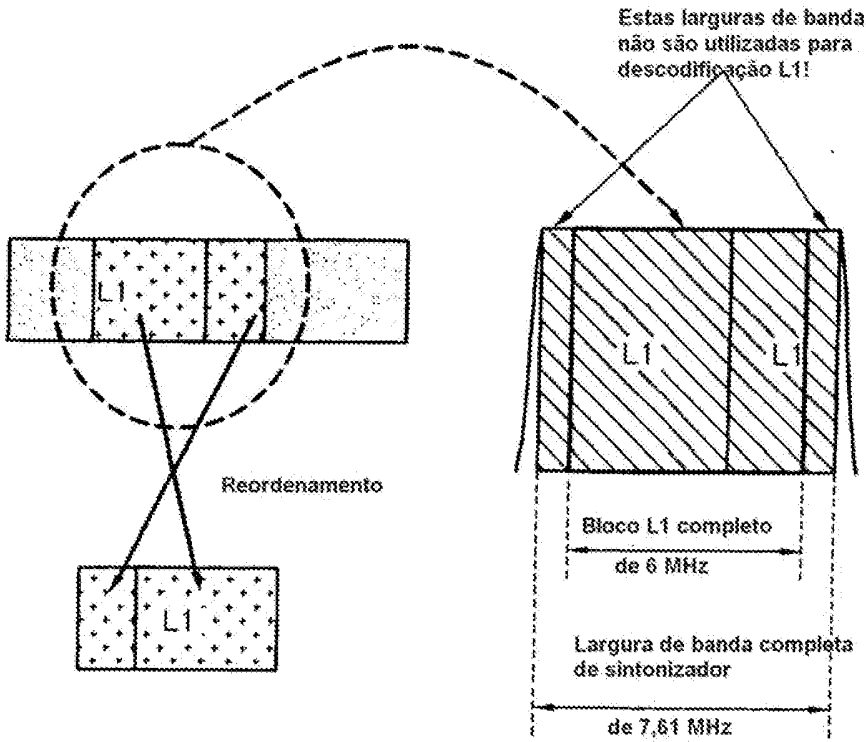


Fig. 22

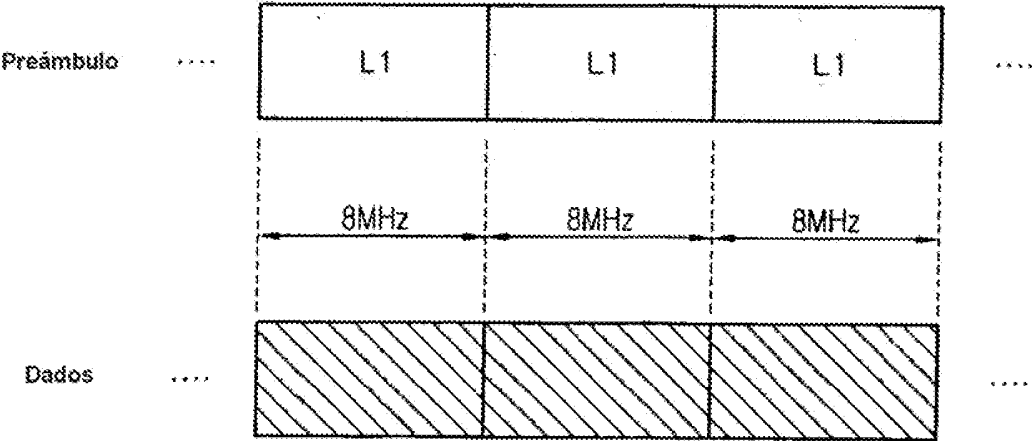


Fig. 23

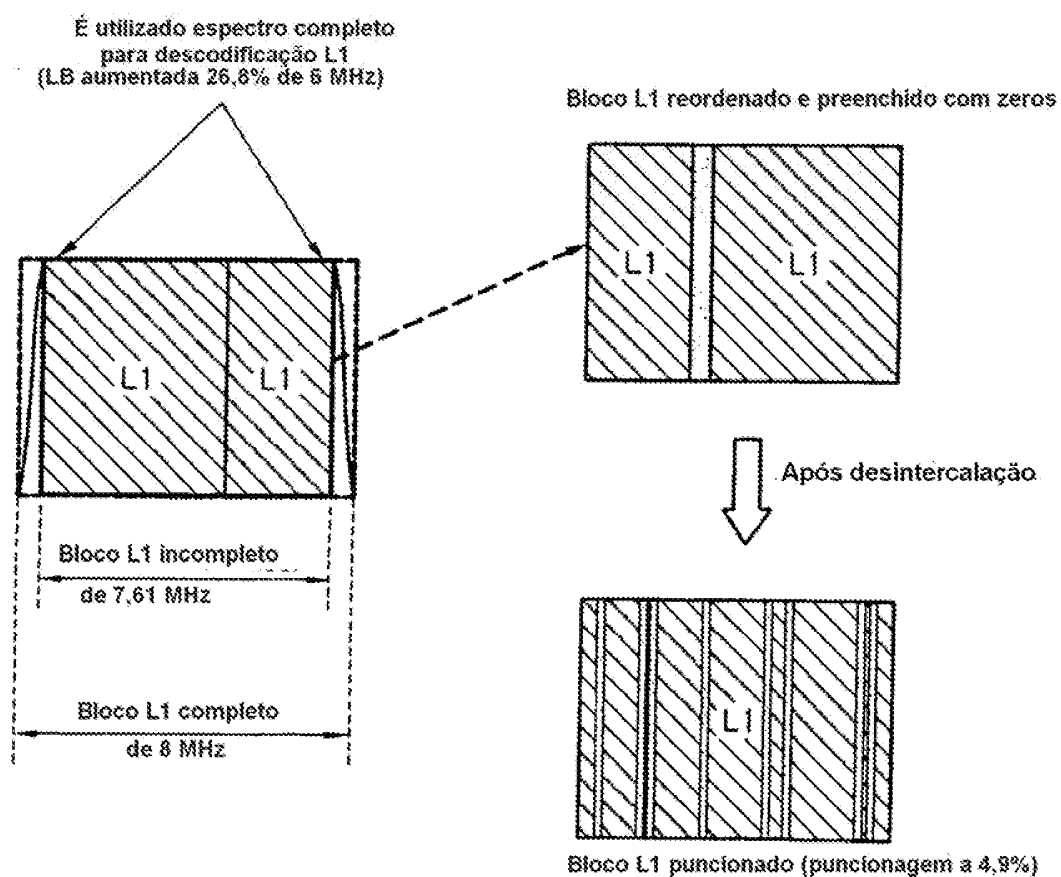


Fig. 24

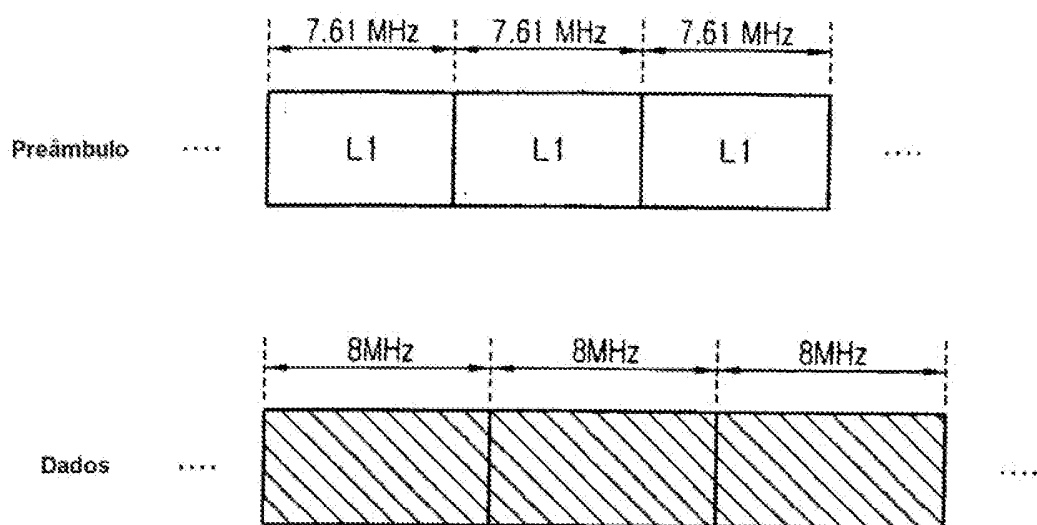


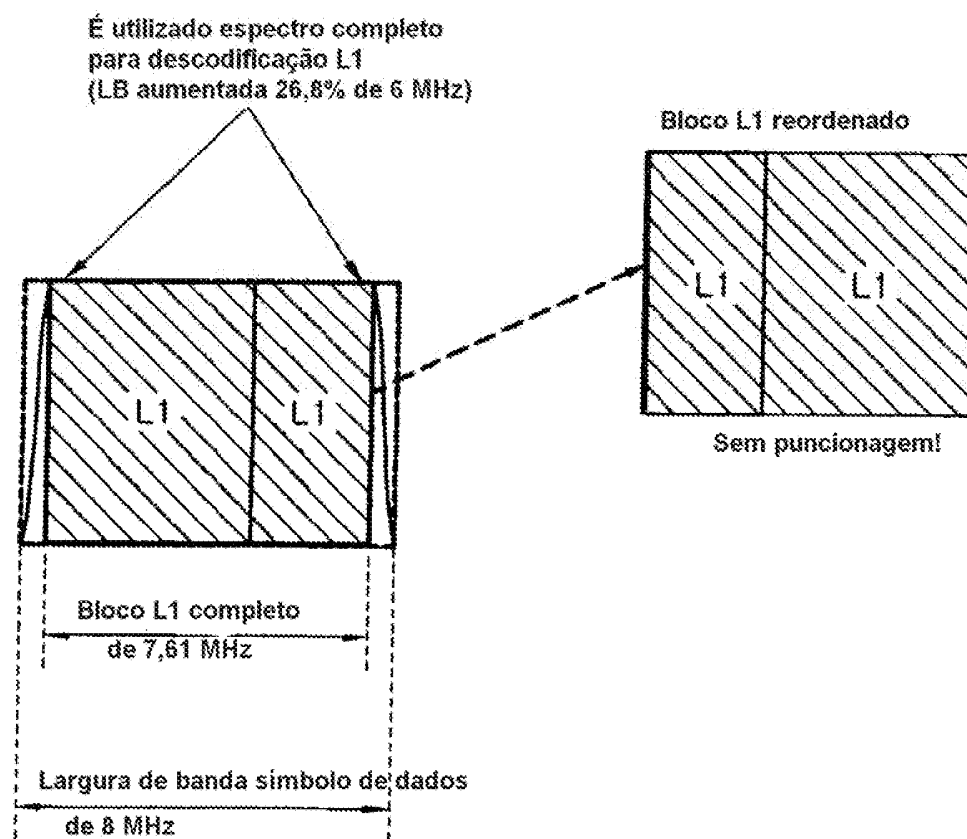
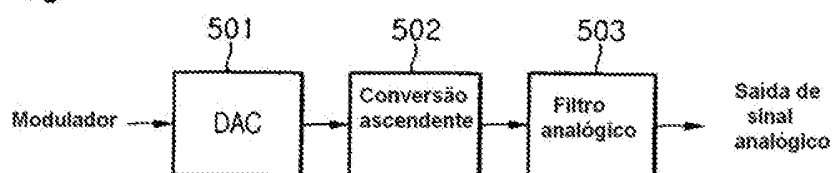
Fig. 25**Fig. 26****Fig. 27**

Fig. 28

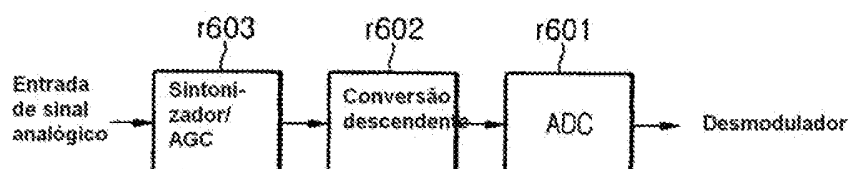


Fig. 29.

