UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

DEPARTAMENTO DE TELEMÁTICA

UMA CONTRIBUIÇÃO À DECODIFICAÇÃO POR DECISÃO SUAVE DE CÓDIGOS DE BLOCO

HÉLIO CÉSAR ALVES SEABRA SALLES

Orientador:
Prof. Dr. Walter da Cunha Borelli

Banca Examinadora:
Prof. Dr. Walter da Cunha Borelli - FEE/UNICAMP
Prof. Dr. Dalton Soares Arantes - FEE/UNICAMP
Prof. Dr. Ricardo Menezes Campello - FEE/UFPe
Prof. Dr. Ivanil S. Bonatti - suplente - FEE/UNICAMP

Este exemplar corresponde à redação final da tese defendida por Hélio César Alves Seabra Salles e aprovada pela Comissão Julgadora em 18 12 90.

Orientador

Tese apresentada à Faculdade de Engenharia Elétrica- UNICAMP, como requisito parcial para obtenção do Título de Mestre em Ciências.

Campinas, 18 de Dezembro de 1990.

A Miriam, Marina e Daniel, minha família.

A meus pais, avós e irmãos.

Não posso deixar que as idéias sejam só idéias. Tenho de empurrá-las até descobrir como torná-las reais.

-Nolan Bushnell- fundador da Atari

A sabedoria é a coisa principal; adquire, pois, a sabedoria; sim, com tudo o que possuis adquire o conhecimento. Exalta-a, e ela exaltará; e, abraçando-a tu, ela te honrará.

-Provérbios 4: 7-8

AGRADECIMENTOS

Agradeço a todos que direta ou indiretamente contribuíram para a realização deste trabalho. Em especial, devo agradecer:

a meus pais MARCOS e ELISSA e meus avós DINAH, HÉLIO e MILA pelo esforço contínuo para minha educação;

a minha esposa MIRIAM e meus filhos MARINA e DANIEL pelo amor e paciência de me suportarem;

a meu orientador WALTER BORELLI pela contínua e cuidadosa orientação;

a meus gerentes LAURO GOMES, pela confiança em meu trabalho, e JURANDIR PITSCH, por sua orientação, ensinamentos e muito frutíferas discussões técnicas;

ao Prof. DALTON SOARES ARANTES e a ANTÔNIO CLÁUDIO FRANÇA PESSOA, pelas exaustivas simulações iniciais de desempenho de decodificadores para uso no projeto SAMSAT, dentro do convênio TELEBRÁS-UNICAMP.

ao técnico FLÁVIO BUENO BRANDÃO, pela sua dedicação, paciência, esforço e cuidado na realização de diversas atividades de laboratório necessárias na execução deste trabalho;

ao estagiário e depois engenheiro ROGÉRIO CARDOSO, que muito contribuiu na depuração das placas e programou todas as rotinas de teste necessárias para validar este trabalho;

ao estagiário ANDRÉ C. MARCHIORI, que fez as rotinas de programação das memórias PROMs e EPROMs usadas no decodificador;

à Sra. SILVANA MARIANI, que cuidadosamente digitou e fez a editoração eletrônica deste trabalho;

à TELEBRÁS, que deu todo o suporte financeiro para a execução deste trabalho;

a todos os colegas do projeto SAMSAT.

RESUMO

Mostramos, ao longo deste trabalho, como projetamos uma arquitetura de máquina digital capaz de realizar a decodificação por decisão suave do código de Golay estendido (24,12), usando o algoritmo dois de Chase, dentro de um sistema de transmissão de dados via satélite com acesso múltiplo por divisão temporal (AMDT). Esta arquitetura é tal que pode ser implementada com a tecnologia de CIs digitais "off-the-shelf" disponível no mercado e atingir taxas de transmissão de até 1 Mbit/s. O custo e tamanho do circuito final são altamente competitivos dentro da classe de desempenho a que pertence esta técnica de codificação-decodificação.

ABSTRACT

We show throughout this thesis how to design a digital machine architecture suitable to decode by soft-decision the extended Golay block code (24,12), using the Chase Algorithm II, to be used as a PCB, component of a TDMA data transmission system by satellite. This architecture may be implemented using off-the-shelf digital ICs and still reach 1 Mbit/s transmission rates. Cost and chip count of final circuitry are highly competitive within the class of performance to which this FEC technique pertains.

CONTEÚDO

CAPÍTULO 1

INTRODUÇÃO

1.1. ESCOPO DA TESE

O trabalho que apresentaremos no que se segue pertence ao campo das Comunicações Digitais ou Transmissão Digital, dentro da área de Teoria da Informação, nos tópicos da Teoria da Codificação de Canal e Decodificação.

Dedicamo-nos, especificamente, ao estudo e ao projeto de uma arquitetura de máquina digital capaz de realizar a Decodificação por Decisão Suave de um código corretor de erros, linear, binário, sistemático e de bloco, chamado Código de Golay Estendido, com parâmetros (n, k, d) iguais a (24, 12, 8), onde n é o comprimento em bits do vetor código, k é o comprimento em bits da palavra a ser codificada e d é a distância mínima de Hamming das palavras-código. Doravante, chamaremos este código de CGE (24, 12).

A técnica de decodificação lança mão de um algoritmo proposto originalmente por CHASE [13] em 1971 e que doravante será chamado Algoritmo 2 de Chase, porque este propôs, junto com este, outros dois algoritmos e os chamou de algoritmos 1, 2 e 3.

A arquitetura a ser proposta aqui é bloco constituinte do modem (modulador-demodulador) de um sistema de transmissão de dados via satélite, com técnica de acesso múltiplo por divisão temporal (AMDT).

Esta arquitetura pode ser implementada (e efetivamente o foi) usando-se como componentes CIs digitais "standard", disponíveis hoje comercialmente. Estes componentes "off-the-shelf" são CIs de integração em baixa e média escalas (SSI e MSI) em tecnologia C-MOS, da série HCMOS compatível com TTL, CIs das séries TTL-LS (Low Schottky) e TTL-S (Schottky), e CIs de memórias RAMs em tecnologia N-MOS.

O hardware resultante pode ser usado para taxas de transmissão de 1068 kbit/s, sendo a máxima velocidade de operação limitada somente pelos tempos de acesso das memórias disponíveis (2,5 Mbit/s para 25 ns de tempo de acesso).

O custo, complexidade e número de CIs do circuito final são altamente competitivos, dentro da classe de desempenho a que pertence esta arquitetura.

Os pontos supracitados (custo, complexidade e número de CIs versus velocidade de operação) merecem destaque pois é onde reside o que julgamos ser a originalidade da tese: apesar de Chase ter concebido, e simulado em computador, estes algoritmos em 1971, não encontramos referências ou trabalhos ou equipamentos que os implementassem em hardware, voltados a taxas de transmissão altas. Uma variante do algoritmo de Chase, proposta por HACKETT [16], foi implementada, por ele mesmo, em software, usando microprocessador, para taxas de transmissão muito baixas.

Além disso devemos ressaltar que a arquitetura aqui proposta utiliza conceitos e técnicas recentemente introduzidas no projeto de máquinas digitais, tais como processamento em pipeline, processos paralelos e circuitos auto-temporizados.

Este trabalho teve início em julho de 1986, quando já se dispunha, no Centro de Pesquisa e Desenvolvimento da TELEBRÁS (CPqD-TELEBRÁS), de uma primeira Especificação de Sistema do sistema AMDT de transmissão de dados via satélite, batizado com o nome de SAMSAT.

Nesta primeira especificação, o CGE (24, 12) e o algoritmo 2 de Chase para a decodificação suave, usando quantização de oito níveis eram adotados com base nas recomendações de um estudo feito por PESSOA e ARANTES [31] dentro de um convênio entre o CPqD-TELEBRÁS e a FEE-UNICAMP.

Resumidamente, o nosso trabalho consistiu nas seguintes fases:

– estudo de algoritmos alternativos para a decodificação por decisão suave de códigos de bloco, usando o CGE (24, 12);

– validação do algoritmo 2 de Chase como melhor alternativa para a decodificação do CGE (24, 12), levando em conta as restrições impostas pela Especificação de Sistema AMDT;

– concepção da arquitetura para o decodificador por decisão suave, usando o algoritmo 2 de Chase;

– projeto lógico do hardware, a nível de circuitos e seleção de CIs;

– montagem do hardware projetado em placa de prototipagem do tipo "wire-wrap";

– "debug" e reprojeto de hardware, com alterações na versão "wire-wrap" do decodificador, para checar a viabilidade do projeto;

– projeto do "lay-out" da placa de circuito impresso, na primeira versão;

– montagem e "debug" de dez placas de circuito impresso, na primeira versão, caracterização de desempenho e testes de integração no sistema para estas dez placas;

– reprojeto de "lay-out" da placa, já na sua versão definitiva, e montagem, "debug" e testes de outras dez placas de circuito impresso, nesta versão definitiva, assegurando assim sua viabilidade como produto industrial.

Da exposição acima, pode-se ver que não demos ênfase ao estudo de outros códigos alternativos ao CGE (24, 12), por duas razões: a primeira é a existência de estudos anteriores que justificavam plenamente sua adoção no sistema AMDT e a segunda é que uma mudança do código e do sistema de codificação alteraria radicalmente a composição e parâmetros do sistema AMDT, tais como estrutura de quadros e surtos, comprimento mínimo de surtos, eficiência do sistema em termos de símbolos, e outros, o que traria impactos consideráveis sobre outras unidades funcionais do sistema e, portanto, sobre a Especificação de Sistema e sobre outros trabalhos que já estavam em andamento. Conseqüentemente, nesta tese, não abordaremos esta temática comparativa sobre códigos. Todavia, mostraremos que o CGE (24, 12) é um bom código para o sistema AMDT e também procuraremos generalizar a aplicação da arquitetura proposta para o decodificador a outros códigos de bloco, com base em seus parâmetros (n, k, d) e com base nos parâmetros do quantizador.

