



República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(11) PI 0920595-0 B1



(22) Data do Depósito: 12/10/2009

(45) Data de Concessão: 26/11/2019

(54) Título: PROCESSO PARA A TRANSFERÊNCIA DE CÓDIGOS DE PROGRAMA PARA A MEMÓRIA DE UM APARELHO DE COMANDO, ESPECIALMENTE PARA VEÍCULOS AUTOMOTORES

(51) Int.Cl.: G06F 9/445.

(30) Prioridade Unionista: 23/10/2008 DE 10 2008 052 955.9.

(73) Titular(es): KNORR-BREMSE SYSTEME FÜR NUTZFAHRZEUGE GMBH.

(72) Inventor(es): UWE FISCHER; ULRICH KANZLER.

(86) Pedido PCT: PCT EP2009007313 de 12/10/2009

(87) Publicação PCT: WO 2010/046043 de 29/04/2010

(85) Data do Início da Fase Nacional: 25/04/2011

(57) Resumo: PROCESSO PARA A TRANSFERÊNCIA DE CÓDIGOS DE PROGRAMA PARA A MEMÓRIA DE UM APARELHO DE COMANDO, ESPECIALMENTE PARA VEÍCULOS AUTOMOTORES A presente invenção refere-se à transferência do código de programa para uma memória de programa de um aparelho de comando, especialmente para veículos automotores, inicialmente serão deletadas todas as células da memória de programação e preferencialmente serão sobrescritas com código de programa inválido (illegal Opcode). Códigos de programa válidos serão comprimidos de acordo com um processo de compressão de dados, isento de perdas e especialmente a codificação Huffman, sendo desta forma transferida para o aparelho de comando, onde serão correspondentemente descomprimidos e registrados na memória do programa do aparelho de comando. O valor para códigos de programa inválidos está memorizado no aparelho de comando como valor (default) ou será no máximo transferido uma vez do aparelho de programação para o aparelho de comando. Com estas medidas, o volume de dados a ser transferido e, portanto, o tempo para a transferência do código de programa será essencialmente reduzido.

Relatório Descritivo da Patente de Invenção para "PROCESSO PARA A TRANSFERÊNCIA DE CÓDIGOS DE PROGRAMA PARA A MEMÓRIA DE UM APARELHO DE COMANDO, ESPECIALMENTE PARA VEÍCULOS AUTOMOTORES".

Descrição

[001] A presente invenção refere-se a um processo para a transferência de códigos de programa para uma memória de programa de um aparelho de comando, de acordo com o preâmbulo da reivindicação 1. Um processo desta natureza já passou a ser conhecido a partir do documento DE nº 10 2006 016 891 A1.

[002] Veículos automotores modernos contêm um ou vários aparelhos de comando eletrônicos, apresentando uma memória de programa que pode ser programada quando estiver montado o aparelho de comando, isto é, poderá ser alimentado com códigos de programa. A memória de programa, em caso normal, é uma chamada memória Flash que frequentemente também está integrada em um micro controlador.

[003] Os aparelhos de comando convencionais contêm uma interface, também denominada interface de diagnóstico, que é usada para a transferência de comandos e dados. Como interface é frequentemente usado um chamado Bus "CAN", um protocolo de diagnóstico convencional UDS, de acordo com a ISO 14229. Estes são apenas mencionados como exemplo, porque existe uma grande variedade de outras interfaces e protocolos.

[004] No aparelho de comando geralmente também está integrado um software de diagnóstico que através da interface de diagnóstico oferece comandos para deletar e para programar a memória de programa. Este software frequentemente é designado em inglês como "Bootloader".

[005] Para a comunicação com o aparelho de comando é usado

um aparelho de programação que através da peça contrária necessária é disponibilizado com a interface de diagnóstico no aparelho de comando. Um aparelho de programação desta maneira frequentemente também é designado como aparelho de diagnóstico ou teste de diagnóstico. Na prática, este aparelho de programação frequentemente é um computador pessoal CP como, por exemplo, um Laptop.