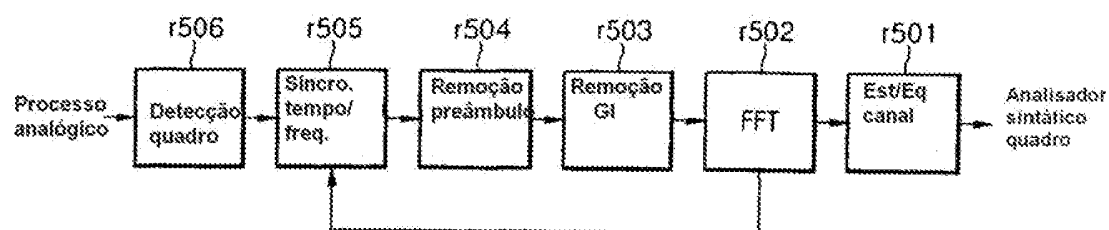


Fig. 30

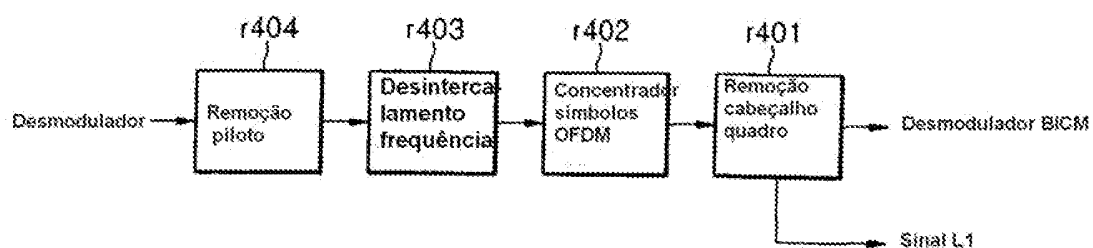


Fig. 31

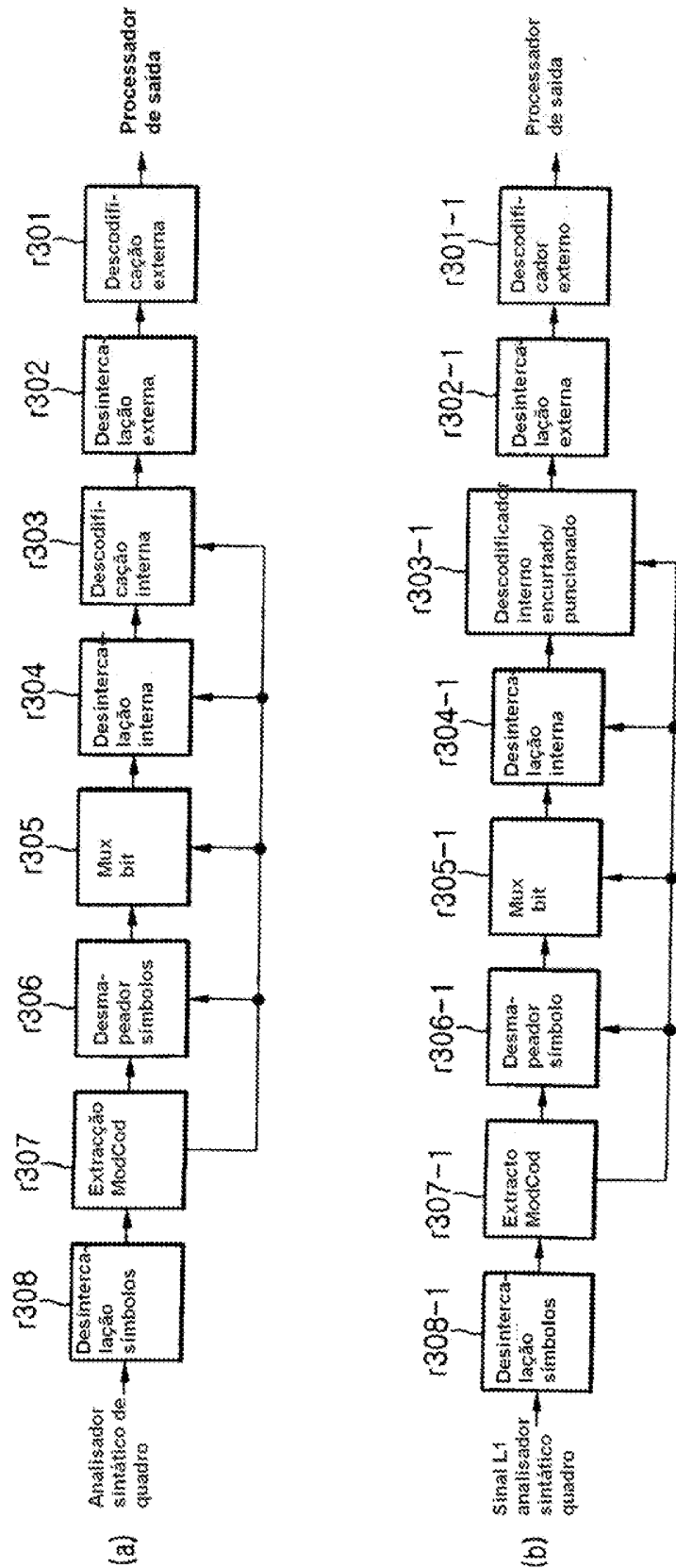


Fig. 32

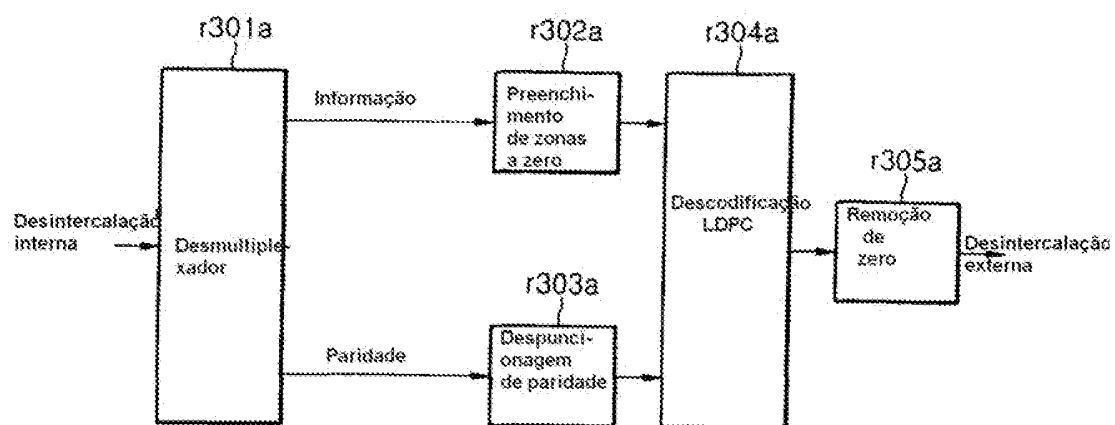


Fig. 33

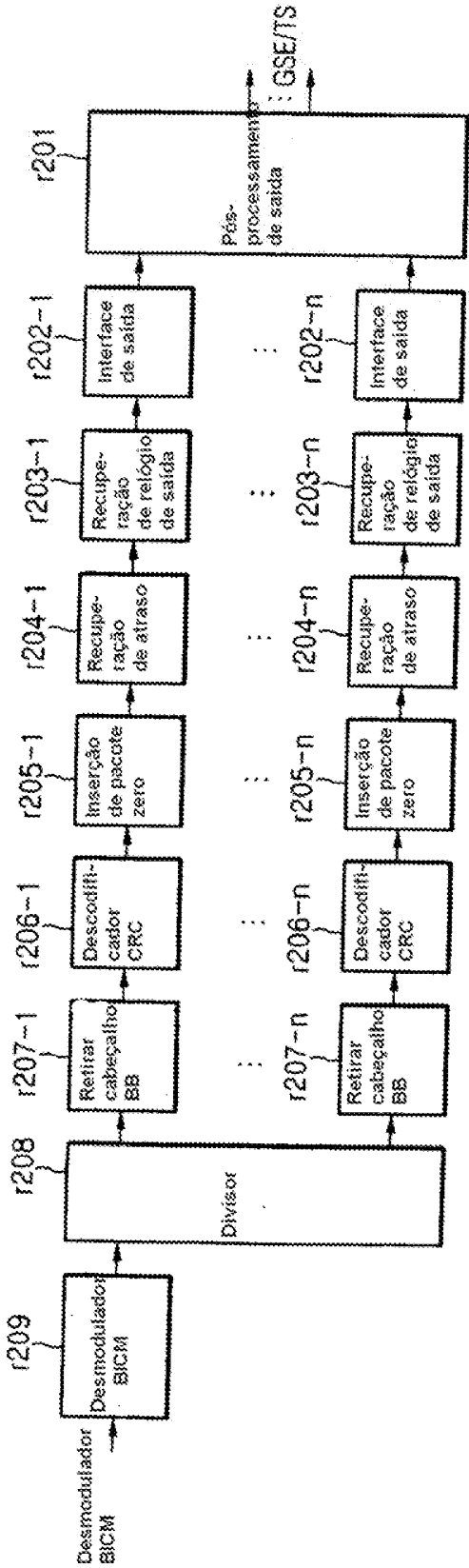


Fig. 34

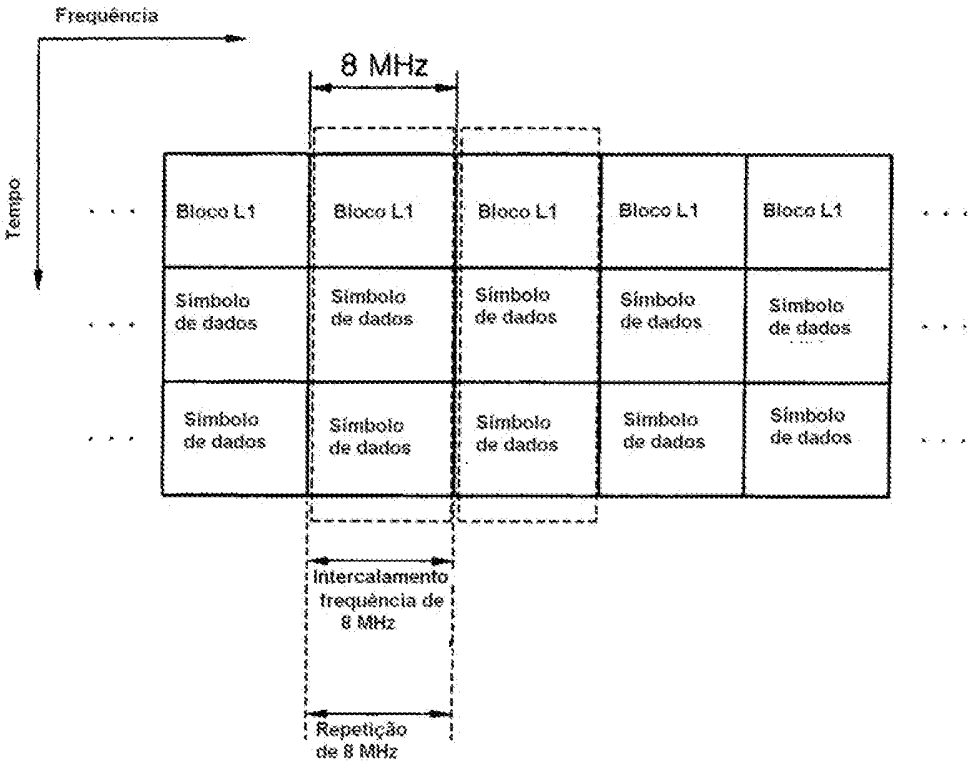


Fig. 35

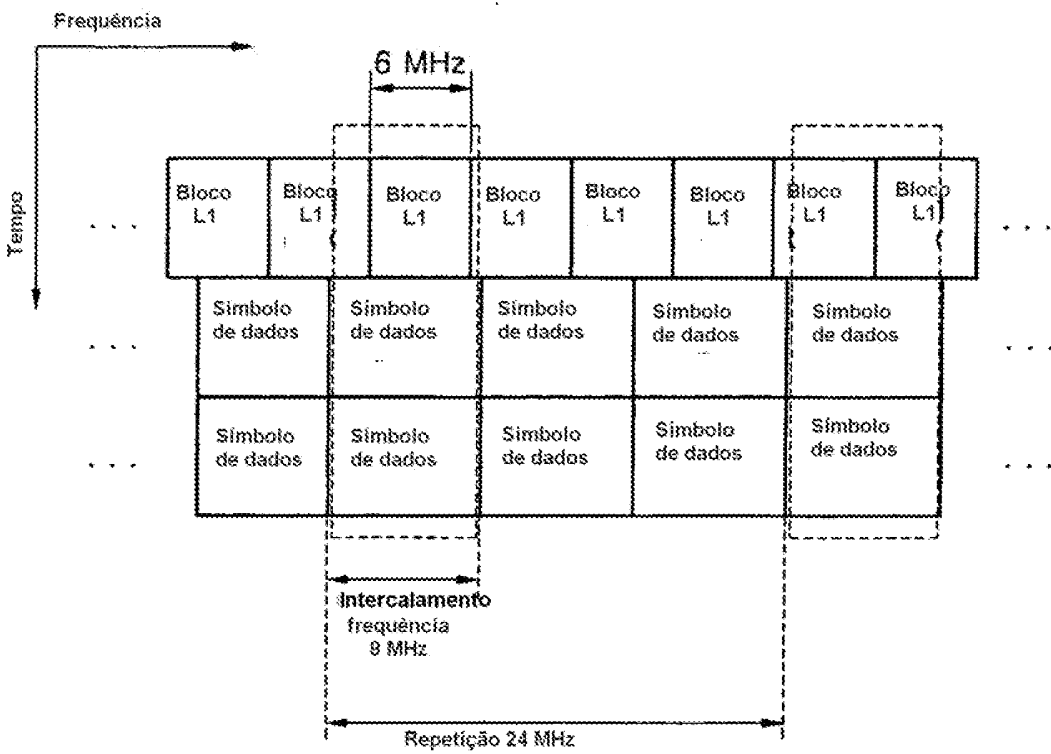


Fig. 36

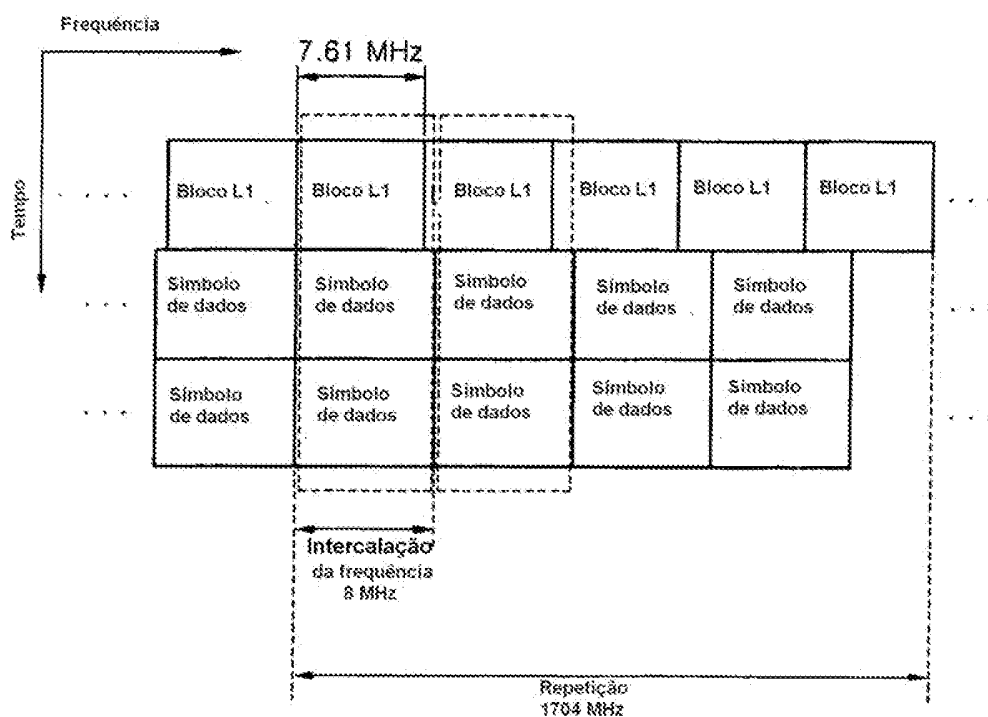


Fig. 37

Campo	Bits	
L1_span	12	Número de portadoras fracionadas por bloco L1 dentro de um símbolo OFDM (Max = 7,61 MHz)
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_plp	8	número de PLPs
num_notch	5	número de bandas de entalhe
for dslice {		
chbon_index	3	índice de canal ligado
dslice_start	9	início da fatia de dados dentro de um canal (8 MHz)
dslice_width	9	largura da fatia de dados
}		
for plp {		
dslice_id	8	ID em fatias de dados
plp_id	8	PLP ID
plp_type	1	tipo de PLP (comum/dados)
plp_payload_type	5	tipo de carga útil PLP (TS, GS,...)
}		
for notch {		
chbon_index	3	índice de canal ligado
notch_start	9	início da banda de entalhe dentro de um canal (8 MHz)
notch_width	9	largura da banda de entalhe
}		
gi	1	modo de intervalo de guarda
sframe_id	16	ID do superquadro
frame_id	16	ID do quadro
reserved	0	rfu
crc32	32	CRC32
Total	11760	

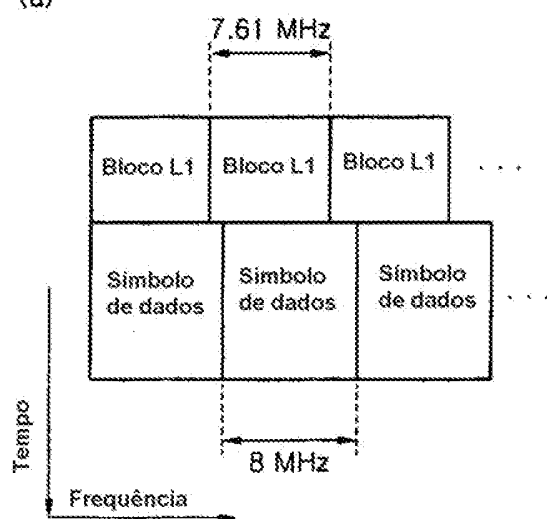
O número de bits de informação L1 varia de acordo com várias configurações/condições

Fig. 38

Info L1 (bits)	11760	 Dimensão máx.
Bloco L1 (bits)	23520	
Símbolos 16-QAM	5880	
Portadoras totais	3408	
Distância SP do preâmbulo	6	
Portadoras de dados	2840	
LDPC curto	1.45	
Símbolos OFDM	2.07	

Fig. 40

(a)



(b)

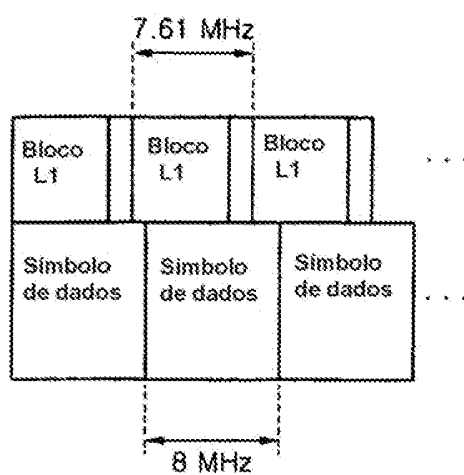


Fig. 41

Field	Bits	
L1_column	9	Número de portadoras fraccionadas por bloco L1 dentro de um símbolo OFDM (máx = 7,61 MHz)
L1_row	3	número de símbolos OFDM fraccionados por bloco L1
num_chbon	3	número de canais ligados
num_plp	8	número de fatias de dados
num_dslice	8	número de PLP's
num_notch	5	número de bandas de entalhe
for dslice {		
chbon_index	3	índice de canal ligado
dslice_start	9	início da fatia de dados dentro de um canal (8 MHz)
dslice_width	9	largura da fatia de dados
}		
for plp {		
dslice_id	8	ID em fatias de dados
plp_id	8	PLP ID
plp_type	1	tipo de PLP (comum/dados)
plp_payload_type	5	tipo de carga útil PLP (TS, GS,...)
}		
for notch {		
chbon_index	3	índice de canal ligado
notch_start	9	início da banda de entalhe dentro de um canal (8 MHz)
notch_width	9	largura da banda de entalhe
}		
gi	1	modo de intervalo de guarda
sframe_id	16	ID do superquadro
frame_id	16	ID do quadro
reserved	16	rfu
crc32	32	CRC32
Total	11776	

O número de bits de informação L1 varia de acordo com várias configurações/condições

Fig. 42

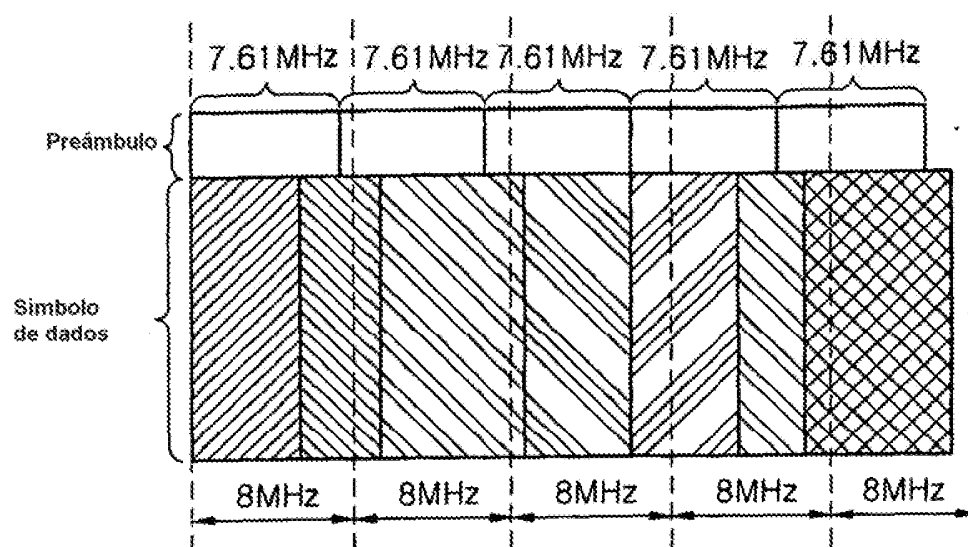


Fig. 43

Tamanho bloco LDPC	Modulação (bps/Hz)	Símbolos QAM	Informação complementar ModCod
64800	4	16200	0.28%
	6	10800	0.42%
	8	8100	0.56%
	10	6480	0.69%
	12	5400	0.83%
16200	4	4050	1.11%
	6	2700	1.67%
	8	2025	2.22%
	10	1620	2.78%
	12	1350	3.33%

Fig. 44

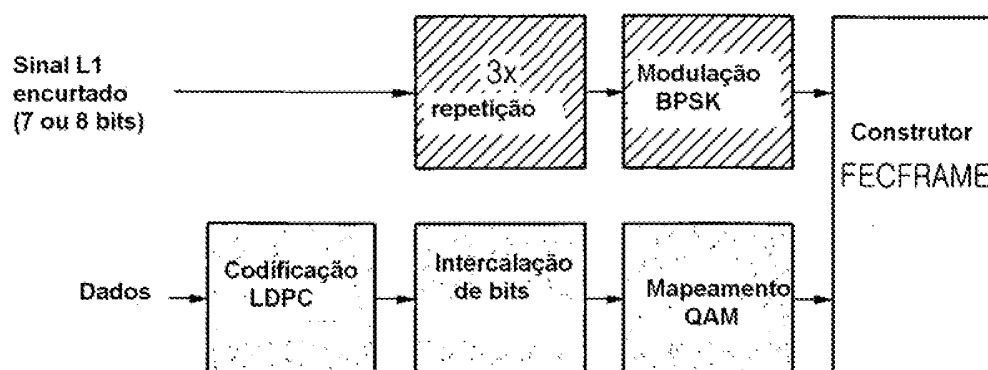


Fig. 45

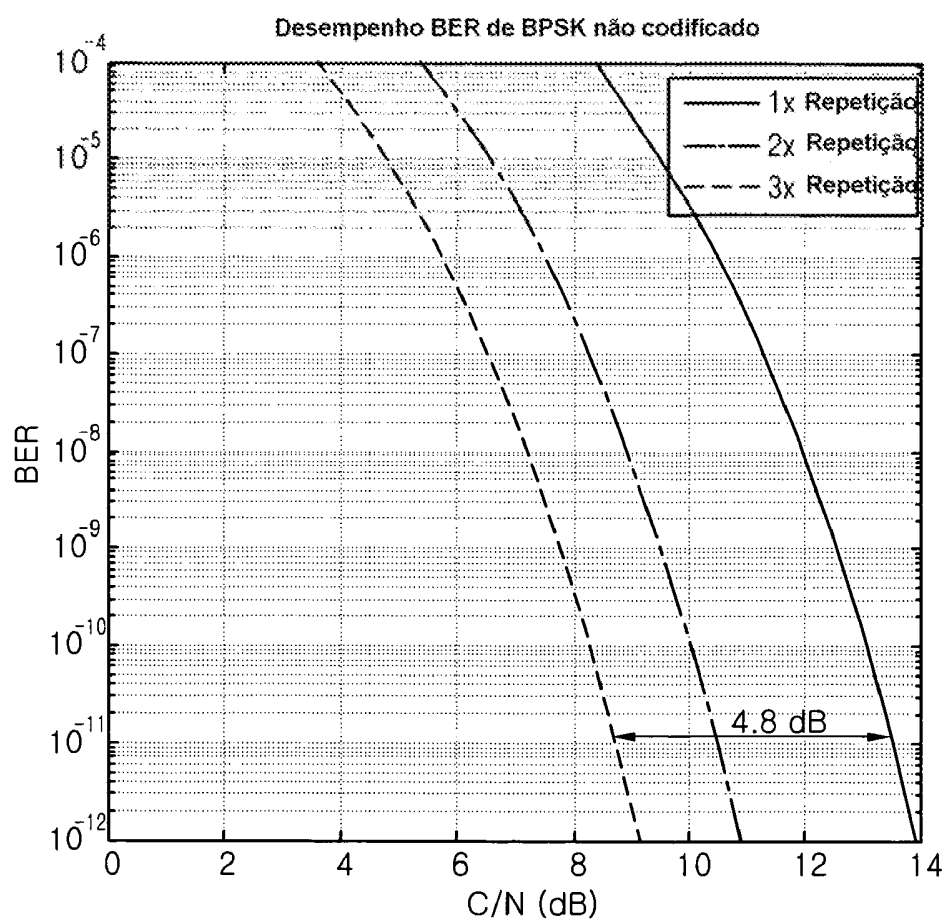


Fig. 46

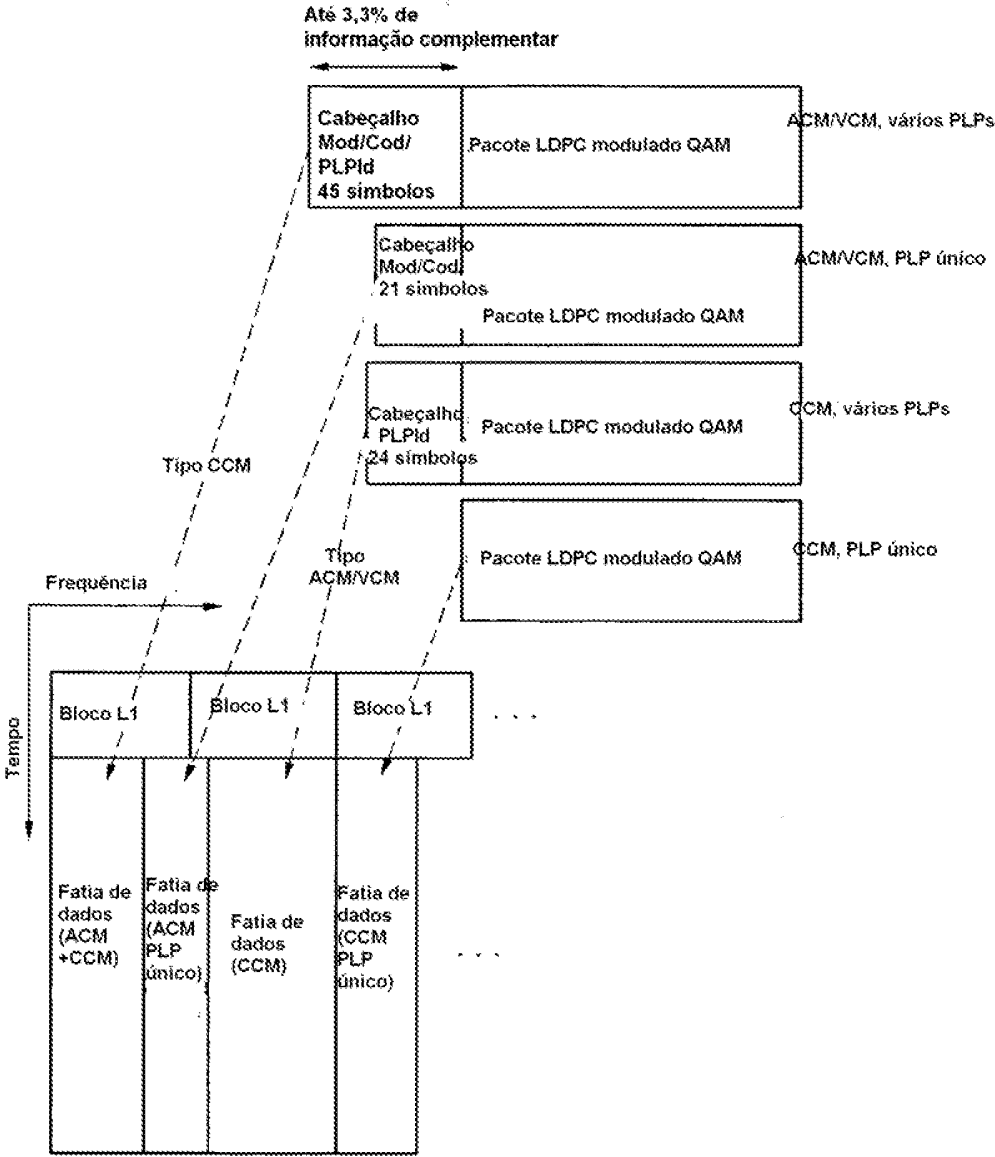


Fig. 47

Campo	Bits	
L1_size	14	Tamanho do bloco L1 (bits)
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_plp	8	número de PLPs
num_notch	5	número de bandas de entalhe
for dslice {		
chbon_index	3	índice de canal ligado
dslice_start	9	início de fatia de dados dentro de um canal (8 MHz)
dslice_width	9	largura da fatia de dados
dslice_time_intrlv	1	intercalação no tempo ligado/desligado
dslice_type	1	tipo de fatia de dados (CCM, ACM/VCM)
for plp {		
dslice_id	8	ID em fatias de dados
plp_id	8	ID PLP
plp_type	1	tipo PLP
plp_payload_type	5	carga útil PLP
if dslice_type=CCM {		
plp_mod	3	tipo de modulação PLP
plp_fec_type	1	tipo PLP FEC (longo/curto)
plp_cod	3	taxa de código PLP
}		
}		
}		
for notch {		
chbon_index	3	índice de canal ligado
notch_start	9	início da banda de entalhe dentro de um canal (8 MHz)
notch_width	9	largura da banda de entalhe
}		
gi	1	modo de intervalo de guarda
sframe_id	16	ID de superquadro
frame_id	16	ID do quadro
reserved	16	rfu
crc32	32	CRC32
Total	14082	

O tamanho do bloco L1 é transmitido no primeiro bloco LDPC (comprimento mais curto fixo = 192 bits)