Para melhor situar o tema de nosso trabalho, apresentamos na próxima seção o problema geral da Codificação de Canal e sua decodificação associada, seguido de uma de suas maiores aplicações, que é a otimização de canais no transponder e redução de custos das estações terrenas em sistemas de transmissão digital via satélite. Isto será feito de forma resumida e aproveitaremos para relembrar alguns conceitos básicos. Em seguida, em outra seção, exporemos a divisão dos capítulos deste trabalho.

A próxima seção, revisão de conceitos básicos, foi incluída apenas para orientar quem porventura não esteja acostumado aos conceitos de códigos corretores de erro e transmissão via satélite, podendo-se ir para a seção 1.3 caso já se esteja acostumado aos conceitos e nomenclaturas.

1.2. REVISÃO DE CONCEITOS BÁSICOS

1.2.1. CÓDIGOS CORRETORES DE ERRO

A teoria da codificação de canal tem como objetivo primordial a procura de métodos eficazes para transmitir e receber mensagens digitais através dos diversos canais existentes (meio físico de transmissão) de forma a minimizar a distorção das mensagens recebidas, i.e., de forma a se ter na recepção o menor número possível de erros nas mensagens originalmente transmitidas.

Assim, os códigos corretores de erro baseiam-se em técnicas de processamento de sinais digitais que aumentam a confiabilidade das mensagens recebidas e, portanto, a qualidade da comunicação também é aumentada.

Todos os códigos corretores de erro, do mais simples ao mais complexo, são obtidos usando-se dois fatos básicos, a saber:

a) uso de redundância, i.e., toda mensagem digital codificada possui, além dos bits de informação, dígitos adicionais (redundantes), para aumentar as diferenças entre as várias mensagens; assim, dificulta-se a ação corruptora do ruído de transformar uma mensagem em outra;

b) uso do efeito de redução do valor médio de ruído através da técnica de se fazer com que os dígitos de redundância dependam de vários bits de informação espalhados no tempo; assim, dificulta-se a ação do ruído, já que é mais provável que ele possa interferir em dois ou mais símbolos adjacentes do que em símbolos distantes no tempo.

Como já dissemos na seção anterior, estamos especificamente interessados nos códigos corretores de erros lineares, binários, sistemáticos e de bloco. Esta seção apresenta de forma resumida o que é esta subdivisão dos códigos corretores de erro. A teoria geral dos códigos corretores de erro pode ser encontrada nas referências ([11], [14], [27], [33], [34] e [44]).

Genericamente, os sistemas de comunicação que empregam códigos corretores de erro podem ser representados resumidamente pelo diagrama de blocos da figura 1.1. abaixo:

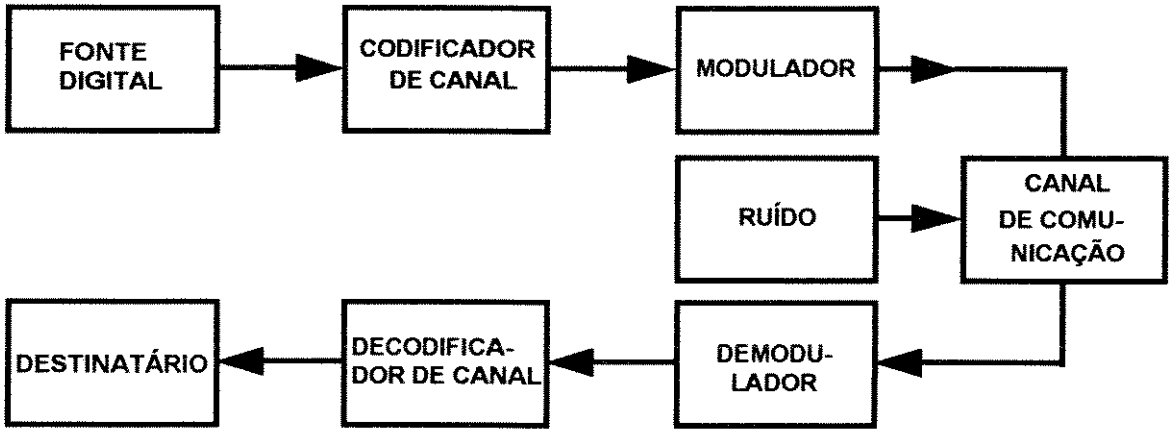


FIGURA 1.1. DIAGRAMA DE BLOCOS DE UM SISTEMA DE COMUNICAÇÕES EMPREGANDO CÓDIGO CORRETOR DE ERROS

Para o presente trabalho, as definições dadas a seguir são suficientes para nosso propósito de rever o que é um código corretor de erros linear, binário, sistemático e de bloco. Foram baseadas na ref. [33].

DEFINIÇÃO 1: CÓDIGO (CORRETOR DE ERROS)

Um código corretor de erros é obtido através de um mapeamento de uma seqüência de k símbolos de informação em uma outra seqüência, mais longa, com n dígitos ($n > k$), chamada palavra-código.

Normalmente, dados n e k, o mapeamento é feito de forma a maximizar a distância de Hamming entre duas palavras-código quaisquer (ver definição 2 a seguir).

Os comprimentos n e k podem ser tão grandes quanto se queira, inclusive semi-infinitos.

DEFINIÇÃO 2: DISTÂNCIA DE HAMMING E PESO DE HAMMING

A distância de Hamming entre duas ênuplas quaisquer $\mathbf{C_1}$ e $\mathbf{C_2}$, $d_H[\mathbf{C_1},\mathbf{C_2}]$, é o número de posições em que as mesmas diferem. Assim, no caso binário, $d_H[\mathbf{C_1},\mathbf{C_2}]$ pode ser expressa pela equação abaixo:

$$d_H[\mathbf{C_1},\mathbf{C_2}] = \sum_{i=1}^{n}(C_{1i} \oplus C_{2i}) \qquad (1.1)$$

onde:

C_{1i}, C_{2i} – i-ésimo dígito da palavra $\mathbf{C_1}$ ou $\mathbf{C_2}$.
$\oplus$ – operação booleana **"** ou exclusivo**"**

O peso de Hamming, W_H, de uma palavra qualquer é, por definição, o número de dígitos não-nulos da palavra. Portanto, para uma palavra binária de comprimento n, C_1, o peso de Hamming pode ser expresso pela equação a seguir:

$$W_H[\mathbf{C_1}] = \sum_{i=1}^{n} C_{1i} \qquad (1.2)$$

A distância de Hamming e o peso de Hamming são muito úteis na análise dos códigos corretores de erro.

DEFINIÇÃO 3: CÓDIGO DE BLOCO

Um código corretor de erros é definido como sendo um código de bloco se, para cada seqüência (ou bloco) de k símbolos de informação presentes à entrada do codificador, a seqüência de n dígitos presentes à saída do codificador depende somente dos k símbolos presentes à entrada. Ou seja, a palavra-código de comprimento n é função exclusiva da palavra de informação de comprimento k.

A nomenclatura para este código de bloco é usualmente feita dizendo-se que temos um código (n, k) com taxa de codificação R igual a:

$$R = k/n \tag{1.3}$$

Existe outra grande família de códigos, chamados códigos de árvore, onde, para cada k símbolos de informação de entrada, o codificador gera n dígitos à saída que dependem não só dos k símbolos presentes à entrada mas também de símbolos de informação anteriores. Então, para o caso de símbolos binários de entrada e saída, vistos pelo ângulo da teoria de circuitos lógicos, podemos afirmar que o codificador para um código de bloco é um circuito com estado único, ou seja, pode ser um circuito combinatório, enquanto o codificador para um código de árvore é uma máquina de múltiplos estados, ou seja, é sempre um circuito seqüencial.

A definição ([33], [14]) de um código de bloco é a seguinte: código de bloco é um conjunto de M seqüências, com comprimento n finito, de símbolos de um canal q-ário. Ou seja, é um conjunto de M ênuplas q-árias. Normalmente, temos a seguinte condição para M:

$$M = q^k \qquad , k < n \tag{1.4}$$

DEFINIÇÃO 4: CÓDIGO BINÁRIO

Um código corretor de erros é binário se os símbolos das palavras-código são binários. Para este alfabeto binário, q=2.

DEFINIÇÃO 5: DISTÂNCIA MÍNIMA E CAPACIDADE CORRETORA DOS CÓDIGOS DE BLOCO BINÁRIOS

A distância mínima (de Hamming) de um código de bloco é definida como sendo a mínima distância de Hamming entre quaisquer duas palavras-código. É denotada por $dMIN$ ou d , indistintamente, ao longo deste trabalho.

Para um código qualquer define-se a capacidade corretora como sendo igual a t se este código é capaz de ser decodificado de forma que seja possível se fazer a correção de todos os padrões de erro com t ou menos erros em uma palavra-código qualquer.

No caso de um código de bloco pode-se mostrar que sua capacidade corretora é t se e somente se sua distância mínima de Hamming, $dMIN$, respeitar a seguinte condição:

$$d_{MIN} = 2 \cdot t + 1 \tag{1.5}$$

DEFINIÇÃO 6: CÓDIGOS DE BLOCO LINEARES:

Formalmente, um código de bloco linear (n,k) é definido como sendo um sub-espaço vetorial, de dimensão k, do espaço vetorial formado por todas as ênuplas q-árias ([33], cap. 3).

Portanto, todo código linear sempre possui a palavra nula **0**, e toda e qualquer combinação linear de palavras-código gera outra palavra-código e qualquer palavra-código multiplicada por qualquer escalar pertencente ao campo (alfabeto de símbolos) também é palavra-código.

Por causa disto, para um código de bloco linear, a distância mínima do código é igual ao mínimo peso obtido entre todas as palavras-código exceto a palavra nula.

Um código de bloco linear **C** pode ser completamente definido por uma matriz geradora **[G]** de dimensão (kxn) composta por uma base vetorial qualquer, de k vetores, para o sub-espaço vetorial formado pelo conjunto de palavras-código de **C**.