[006] Na programação de aparelhos de comando são transferidos volumes de dados bem grandes para códigos de programa, o que com o processo atual demanda muito tempo. Aparelhos de comando de veículos automotores frequentemente são aparelhos de comando universais que são programados na produção do veículo automotor, no fim da série individualmente para o respectivo veículo automotor. O tempo necessário para a programação do aparelho de comando é, portanto, um fator crítico para a produção do veículo automotor e deverá ser o mais curto possível.

[007] Também durante a operação do veículo automotor, na prática, terão de serem alterações (os chamados Upgrades) do software que normalmente é realizado em uma oficina. Também aqui, o tempo necessário para a nova programação do aparelho de comando deve ser o mais curto possível.

[008] O documento inicialmente citado DE nº 10 2006 016 891 A1 se ocupa com a alteração posterior do software de um aparelho de comando e diferencia entre um software de série e um software "Upload". Para diminuir o esforço para o recarregamento do software "Upload" é ali proposto prever duas regiões de memória separadas, ou seja, uma para o software de série e uma para o software Upload. Para ampliar a funcionalidade do aparelho de comando será depois necessário apenas transferir o software Upload na região da memória separada e não o software completo.

[009] Para a programação inicial do aparelho de comando, não

resultam assim quaisquer vantagens.

[0010] Constitui objeto da presente invenção, aprimorar o processo da espécie inicialmente citada, no sentido de que o tempo para a transferência de códigos de programa para a memória do aparelho de comando seja essencialmente reduzido.

[0011] Esta tarefa será solucionada pelas características indicadas na reivindicação 1. Modalidades vantajosas e ampliações da invenção podem ser depreendidas das reivindicações dependentes.

[0012] A ideia básica da presente invenção reside em que antes de registrar os códigos do programa na memória no aparelho de comando, todas as células da memória seriam ajustadas para um valor padrão (default); em seguida, os códigos de programa seriam transferidos pelo aparelho da programação, dentro do aparelho de programação, seriam submetidos a um processo de compressão de dados, isento de perdas, de preferência de acordo com o código Huffman,

[0013] em seguida, os códigos de programa comprimidos seriam transferidos para o aparelho de comando,

[0014] em seguida, os códigos de programa seriam descompactados e

[0015] finalmente, os códigos de programa compactados seriam memorizados na memória do aparelho de comando.

[0016] Pelo ajuste de todas as células da memória para o valor padrão é assegurado que não esteja presente na memória nenhum código de programa falho. Pela transmissão dos códigos de programa comprimidos, o volume de dados a ser transferido será essencialmente reduzido e o mesmo ocorre também com o tempo da transferência. Pela utilização de um processo de compressão de dados, sem perda, também não se apresentam perdas de dados. Para a compressão e descompressão, preferencialmente deverá ser usado uma codificação Huffman que alcança um fator de compressão muito elevado.

[0017] Segundo uma modalidade vantajosa da invenção, no ajuste de todas as células da memória para um valor padrão, este valor padrão será escolhido de tal maneira que contém um código de programa inválido (o chamado illegal Opcode). Em muitos microprocessadores, o seu valor está memorizado no microprocessador. De acordo com o microprocessador usado ou linguagem de programação usada, pode também ser outro valor que depois, em uma fase de inicialização, antes da transferência dos códigos de programa, será transferido pelo aparelho de programação para o aparelho de comando.

[0018] Segundo uma variante da invenção, é possível transferir todo o conteúdo da memória do aparelho de comando, do aparelho de programação para o aparelho de comando, ou seja, também o conteúdo de células de memória não usadas. Todavia, é preferido transferir apenas códigos de programas que forem necessários porque, células de memória não usadas ou regiões de endereços não usados, estarão previamente ocupadas com valor padrão. Desta maneira, o volume de dados a ser transferido será ainda adicionalmente reduzido.

[0019] A seguir, a invenção será explicitada em conexão com um exemplo de execução. As figuras mostram:

[0020] figura 1 um esquema principal de um aparelho de comando e de um aparelho de programação;

[0021] figura 2 um diagrama de fluxo dos passos do processo que decorrem na invenção;

[0022] figura 3a até figura3g uma diferenciação dos diferentes passos para a compressão de dados;

[0023] figura 4 uma representação de passos do processo da compressão de dados, através de um vetor auxiliar; e

[0024] figura 5 uma representação de uma árvore de decodificação.