Intercalação no tempo ligado/desligado para suportar modo de latência curto

Tipo de fatia de dados para redução da informação complementar da sinalização L1

Os campos Mod/Cod são transmitidos no preâmbulo somente para tipo CCM

Fig. 48

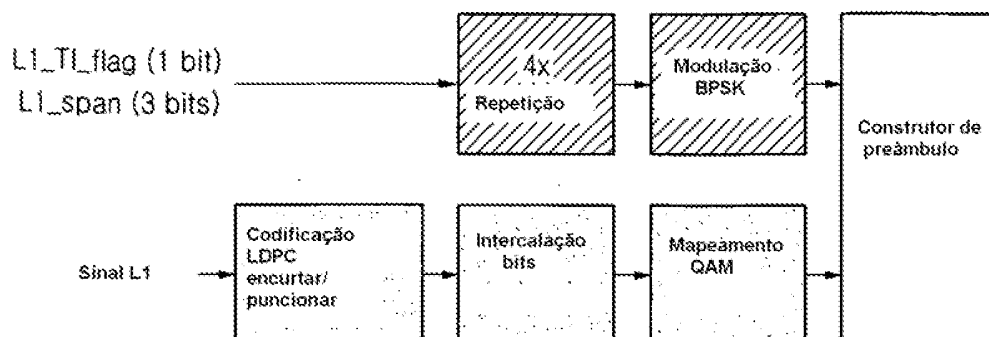


Fig. 49

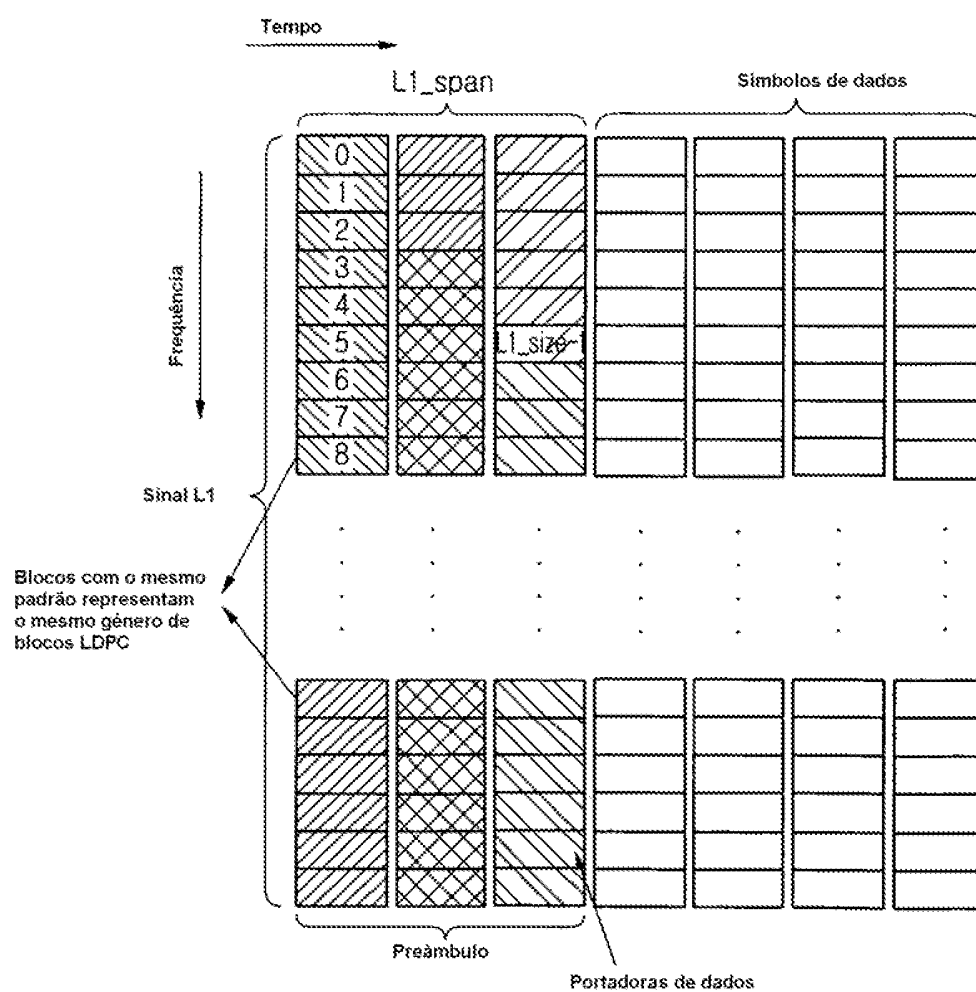


Fig. 50

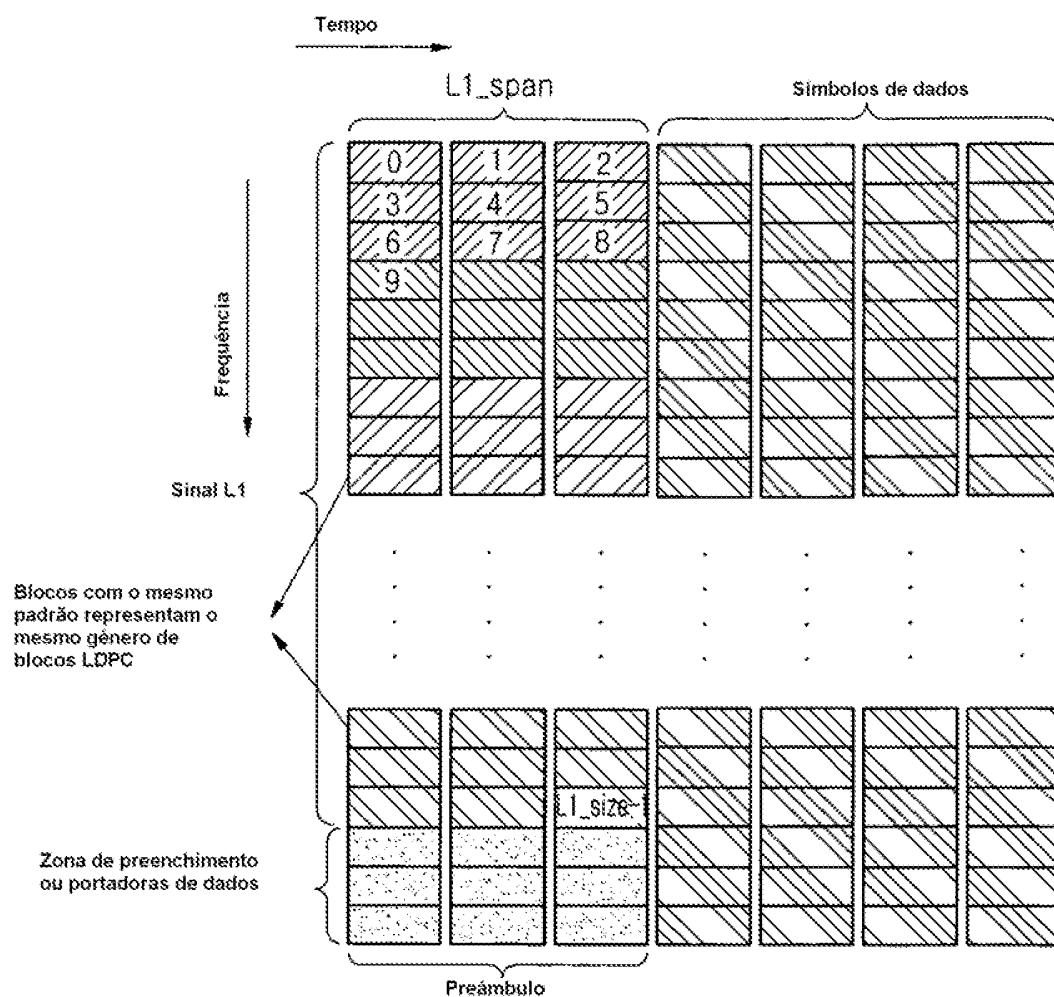


Fig. 51

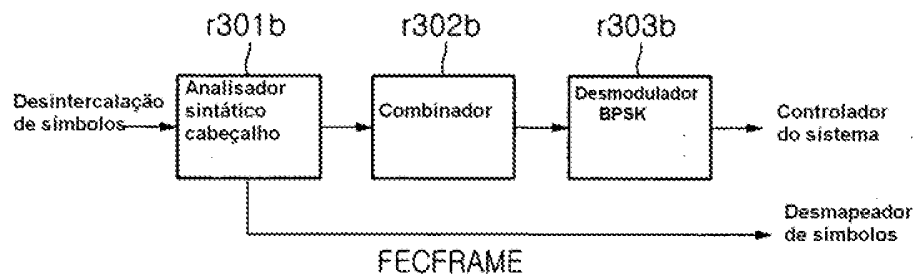


Fig. 52

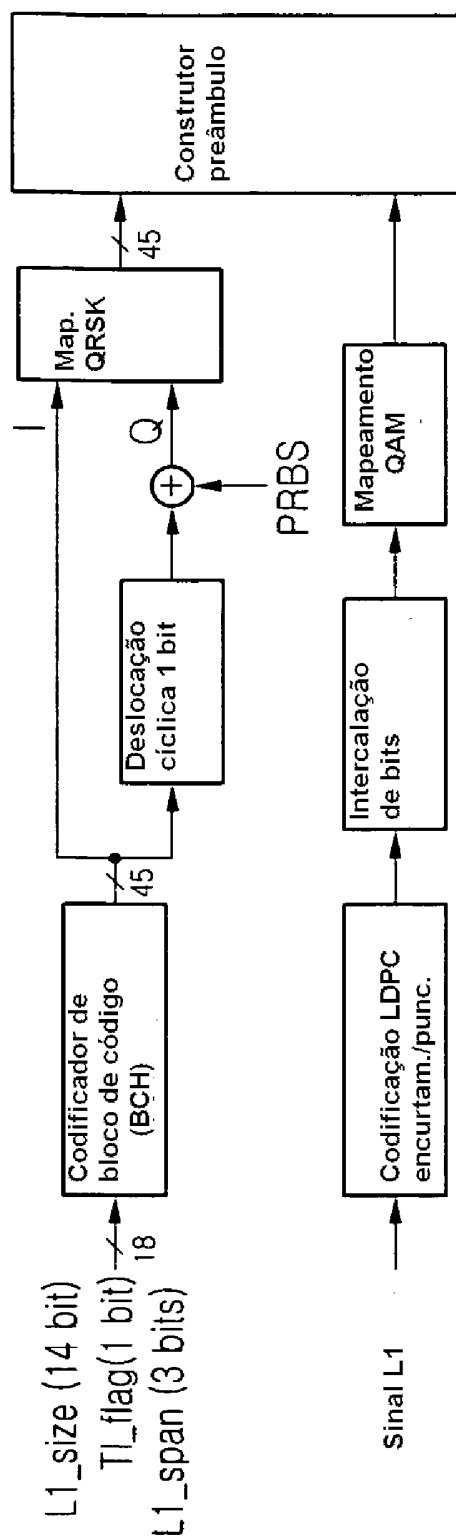


Fig. 53

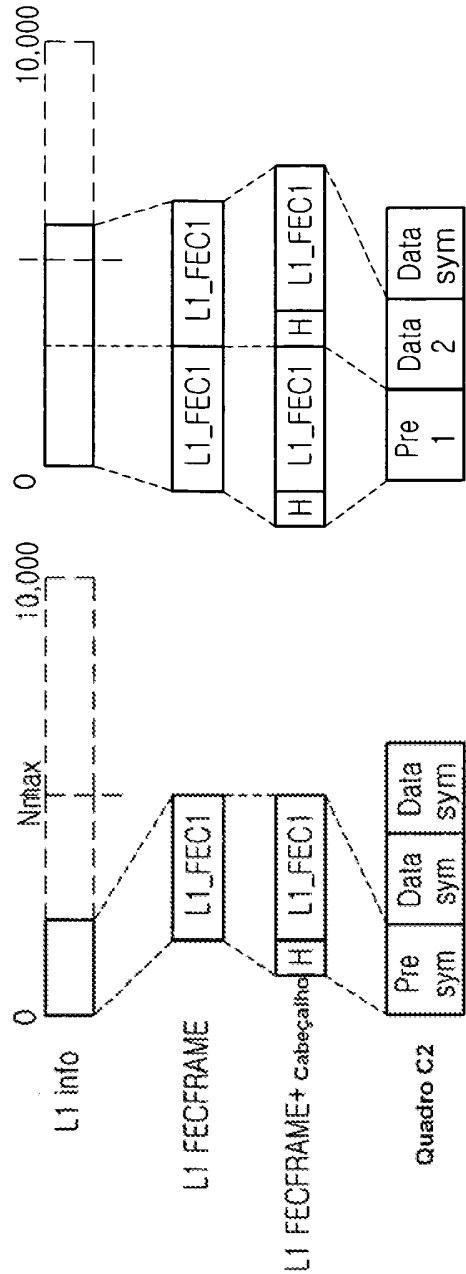


Fig. 54

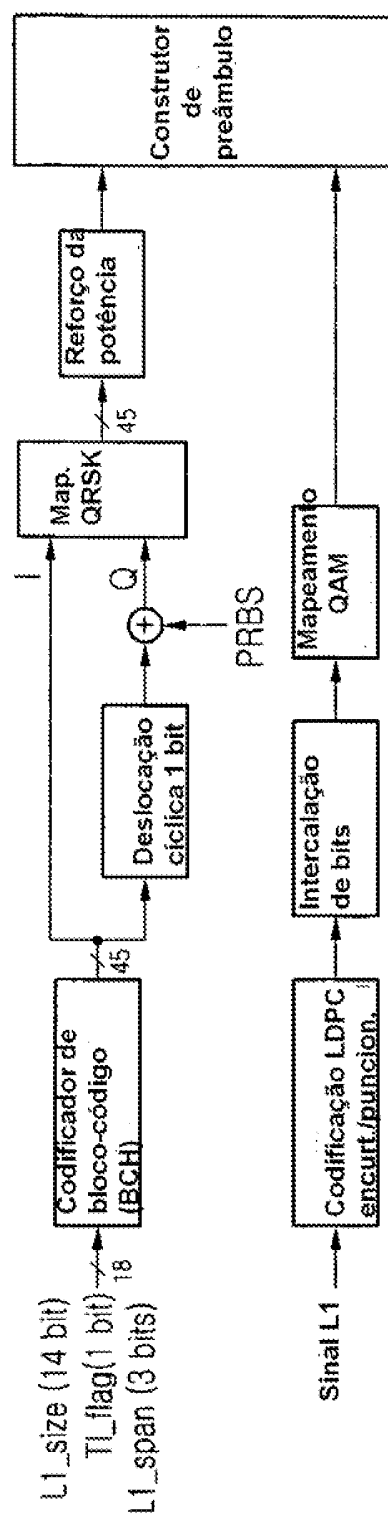


Fig. 55

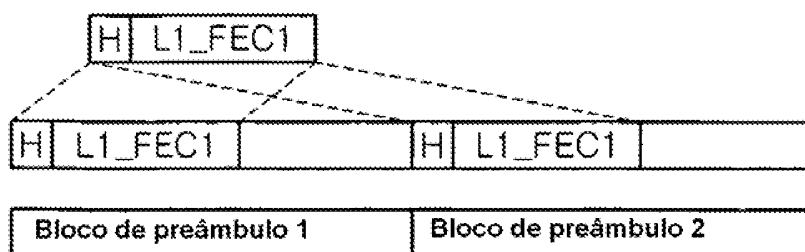


Fig. 56

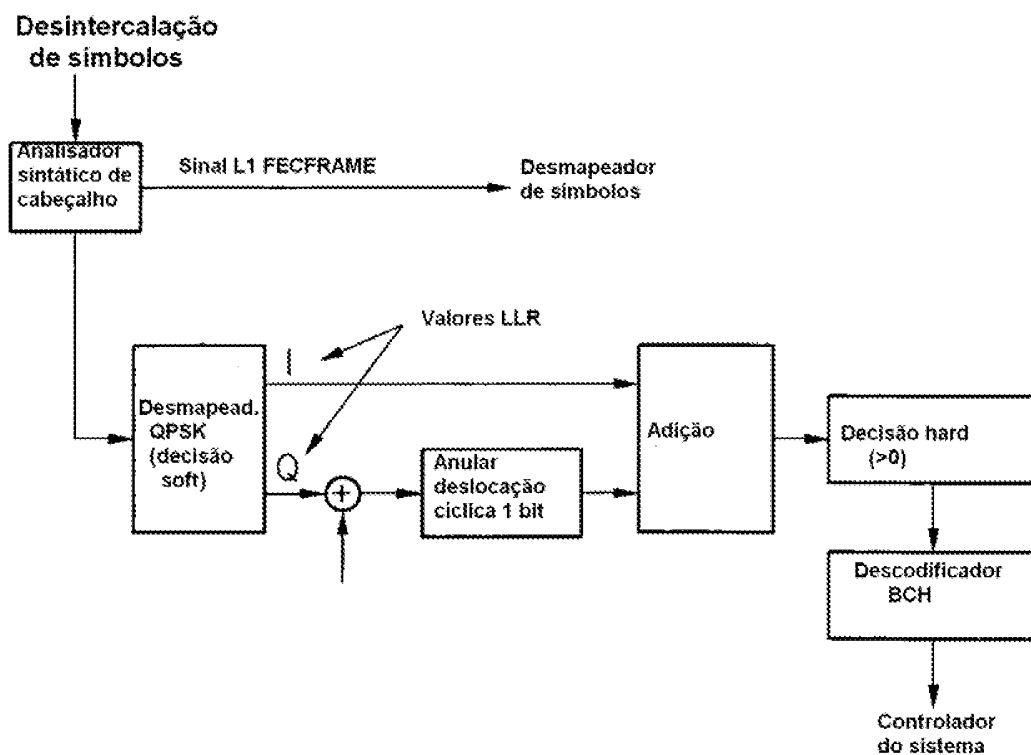


Fig. 57

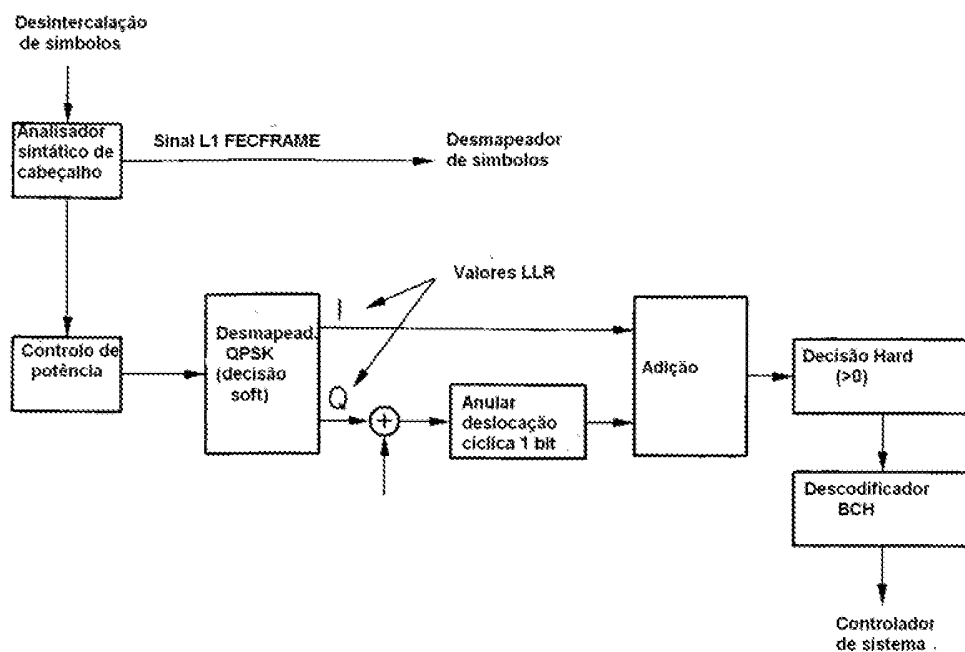


Fig. 58

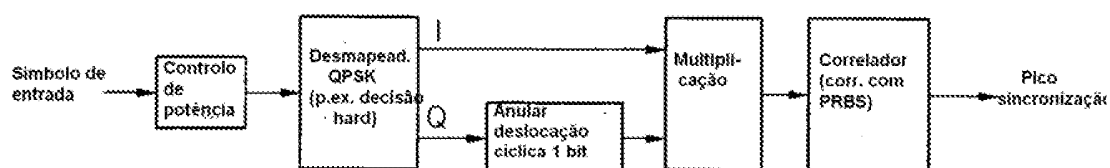


Fig. 59

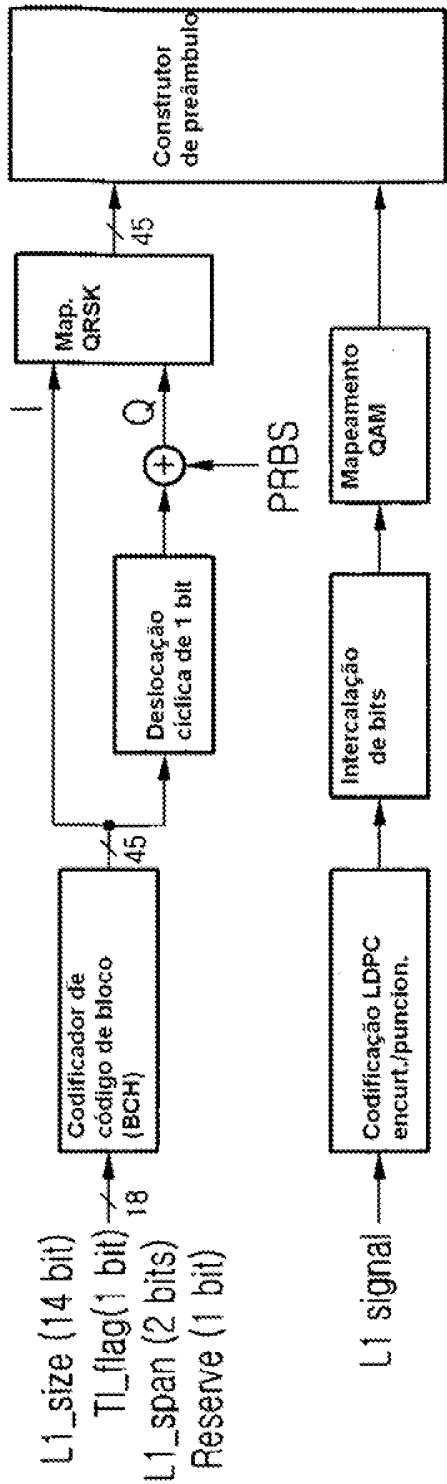


Fig. 60

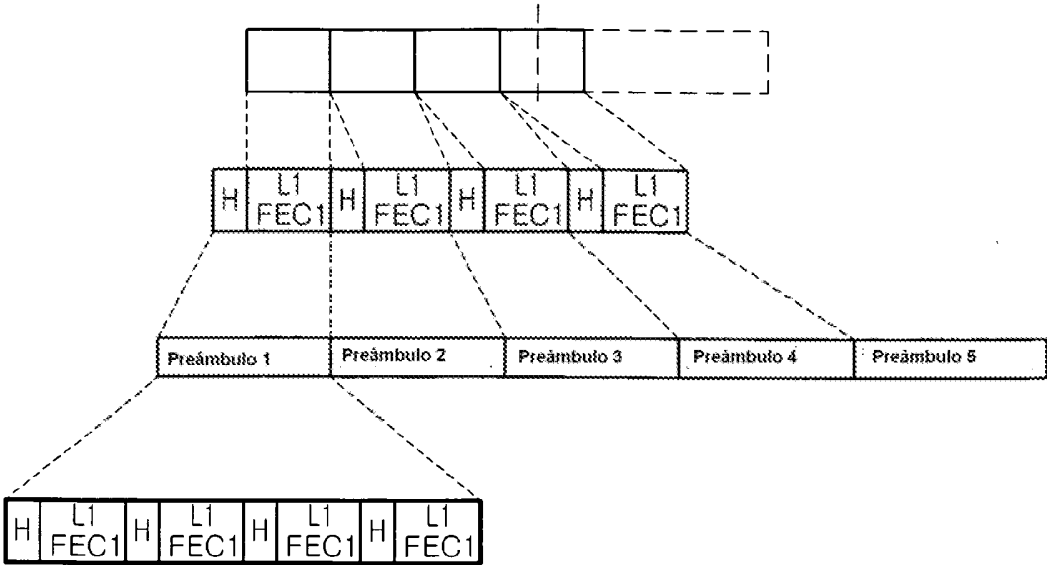


Fig. 61

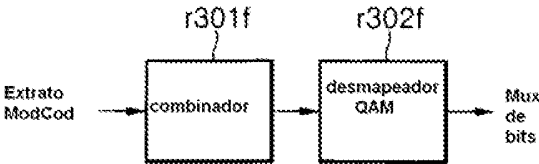


Fig. 62

L1_size	14	Tamanho do bloco L1 (bits)
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_plp	8	número de PLPs
num_notch	5	número de bandas de entalhe
for dslice {		
chbon_index	3	índice de canal ligado
dslice_start	9	início de fatia de dados dentro de um canal (8 MHz)
dslice_width	9	largura da fatia de dados
dslice_time_intrlv	1	intercalação no tempo ligado/desligado
dslice_type	1	tipo de fatia de dados (CCM, ACM/VCM)
}		
for plp {		
dslice_id	8	ID em fatias de dados
plp_id	8	ID PLP
plp_type	1	tipo PLP (comum/dados)
plp_payload_type	5	carga útil PLP (TS, GS, ...)
if dslice_type=CCM {		
plp_mod	3	tipo de modulação PLP
plp_fec_type	1	tipo PLP FEC (longo/curto)
plp_cod	3	taxa de código PLP
plp_start	21	endereço de início PLP
}		
}		
for notch {		
chbon_index	3	índice de canal ligado
notch_start	9	início da banda de entalhe dentro de um canal (8 MHz)
notch_width	9	largura da banda de entalhe
}		
gi	1	modo de intervalo de guarda
sframe_id	16	ID de superquadro
frame_id	16	ID do quadro
reserved	16	rfu
crc32	32	CRC32
Total	19458	

O número de bits de informação L1
varia de acordo com varias configurações/condições

Fig. 62

L1_size	14	Tamanho do bloco L1 (bits)
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_plp	8	número de PLPs
num_notch	5	número de bandas de entalhe
for dslice {		
chbon_index	3	índice de canal ligado
dslice_start	9	início de fatia de dados dentro de um canal (8 MHz)
dslice_width	9	largura da fatia de dados
dslice_time_intrlv	1	intercalação no tempo ligado/desligado
dslice_type	1	tipo de fatia de dados (CCM, ACM/VCM)
}		
for plp {		
dslice_id	8	ID em fatias de dados
plp_id	8	ID PLP
plp_type	1	tipo PLP (comum/dados)
plp_payload_type	5	carga útil PLP (TS, GS, ...)
if dslice_type=CCM {		
plp_mod	3	tipo de modulação PLP
plp_fec_type	1	tipo PLP FEC (longo/curto)
plp_cod	3	taxa de código PLP
plp_start	14	endereço de início PLP
}		
}		
for notch {		
chbon_index	3	índice de canal ligado
notch_start	9	início da banda de entalhe dentro de um canal (8 MHz)
notch_width	9	largura da banda de entalhe
}		
gi	1	modo de intervalo de guarda
sframe_id	16	ID de superquadro
frame_id	16	ID do quadro
reserved	16	rfu
crc32	32	CRC32
Total	17666	

O número de bits de informação L1
varia de acordo com varias configurações/condições

Fig. 64

Comprimento LDPC	Tipo QAM	Símbolos QAM
64800	16 QAM	16200
64800	64 QAM	10800
64800	256 QAM	8100
64800	1024 QAM	6480
64800	4096 QAM	5400
16200	16 QAM	4050
16200	64 QAM	2700
16200	256 QAM	2025
16200	1024 QAM	1620
16200	4096 QAM	1350