Como todo sub-espaço, o código C tem um espaço nulo, cuja matriz geradora, denotada por **[H]**, de dimensão nx(n–k), é tal que o produto

$$[\mathbf{G}][\mathbf{H}]^T = [\mathbf{0}] \tag{1.6}$$

onde **[0]** é a matriz nula kx(n-k). **[H]** é chamada de matriz de verificação de paridade de **C**, onde cada linha da matriz **[H]** é uma equação generalizada de verificação de paridade.

DEFINIÇÃO 7: CÓDIGO SISTEMÁTICO

Um código de bloco é sistemático se e somente se os k primeiros dígitos da palavra-código de n dígitos forem iguais aos k bits de informação da seqüência de entrada.

Os (n–k) dígitos restantes são dígitos de paridade obtidos através de combinações lineares de bits de informação.

Assim, a matriz [G] de um código sistemático sempre será da forma:

$$[\mathbf{G}] = [\mathbf{I_k} | \mathbf{P}] \tag{1.7}$$

onde

$[\mathbf{I_k}]$ = matriz identidade kxk
[P] = matriz kx(n–k)

DEFINIÇÃO 8: CÓDIGO CÍCLICO E POLINÔMIO GERADOR

Um código de bloco é *cíclico* se e somente se toda e qualquer palavra-código $\mathbf{C_1} = (a_{n-1}, a_{n-2}, \ldots, a_2, a_1, a_0)$ gerar outra palavra-código $\mathbf{C_2}$ através de um deslocamento unitário e cíclico à direita de seus componentes, i.e., $\mathbf{C_2} = (a_0, a_{n-1}, \ldots, a_3, a_2, a_1)$.

Neste trabalho, apesar de não usarmos o código de Golay na forma cíclica, mas sim o código de Golay estendido CGE (24, 12) em sua forma sistemática, é importante ressaltarmos sua relação com códigos cíclicos e polinômios geradores pois

estes últimos ainda podem ser aproveitados para minimizar o hardware do codificador ou do cálculo da síndrome.

Todo código cíclico pode ser caracterizado por um *polinômio gerador* $g(x)$, que se define como sendo um polinômio que é fator de todas as palavras-código não-nulas, quando postas em suas formas polinomiais

$$C_i(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0$$

DEFINIÇÃO 9: SÍNDROME E DECODIFICAÇÃO POR MÍNIMA DISTÂNCIA

Define-se síndrome como sendo um vetor de dimensão $(n-k)$ obtido do produto de uma ênupla **V** qualquer com a matriz verificação de paridade transposta:

$$\mathbf{S} = \mathbf{V}[\mathbf{H}]^T \tag{1.8}$$

Assim, teremos que **S** é nulo se e somente se **V** for palavra-código.

Para decodificadores de mínima distância de códigos de bloco lineares, a síndrome é única para cada padrão de erro corrigível e pode-se, portanto, realizar a decodificação através de uma tabela que relacione síndrome e padrão de erros.

DEFINIÇÃO 10: ARRANJO PADRÃO E CLASSES LATERAIS

Seja um código binário linear **C** (n,k) e suas palavras-código $\mathbf{0}$, $\mathbf{c}_1$, $\mathbf{c}_2$, ..., $\mathbf{c}_{2k}$.

Define-se *arranjo-padrão* para este código como sendo uma tabela decodificadora formada por todas as ênuplas binárias dispostas da seguinte maneira:

1- as palavras-código são dispostas na primeira linha, com a palavra nula à esquerda.

2- das ênuplas restantes, escolhe-se uma qualquer, representada aqui por $\mathbf{d}_1$, para ser colocada na segunda linha, logo abaixo da palavra nula $\mathbf{0}$. Completa-se esta linha com as ênuplas correspondentes às somas de $\mathbf{d}_1$ com cada palavra-código $\mathbf{c}_i$ colocadas sob estas mesmas palavras-código $\mathbf{c}_i$. Normalmente, para uma tabela decodificadora bem construída, a palavra $\mathbf{d}_1$ escolhida seria uma das mais prováveis de serem recebidas caso fosse transmitida a palavra $\mathbf{0}$, i.e., uma cujo peso fosse baixo.

3- repete-se o passo acima, formando diversas outras linhas com outras ênuplas $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, etc. até que todas as ênuplas possíveis apareçam uma única vez no arranjo.

A cada linha do arranjo-padrão dá-se o nome de *classe lateral* ou *coset*, sendo que cada ênupla da primeira coluna é chamada *líder de classe lateral*.

1.2.1.1. O CÓDIGO DE GOLAY

Dada sua importância teórica e sua importância prática na implementação do sistema AMDT, vamos rever agora as principais características do código de Golay ([33], [14], [27], [28], [34], [44], [21]).

Em 1949, Golay descobriu o único código perfeito e não-trivial binário além dos de Hamming, conhecido até hoje. Um código linear de bloco é dito perfeito se, para algum t inteiro, ele possuir todos os padrões de erro com pesos menores ou iguais a t, e nenhum outro, como líderes de coset (também conhecidos como líderes de classes laterais) ([33], cap. 4). Códigos triviais são os códigos de repetição simples (2m+1,1) e os códigos de um único dígito de paridade (n,n–1).

Golay, procurando códigos perfeitos, verificou que a igualdade (1.9) abaixo:

$$\binom{23}{0}+\binom{23}{1}+\binom{23}{2}+\binom{23}{3}=2^{11} \qquad (1.9)$$

indicava que era possível a existência de um código perfeito (23, 12) que corrigiria todos os padrões de t=3 erros ou menos, tendo, por isso, d_{MIN}=d=7. A igualdade refere-se à distribuição dos padrões de erros dentro do arranjo-padrão de classes laterais (cosets) do código ([14], cap. 1; [33], cap. 2).

O código de Golay (CG) (23, 12) pode ser estendido ao receber um dígito de verificação de paridade total, calculado pela soma módulo 2 (X-OR) dos 23 dígitos e aumentar a distância mínima para d_{MIN}=7+1=8 e gerar o código de Golay estendido (CGE) (24, 12).

Peterson & Weldon ([33], cap. 5) fornecem a distribuição de peso das palavras-código de CG (23, 12) e CGE (24, 12), mostrada na tabela 1.1.

Peso W_H (C_i)	Nº de Palavras com este Peso	
	CG (23, 12)	CGE (24,12)
0	1	1
7	253	-
8	506	759
11	1288	-
12	1288	2576
15	506	-
16	253	759
23	1	-
24	-	1
Total = 2^{12}	4096	4096

TABELA 1.1. PESO DE HAMMING DAS PALAVRAS-CÓDIGO

O CGE (24, 12), com d maior que o CG (23, 12), não é perfeito mas quase-perfeito, pois tem como líderes de classe lateral todos os padrões de erros triplos ou menos e alguns padrões de erro com peso 4. De qualquer forma, tanto o CG (23, 12) como o CGE (24, 12) são ÓTIMOS para um canal BSC porque são, no mínimo, quase-perfeitos (qualquer código quase-perfeito é ÓTIMO para o BSC, [33, cap. 3]).

Como o BSC é a versão com quantização abrupta de um canal AWGN, que é o modelo aplicável a um canal via satélite linear como o nosso caso, o CG (23, 12) e o CGE (24, 12) são ÓTIMOS para um canal linear de satélite.

Apesar de Golay ter determinado seu código (23, 12) usando força bruta, podemos derivá-lo de forma mais elegante usando o conceito de códigos cíclicos e álgebra sobre Corpos de Galois GF (2^q) ([33], cap. 8; [11], cap. 2). O procedimento de construção fornece um código BCH corretor de erros duplos cujos polinômios geradores são ([33], [11]):

$$g_1(x) = X^{11} \oplus X^9 \oplus X^7 \oplus X^6 \oplus X^5 \oplus X \oplus 1 \tag{1.10}$$

ou

$$g_2(x) = X^{11} \oplus X^{10} \oplus X^6 \oplus X^5 \oplus X^4 \oplus X^2 \oplus 1 \tag{1.11}$$

No caso de nosso sistema AMDT, o polinômio escolhido para gerar as palavras-código para o CG (23, 12) é o dado por (1.10). O código CGE (24, 12) usado no AMDT é dado, então, por:

$$\begin{cases} \mathbf{c} = (\mathbf{b}, p_{12}) \\ b(x) = a(x)\, g_1(x) \\ g_1(x) = X^{11} \oplus X^9 \oplus X^7 \oplus X^6 \oplus X^5 \oplus X \oplus 1 \\ p_{12} = b_1 \oplus \cdots \oplus b_{23} = \sum_{i=1}^{23} b_i \end{cases} \tag{1.12}$$

É importante notarmos que $g_1(x)$ e $g_2(x)$ foram obtidos especificando-se um código BCH *corretor de erros duplos* apenas. Apesar disso, o código CG (23, 12) tem d_{MIN}=7 e t=3, permitindo correção de erros triplos. A conclusão, importante, é que se usarmos um decodificador BCH só conseguiríamos corrigir erros simples e duplos. Para se conseguir corrigir erros triplos também, o decodificador binário pode ser, por exemplo, uma tabela decodificadora (look-up table) ou um decodificador de Kasami [21].

O codificador então pode ser facilmente realizado para ter saída na forma sistemática usando um registrador de deslocamento com realimentação (FSR's) de

(k–1)=11 estágios com os circuitos descritos em ([33], cap. 7; [11]) obtendo uma palavra-código

$$\mathbf{c} = (c_1 \cdots c_{24}) = (a_1 \cdots a_{12}, p_1 \cdots p_{11}, p_{12}) \tag{1.13}$$

onde (a_1...a_{12})=**a** são os 12 dígitos de informação, vetor de entrada; (p_1...p_{11}) os dígitos de paridade do CG (23, 12) obtidos via multiplicação de **a** por $g_1(x)$ no FSR e p_{12} o dígito de paridade total.