[0025] O aparelho de comando eletrônico 1 da figura 1, designado

com ECU (Electronic control unit - unidade de controle eletrônica) possui uma memória de programa 2, conformada como memória que pode ser deletada e que possibilita proceder a registros. O aparelho de comando 1, através de uma interface 3, pode ser unido com um aparelho de programação 4, que pode ser, por exemplo, um PC como, por exemplo, um laptop. No aparelho de programação 4, estão memorizados códigos de programa que através da interface 3 devem ser transferidos para o aparelho de comando 1. Os códigos de programa estão previstos normalmente como arquivo binário ou arquivo hexa.

[0026] A "programação" do aparelho de comando 1 se verifica por serem transferidos os códigos de programa, a partir do aparelho de programa 4 para a unidade de comando 1, onde serão memorizados na memória do programa 2.

[0027] Até agora, a programação ou a reprogramação do aparelho de comando se realizava pelo fato de que todo o programa, com os diferentes códigos de programa, foi registrado na memória do programa 2. Células da memória, eventualmente antes ocupadas na memória de programa 2, serão sobrescritas no caso.

[0028] O processo representado na figura 2, de acordo com a invenção, trabalha como segue: em um primeiro passo S1, o aparelho de programa 4, através da interface 3, envia inicialmente um comando para a unidade de comando 1, o chamado "Bootloader" do aparelho de comando 1, com o qual a memória do programa 2 será deletada. De preferência, este processo de deletar implica em que todas as células da memória de programa 2 serão escritas com códigos de programa inválidos (illegal Opcode). Este código pode estar previamente ocupado como valor padrão (default) na unidade de comando 1. De acordo com o microcomputador usado na unidade de comando 1, também é possível, todavia, transferir este código correspondente do aparelho de comando 3 para o aparelho de comando 1.

[0029] Em um segundo passo S2, os códigos de programa memorizados no aparelho de programa 4 serão submetidos a um processo de compressão isento de perda, ou seja, serão codificados ou comprimidos.

[0030] Os requisitos formulados ao processo da compressão são o consumo mínimo de recursos, especialmente tempo de cálculo, memória de programa e memória de trabalho.

[0031] Muitos módulos de memória existentes somente podem ser programados sem vãos, ou seja, espaços. Em outros módulos da memória terão de ser escritas, no mínimo, regiões predeterminadas, sem espaços ou vãos. Daí resulta o requisito de que os dados descomprimidos, inicialmente precisam estar previstos ao menos em determinadas regiões, isentos de espaços ou vãos. A manipulação desses dados descomprimidos é mais simples quando chegarem isentos de espaços por toda a região da memória a ser programada. Como a memória de trabalho existente do aparelho de comando 1 normalmente é muito menor do que a memória de programa 2, não é possível fazer a descompressão de todos os dados, antes da programação. Portanto, torna-se necessário que os dados a serem transferidos cheguem isentos de espaços ou vãos.

[0032] Para a compressão sem perdas dos códigos de programa a serem transferidos, mostrou ser especialmente adequada à codificação Huffman, já conhecida, na qual, contrário a muitos outros processos de codificação, os dados a serem transmitidos chegam sem espaços, isto é, sem vãos. Com esta codificação ou compressão, poderá ser concretizada uma compressão e descompressão, isentas de perdas dos dados a serem transferidos. No caso, em princípio, frequentemente dados existentes serão codificados em palavras codificadas curtas e dados com menor frequência serão codificados em palavras codificadas mais extensas.

[0033] A eficiência do processo depende da distribuição dos dados a serem transferidos. Quanto mais frequentes estiverem presentes determinados dados, tanto mais frequentes serão usados códigos curtos, sendo, portanto, logrado um fator de compressão mais elevado.

[0034] Através de processos de classificação, será produzida uma "árvore", por meio da qual pode ser formado um "dicionário" para a codificação. A partir desta árvore, poderá ser formulada uma árvore correspondente para a descompressão. Esta árvore ou terá de ser também transferida para a unidade de comando 1 ou poderá ali já estar fixada. Nesta hipótese, naturalmente também para a codificação terá de ser usado o mesmo dicionário.