Fig. 65

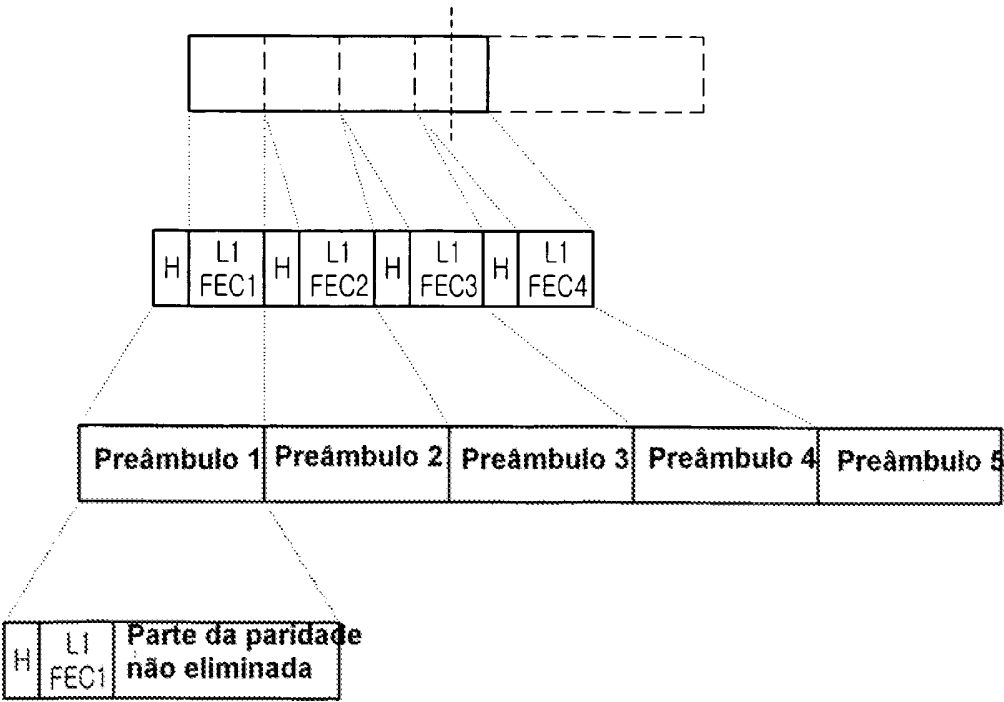


Fig. 66

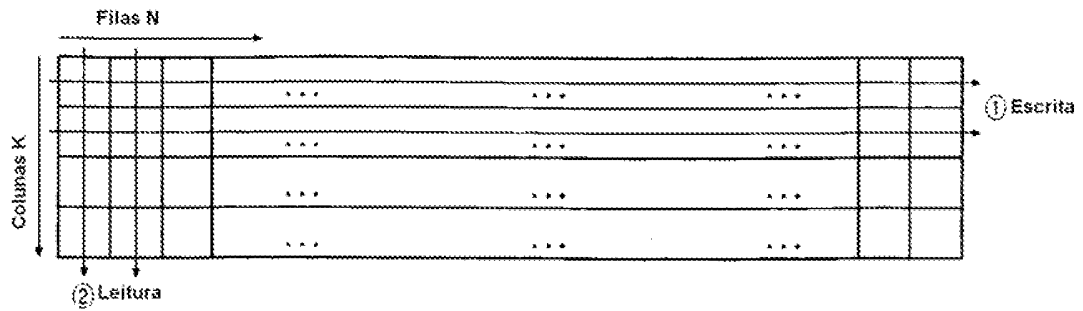
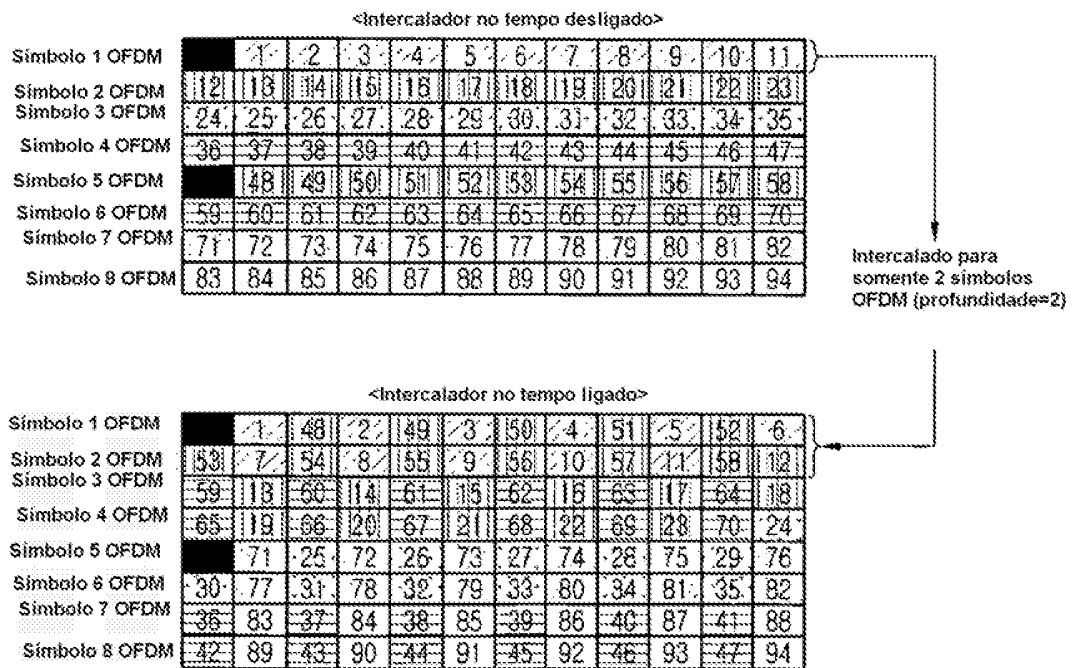


Fig. 67



Pilotos dispersos

Fig. 68

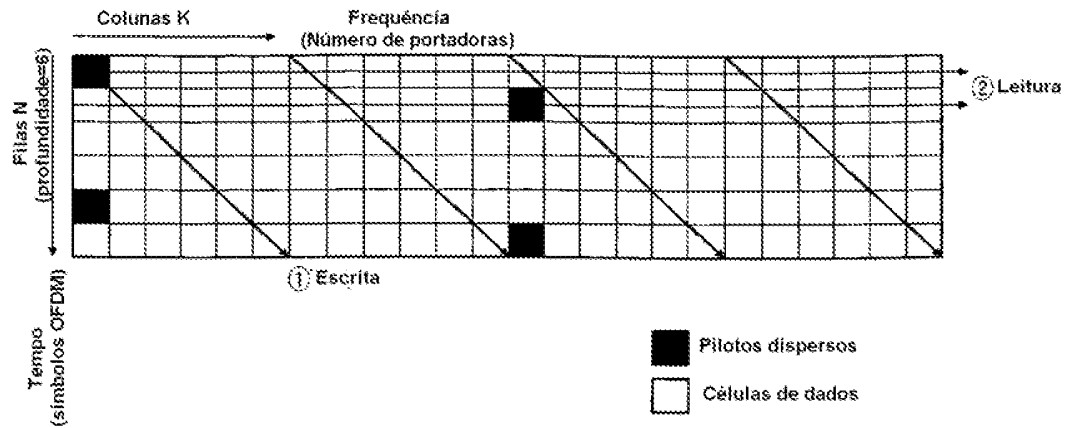


Fig. 69

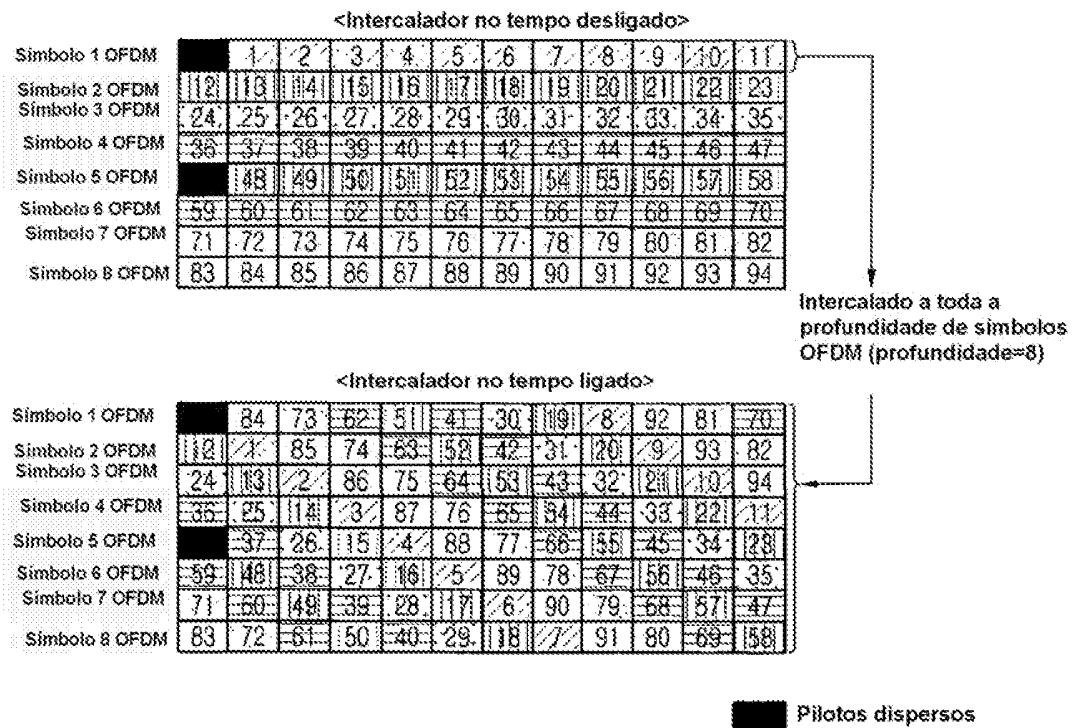


Fig. 70

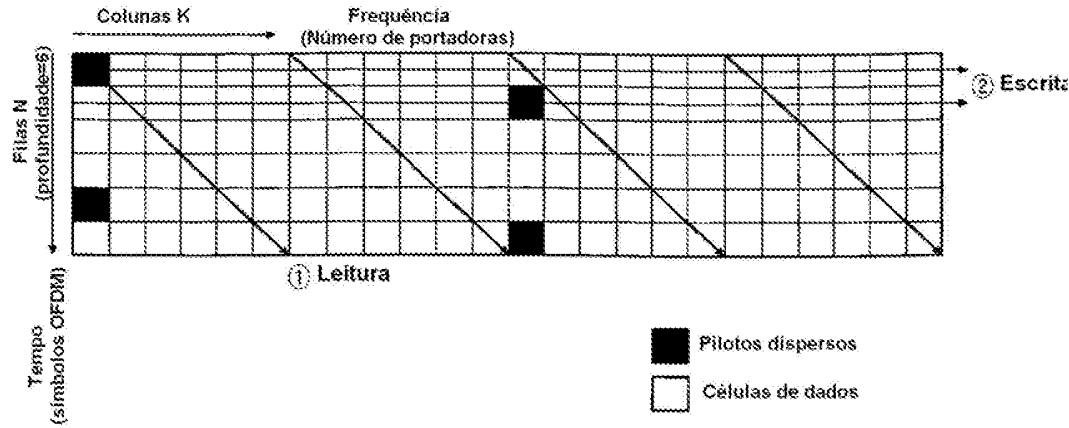


Fig. 71

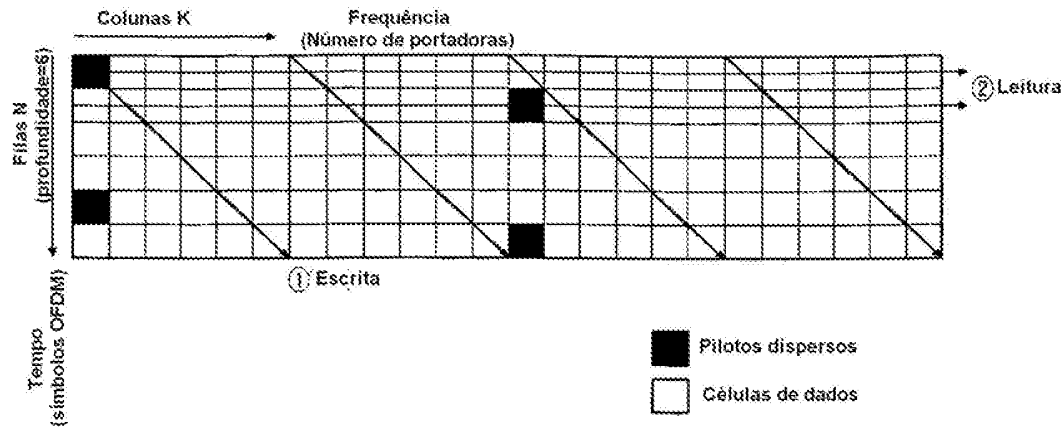


Fig. 72

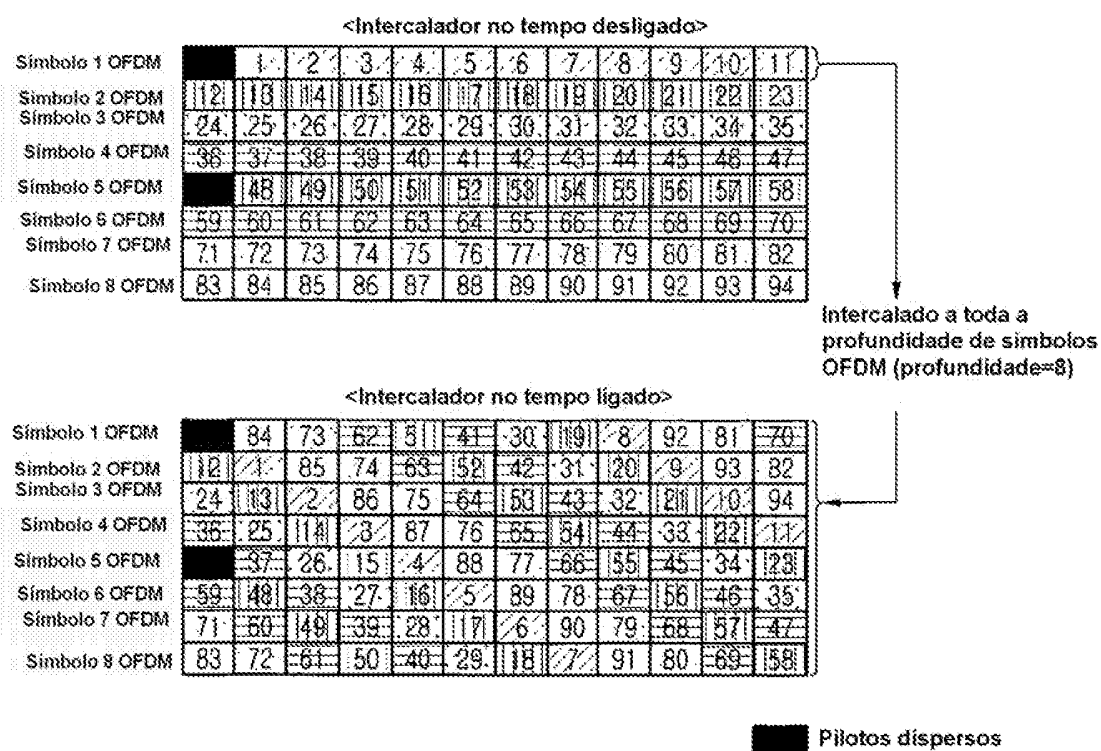
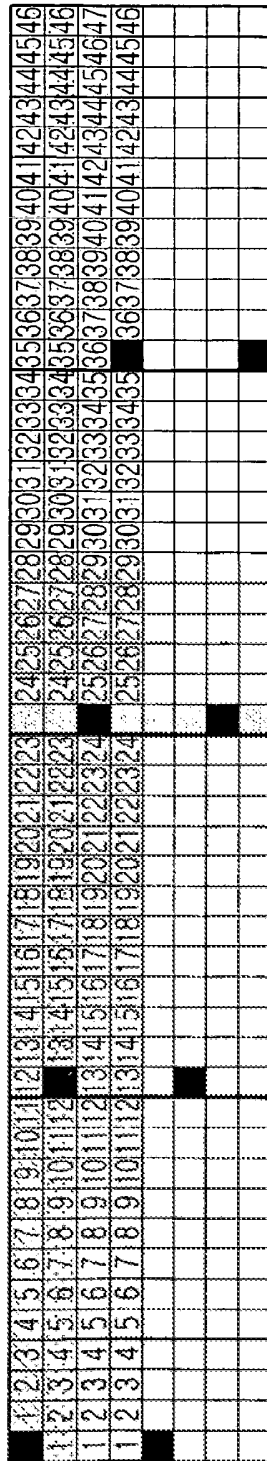
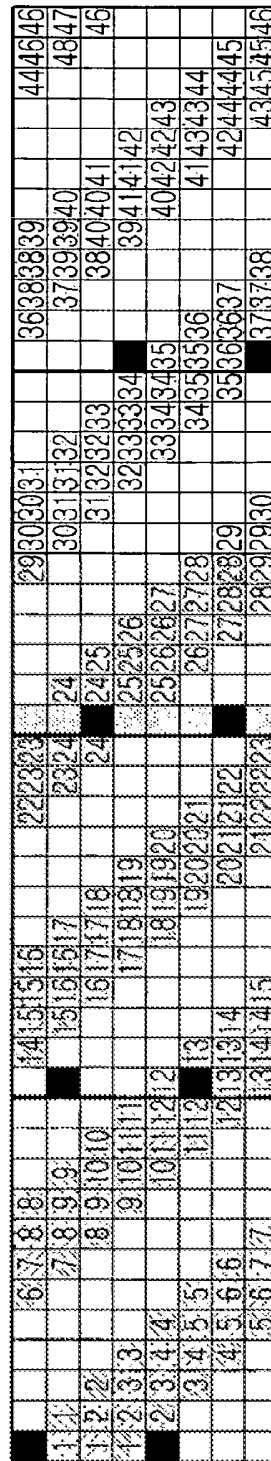


Fig. 73

Após aplicar intercalação
no tempoAntes de aplicar intercalação
no tempo

RA=0; CA=0;

while I<nCELL loop

 If addr = pilot position

 RA = [RA+1] mod NT;

 CA = [CA+1] mod ND;

 end if ;

WRITE:

RA = [RA+1] mod NT;

CA = [CA+1] mod ND;

 if CA = 0

 RA = RA+1;

 end if ;

end loop;