Será útil, para a explicação dos processos do decodificador obtermos as matrizes $[\mathbf{G_1}]$ e $[\mathbf{H_1}]$ do CG (23, 12) e as matrizes [G] e [H] do CGE (24, 12).

A matriz $[\mathbf{G_1}]$ do CG (23, 12) – veja eq. 1.16 – pode ser obtida a partir das classes de resíduo representadas pelos polinômios X^Z. $g_1(x)$, $0 \le Z \le k$, e é uma matriz kxn = 12x23.

Para colocá-la em sua forma escalonada reduzida (ou canônica), que é a matriz geradora do código posto sob forma sistemática, sabemos, pelo teorema do resto de Euclides, que

$$X^i - r_i(x) = g_1(x)\, q_i\,(x) \qquad ,n-k \le i \le n-1 \tag{1.14}$$

é palavra-código onde só o termo em grau "i" é não-nulo para $i >> n-k-1$.
Então, podemos fazer (X^i–r) como j-ésima linha de $[\mathbf{G_1}]$, $i = n-j$.
A forma de $[\mathbf{G_1}]$ ficará sendo

$$[\mathbf{G_1}] = [\mathbf{I_k}\,|\,-\mathbf{R}] \tag{1.15}$$

$$[\mathbf{G_1}] = \begin{bmatrix}
1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1
\end{bmatrix} \tag{1.16}$$

onde $\mathbf{I_k}$ é a matriz identidade (kxk) e $-\mathbf{R}$ é uma matriz kx(n-k) onde a j-ésima linha é o vetor dos coeficientes em $-r_{n-j}(x)$. Procedendo assim, $[\mathbf{G_1}]$ posta na forma escalonada reduzida é dada em (1.17) e vale para o CG (23, 12) posto em forma sistemática:

$$[\mathbf{G_1}] = \begin{bmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1\\
0&1&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1\\
0&0&1&0&0&0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&1&0&1\\
0&0&0&1&0&0&0&0&0&0&0&0&1&1&0&0&0&1&1&1&0&1&1\\
0&0&0&0&1&0&0&0&0&0&0&0&1&1&0&0&1&1&0&1&1&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&1&1&0&0&1&1&0&1&1&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&1&1&0&0&1&1&0&1&1&1\\
0&0&0&0&0&0&0&1&0&0&0&0&1&0&1&1&0&1&1&1&1&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&1&1&0&1&1&1&1&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&1&1&0&1&1&1&1&1\\
0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&1
\end{bmatrix} \qquad (1.17)$$

O código de Golay estendido CGE (24, 12) pode ser modelado como na fig. 1.2 a seguir:

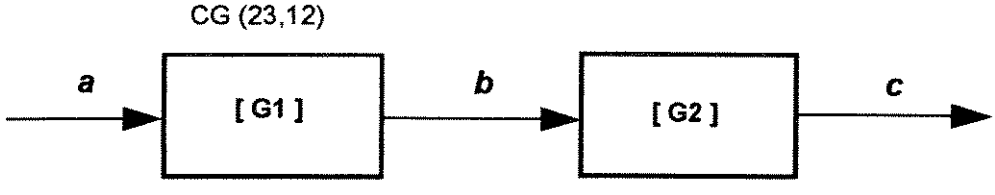


FIGURA 1.2. CODIFICADOR PARA O CGE (24,12) (MODELO TEÓRICO)

Temos então, pela estrutura da palavra código **c** do CGE (24, 12), que:

$$\mathbf{c} = \mathbf{b}[\mathbf{G_2}] = \left(\mathbf{b}, \sum_{i=1}^{23} b_i\right) \qquad (1.18)$$

Donde

$$[\mathbf{G_2}] = \left[\mathbf{I}_{23} \,\middle|\, \mathbf{1}^T \right] \tag{1.19}$$

é uma matriz 23x24.
Como **b** = **a** $[\mathbf{G_1}]$, teremos:

$$\mathbf{c} = \left(\mathbf{a}[\mathbf{G_1}] \right) \times [\mathbf{G_2}] \tag{1.20}$$

onde **a** é o vetor de informação [1x12], **c** a palavra-código [1x24], $[\mathbf{G_1}]$ a matriz geradora do CG (23, 12) dada por (1.17), [12x23], e $[\mathbf{G_2}]$ é uma matriz [23x24] constituída por uma sub-matriz identidade $\mathbf{I}_{23}$, [23x23], e um vetor coluna de 1s.

A matriz geradora [G] do CGE (24, 12) não nos interessa para efeito da discussão do decodificador. Entretanto, é muito importante obtermos a matriz de verificação de paridade [H] do CGE (24, 12).

A matriz $[\mathbf{H_1}]$ de verificação de paridade do CG (23, 12) posto na forma sistemática é, como sabemos ([33], [27], [11], [12]):

$$[\mathbf{G_1}] = \left[\mathbf{I}_k \,|\, \mathbf{P} \right] \;\Rightarrow\; [\mathbf{H_1}] = \left[-\mathbf{P}^T \,|\, \mathbf{I}_{n-k} \right] \tag{1.21 a}$$

Então, a matriz $[\mathbf{H_1}]$ do CG (23, 12) será $(n\text{-}k)$xn = 11x23 e valerá, por inspeção em (1.17):

$$[\mathbf{H_1}] = \begin{bmatrix}
1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0 \\
0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0 \\
1&1&0&0&0&1&1&1&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0 \\
0&1&1&0&0&0&1&1&1&0&1&1&0&0&0&1&0&0&0&0&0&0&0 \\
1&1&0&0&1&0&0&0&1&1&1&1&0&0&0&0&1&0&0&0&0&0&0 \\
1&0&0&1&1&1&0&1&0&1&0&1&0&0&0&0&0&1&0&0&0&0&0 \\
1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\
0&1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0 \\
0&0&1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0 \\
0&0&0&1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0 \\
1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&1
\end{bmatrix} \tag{1.21 b}$$

No caso do CGE (24, 12), a matriz [H] é obtida acrescentando-se um vetor-linha 1 e a matriz é 12x24, dada em (1.21) (c).

$$[\mathbf{H}] = \begin{bmatrix} 1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0 \\ 1&1&0&0&0&1&1&1&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&1&1&0&0&0&1&1&1&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0 \\ 1&1&0&0&1&0&0&0&1&1&1&1&0&0&0&0&1&0&0&0&0&0&0&0 \\ 1&0&0&1&1&1&0&1&0&1&0&1&0&0&0&0&0&1&0&0&0&0&0&0 \\ 1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&1&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&1&0&0 \\ 1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&1&0 \\ 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \end{bmatrix} \qquad (1.21\text{ c})$$

No decodificador, esta matriz [H] é usada na determinação dos vetores-síndrome que indicarão as posições dos erros nas palavras recebidas, já que

$$\mathbf{s} = \mathbf{r}[\mathbf{H}]^T \qquad (1.21\text{ d})$$

1.2.1.2. PROPRIEDADES ADICIONAIS DO CÓDIGO DE GOLAY

Algumas propriedades adicionais do código de Golay, interessantes tanto do ponto de vista teórico quanto do prático serão vistas nesta seção.

Observando os espectros de peso dos códigos de Golay CG (23, 12) e CGE (24, 12), dados na tabela 1.1, podemos concluir que eles apresentam simetria de pesos em torno do valor médio (n / 2). Esta simetria indica que, se a palavra **c** pertence ao código, então o complemento booleano $\bar{\mathbf{c}}$ dígito-a-dígito de **c** também pertence ao código (i.e., se $\mathbf{c} \in \mathbf{C} \Rightarrow \bar{\mathbf{c}} \in \mathbf{C}$).

Para provar esta propriedade, recordamos que, em álgebra binária, a inversão booleana é dada por:

$$\bar{c}_i = (1 + c_i) \qquad (1.22)$$

Como qualquer um dos polinômios geradores, $g_1(x)$ ou $g_2(x)$ – ver eqs. (1.10) e (1.11) – que dão origem a matrizes geradoras com um número ímpar de 1's em cada uma das 23 colunas, podemos, sem perda de generalidade, usar a matriz G canônica

obtida de $g_1(x)$, dada em (1.17). Esta matriz tem um **"1"** por coluna na sub-matriz identidade e sete **"1's"** em cada coluna da sub-matriz paridade **[P]**.

Assim, para uma palavra-código **c** obtida de uma palavra **a** de entrada, temos:

$$\begin{aligned} c_i &= a_i & 1 \le i \le 12 \\ c_i &= p_{(i-12)} & 13 \le i \le 23 \end{aligned} \qquad (1.23)$$

com as equações de paridade p_{i-12} obtidas da matriz $[\mathbf{G_1}]$ dada na eq. (1.17):

$$\begin{aligned} p_1 &= a_1 + a_2 + a_3 + a_4 + a_5 + a_8 + a_{11} \\ &\vdots \\ p_{11} &= a_1 + a_2 + a_3 + a_4 + a_7 + a_{10} + a_{12} \end{aligned}$$

Para uma palavra $\bar{\mathbf{a}}$ de entrada $\bar{\mathbf{a}} = (\mathbf{1}+\mathbf{a}) = [(1+a_1),...,(1+a_{12})]$, teremos:

$$c_i^* = 1 + a_i = \bar{a}_i = \bar{c}_i \qquad 1 \le i \le 12$$

E também, a partir da matriz $[\mathbf{G_1}]$ dada em (1.17):

$$\begin{aligned} p_1^* &= (1+a_1)+(1+a_2)+(1+a_3)+(1+a_4)+(1+a_5)+(1+a_8)+(1+a_{11}) = (1+1+1+1+1+1+1)+p_1 = \bar{p}_1 \\ &\vdots \\ p_{11}^* &= (1+a_1)+(1+a_2)+(1+a_3)+(1+a_4)+(1+a_7)+(1+a_{10})+(1+a_{12}) = (1+1+1+1+1+1+1)+p_{11} = \bar{p}_{11} \end{aligned}$$

Então, para uma entrada $\bar{\mathbf{a}}$, temos como palavra-código:

$$\begin{aligned} c_i^* &= \bar{c}_i & 1 \le i \le 12 \\ c_i^* &= \bar{p}_{(i-12)} & 13 \le i \le 23 \end{aligned} \qquad (1.24)$$

No caso do código estendido CGE (24, 12), temos:

$$c_{24} = p_{12} = a_1 + a_2 + \cdots + a_{12} + p_1 + p_2 + \cdots + p_{11} \qquad (1.25)$$

Portanto, para entrada $\bar{\mathbf{a}}$:

$$c_{24}^{*} = p_{12}^{*} = (1+a_1)+(1+a_2)+\cdots+(1+a_{12})+(1+p_1)+(1+p_2)+\cdots+(1+p_{11}) =$$
$$= \left(\sum_{i=1}^{23} 1\right) + p_{12} = \overline{p}_{12} = \overline{c}_{24} \qquad (1.26)$$

Conclui-se então que:

Para os códigos CG (23,12) e CGE (24, 12), se $\overline{\mathbf{c}} \in \mathbf{C}$ então $\mathbf{c} \in \mathbf{C}$. C.Q.D.