[0035] Uma variante adaptada, na qual sempre a árvore do código é novamente gerada e transferida, oferece melhores resultados na compressão e requer mais memórias de trabalho na unidade de comando 1, mas menor número de memórias de programa, onde for integrado o dicionário. De acordo com o caso de uso, poderá ser mais vantajosa a variante adaptiva ou não adaptiva.

[0036] Na variante adaptiva, a estrutura da árvore de decodificação será transferida para uma região não comprimida, no começo do fluxo de dados. Em seguida, virá a árvore de decodificação e depois os dados comprimidos.

[0037] De uma maneira lógica, a codificação se verifica para grandezas de dados que representam um múltiplo e um byte (8 Bits). O tamanho da árvore de decodificação cresce acentuadamente com o comprimento (L) das palavras a serem codificadas. A extensão dos sinais codificados, segundo o processo Huffman, pode ser maior para sinais raros do que o comprimento para o sinal não codificado. Como especialmente em uma codificação de 16 Bits, na prática, não ocorre em todos os sinais, a árvore de decodificação poderá ficar correspondentemente menor. Mesmo assim se verifica que regiões de dados

menores serão mais bem codificadas com 8 – bits.

[0038] A codificação de 16 Bits demonstrou em ensaios um fator de compressão mais elevado. Isto, todavia, somente é eficaz a partir de uma quantidade mínima de dados a serem transferidos, porque a própria árvore de decodificação também precisa ser transferida.

[0039] Na variante não adaptiva, de acordo com o local da memorização para a árvore de decodificação, terá de ser disponibilizada na memória de programa não volátil.

[0040] Em seguida, a codificação será explicada com base em um exemplo de execução, em relação com as figuras 3 a 5.

[0041] Inicialmente, para todos os sinais incidentes será determinada a sua frequência o que é explicado no seguinte exemplo.

Frequência	0	0	0	5	8	0	9	22
Sinal	0	1	2	3	4	5	6	7

[0042] No próximo passo serão classificados os sinais, de acordo com a frequência (figura 3a), o que pode ocorrer em uma sequência de classificação decrescente ou crescente. Na sequência decrescente resulta do exemplo acima:

Frequência	22	9	8	5	0	0	0	0
Sinal	7	6	4	3	0	1	2	5

[0043] Os sinais e os respectivos em "0", "1", "2" e "5" com a frequência "0" podem ser eliminados.

[0044] As frequências dos dois sinais com a frequência menor serão somadas e produzem uma nova "folha" na árvore (figura 3b e figura 3c). Este será aplicado como elemento adicional em uma disposição de estruturas correspondentes e os índices, ou seja, os indicadores serão registrados de acordo com a implementação. No caso, em uma implementação exemplificada "folhas" não são realmente classificadas, porém serão depositadas em um vetor auxiliar, em forma de indicadores, ou seja, índices das folhas (figura 4). Serão apenas classificadas

aquelas que forem diminuir o trabalho na classificação.

[0045] Na geração de uma nova folha, esta será acrescentada e os dois últimos índices serão removidos e no lugar deles será registrado o índice da nova folha (figuras 3d e 3e).

[0046] Isto é repetido durante tanto tempo, até que permanece ainda um último registro que é direcionado para a ponta da árvore (figuras 3f e 3g).

[0047] Nas figuras 3a, 3d e 3f está representada em forma linear uma árvore correspondente, porque nesta forma é fácil de ser implementado. Com o auxílio desta árvore, podem agora serem geradas as tabelas de codificação e decodificação.

[0048] Baseado no exemplo acima e da figura 3 auto-explicativa resultará então a seguinte tabela de codificação.

Sinal	Código	Extensão
0	-	
1	-	
2	-	
3	0-1-0	3
4	1-1-0	3
5	-	
6	0-0	2
7	1	1

[0049] Por exemplo, para determinar o código para "3", será necessário passar através de 0-1-0, isto é, o comprimento é 3. Daí resulta para os diferentes sinais, o seguinte fator de compressão.

Sinal	Extensão	Número	Bits
3	3	5	15
4	3	8	24
6	2	9	18
7	1	22	22
			79 total

[0050] Aqui são necessários, portanto, 79 Bits para codificar os sinais. No caso de uma codificação de 8 Bits seriam necessários, por sua vez, 352 sinais. Desta maneira, resulta um fator de compressão aproximado de 0,22.