■ Piloto disperso

□ Piloto contínuo

RA = Endereço da linha

CA = Endereço da coluna

nCell = # de células dentro do
bloco TI

Informação complementar
da memória = 2%

Fig. 74

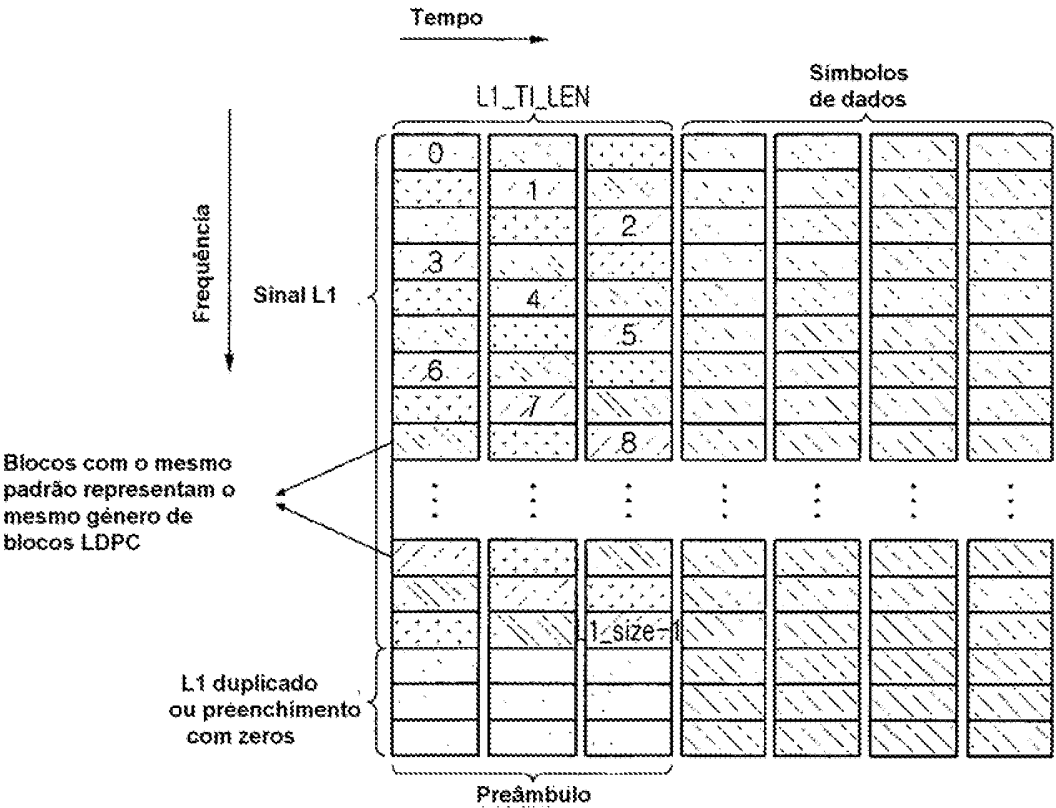


Fig. 75

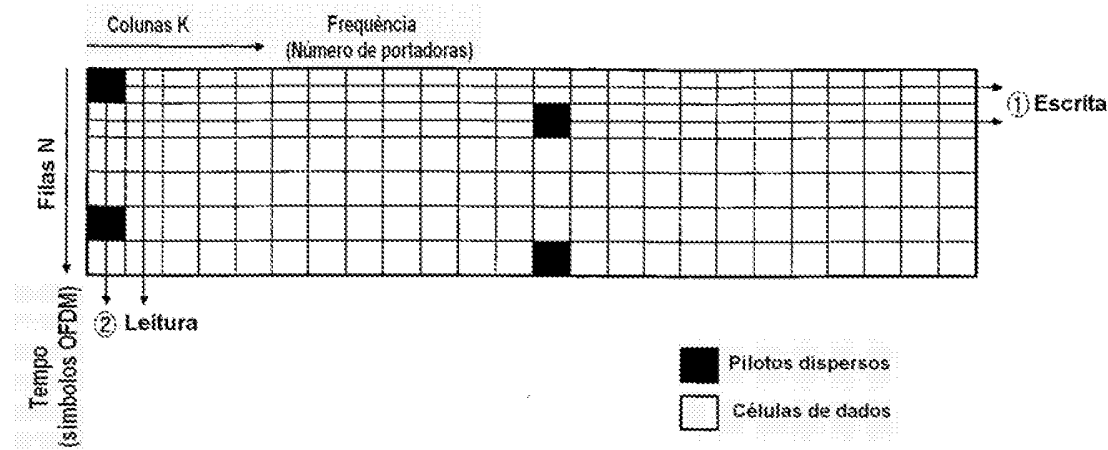


Fig. 76

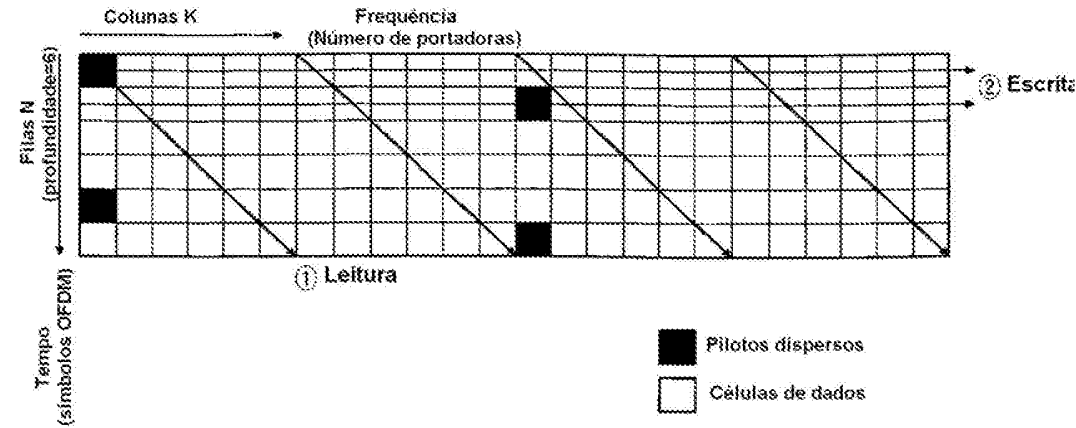


Fig. 80

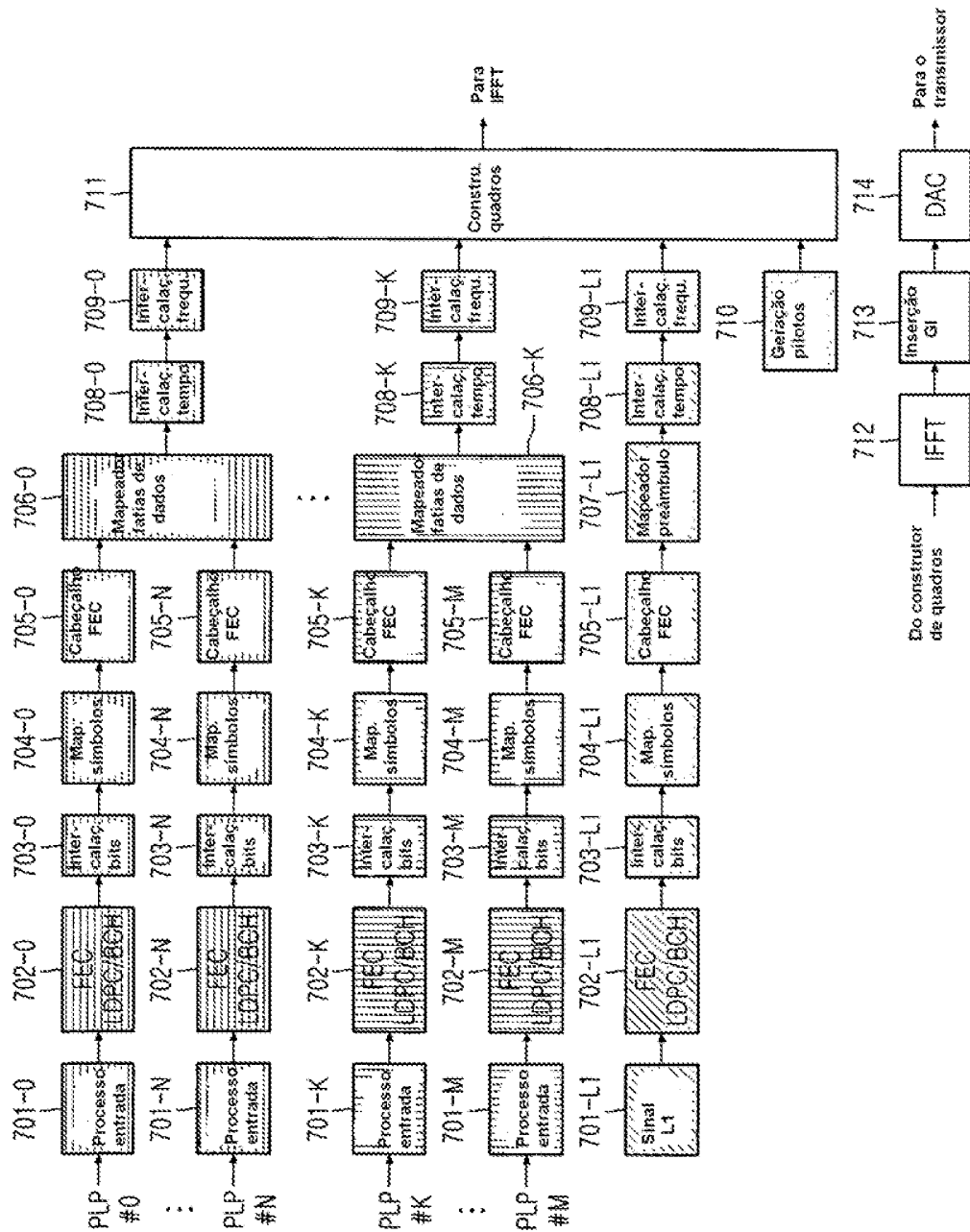


Fig. 81

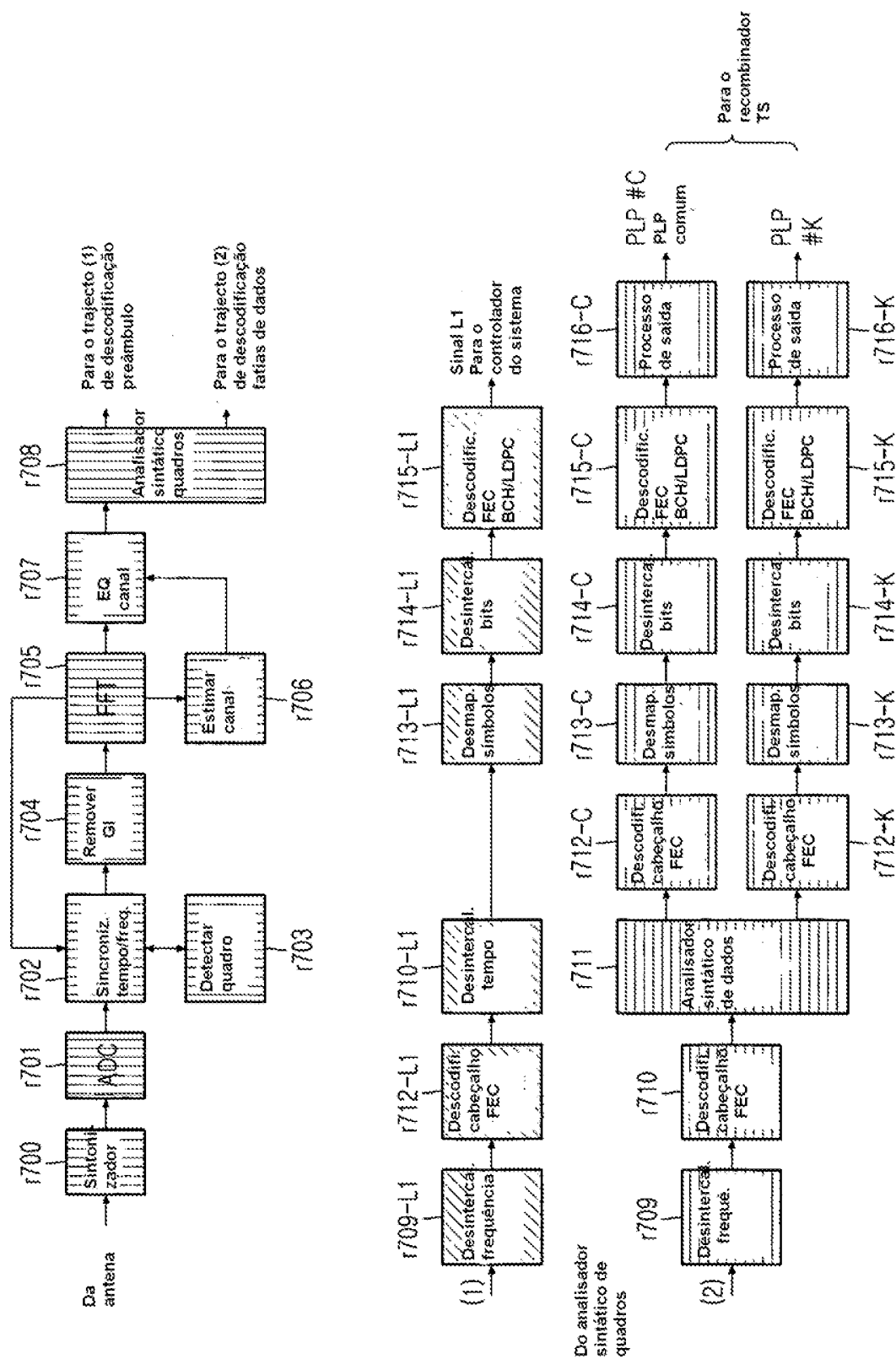


Fig. 82

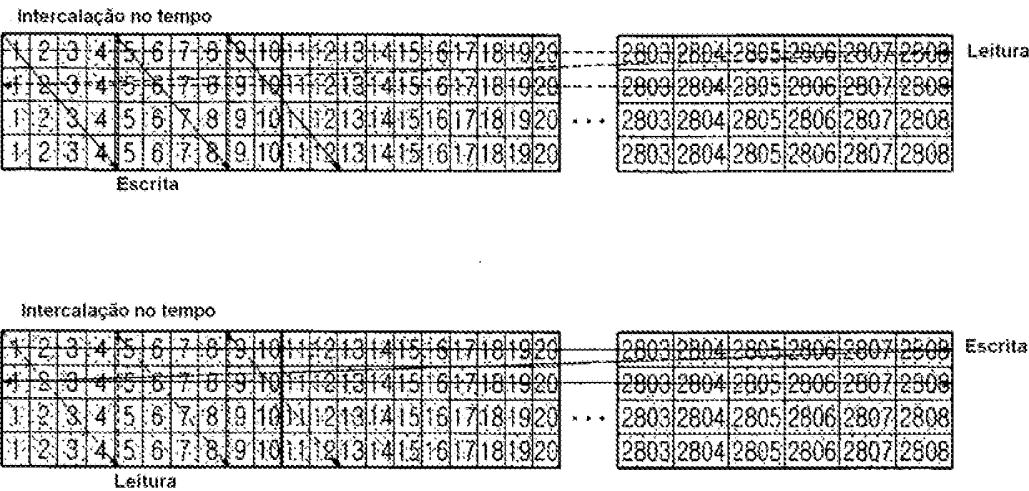


Fig. 83

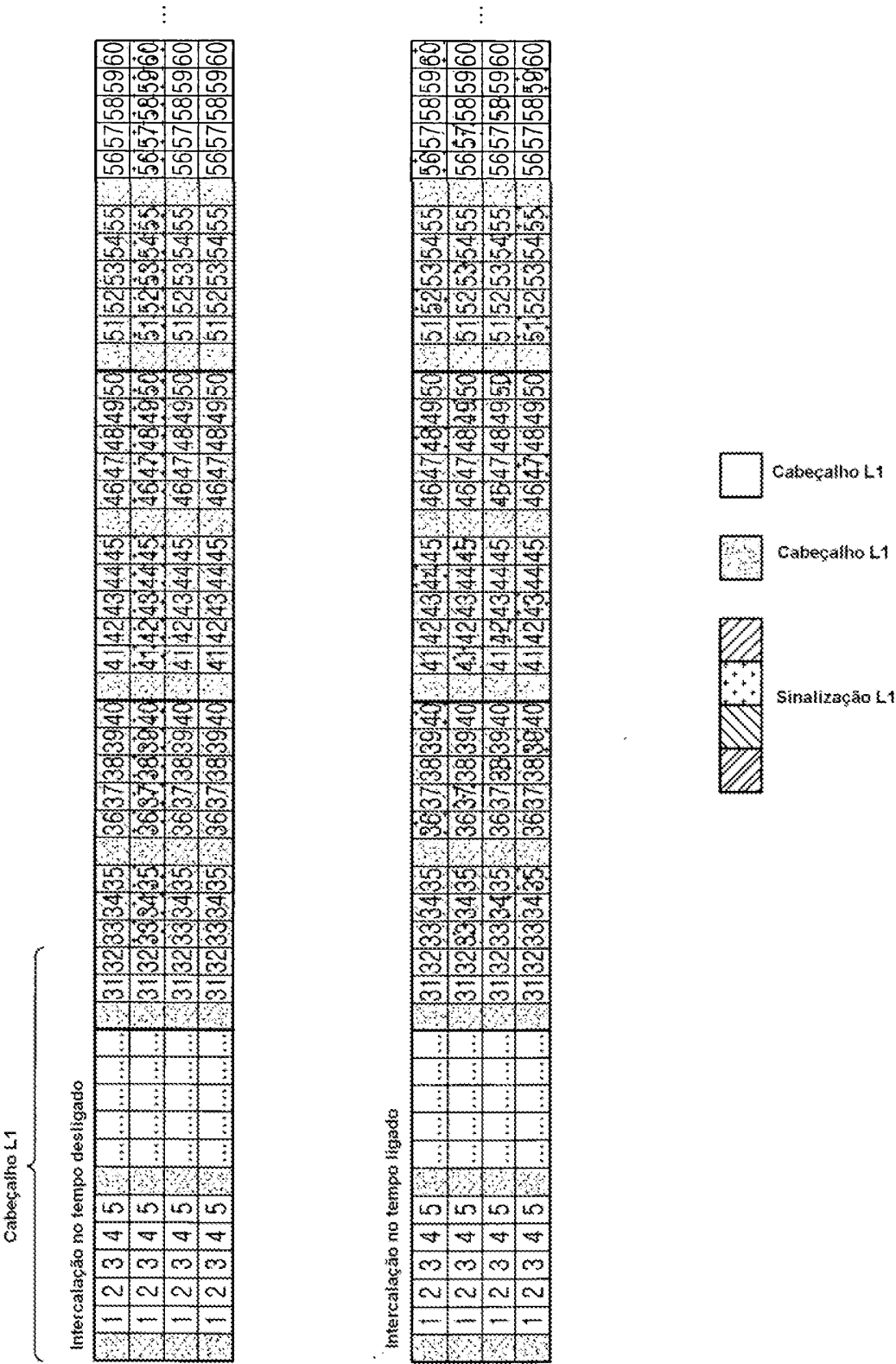


Fig. 84

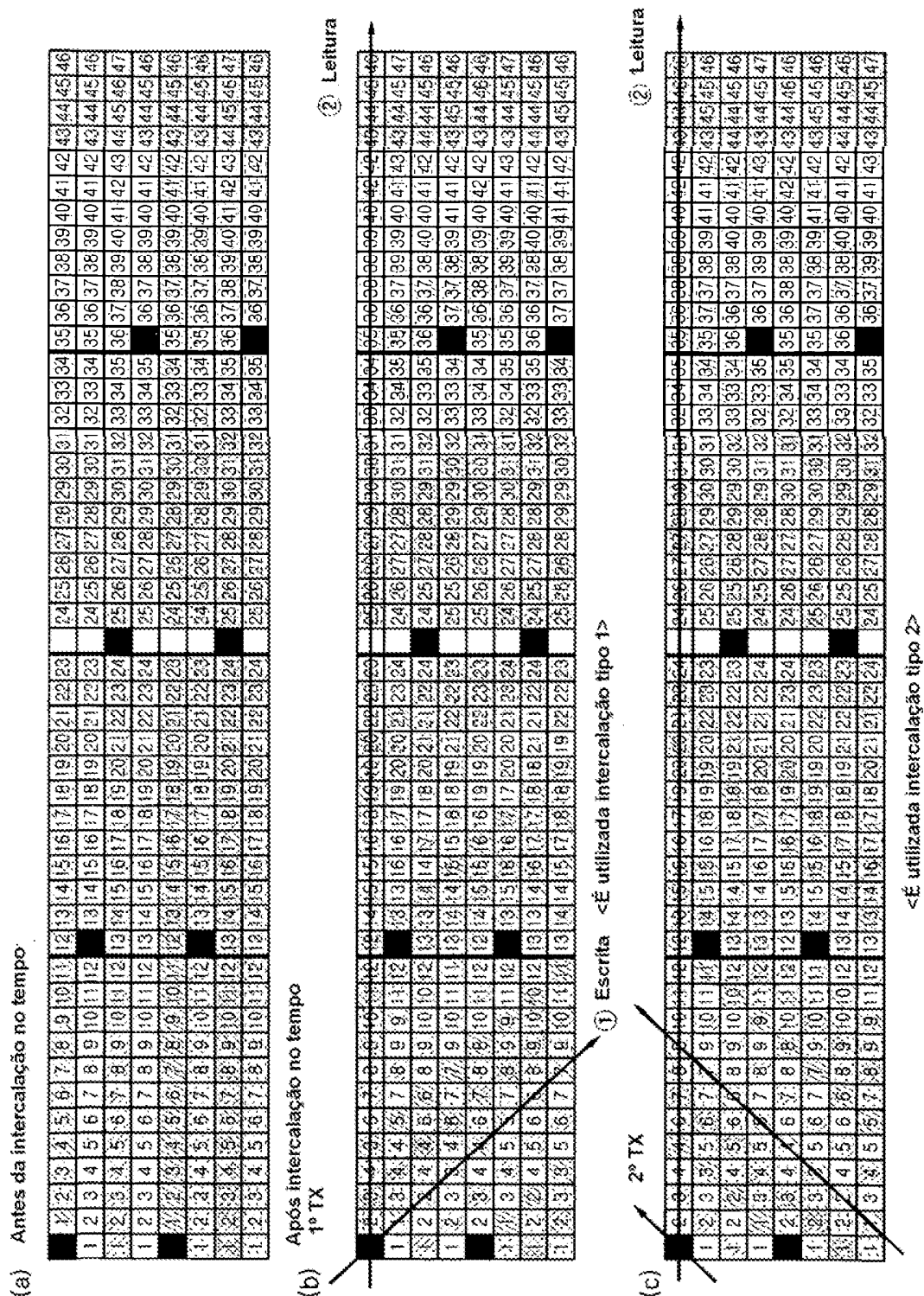


Fig. 85

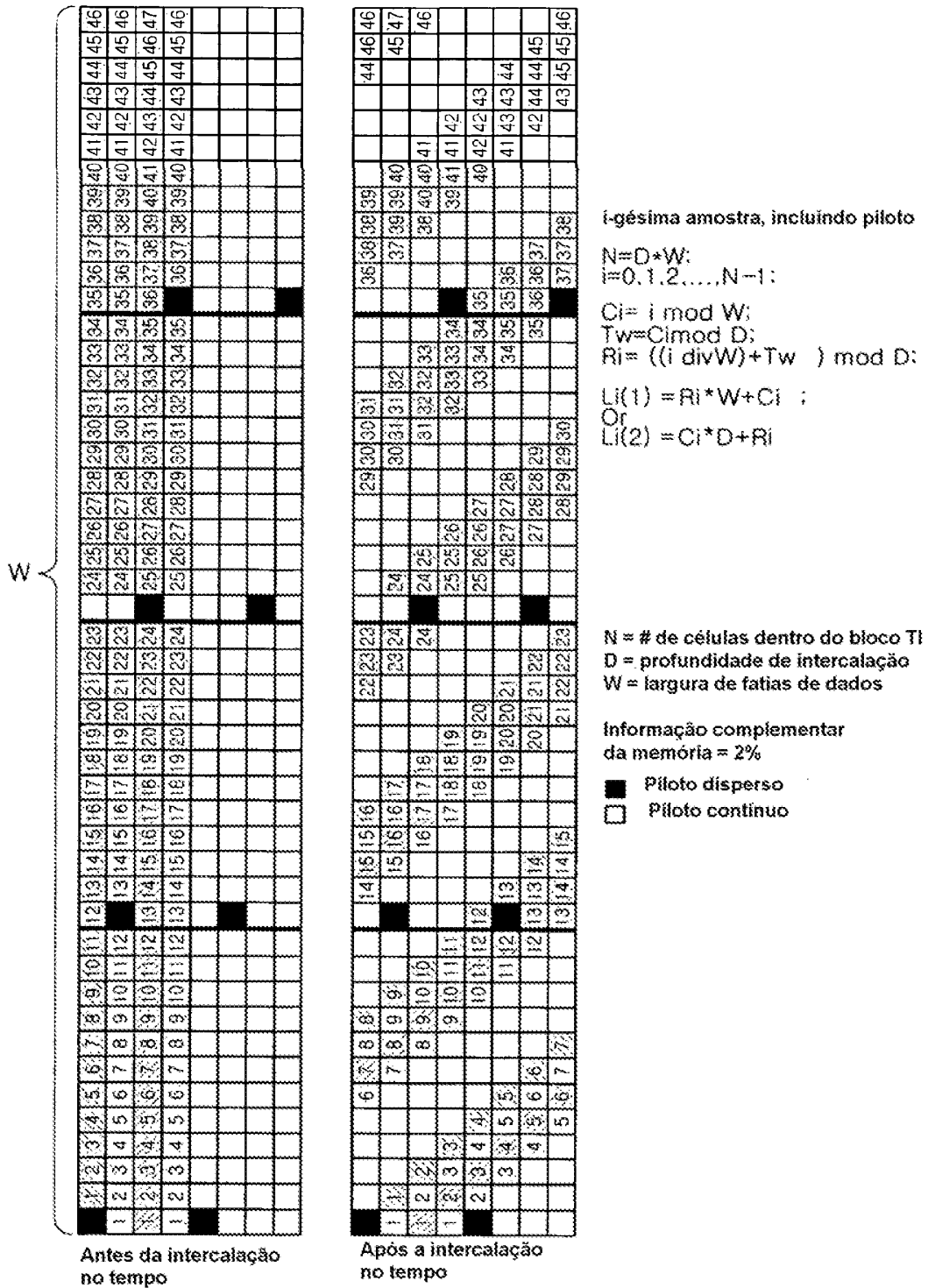


Fig. 86

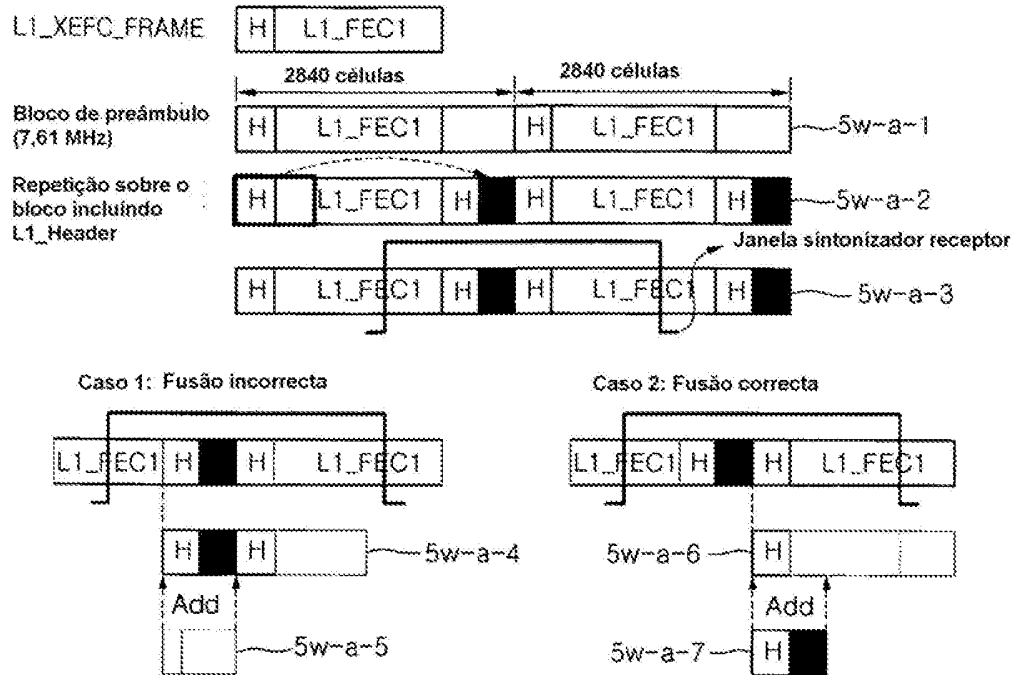


Fig. 87

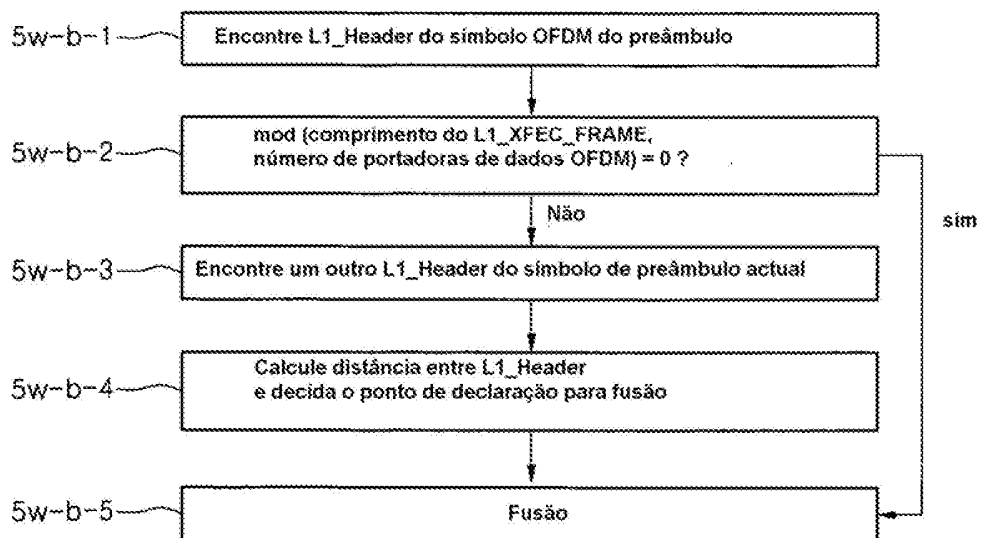


Fig. 88

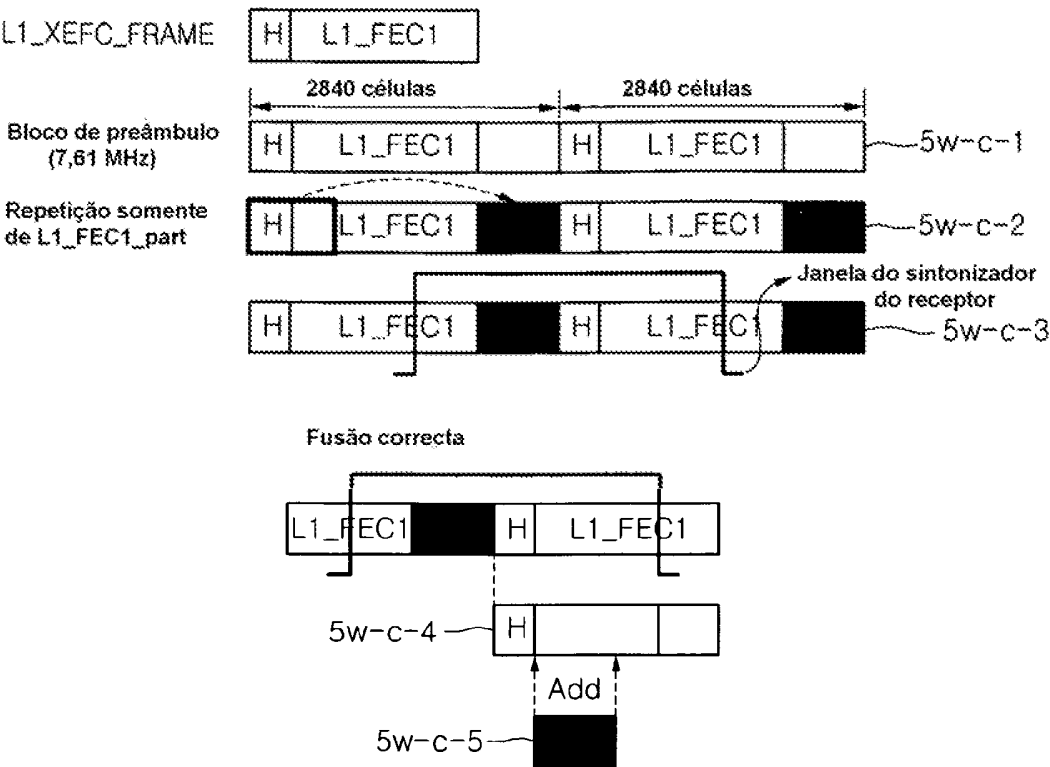


Fig. 90

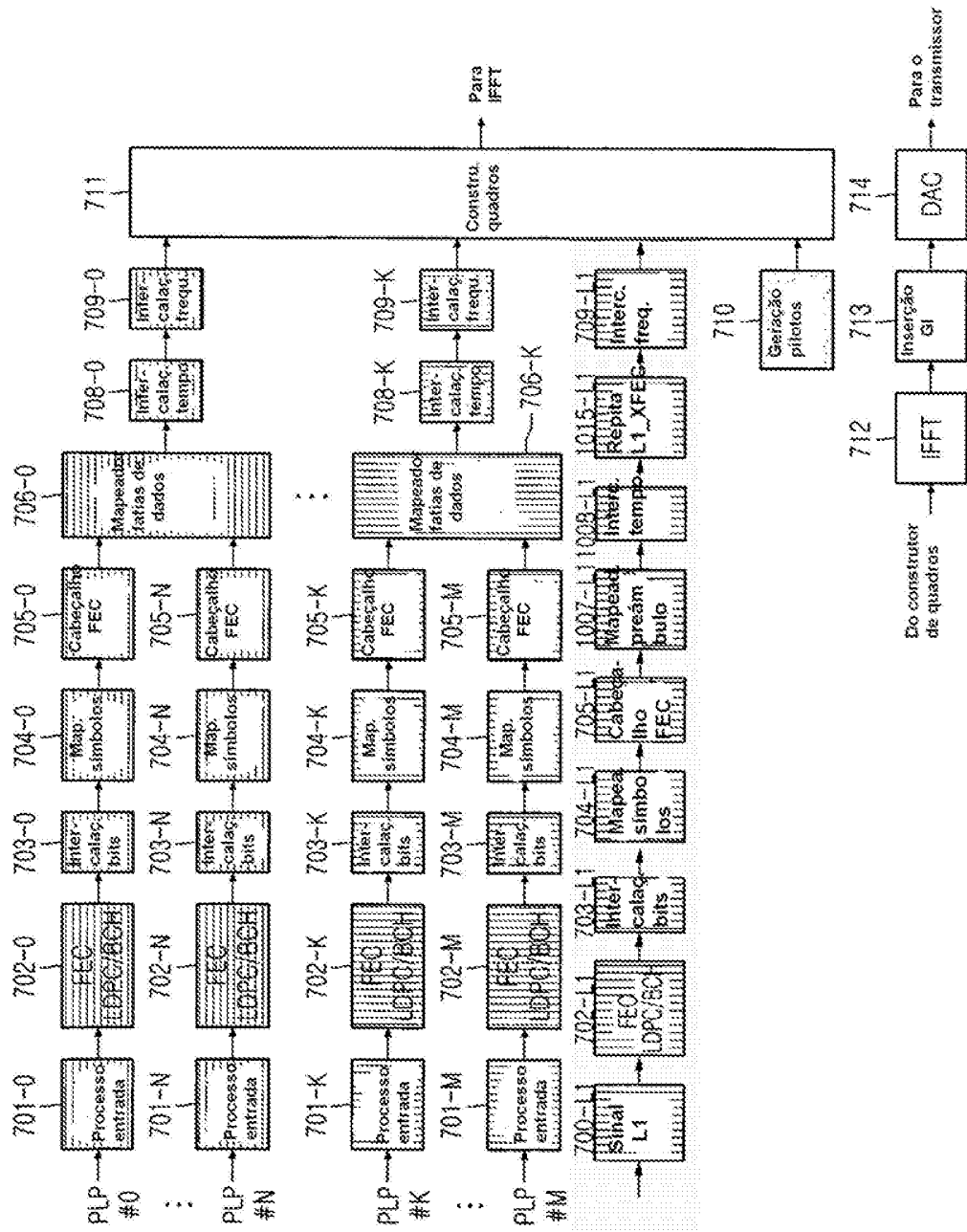


Fig. 91

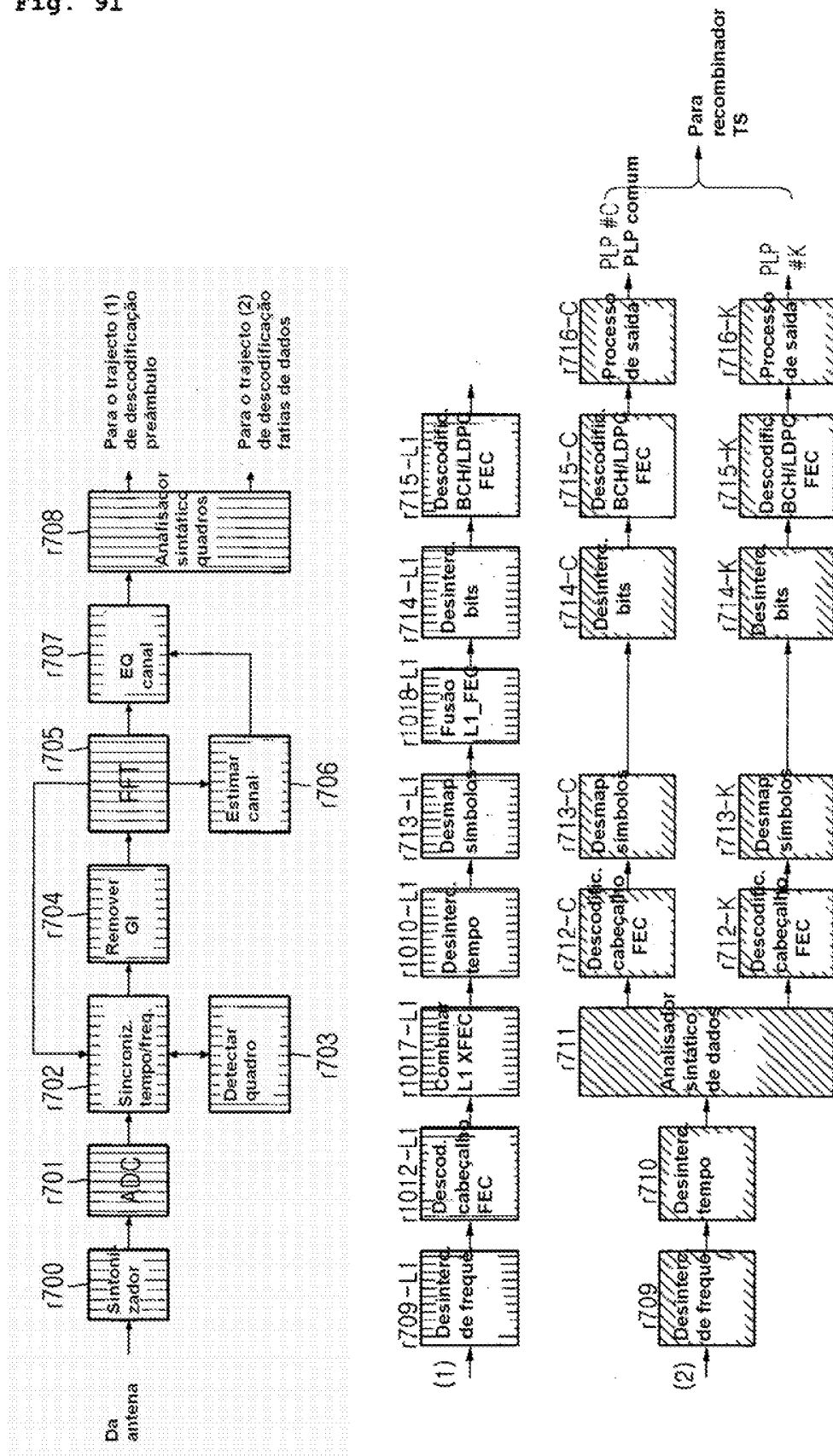


Fig. 93

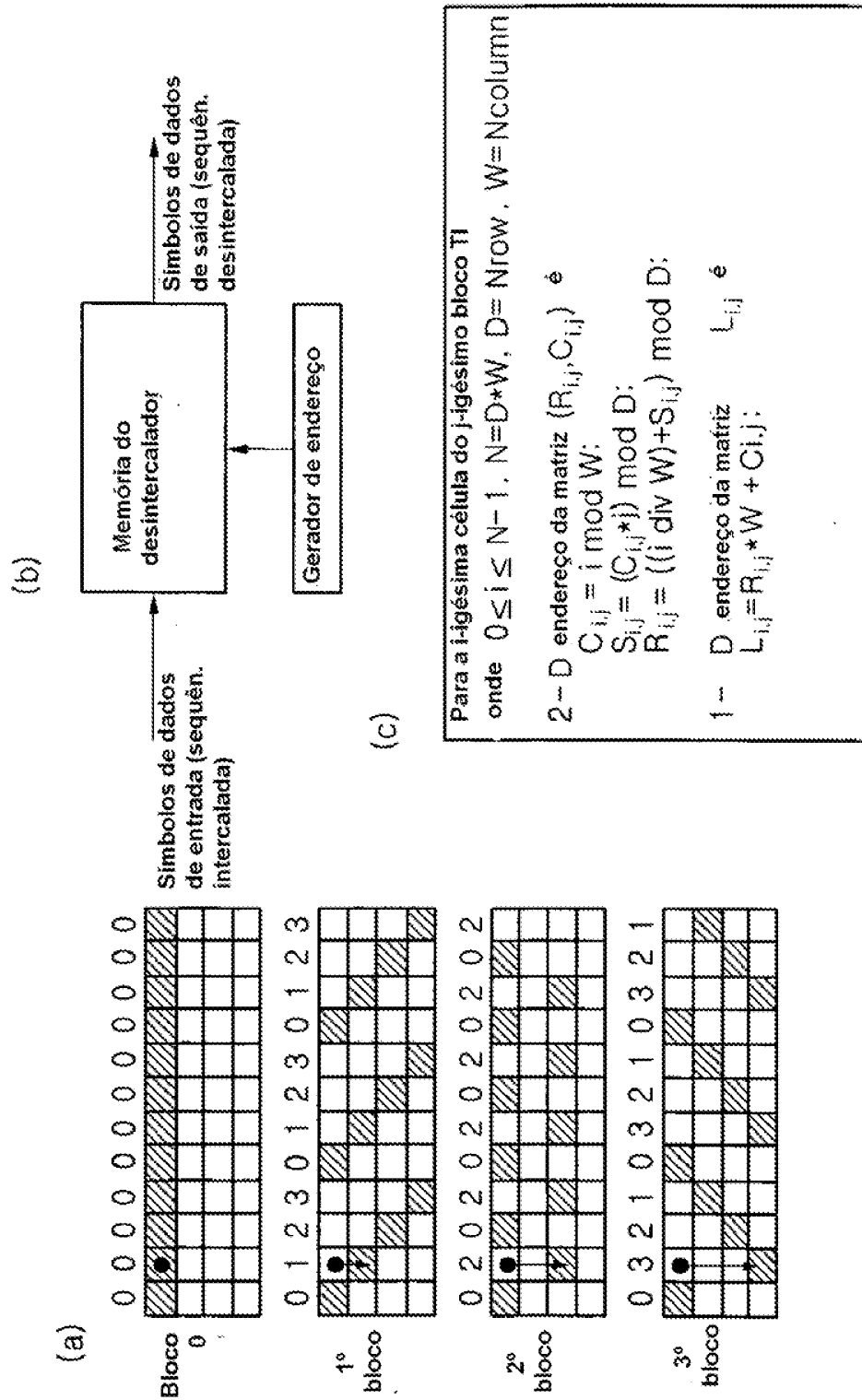


Fig. 94

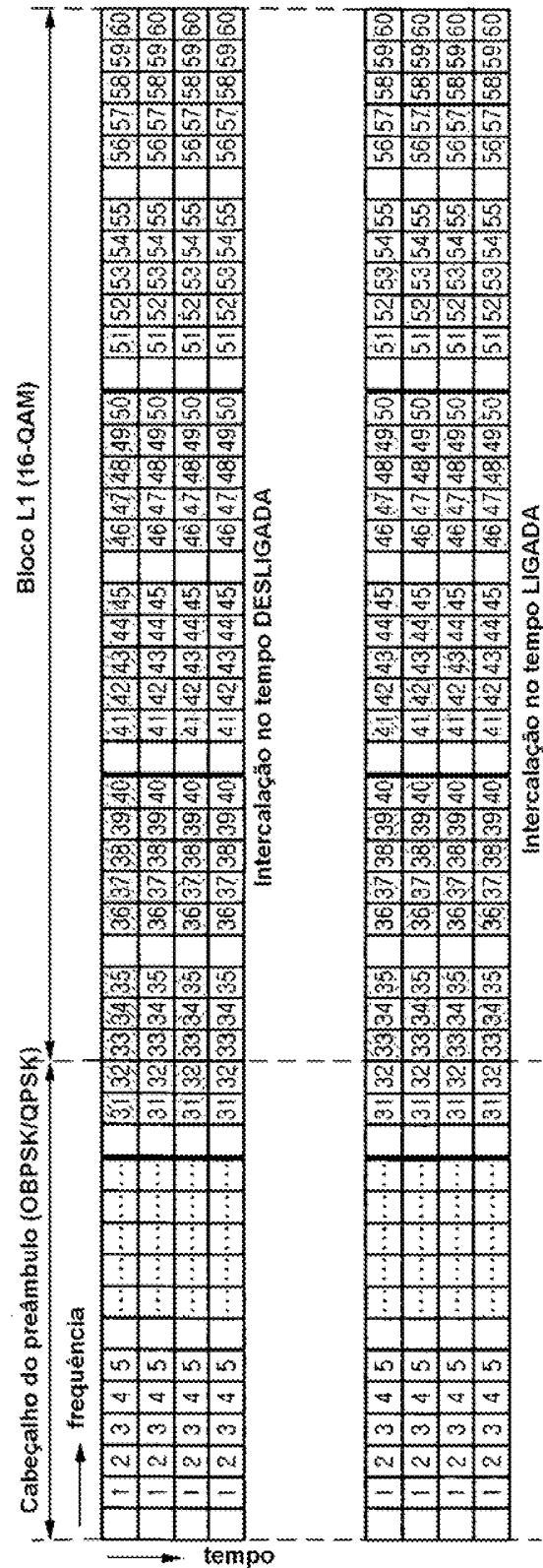


Fig. 95

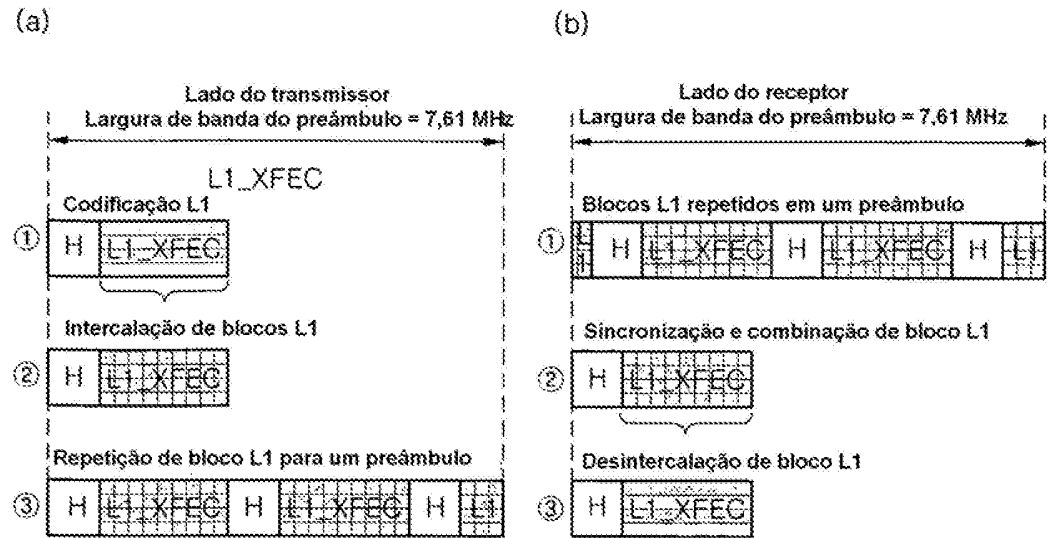


Fig. 96

Valor	Profundidade TI
00	Sem TI
01	Mais adequado
10	4 símbolos OFDM
11	8 símbolos OFDM

Quando TI_flag(1 bit) é utilizado
TI para bloco L1 curto
Para melhor desempenho TI

Fig. 97

(a) Sinalização e estrutura do cabeçalho

Nº	Características chave	Campos de sinalização	Preenchimento a zeros	Cabeçalho L1 por desempenho	Desempenho bloco L1 TI	Inform. complement. adicional
1	Somente marca TI lig./deslig.	L1 (15), TI (1), no CRC	Não	Normal	Mau para bloco L1 curto	Não
2	L1_size_bits / 8	L1 (12), TI (2), CRC (2)	Max 7 bits	Melhor	Bom	Não
3	L1_size_cells	L1 (13), TI (2), CRC (1)	Max 3 bits	Bom	Bom	Não
4	L1_size_bits por símbolo	L1 (13), TI (2), CRC (1)	Max 4 bits	Bom	Bom	Não
5	L1_size_cells por símbolo	L1 (11), TI (2), CRC (2), Rfu (1)	Max 19 bits	Melhor	Bom	Não
6	Dois cabeçalhos (32 bits)	L1 (15), TI (2), CRC (4), Rfu (11)	Não	Melhor	Bom	0,01% (negligenciável)

(b) Método de preenchimento a zeros

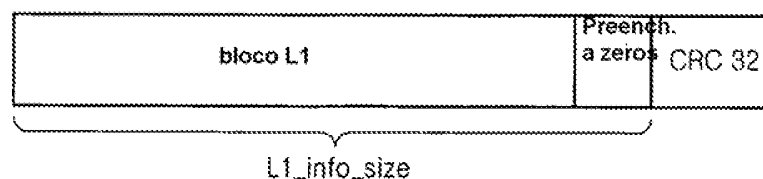


Fig. 98

Campo	Bits	Descrição
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_notch	4	número de bandas de entalhe
for i=0..num_dslice-1 {		
dslice_id	8	identificador de fatia de dados
dslice_start	12	início da fatia de dados dentro do canal C2 (Max 64 MHz)
dslice_width	9	largura da fatia de dados (Max 7,61 MHz)
dslice_ti_depth	2	profundidade do intercalador no tempo
dslice_type	1	tipo de fatia de dados (tipo 1, tipo 2)*
dslice_pwr_allocation	2	atribuição de potência de fatia de dados
num_plp	8	número de PLPs
for i=0..num_plp-1 {		
plp_id	8	identificador PLP
plp_type	1	tipo de PLP (comum/dados)
PSI/SI reprocessing	1	marca de reprocessamento PSP/SI
plp_payload_type	5	tipo de carga útil PLP (TS, GS, ...)
if dslice_type=='0' {		
plp_modcod	5	modulação/codificação PLP
plp_start_addr	14	endereço de início PLP
}		
}		
for i=0..num_notch-1{		
notch_start	12	início da banda de entalhe dentro do canal C2 (Max 64 MHz)
notch_width	9	largura da banda de entalhe (Max. 7,61 MHz)
}		
gi	2	modo de intervalo de guarda