Esta propriedade se aplica a qualquer código de bloco binário que tenha o vetor **1** como palavra-código, que equivale a distribuição simétrica de pesos em torno de (n/2) e que equivale a ter número ímpar de 1's nas colunas da matriz G.

Da propriedade descrita acima decorre outra propriedade:

A decodificação dos códigos de Golay CG (23, 12) e CGE (24,12) é transparente à ambigüidade de fase associada à demodulação de um sinal BPSK. Esta propriedade leva, em termos práticos, a esquemas de decodificação mais simples, que não exigem a resolução (remoção) desta ambigüidade.

Isto constitui uma vantagem aparente porque, na maioria dos casos de transmissão digital, a polaridade dos dígitos (dada pela coerência de fases entre transmissão e recepção) importa muito.

Assim, nos casos de transmissão contínua, o único meio de remover a ambigüidade de fase de sinais BPSK seria a codificação diferencial do sinal digital em banda básica, resultando em uma modulação BPSK diferencial que, como se sabe, tem um desempenho abaixo do BPSK coerente dado que a taxa de erro de bit é o dobro da taxa de erro de símbolos. Em outras palavras, o ganho de codificação assintótico do código de Golay é reduzido, na prática, pelo uso de codificação diferencial.

Já nos casos de transmissão em surtos, como é o nosso caso, este problema pode ser evitado através do uso de um preâmbulo adequado, que permita a remoção da ambigüidade de fase. Aproveita-se assim a existência do preâmbulo não só para suas funções primordiais (sinalização de início de transmissão, recuperação de portadora e recuperação de relógio de símbolos) mas também para maximizar o desempenho quanto a ruído do sistema.

Para esquemas de modulação QPSK, que é o nosso caso, aplicam-se os mesmos princípios acima, com a ressalva de que existem 4 condições de ambigüidade de fase entre transmissão e recepção ao invés de apenas 2 do BPSK. A tabela 1.2. apresenta as condições de ambigüidades de fase para QPSK:

γ	A	B
0°	P	Q
90°	$\overline{Q}$	P
180°	$\overline{P}$	$\overline{Q}$
270°	Q	$\overline{P}$

TABELA 1.2. AMBIGÜIDADES DE FASE PARA QPSK

onde γ é a fase relativa entre a portadora recuperada e a portadora de transmissão, A e B são os canais em fase e quadratura da recepção e P e Q os canais em fase e quadratura da transmissão.

1.2.1.3. EXEMPLO DE APLICAÇÃO DO ALGORITMO DE CHASE

Chase [13] e Pessoa e Arantes [31] verificaram por simulação em computador qual seria o desempenho dos decodificadores baseados nos algoritmos 1, 2 e 3 de Chase em um canal AWGN e com o código de Golay estendido (24, 12); os resultados obtidos por Chase [13] estão na Fig. 1.3. a seguir, assumindo modulação 2-PSK.

Pela Fig. 1.3., pode-se concluir que:

a) para $P_b=10^{-4}$ o desempenho do decodificador usando ou o algoritmo 1 ou o 2 tem um ganho de 1,6 dB sobre o decodificador binário (decisão abrupta) e um ganho de 3,0 dB aproximadamente sobre uma transmissão não-codificada [31];

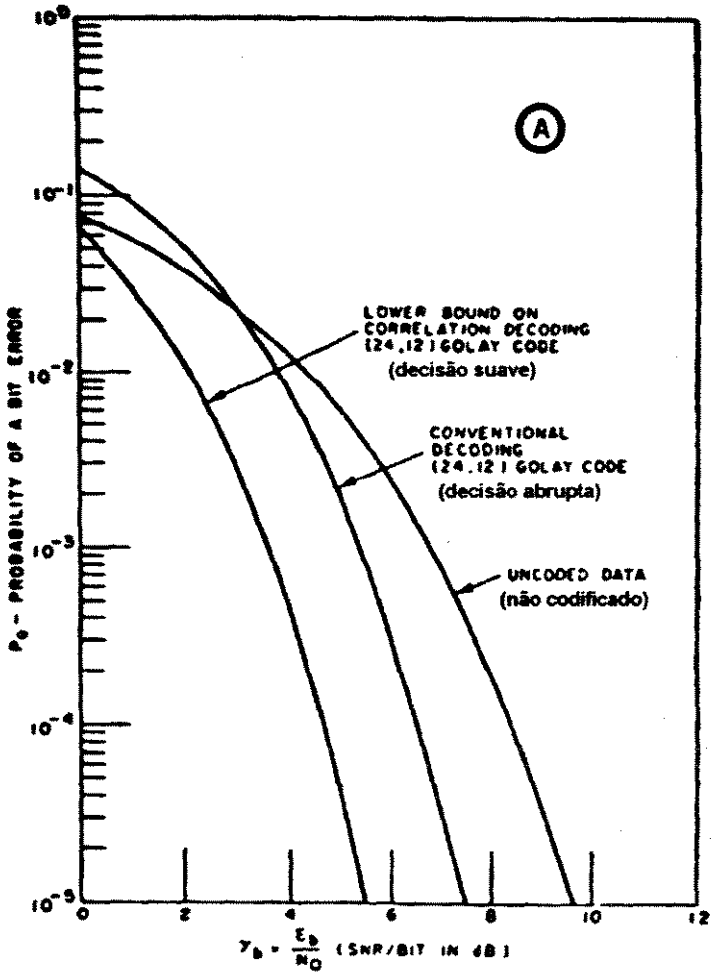
b) os ganhos assintóticos são de 3 dB para decodificação binária e de 6 dB para decodificação por correlação [13];

c) para $P_b=10^{-5}$ o ganho do decodificador binário é de 2 dB sobre a transmissão não-codificada e o do decodificador por correlação é de 4 dB [13];

d) o desempenho dos algoritmos 1 e 2 são exatamente iguais mas o algoritmo 1 usa 10626 padrões de teste conforme veremos no capítulo 4, enquanto o algoritmo 2 usa apenas 16 padrões de teste já que d=8 para o código estendido de Golay (24, 12).

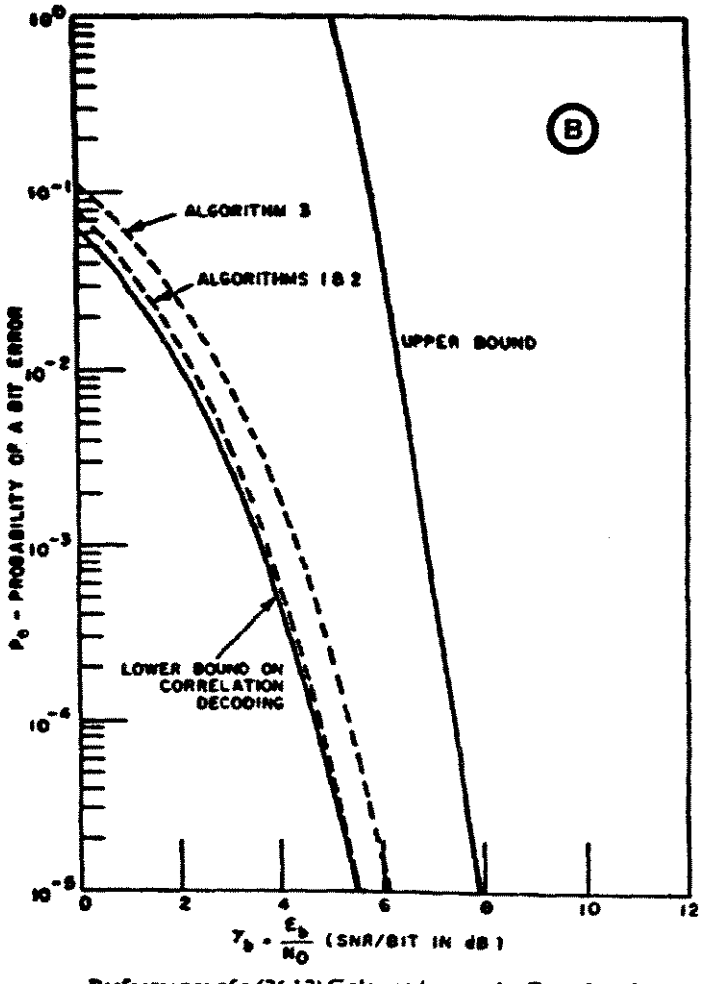
e) o desempenho do algoritmo 3 é 0,4 dB pior que o dos algoritmos 1 e 2 para $P_b=10^{-5}$, e usa 5 padrões de teste **Tj** com **Tj** tendo 0, 1, 3, 5, e 7 dígitos **"1"**, nas 7 posições menos confiáveis. Neste caso específico, onde n=24 ainda é relativamente pequeno, às vezes vale a pena usar 16 padrões de testes do algoritmo 2 ao invés de se usar os 5 do algoritmo 3 para ganhar 0,4 dB e ficar com um desempenho 0,1 dB pior que o melhor decodificador possível, que é o de correlação.