[0051] A árvore de decodificação necessária para decodificar será formada pela ponta da árvore (figura 5). Para sinais não existentes, não serão gerados lançamentos. Nesta árvore trata-se novamente de uma relação encadeada. Os códigos a serem decodificados serão analisados da direita para a esquerda, de acordo com a tabela de codificação acima. Nesta direção também serão transmitidos os códigos. No exemplo acima será transferido, portanto, o 4 como 0-1-1.

[0052] Para poder reconhecer se é preciso decodificar em sequência, ou se foi encontrado um sinal válido, as raízes na árvore serão correspondentemente caracterizadas. Para tanto, será suficiente um Bit adicional, por cada lançamento, a fim de diferenciar um sinal válido, de um indicador. Os lançamentos N1 e N2 contêm correspondentemente indicadores ou um sinal decodificado, o que é mostrado na figura 3. O percurso esquerdo significa ali "0" e o direito "1". Esta alocação terá de ser feita de modo idêntico para a codificação. Se no exemplo da figura 3 for recebida a sequência 0-1-1, se acompanhará na decodificação (figura 5) os lançamentos N1, N2, partindo da ponta da árvore (folha esquerda) e será recebido como resultado, o "4".

[0053] O número das folhas na árvore de decodificação é menor ou igual ao número dos sinais a serem codificados. No exemplo mostrado, serão suficientes três folhas para uma árvore de decodificação de quatro códigos. Na implementação escolhida, os lançamentos N1, N2 ou contêm um indicador ou índice, ou contêm os sinais codificados. Para o número dos Bits dos índices resultará, portanto:

[0054] $B = \text{Log}(L) / \text{Log}(2) + 1 = 9$ (para codificação de 8 Bits).

[0055] Para arredondar ao próximo número maior inteiro. No caso,

L é o comprimento das palavras a serem codificadas (por exemplo, 8 ou 16).

[0056] O Bit adicional é necessário para diferenciar índices de sinais válidos. Para cada folha são necessários os lançamentos N1 e N2. Desta maneira, resultará um comprimento mínimo da tabela de decodificação de $3 \times 2 \times 9 = 54$ Bits.

[0057] Adicionalmente, ainda terá de ser transmitida a informação sobre a extensão da tabela de decodificação, bem como o número dos Bits B por cada lançamento para a unidade de comando.

[0058] Retornando à figura 2, os códigos de programa assim comprimidos, segundo Huffman, no passo S3, serão transferidos do aparelho de programação 4, através da interface 3, para a unidade de comando 1, em seguida, no passo S4, será feita a correspondente descompressão e no passo S5 será feito o lançamento na memória de programa 2. Finalizando, ainda poderá ser feito um teste dos códigos de programa transmitidos, por exemplo, através de um exame de checar de soma ou um exame redundante cíclico (CRC; cyclic redundancy check) (comparar passo S7). Caso desejado, o resultado deste exame poderá ser informado em processo de retorno, em um passo S8, para o aparelho de programação 4.

[0059] O processo de compressão descrito é tecnicamente simples, porém muito eficaz. Para a compressão são essencialmente apenas necessários processos de classificação simples. A descompressão requer apenas um acompanhamento simples de lançamentos na árvore de decodificação. A compressão será tanto melhor quanto mais irregular se apresentarem os sinais codificados. Nos códigos de programa, a serem codificados para aparelhos de comando de veículos automotores, se trata de códigos para microprocessadores, nos quais, de acordo com a experiência, determinados códigos (Opcodes) aparecem nitidamente com maior frequência do que outros, de manei-

ra que o processo de compressão descrito pode atuar muito bem. De acordo com o grau da memória de programa poderá ser adicionalmente lograda uma economia considerável pela não transferência de códigos de programas para regiões não usadas. Pelo emprego de códigos de programa (illegal Opcodes) inválidos para estas regiões, se recebe uma segurança adicional na operação, por ocasião de falhas ou erros de software, já que os microprocessadores, no caso de códigos de programa inválidos, realizam uma interrupção que pode ser usada para o reconhecimento da falha.

[0060] Como na prática, a memória de programa do aparelho de comando não está totalmente cheia e apresenta maiores espaços (vãos) na região dos endereços, será vantajoso de preencher também estes espaços vazios com o valor para o entre "illegal Opcode".