num_data_symbols	10	número de dados símbolos OFDM por quadro C2
num_c2_frames	8	número de quadros C2 por superquadro
frame_idx	8	índice do quadro C2 dentro de um superquadro
PAPR	1	marca PAPR
reserved	0	reservado para utilização futura
crc32	32	CRC32

*tipo 1: PLP único com CCM

*tipo 2: Outros

Fig. 99

dslice_ti_depth	Profundidade TI
"00"	1
"01"	4
"10"	8
"11"	16

Fig. 100

dslice_type	tipo
"0"	PLP único com CCM
"1"	Outros

Fig. 101

plp_type	tipo
"0"	plp comum
"1"	plp de dados

Fig. 102

plp_payload_type	tipo de carga útil
"00000"	GFPS
"00001"	GCS
"00010"	GSE
"00011"	TS
others	reservado para utilização futura

Fig. 103

plp_modcod	Modulação	Taxa de código	Bloco LDPC
"00000"	16QAM	4/5	longo
"00001"			curto
"00010"		9/10	longo
"00011"		8/9	curto
"00100"	64QAM	2/3	longo
"00101"			curto
"00110"		4/5	longo
"00111"			curto
"01000"	256QAM	9/10	longo
"01001"		8/9	curto
"01010"		3/4	longo
"01011"			curto
"01100"	1024QAM	5/6	longo
"01101"			curto
"01110"		9/10	longo
"01111"		8/9	curto
"10000"	4096QAM	3/4	longo
"10001"			curto
"10010"		5/6	longo
"10011"			curto
"10100"		9/10	longo
"10101"		8/9	curto
"10110"		5/6	longo
"10111"			curto
"11000"		9/10	longo
"11001"		8/9	curto
outros	reservado para uso futuro	reservado para uso futuro	reservado para uso futuro

Fig. 104

gi	valor
"00"	1/128
"01"	1/64
outros	reservado para uso futuro

Fig. 105

PAPR	valor
"0"	desativar
"1"	ativar

Fig. 106

Campo	Bits	Descrição
network_id	16	ID da rede DVB-C2 actual
c2_system_id	16	ID do sistema C2 dentro da rede DVB-C2
C2_signal_start_frequency	32	Frequência do início do sinal C2
C2_signal_stop_frequency	32	Frequência da paragem do sinal C2
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_notch	4	número de bandas de entalhe
for i=0..num_dslice-1 {		
dslice_id	8	identificador da fatia de dados
dslice_start	12	início de fatia de dados dentro do canal C2 (max. 64 MHz)
dslice_width	9	largura da fatia de dados (máx. 7,61 MHz)
dslice_ti_depth	2	profundidade do intercalador no tempo
dslice_type	1	tipo de fatia de dados (tipo 1, tipo 2)*
dslice_pwr_allocation	2	atribuição da potência da fatia de dados
reserved_1	0	reservado para uso futuro
num_plp	8	número de PLPs
for i=0..num_plp-1 {		
plp_id	8	identificador PLP
plp_type	2	tipo de PLP (comum/dados normais/dados agrupados)
if plp_type=='common' or 'grouped' {		
plp_group_id	8	identificador do grupo PLP
}		
plp_payload_type	5	tipo de carga útil PLP (TS, GS, ...)
PSI/SI reprocessing	1	marca de reprocessamento PSP/SI
reserved_2	0	reservado para uso futuro
if dslice_type=='0' {		
plp_modcod	5	modulação/codificação PLP
plp_start_addr	14	endereço de início PLP
}		
}		
}		
for i=0..num_notch-1 {		
notch_start	12	início da banda de entalhe dentro do canal C2 (max. 64 MHz)
notch_width	9	largura da banda de entalhe (max. 7,61 MHz)
}		
qi	2	modo de intervalo de guarda
num_data_symbols	10	número de dados de símbolos OFDM por quadro C2
num_c2_frames	8	número de quadros C2 por superquadro
frame_idx	8	índice de quadro C2 dentro de um super quadro
PAPR	1	marca PAPR
reserved_3	0	reservado para uso futuro
crc32	32	CRC32