Este exemplo de aplicação do algoritmo 2 de Chase usando o CGE (24, 12) é de grande importância prática para nós dado que a arquitetura apresentada neste trabalho é especificamente voltada para esta aplicação, apesar de poder ser estendida para outros códigos de bloco desde que se disponha do decodificador por decisão abrupta e se adapte os diversos blocos constituintes do decodificador aos parâmetros do código usado (n, k e d), conforme mostraremos no capítulo 5.



Performance curves for the Gaussian channel (2ϕ – PSK).

A- DESEMPENHOS TEÓRICOS



Performance of a (24,12) Golay code over the Gaussian channel (2ϕ – PSK).

B- DESEMPENHO DOS ALGORITMOS

FIGURA 1.3. DESEMPENHO DOS ALGORITMOS [13]

Após esta revisão sucinta de conceitos de códigos corretores de erro, passaremos à revisão de um típico enlace digital via satélite para mostrar conceitos e problemas específicos da transmissão via satélite.

1.2.2. TRANSMISSÃO DIGITAL VIA SATÉLITE

A tecnologia de transmissão via satélite foi uma das tecnologias que mais rapidamente se desenvolveram e uma das que mais rapidamente penetraram no mercado, devido ao enorme aumento de capacidade para comunicações transoceânicas quando comparada com os cabos submarinos.

O satélite geoestacionário de comunicações foi concebido por Arthur Clarke em 1945 como sendo um repetidor de rádio a 36.000 km de altitude e em 1965 o primeiro satélite geoestacionário comercial para telecomunicações, o Intelsat I, já estava em serviço.

Tipicamente, um satélite de telecomunicações geoestacionário é um repetidor ativo de microondas situado em uma órbita sobre o equador, a 36.000 km de altitude. Nesta órbita, o período de translação do satélite em torno da Terra é de 24 horas, igualando o período de rotação da própria Terra. Então, para um observador fixo em um ponto da superfície terrestre, a posição aparente do satélite não muda, i.e., a órbita pode ser chamada geoestacionária. Um satélite de telecomunicações, além de painéis solares, baterias, subsistemas de comando e telemetria para controle de vôo e foguetes e retrofoguetes, possui o subsistema de comunicações, comumente chamado de "payload" (carga útil).

O diagrama em blocos deste "payload" está mostrado na Fig. 1.4.

Este "payload" consiste em um conjunto de repetidores ativos chamados de "transponders". Normalmente, estes transponders ocupam freqüências adjacentes e possuem largura de faixa de 40 MHz aproximadamente.

No caso do Brasil, os satélites BRASILSAT I e BRASILSAT II operam na faixa chamada banda C (3,9 a 6,2 GHz). Cada satélite tem uma banda passante autorizada de 500 MHz, recebe sinais das estações terrenas na faixa de 5925 MHz a 6425 MHz e os retransmite na faixa de 3700 MHz a 4200 MHz. Cada um dos satélites tem 24 transponders, com 36 MHz de largura de faixa cada um. Estes transponders são divididos em 2 grupos de 12 transponders, cada grupo operando com polarização ortogonal à do outro. Nos BRASILSATS, a polarização é linear e, portanto, 12 transponders recebem o feixe de polarização horizontal e os outros 12 recebem o feixe de polarização vertical. As razões para dividir a banda autorizada de 500 MHz em sub-faixas de 36 MHz e usar diversos "transponders" ao invés de um único "transponder" são:

a) economia: a potência e peso do subsistema de energia teriam que ser muito grandes porque as válvulas TWT do satélite teriam que ser de altíssima potência no caso de "transponder" único (para evitar intermodulações no ambiente multi-portadoras), encarecendo muito o satélite e o foguete de lançamento;

b) tecnologia: existem dificuldades técnicas muito grandes para implementar um "transponder" único;

c) confiabilidade: há ganho de confiabilidade com vários transponders.

Do ponto de vista de serviços e da tecnologia de transmissão, a comunicação via satélite apresenta a enorme vantagem de oferecer uma grande quantidade de canais para comunicações a longas distâncias ou transoceânicas a custos razoáveis e menores que em outras opções tecnológicas. Entretanto, a comunicação via satélite apresenta algumas dificuldades, que podem ou não se tornarem inconvenientes, dependendo do serviço.

A primeira dificuldade é o atraso de propagação. Este atraso é inevitável e ocorre devido à limitação física da velocidade da luz na atmosfera e no vácuo. Como em um enlace a informação percorre no mínimo 72.000 km (caso as estações estejam sobre o equador), o tempo de propagação é da ordem de 260 ms a 270 ms e varia conforme a localização das estações. Este atraso em nada afeta o desempenho de transmissão de TV ou voz, mas pode ser inconveniente em alguns serviços de transmissão de dados.

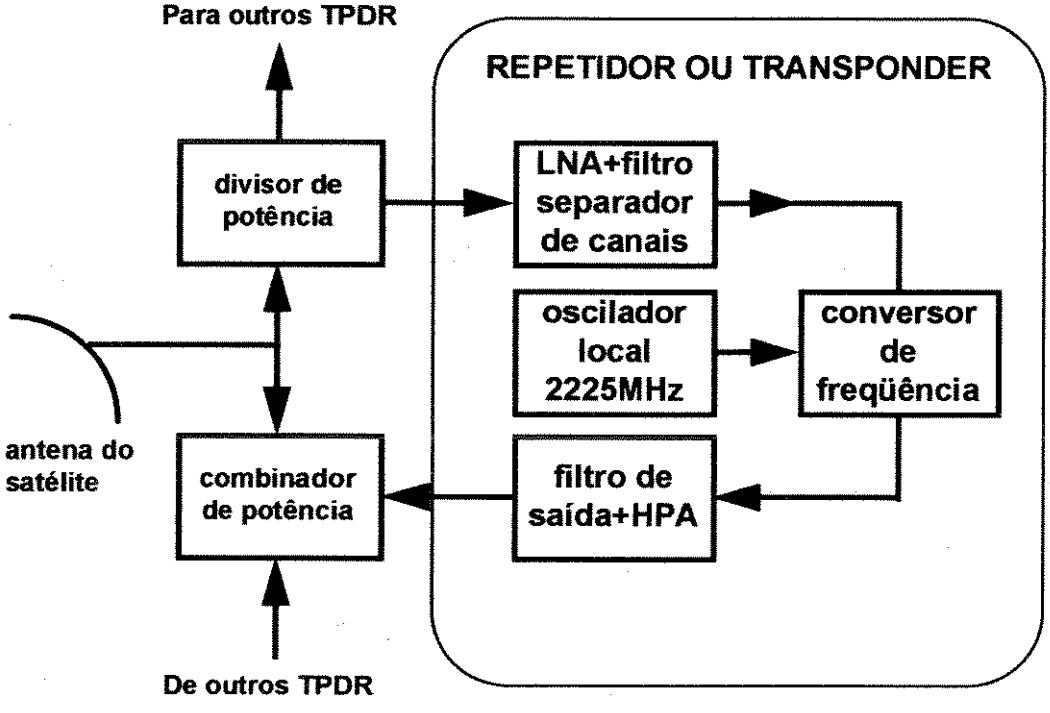
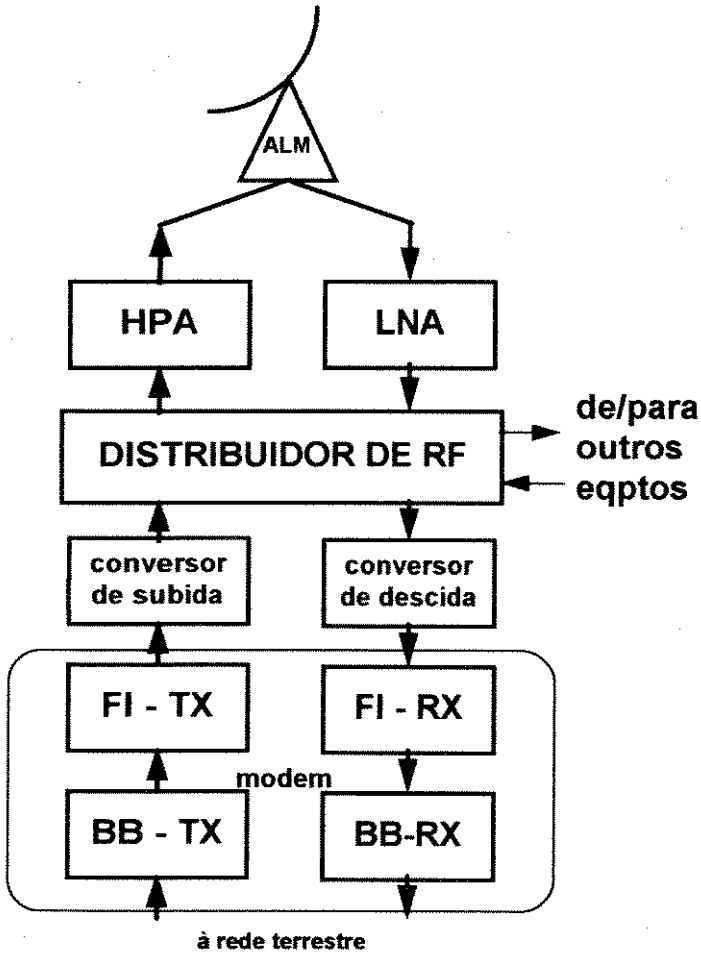


FIGURA 1.4. DIAGRAMA EM BLOCOS DO SUBSISTEMA DE COMUNICAÇÕES DE UM SATÉLITE TÍPICO

A outra dificuldade, contornável, diz respeito às perdas de transmissão e ruído, que obrigam os sistemas via satélite a operarem com tecnologias modernas e mais refinadas que as utilizadas em outros sistemas de telecomunicações.