REIVINDICAÇÕES

1. Processo para a transferência de códigos de programa para uma memória de programa (2) de um aparelho de comando (1), especialmente um veículo automotor, **caracterizado pelo fato de que compreende os seguintes passos:**

a) ligação de uma interface (3) de um aparelho de comando (1) para um aparelho de programação (4), contendo os códigos de programa;

b) ajustar todas as células da memória de programa (2) do aparelho de comando (1) para um valor padrão;

c) compressão do código de programa do aparelho de programação (4), de acordo com um processo de compressão de dados, sem perdas;

d) transferência do código de programa comprimido para o aparelho de comando (1);

e) descompressão do código de programa recebido no aparelho de comando (1); e

f) memorização do código de programa descomprimido na memória de programa (2) do aparelho de comando (1).

2. Processo, de acordo com a reivindicação 1, **caracterizado pelo fato de que** em todas as células de memória não utilizadas da memória de programa (2) do aparelho de comando (1) será registrado um código de programa inválido.

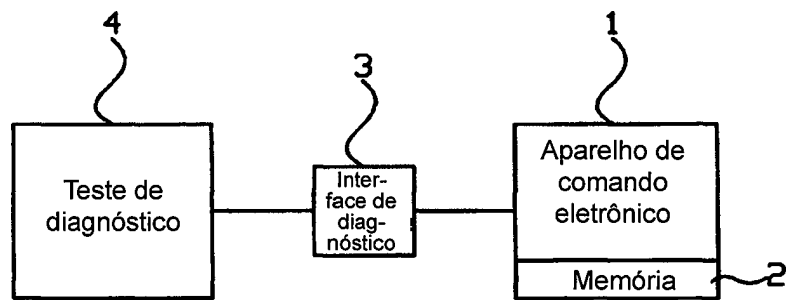
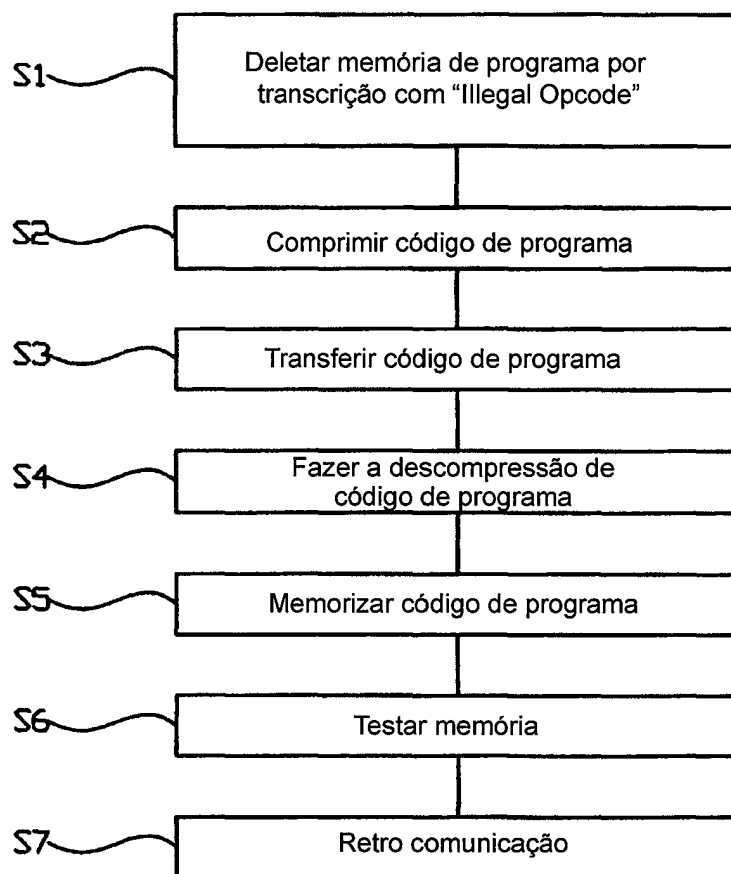
3. Processo, de acordo com a reivindicação 1 ou 2, **caracterizado pelo fato de que** o valor padrão mencionado no passo b), da reivindicação 1 é um código de programa inválido.