*tipo 1: PLP único com CCM

*tipo 2: Outros

Fig. 107

plp_type	tipo
"00"	plp comum
"01"	plp de dados normais
"10"	plp de dados agrupados
outros	reservado para uso futuro

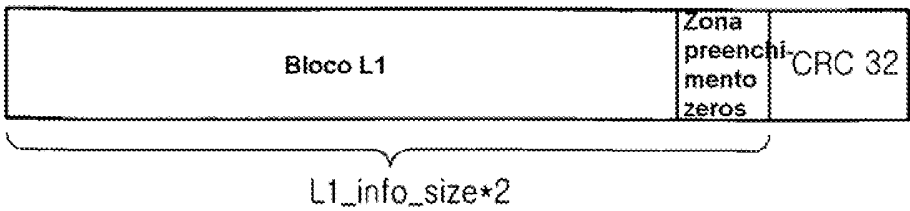
Fig. 108

Campo	Bits	Descrição
network_id	16	ID da rede DVB-C2 actual
c2_system_id	16	ID do sistema C2 dentro da rede DVB-C2
C2_signal_start_frequency	32	frequência do início do sinal C2
num_chbon	3	número de canais ligados
num_dslice	8	número de fatias de dados
num_notch	4	número de bandas de entalhe
for i=0..num_dslice-1 {		
dslice_id	8	identificador de fatia de dados
dslice_start	12	início da fatia de dados dentro do canal C2 (max. 64 MHz)
dslice_width	12	largura da fatia de dados (Max. 64 MHz)
dslice_ti_depth	2	profundidade do intercalador no tempo
dslice_type	1	tipo de fatia de dados (tipo 1, tipo 2)*
num_plp	8	número de PLPs
reserved_1	0	reservado para uso futuro
for i=0..num_plp-1 {		
plp_id	8	identificador PLP
plp_type	2	tipo PLP (comum/dados normais/dados agrupados)
if plp_type=='common' or 'grouped' {		
plp_group_id	8	identificador de grupo PLP
}		
PSI/SI reprocessing	1	marca de reprocessamento PSP/SI
plp_payload_type	5	tipo de carga útil PLP (TS, GS, ...)
reserved_2	0	reservado para uso futuro
if dslice_type=='0' {		
plp_modcod	5	modulação/codificação PLP
plp_start_addr	14	endereço de início PLP
}		
}		
}		
for i=0..num_notch-1 {		
notch_start	12	início da banda de entalhe dentro do canal C2 (max. 64 MHz)
notch_width	9	largura da banda de entalhe (max. 7,61 MHz)
}		
gi	2	modo de intervalo de guarda
PAPR	1	marca PAPR
reserved_3	0	reservado para uso futuro
crc32	32	CRC32
Total	221	

*tipo 1: PLP único com CCM

*tipo 2: Outros

Fig. 109



Valor	Profundidade TI
00	Sem TI
01	Mais adequado
10	4 símbolos OFDM
11	8 símbolos OFDM

Quando TI_flag(1 bit) é utilizado
TI para bloco L1 curto
Para melhor desempenho TI

Fig. 110

Campo	Bits	Descrição
NETWORK_ID	16	ID da rede DVB-C2 actual
C2_SYSTEM_ID	16	ID do sistema C2 dentro da rede DVB-C2
C2_SIGNAL_START_FREQUENCY	32	frequência do início do sinal C2 (unidade = 1 Hz)
NUM_CH3ON	3	número de canais ligados
NUM_DSLICE	8	número de fatias de dados
NUM_NOTCH	4	número de bandas de entalhe
for i=0.NUM_DSLICE-1 {		
DSLICE_ID	8	identificador da fatia de dados
DSLICE_START	12	início da fatia de dados dentro do canal C2 (máx. 64 MHz) (unidade = 12 células OFDM)
DSLICE_WIDTH	12	largura da fatia de dados (máx. 64 MHz) (unid. = 12 células OFDM)
DSLICE_TI_DEPTH	2	profundidade do intercalador no tempo
DSLICE_TYPE	1	tipo de fatias de dados (PLP único com CCM, outros)
DSLICE_NUM_PLP	8	número de PLPs na fatia de dados actual
for i=0.DSLICE_NUM_PLP-1 {		
PLP_ID	8	identificador PLP
PLP_TYPE	2	tipo de PLP (comum/dados normais/dados agrupados)
PLP_PAYLOAD_TYPE	5	tipo de carga útil PLP (GFPS, GCS, GSE, TS, RFU)
if(PLP_TYPE=='00' or '01' {		
PLP_GROUP_ID	8	identificador de grupo PLP
}		
if(DSLICE_TYPE=='0' {		
PLP_START	14	endereço de início PLP (unidade = 1 célula OFDM)
PLP_MODCOD	5	modulação/codificação PLP
}		
PSI/SI_REPROCESSING	1	marca de reprocessamento PSP/SI
RESERVED_1	8	reservado para uso futuro
}		
RESERVED_2	8	reservado para uso futuro
}		
for i=0.NUM_NOTCH-1 {		
NOTCH_START	12	início da banda de entalhe dentro do canal C2 (máx. 64 MHz) (unidade = 12 células OFDM)
NOTCH_WIDTH	9	largura da banda de entalhe (máx. 7,61 Mhz)(unidade = 12 células OFDM)
RESERVED_3	8	reservado para uso futuro
}		
GI	2	modo de intervalo de guarda
PAPR	1	marca PAPR
L1_PART2_CHANGE_COUNTER	8	número de quadros C2 à frente que a L1_part2 irá alterar excepto para PLP_START (unidade = 1 quadro C2)
RESERVED_4	16	reservado para uso futuro
CRC32	32	CRC32
Total	269	

Fig. 111

DSLICE_TI_DEPTH	
Valor	Profundidade TI
00	1 símbolo OFDM
01	4 símbolos OFDM
10	8 símbolos OFDM
11	16 símbolos OFDM

PAPR	
Valor	PAPR
0	desativar
1	ativar

DSLICE_TYPE	
Valor	Tipo de fatia de dados
0	PLP único com CCM
1	Outros

GI	
Valor	Fracção de intervalo de guarda
00	1/128
01	1/64
10 a 11	Reservado para uso futuro

PLP_TYPE	
Valor	Tipo PLP
00	PLP comum
01	PLP de dados agrupados
10	PLP de dados normais
11	Reservado para uso futuro

PLP_PAYLOAD_TYPE	
Valor	Tipo de carga útil
00000	GFPS
00001	GCS
00010	GSE
00011	TS
00100 a 11111	Reservado para uso futuro

PLP_MODCOD			
Valor	Modulação	Taxa de código	Tipo FEC
00000	16QAM	4/5	64K LDPC
00001			16K LDPC
00010		9/10	64K LDPC
00011		8/9	16K LDPC
00100	64QAM	2/3	64K LDPC
00101			16K LDPC
00110		4/5	64K LDPC
00111			16K LDPC
01000	256QAM	9/10	64K LDPC
01001		8/9	16K LDPC
01010		3/4	64K LDPC
01011			16K LDPC
01100		5/6	64K LDPC
01101			16K LDPC
01110		9/10	64K LDPC
01111		8/9	16K LDPC
10000	1024QAM	3/4	64K LDPC
10001			16K LDPC
10010		5/6	64K LDPC
10011			16K LDPC
10100		9/10	64K LDPC
10101			16K LDPC
10110	4096QAM	5/6	64K LDPC
10111			16K LDPC
11000		9/10	64K LDPC
11001		8/9	16K LDPC
11010 a 11111	Reservado para uso futuro		

Fig. 112

Campo	Bits	Descrição
NETWORK_ID	16	ID da rede DVB-C2 actual
C2_SYSTEM_ID	16	ID do sistema C2 dentro da rede DVB-C2
C2_SIGNAL_START_FREQUENCY	32	Frequência do início do sinal C2 (unidade = 1 Hz)
GI	2	Modo de intervalo de guarda
L1_PART2_CHANGE_COUNTER	8	Número de quadros C2 à frente que a L1_part2 irá alterar excepto para PLP_START (unidade = 1 quadro C2)
NUM_CHBON	3	Número de canais ligados
NUM_DSLICE	8	Número de fatias de dados
NUM_NOTCH	4	Número de bandas de entalhe
for i=0..NUM_DSLICE-1 { DSLICE_ID	8	Identificador da fatia de dados
DSLICE_START	12(11)	Início da fatia de dados dentro do canal C2 (max. 64 MHz) (unidade = 12 ou 24 células OFDM para 1/64, 1/128 GI respectivamente.)
DSLICE_WIDTH	12(11)	Largura da fatia de dados (max. 64 MHz) (unidade = 12 ou 24 células OFDM para 1/64, 1/128 GI respectivamente.)
DSLICE_TL_DEPTH	2	Profundidade do intercalador no tempo
DSLICE_TYPE	1	Tipo de fatia de dados (PLP único com CCM, outros)
DSLICE_CONST_FLAG	1	Indicador para fatia de dados constante
DSLICE_NOTCH_FLAG	2	Indicador de entalhe contíguo
DSLICE_NUM_PLP	8	Número de PLPs na fatia de dados actual
for i=0..DSLICE_NUM_PLP-1 { PLP_ID	8	Identificador PLP
PLP_TYPE	3	Tipo PLP (comum/dados agrupados/dados normais/dados integrado)
PLP_PAYLOAD_TYPE	5	Tipo de carga útil PLP (GFPS, GCS, GSE, TS, RFU)
if PLP_PAYLOAD_TYPE=00 or 01 { PLP_GROUP_ID	8	Identificador de grupo PLP
} else if PLP_PAYLOAD_TYPE=0 { PLP_START	14	Endereço de início PLP (unidade = 1 célula OFDM)
PLP_MODCOD	5	Modulação/codificação PLP
}		
PSI/SI_REPROCESSING	1	Marca de reprocessamento PSP/SI
RESERVED_1	8	Reservado para uso futuro
}		
RESERVED_2	8	Reservado para uso futuro
}		
for i=0..NUM_NOTCH-1 { NOTCH_START	12(11)	Início da banda de entalhe dentro do canal C2 (max. 64MHz) (unid. = 12 ou 24 células OFDM para 1/64, 1/128 GI respectivamente)
NOTCH_WIDTH	9(8)	Largura da banda de entalhe (max. 7,61MHz) (unid. = 12 ou 24 células OFDM para 1/64, 1/128 GI respectivamente)
RESERVED_3	8	Reservado para uso futuro
}		
PAPR	1	Marca PAPR
RESERVED_4	16	Reservado para uso futuro
CRC32	32	CRC32
Total	228	

Fig. 113

Valor	Tipo PLP
000	PLP comum
001	PLP de dados agrupados
010	PLP de dados normais
011	PLP de dados integrados
100 to 111	Reservado para uso futuro

Fig. 114

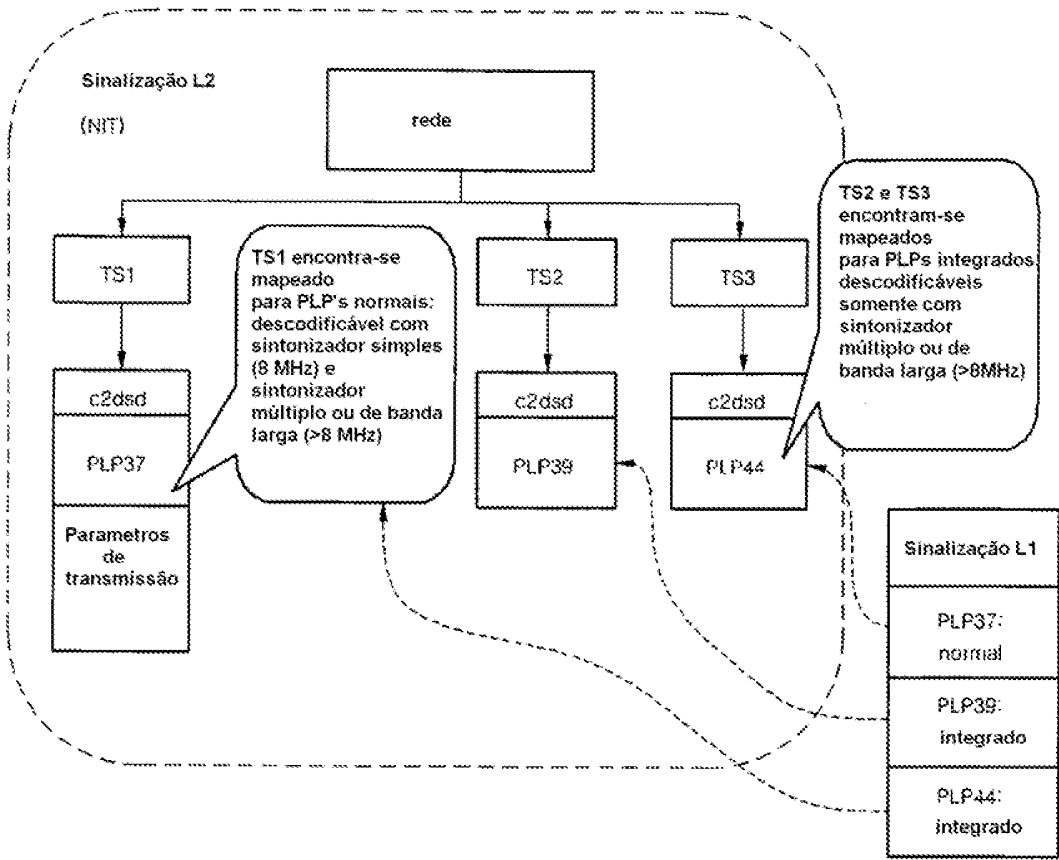


Fig. 115

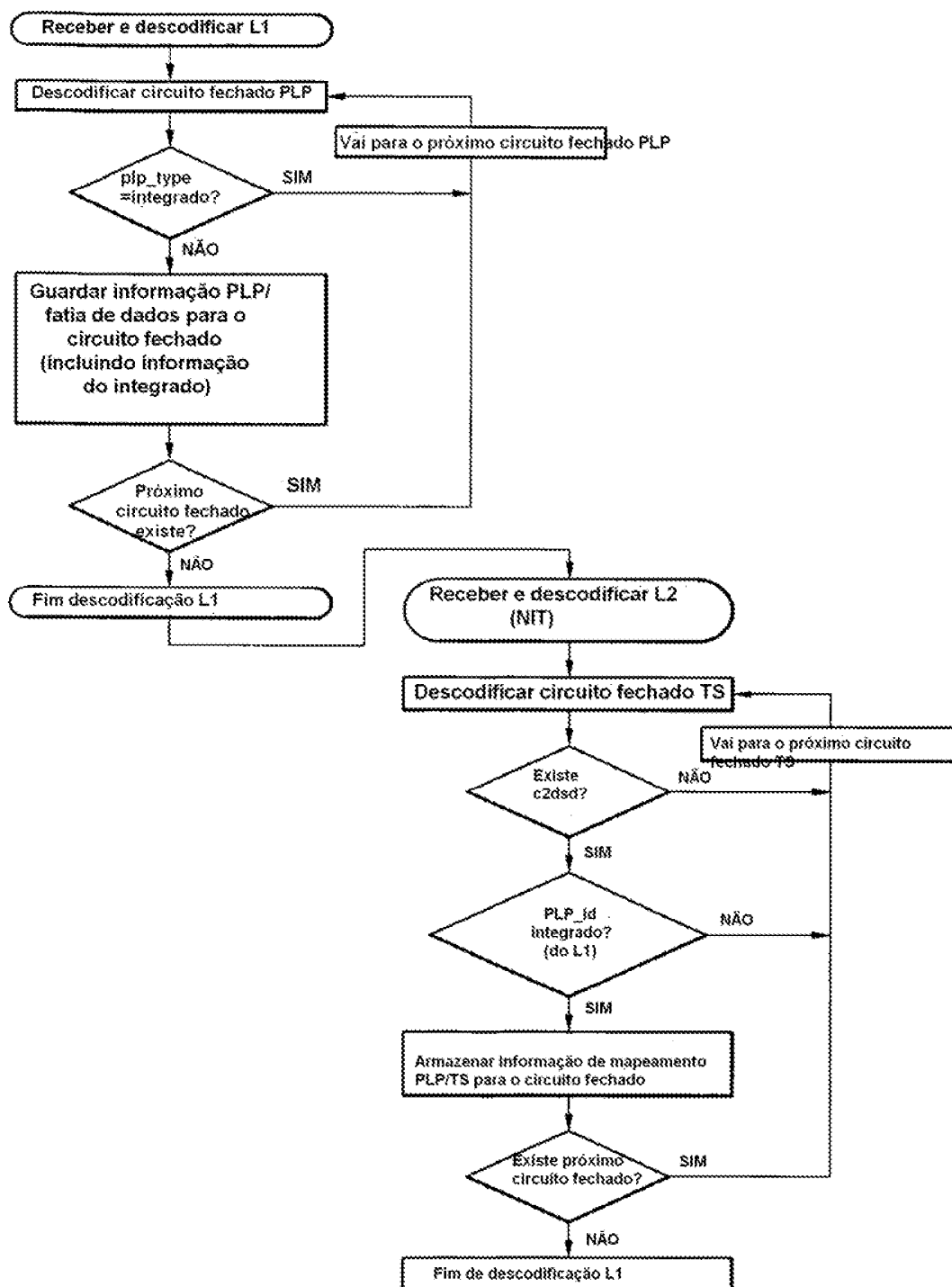


Fig. 116

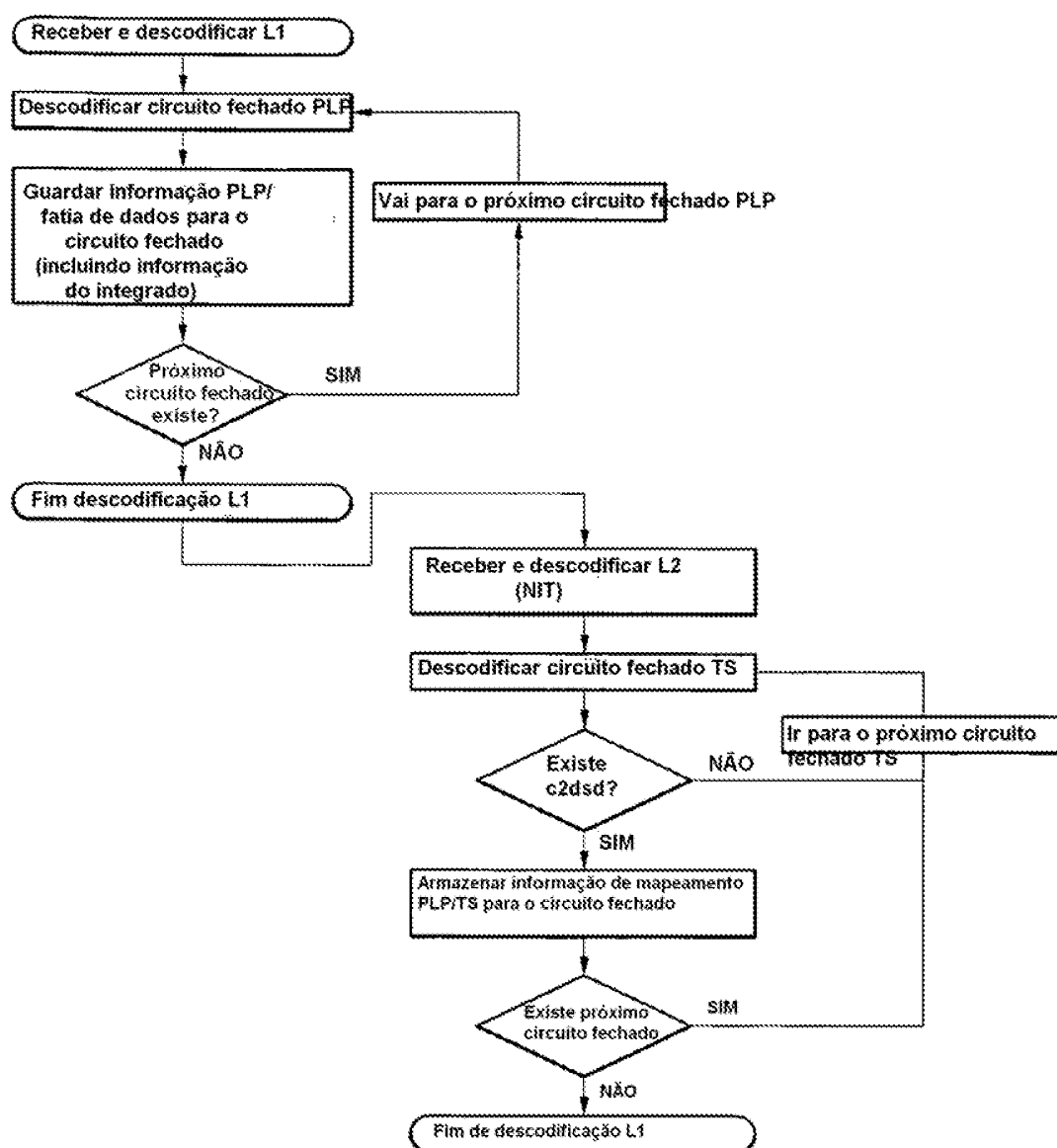


Fig. 117

Sintaxe	Número de bits	Identificador
C2_delivery_system_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
descriptor_tag_extension	8	uimsbf
plp_id	8	uimsbf
C2_system_id	16	uimsbf
if (descriptor_length > 5){		
C2_System_tuning_frequency	32	bslbf
active OFDM symbol duration	3	bslbf
guard_interval	3	bslbf
reserved	2	bslbf
}		
}		
)		

Fig. 118

Active_OFDM_symbol_duration	Descrição
000	448µs (Modo FFT 4k para sistemas CATV de largura de banda de 8MHz)
001	597µs (Modo FFT 4k para sistemas CATV de largura de banda de 6MHz)
010 to 111	Reservado para uso futuro

Fig. 119

guard_interval	Valores de intervalo de guarda
000	1/128
001	1/64
010 to 111	Reservado para uso futuro

Fig. 120

	Bits	
NETWORK_ID	16	ID da rede DVB-C2 actual
C2_SYSTEM_ID	16	ID do sistema C2 dentro da rede DVB-C2
C2_SIGNAL_START_FREQUENCY	32	Frequência do início do sinal C2 (unidade = 1Hz)
GI	2	Modo de intervalo de guarda
L1_PART2_CHANGE_COUNTER	8	Número de quadros C2 a frente que a L1_part2 irá alterar excepto para PLP_START (unidade = 1 quadro C2)
NUM_CHBON	3	Número de canais ligados
NUM_DSlice	8	Número de fatias de dados
NUM_NOTCH	4	Número de bandas de entalhe
for i=0..NUM_DSlice {		
DSlice_ID	8	Identificador da fatia de dados
DSlice_TUNE_POS	12(11)	Posição de sintonia para a fatia de dados actual (unidade = 12 ou 24 células OFDM ou 1/64, 1/128 GI respectivamente)
DSlice_OFFSET_LEFT	9(8)	Desvio para o bordo esquerdo da fatia de dados da posição de sintonia (unid.=12 ou 24 cél. OFDM p/ 1/64, 1/129 GI respectiv.). Assinalado
DSlice_OFFSET_RIGHT	9(8)	Desvio para o bordo direito da fatia de dados da posição de sintonia (unid.=12 ou 24 cél. OFDM p/ 1/64, 1/129 GI respectiv.). Assinalado
DSlice_TI_DEPTH	2	Profundidade do intercalador no tempo
DSlice_TYPE	1	Tipo de fatia de dados (PLP único com CCM, outros)
DSlice_CONST_FLAG	1	Indicador para fatia de dados constante
DSlice_NOTCH_FLAG	1	Indicador de entalhe contíguo
DSlice_NUM_PLP	8	Número de PLPs na fatia de dados actual
for i=0..DSlice_NUM_PLP-1 {		
PLP_ID	8	Identificador PLP
PLP_BUNDLED_FLAG	1	Marca PLP agrupada
PLP_TYPE	2	Tipo de PLP (comum/ dados agrupados/ dados normais)
PLP_PAYLOAD_TYPE	5	Tipo de carga útil (GFPS, GCS, GSE, TS, RFU)
if PLP_TYPE=='00' or '01' {		
PLP_GROUP_ID	8	Identificador de grupo PLP
}		
if DSlice_TYPE =='0' {		
PLP_START	14	Endereço de início PLP (unidade = 1 célula OFDM)
PLP_MODCOD	5	Modulação/codificação PLP
}		
PSI/SI_REPROCESSING	1	Marca de reprocessamento PSI/SI
RESERVED_1	8	Reservado para uso futuro
}		
RESERVED_2	8	Reservado para uso futuro
}		
for i=0..NUM_NOTCH-1 {		
NOTCH_START	12(11)	Início da banda de entalhe dentro do canal C2 (max 64 MHz) (unid.=12 ou 24 célul. OFDM para 1/64, 1/128 GI respectivamente)
NOTCH_WIDTH	9(8)	Largura da banda de entalhe (máx. 7,61 MHz) (unid.=12 ou 24 célul. OFDM para 1/64, 1/128 GI respectivamente)
RESERVED_3	8	Reservado para uso futuro
}		
PAPR	1	Marca PAPR
RESERVED_4	16	Reservado para uso futuro
CRC_32	32	CRC32
Total	227	