Assim, tipicamente, as estações terrenas precisam utilizar amplificadores de potência de microondas (HPAs "high power amplifiers") na faixa de 1 W (para um canal de voz) a 600 W (transmissão de TV), amplificadores com baixo ruído (LNAs – "low-noise amplifiers") na faixa de 180 °K (recepção de TV doméstica) a 40 °K (estação de altíssima capacidade) e antenas parabólicas de alto desempenho com diâmetros entre 2 m (recepção de TV doméstica) e 30m (estação de altíssima capacidade), na banda C. A fig. 1.5. a seguir mostra uma estação terrena de transmissão via satélite típica em diagrama de blocos simplificado, omitindo equipamentos redundantes e de comutação (que aumentam a confiabilidade da estação).



LEGENDAS:

ALM – alimentador
HPA - amplificador de potência
LNA - amplificador com baixo ruído
EQPTO. DE RF - equipamento de RF (combinadores, divisores de potência, filtros em 4 e 6 GHz)
CONV. SUB. - conversor de subida (translação de FI para 6 GHz)
CONV. DESC. - conversor de descida (translação de 4 GHz para FI)
FI-TX - equipamentos de freqüência intermediária (tipicamente 70 MHz) de transmissão (moduladores, filtros, sintetizadores, amplificadores).
FI-RX - idem, para recepção (demoduladores, CAG, CAF, recuperação de portadora).
BB-TX - equipamento de banda básica de transmissão (amplificadores, filtros, pré-ênfase, embaralhadores, codificadores de voz, código corretor de erros, codificadores de vídeo, interfaces).
BB-RX - equipamento de banda básica de recepção (amplificadores, filtros, de-ênfase, desembaralhadores, decodificadores D/A de vídeo ou voz, decodificador do código corretor de erros, interfaces, recuperadores de relógio).

FIGURA 1.5. ESTAÇÃO TERRENA TÍPICA – DIAGRAMA EM BLOCOS SIMPLIFICADO

Quanto ao uso de transmissão digital nas comunicações via satélite, as vantagens e desvantagens em relação aos sistemas analógicos são basicamente as mesmas que se tem em outros tipos de sistemas de transmissão. A transmissão digital via satélite tem, entre outras, as seguintes vantagens sobre a transmissão analógica:

a) transparência dos sinais digitais: o modem, exceto em poucas funções, tem os mesmos circuitos, qualquer que seja o serviço;

b) hierarquia facilitada (para taxas de transmissão);

c) permite uso de AMDT (acesso múltiplo por divisão temporal), AMDF (idem, em freqüência) ou AMDC (idem, por código); nos sistemas analógicos só temos AMDF;

d) permite privacidade através de criptografia, o que é importante em algumas comunicações via satélite, já que os sinais transmitidos via satélite podem ser captados por qualquer estação;

e) economia nos equipamentos, que usam funções que podem ser comuns a todos os serviços, barateando o custo;

f) maior resistência a ruídos e interferências, podendo operar com relações sinal-ruído mais baixas do que os sistemas analógicos e também menor recuo dos amplificadores de potência, que passam a operar mais próximo da saturação, barateando ainda mais o custo das estações.

As maiores desvantagens são:

a) o atraso de propagação pode interferir em alguns serviços (dados iterativos, p. ex.) e em alguns métodos de acesso (AMDT);

b) normalmente, os sistemas digitais ocupam mais faixa que os sistemas analógicos equivalentes, o que também ocorre na transmissão via satélite.

Normalmente, a decisão de se usar sistemas de transmissão digital ou analógica leva em conta, em função do serviço, se o transponder operará limitado em faixa ou em potência. Os sistemas digitais, apesar de ampliarem a faixa requerida para um mesmo número de usuários, requerem relações sinal-ruído muito mais baixas para uma mesma qualidade de serviço.

Finalmente, citaremos agora as principais perdas que ocorrem nas transmissões via satélite e as principais interferências (ruído) na faixa de 4/6 GHz (banda C).

A atenuação de espaço livre é a maior perda, devido às enormes distâncias. Para uma antena de estação de 30 m de diâmetro e antena do satélite de 1,5 m de diâmetro, na banda C, com freqüências de subida e de descida iguais a 6,175 GHz e 3,95 GHz, as perdas de subida e de descida são de aproximadamente 95 dB e 104 dB, respectivamente ([29], cap. 4).

A atenuação por chuvas é outro fator de perda. Tipicamente, uma chuva forte (16 mm/hora), em 6 GHz, dá 1 dB de perda para ângulos de elevação de 90° e 3 dB para elevação de 15° ([29], cap. 4).

Outras atenuações são devidas a nuvens e nevoeiros, neve, granizo, etc. Estas atenuações também são pequenas (1 dB ou menos) e dependem da freqüência e ângulo de elevação [29].

Outra perda está associada à absorção atmosférica devida ao oxigênio molecular, vapor d'água e elétrons livres. Esta perda, na banda C, também é pequena e função da elevação da antena. Para os ângulos de elevação das antenas no Brasil, ela é desprezível na banda C [29].

Outro efeito atmosférico causador de perdas é a despolarização do feixe na ionosfera, que, além da perda em si, gera uma interferência entre as polarizações ortogonais. Este efeito também é desprezível na banda C.

Além destas perdas, a atmosfera causa um efeito chamado cintilação, que ocorre em determinada hora do dia (início da noite, durante 2 a 3 horas aproximadamente). A cintilação acarreta variações rápidas e de pequena amplitude no sinal em torno de 3 dB para cima e para baixo do nível nominal, como se fossem "micro-desvanecimentos". Este fenômeno vem ocorrendo com muita freqüência no Brasil e com amplitudes de variação maiores do que a literatura fornece para regiões como Estados Unidos e Europa.

Finalmente, existe uma perda controlável mas não desprezível, que é a perda por desapontamento (misalignment) de antenas. Esta perda é maior para antenas grandes, com feixe de abertura reduzida e é provocada pelo movimento do satélite e por ventos. Pode ser da ordem de até 0,5 dB.

Quanto aos tipos de interferência e ruído que afetam a comunicação via satélite, a maior delas é, sem dúvida alguma, o sol. Na faixa de 4/6 GHz, a temperatura de ruído varia entre 10.000 °K a 1.000.000 °K, dependendo da atividade solar. Para temperaturas de ruído desta ordem, o sinal fica ininteligível e o sistema sofre realmente uma interrupção. Graças à mecânica das órbitas, esta interrupção de serviço é previsível e dura em torno de 10 minutos durante 5 dias consecutivos semestralmente. Assim, pode-se contornar o problema com chaveamento de satélites ou uso de rotas alternativas [29].

A maior fonte de ruído (após o sol) no lance de subida é a Terra e o ruído na banda C é plano e vale em média 254 °K ([29], cap. 7).

Existem ainda os ruídos galáctico e cósmico, que são desprezíveis acima de 1 GHz [29], ruído de descargas eletrostáticas e raios, também desprezível [29] e o ruído do céu, que vale aproximadamente 30 °K [29].

Além disso, a mesma atmosfera, que absorve radiação, a reemite sob forma de ruído, que varia em função da freqüência e da elevação da antena. Assim, o ruído na banda de 4/6 GHz originado pelo vapor d'água, oxigênio e elétrons está na faixa de temperatura de ruído de 3 °K a 10 °K para elevações de 90° a 15° [29].

O mesmo ocorre com a chuva, granizo, neve, nevoeiros e nuvens. Os ruídos gerados por reemissões podem, segundo [29], variar, na banda C, de 10 °K (elevação de 90°) a 30 °K (elevação de 15°) para nevoeiro ou nuvens e de 10 °K (90° de elevação) a 40 °K (15° de elevação) para chuvas fortes.

Existe ainda o ruído da lua e o ruído produzido pelo ser humano. Este último é desprezível na banda C pois a emissão de máquinas elétricas, carros, etc, ocorre abaixo de 1 GHz. Apenas os enlaces de microondas terrestres podem atrapalhar nesta faixa, porém, com estudos de engenharia de sistemas adequados, o problema é contornável (colocando-se, por exemplo, as estações terrenas dentro de um vale cercado de montanhas, o risco de interferência é minimizado).

No caso do lance de descida, temos que considerar também o ruído próprio do equipamento de recepção, que tem uma temperatura de ruído devida à soma das temperaturas de ruído da antena (60° K típico), da guia de onda e do LNA (40° K a 180° K típicos).

Então, a partir dos perfis típicos de ruídos, usa-se tipicamente, para o cálculo de sistemas, uma temperatura de ruído total de 1000 °K para o lance de subida (inclui antena e LNA do satélite) e de 250 °K para o lance de descida (inclui antena e LNA da estação terrena), ao se operar na faixa de 4/6 GHz com estações terrenas com antenas de 12m de diâmetro, utilizando o feixe global do satélite [29].

O dimensionamento exato de um enlace é feito levando-se em conta todos estes fatores e, no caso da transmissão digital, a taxa de erro desejada, que fornecerá a relação sinal-ruído desejada. Convém notarmos que o ruído na transmissão via satélite em banda C é predominantemente branco, gaussiano e aditivo (porque os ruídos vêm de várias fontes independentes e com densidade uniforme na faixa de 4 a 6 GHz).

Na próxima seção, veremos um dos mais eficientes sistemas de múltiplo acesso ao transponder de um satélite, o acesso múltiplo por divisão temporal (AMDT).

1.3. O SISTEMA AMDT

Os sistemas de transmissão digital via satélite que utilizam técnicas AMDT (acesso múltiplo por divisão temporal, TDMA em inglês) vêm se tornando cada vez mais comuns e importantes no cenário das comunicações internacionais e domésticas via satélite graças às suas características altamente vantajosas para as comunicações via satélite [29].