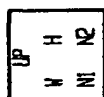
4. Processo, de acordo com a reivindicação 2 ou 3, **caracterizado pelo fato de que** o código de programa inválido é transferido do aparelho de programação (4) para o aparelho de comando (1).

5. Processo, de acordo com qualquer uma das reivindica-

ções 1 a 4, **caracterizado pelo** fato de que do aparelho de programação (4) serão transferidos apenas os códigos de programa válidos, em sequência, para o aparelho de comando (1), enquanto que o valor do código de programa inválido, no máximo, será transferido uma vez.

6. Processo, de acordo com qualquer uma das reivindicações 1 a 5, **caracterizado pelo** fato de que a compressão e a descompressão dos códigos de programa se verificará de acordo com a codificação Huffman.

Fig. 1Fig. 2



1- Classificado em sequência decrescente, conforme a frequência. Códigos não usados, não precisam ser considerados.

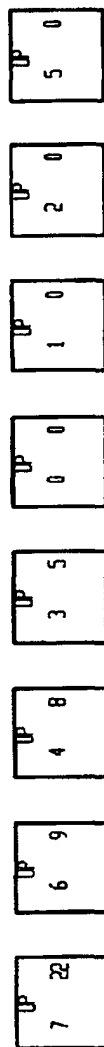


Fig. 3a

2- Os dois códigos com a menor frequência serão reunidos em uma nova folha que contém como frequência a soma das diferentes frequências dos sinais

Representação linear.

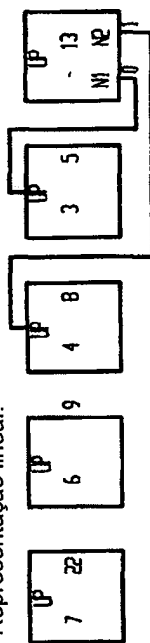


Fig. 3b

Representação da árvore

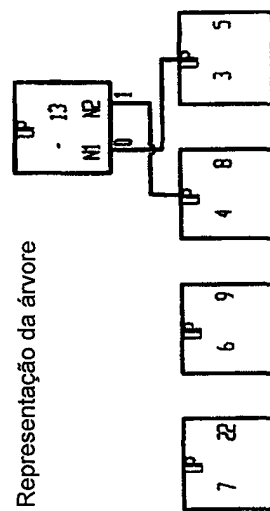


Fig. 32

3. A Nova folha é integrada na classificação.
As duas folhas direitas, com as frequências menores, serão novamente unidas em uma folha nova.