Fig. 121

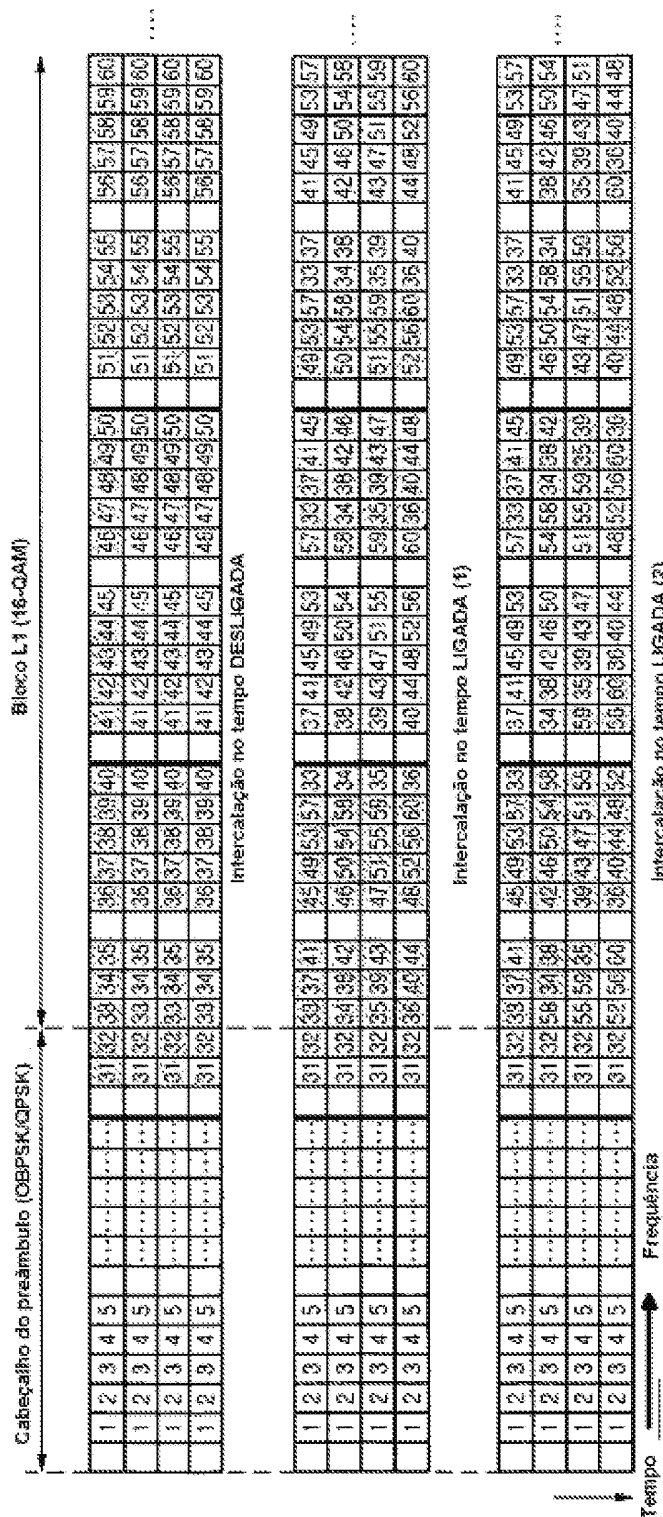


Fig. 122

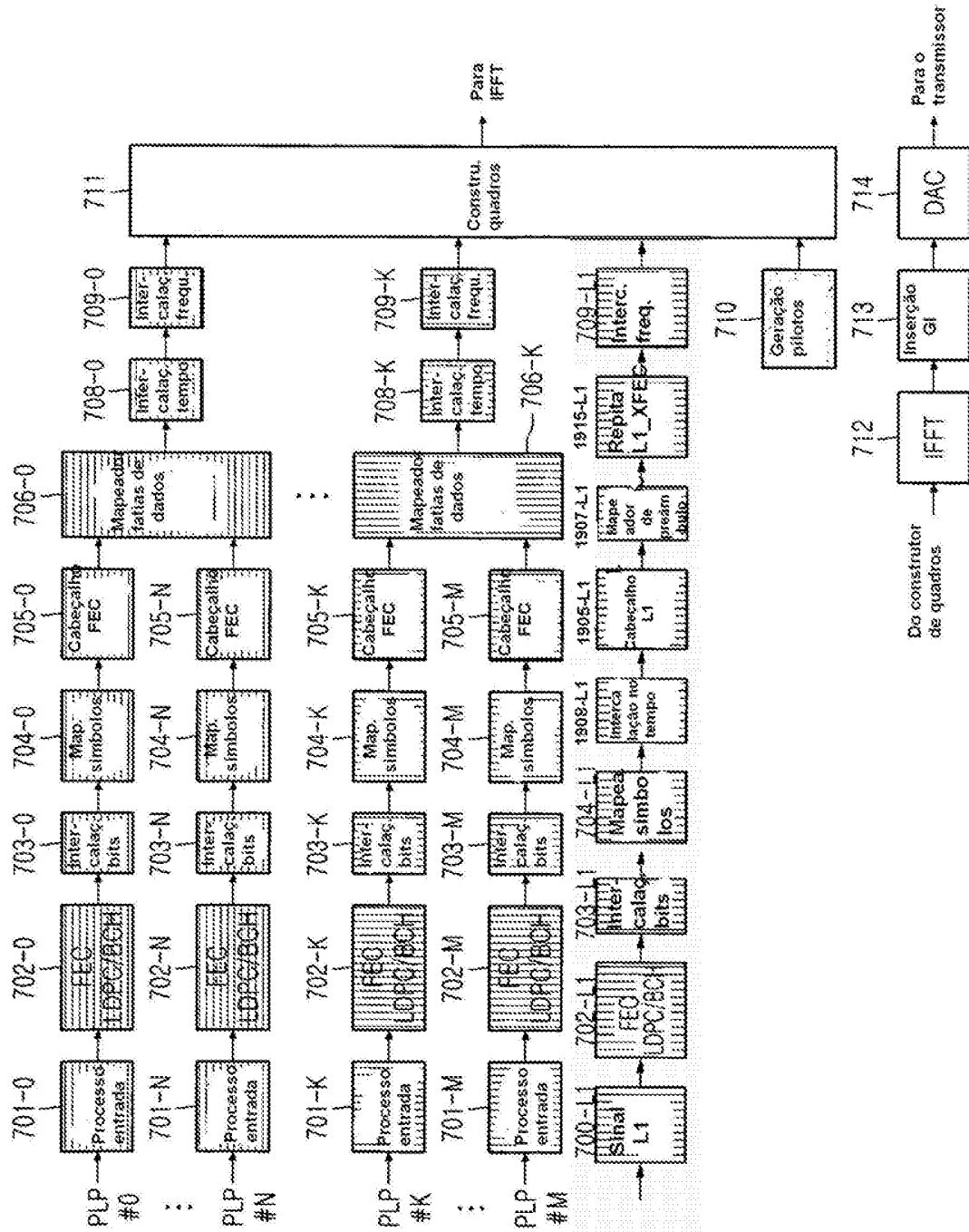


Fig. 123

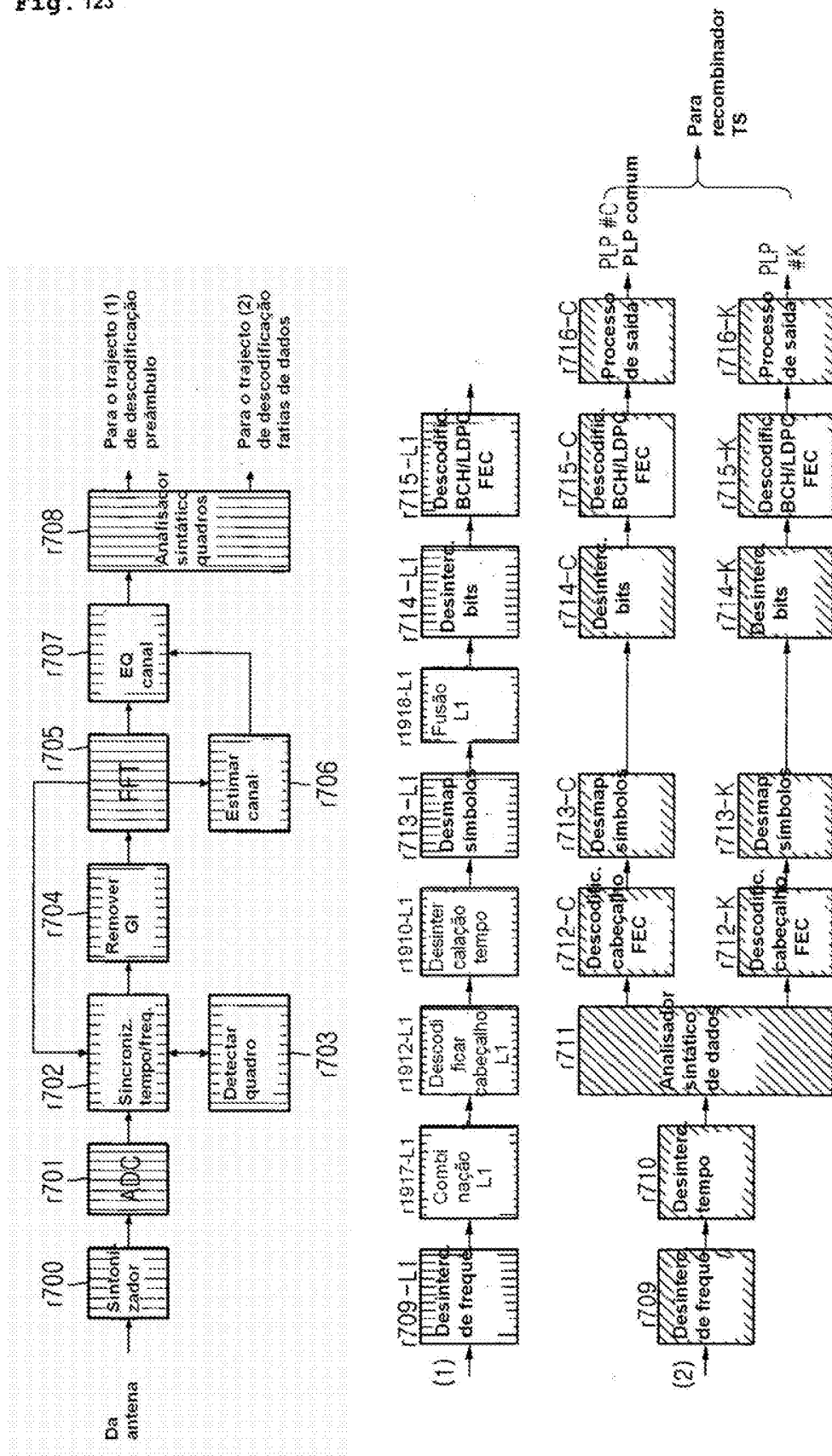
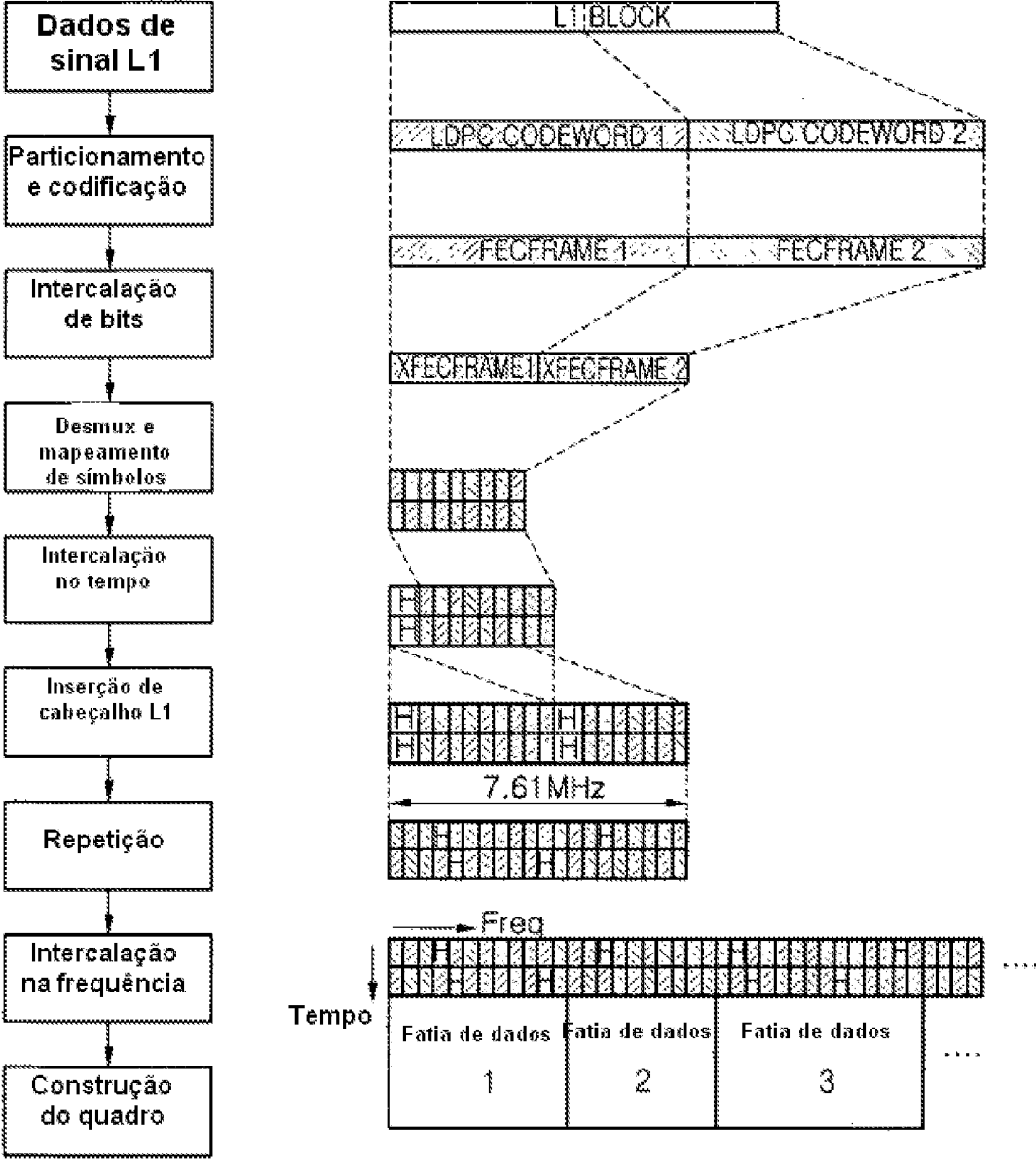


Fig. 124



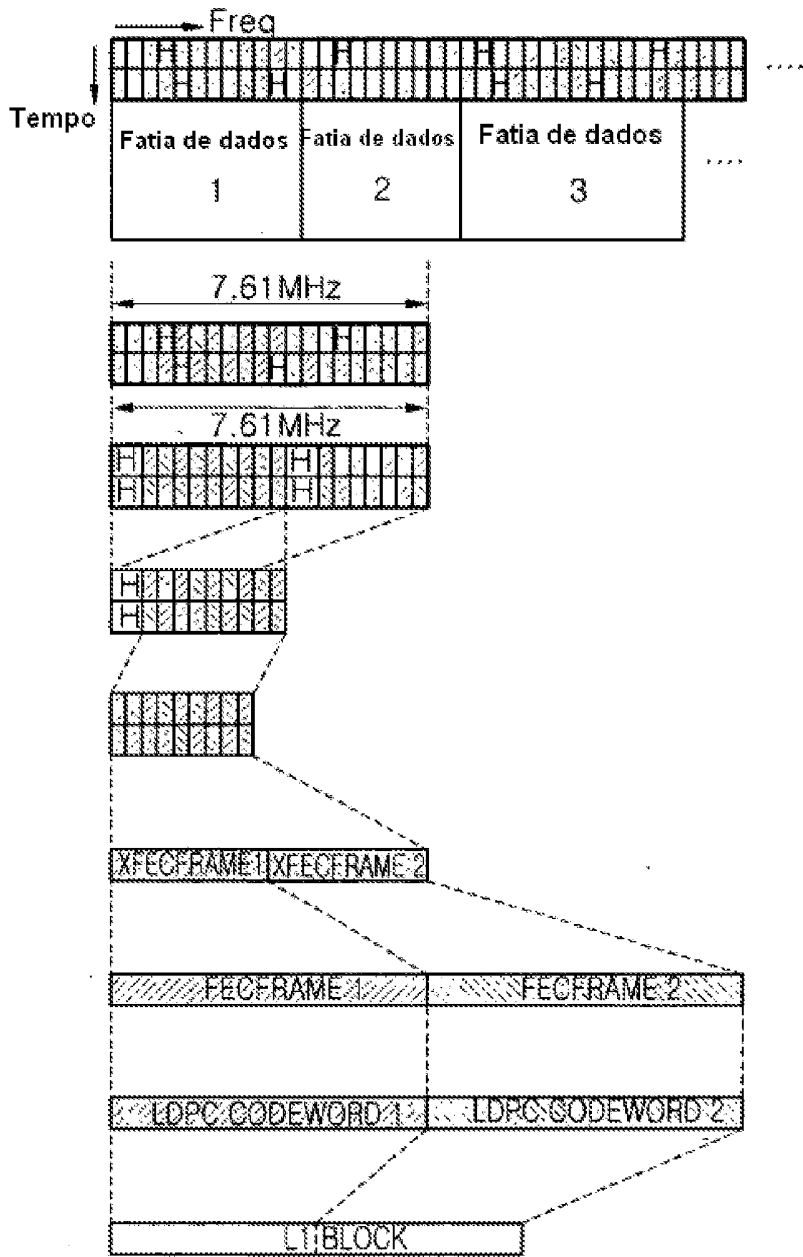
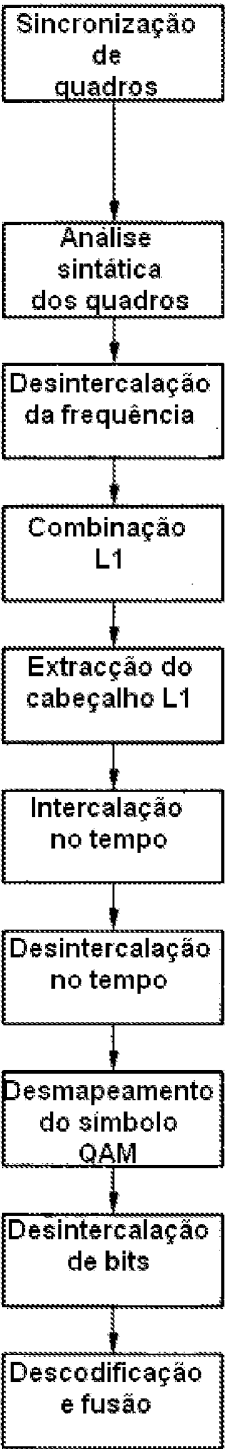


Fig. 126

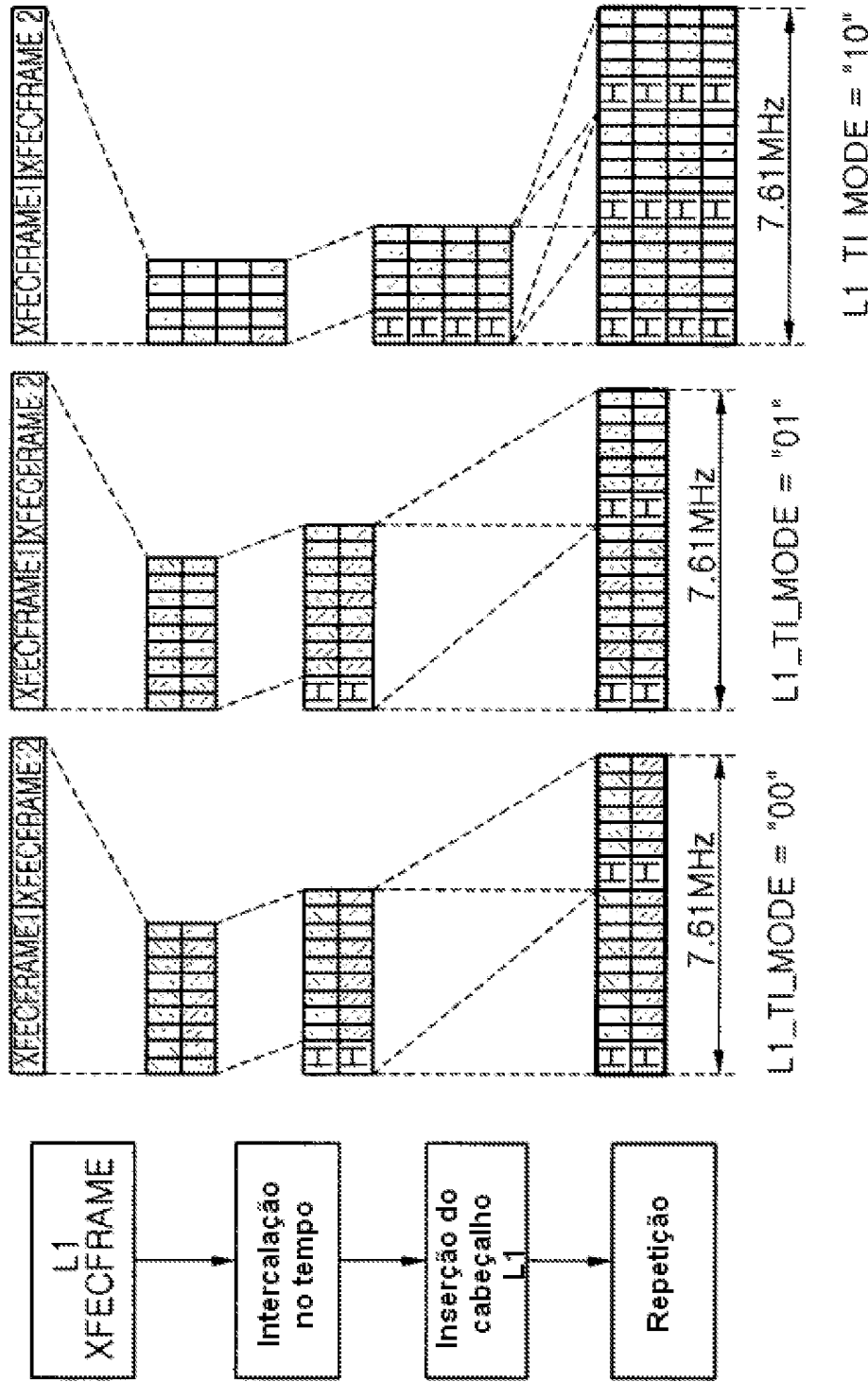


Fig. 127