Representação linear

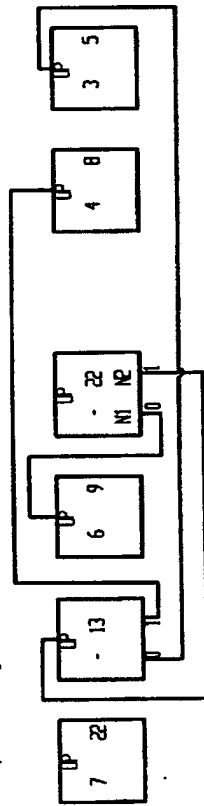


Fig. 3d

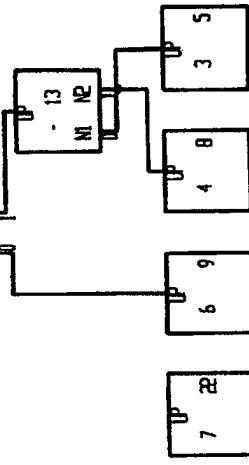
Representação
da árvore

Fig. 3e

4. Agora, todas as folhas são enlaçadas desde a raiz até a ponta da árvore. A árvore está pronta.

Representação linear

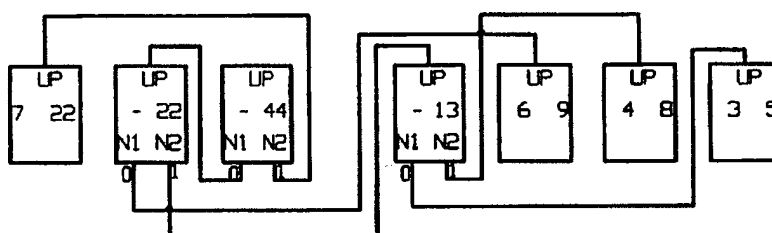


Fig. 3f

Representação da árvore

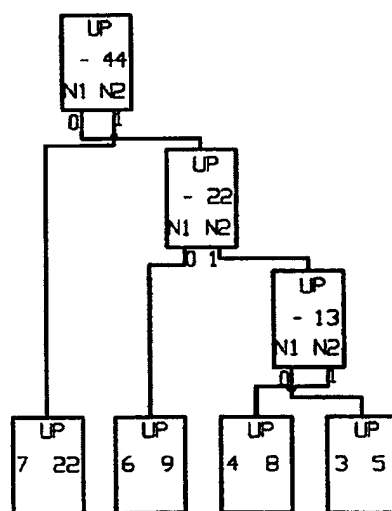


Fig. 3g

Para acelerar os processos de classificação foi selecionada uma implementação por meio do vetor auxiliar de indicadores, ou seja, de índices. Para acelerar o processamento, não será classificada a própria árvore, porém os índices, ou seja, os indicadores no vetor auxiliar.

Situação após o passo

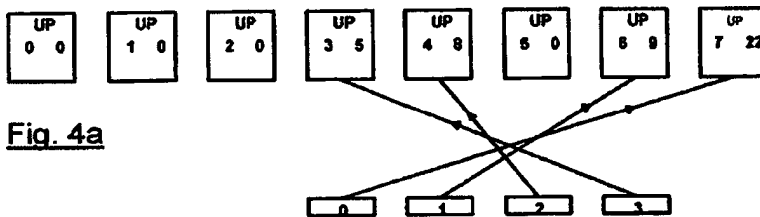


Fig. 4a

Situação após o passo
Códigos não usados não são representa

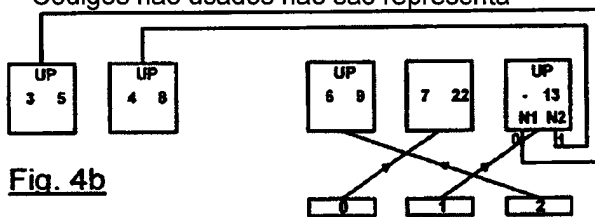


Fig. 4b

Situação após o passo

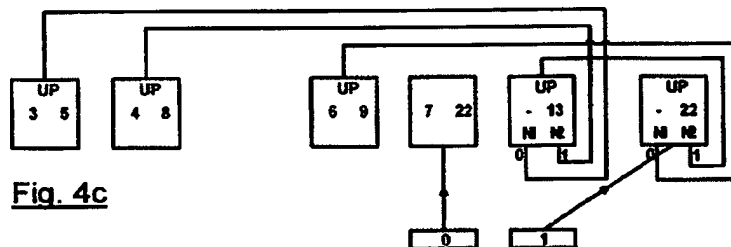


Fig. 4c

Situação após o passo

Permanece como resíduo o indicador na ponta da árvore

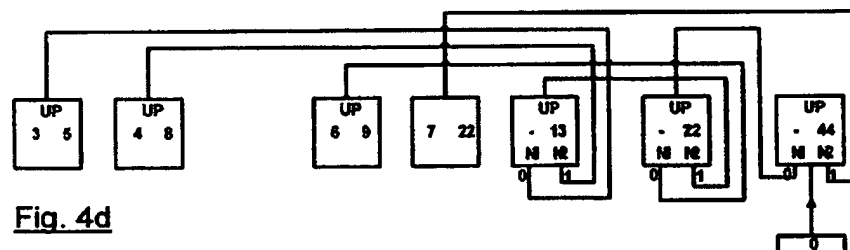


Fig. 4d

Árvore Huffman para decodificação "0" significa que terá de ser seleccionado o percurso direito, "1" significa que terá de ser seleccionado o percurso esquerdo. Esta alocação deverá ser feita de modo idêntico à codificação.

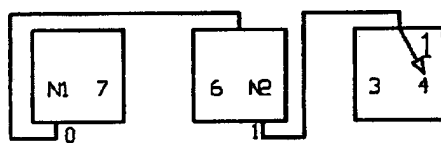


Fig. 5