Desde o início da década de 80 o Canadá possui 2 serviços AMDT para comunicações domésticas: o "High-Capacity TDMA" e o "Thin-Route TDMA"; nos Estados Unidos, um sistema AMDT de comunicações orientadas a realização de negócios entre empresas é operado pela companhia privada SBS e, recentemente, o INTELSAT introduziu seu sistema, operando comercialmente desde o início de 1985 (um sistema experimental para estudos de viabilidade técnica e comercial já operava desde 1974). O CPqD-TELEBRÁS desenvolveu um sistema AMDT – o SAMSAT – cujo teste de campo realizado pela EMBRATEL, mais os testes sistêmicos de laboratório foram concluídos com sucesso.

Como um satélite é um repetidor ativo de microondas estacionário a 36.000 km de altura e, portanto, muito caro, este recurso deve ser compartilhado pelo maior

número de usuários possível para que possa ser viável economicamente. Assim, a técnica AMDT provê um meio para tal compartilhamento: as diversas estações que compõem uma rede transmitem seqüencialmente, em instantes distintos, *surtos* de dados em uma cadência de repetição bem controlada para não aglomerar surtos no satélite que os retransmite às estações terrenas, formando um *quadro* repetitivo, como mostra a Fig. 1.6. Ocorre assim uma multiplexação no tempo entre as diversas estações, onde cada surto (burst) de dados é transmitido só por uma estação, mas todas podem receber todo o quadro (frame). Para controlar a cadência de transmissão e o sincronismo do sistema como um todo, deve existir uma estação de referência (ER), que comande remotamente todas as outras estações (ETS).

Originalmente, os sistemas AMDT foram idealizados para aumentar a capacidade de canalização dos TRANSPONDERS do satélite, porque, nos satélites antigos, a potência de transmissão das válvulas TWT de saída dos TRANSPONDERS (repetidores com largura de faixa BW=36MHz) era limitada a poucos watts e sobrava faixa no espectro de 500 MHz dos satélites. A potência de saída era limitada por causa do custo de lançamento por quilo de carga útil "payload" de eletrônica, por causa da baixa eficiência das baterias solares e da razão ganho por peso das antenas dos satélites. A tecnologia de comunicações via satélite na época era mera evolução da vigente nos "LINKS" de microondas terrestres, ou seja, a técnica de compartilhamento era o FDMA (frequency-division multiple-access ou multiplexação em freqüência). Com isso, criou-se uma limitação básica na máxima qualidade de sinal (máxima relação sinal-ruído, SNR ou S/N) que podia ser obtida, devido a dois fatores conflitantes: para amortizar o custo, um número maior de usuários deveria existir, mas, aumentar o número de usuários implicaria em aumentar a intermodulação entre portadoras (dada a pequena potência disponível, com rápida saturação da TWT), reduzindo a SNR final, podendo cair abaixo do mínimo nível de qualidade exigido pelo usuário. A solução para este problema era óbvia: diminuir o número de portadoras existentes em um transponder e aumentar o número de usuários significava passar de multiplexação em freqüência para multiplexação no tempo. Concebeu-se assim os sistemas AMDT de "primeira geração", cujas características eram:

a) apenas uma portadora ocupando todo o transponder de 36 MHz, com alta taxa de dados (60 Mbit/s usando BPSK ou 120 Mbit/s usando QPSK);

b) poucas estações terrenas compartilhando o serviço, que se comportavam como um nó de rede com milhares de portas acopladas a esse nó, e portanto aglutinadas geograficamente;

c) quadro fixo, com só um PTT (plano de tempo de transmissão).

Contudo, o cenário de comunicações foi evoluindo até chegar aos anos 80. Hoje, a eficiência dos painéis solares é muito superior que a de antes; com novos materiais, as antenas podem ser maiores (e então com maior ganho) e mais leves que as primeiras antenas dos satélites iniciais; o custo de lançamento por quilograma de "payload" é menor e a eficiência das TWTs é maior. Nas comunicações por via terrestre, houve um súbito aumento de demanda com a

explosão das comunicações de dados de computador, texto e negócios; viabilização de canais de banda larguíssima (fibra ótica); digitalização de voz e imagens e a exigência, por parte dos usuários, de maiores facilidades e flexibilidades de interconexão e maior qualidade de sinal. Tudo isto fez com que, hoje, o recurso limitado no satélite deixasse de ser a potência disponível e passasse a ser faixa de utilização (limitada a 500 MHz na faixa de 6/4 GHz).

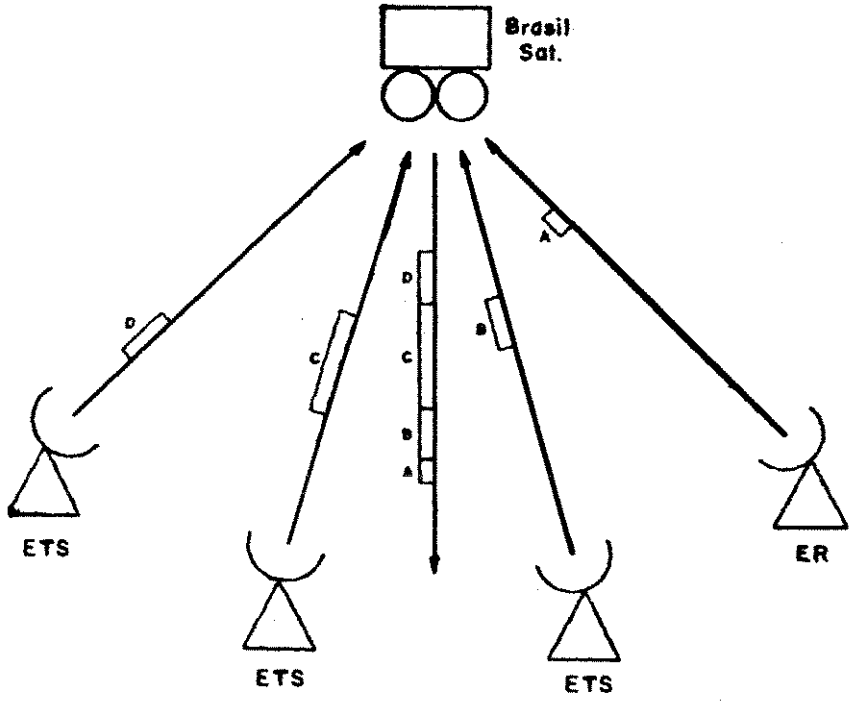


FIGURA 1.6. UM SISTEMA AMDT TÍPICO

Surgiram, então, a técnica de reuso de freqüência, com o aproveitamento da ortogonalidade de polarizações nas antenas (polarização horizontal e vertical ou circular à direita e à esquerda); aumento das freqüências de enlace (de 6/4 GHz para 11-9 GHz, 14-12 GHz, migrando para 20-18 GHz) e conseqüente aumento da largura de faixa total disponível (de 500 MHz para 580 MHz e hoje em 800 MHz) e uma mudança na tecnologia de estações terrenas, no sentido de torná-las menores, mais baratas, mais flexíveis e mais portáteis. Como o satélite é hoje o meio de transmissão à distância mais versátil porque pode: fazer radiodifusão ("BROADCASTING") e comunicações ponto-a-ponto, atingir regiões onde os enlaces terrestres seriam inviáveis economicamente (regiões inóspitas ou de baixa densidade populacional), e cobrir uma vasta região (que pode teoricamente chegar a 1/3 da superfície da Terra), atingindo com isto o maior número de usuários possíveis, o esforço no sentido de estações menores, baratas, flexíveis e transportáveis é de suma importância para competir com outros meios de

transmissão que hoje são mais baratos (como fibra óptica), mas não tão versáteis ou de ampla cobertura ou, ainda, transportáveis e móveis.

Inevitavelmente, a tecnologia AMDT caminhou neste sentido e o sistema desenvolvido hoje no CPqD é um AMDT com as seguintes características:

a) uso de múltiplas freqüências compartilhadas no tempo pelas estações, já que a potência disponível no satélite assim permite. São 16 freqüências diferentes.

b) com isto, pode haver um aumento da capacidade de interconexão entre estações já que podem continuamente, transmitir surtos em instantes distintos em freqüências diferentes. Cria-se, então, uma "central de comutação virtual no espaço" (virtual porque a inteligência não está "on-board" mas na Terra, na ER);

c) esta "central" aumenta a flexibilidade do sistema, já que os PTTs são programáveis e podem ser trocados periodicamente (desde que não seja quadro a quadro) dentro de um tempo que seja suficientemente longo;

d) estações pequenas, baratas e flexíveis, mas não tão facilmente transportáveis;

e) capacidade total de tráfego de 8,192 Mbit/s, distribuídos em quadros a 512 kbit/s com 60 ms de duração por 16 freqüências distintas, f_0 a f_{15}, como mostra a Fig. 1.7. O sistema pode ter até 126 ETS operando, uma estação de referência primária ER0 e uma outra estação de referência, secundária (reserva ou "stand-by"), ER1;

f) velocidade mínima de porta de usuário de 1,2 kbit/s e máxima de 48 kbit/s e velocidade mínima de porta de tronco de rede de 64 kbit/s e máxima de 384 kbit/s.

O "espaço não-utilizado" mostrado na Fig. 1.7 é inerente a todo sistema AMDT multifreqüencial e é um intervalo de tempo para que todas as ETS "ouçam" o canal comum de sinalização (CCS) da ER para ETS (na freqüência f_0). O CCS serve para controle do sistema, supervisão das ETS e monitoração de status (alarmes) entre ER e ETS. O CCS ETS para ER é acessado por contenção e detecção de colisão, i.e., qualquer ETS pode enviar sinais à ER neste CCS ETS para ER (em f1) e pode haver colisão de transmissões de 2 ETS quaisquer. Cada surto (não mostrado na Fig. 1.7) dentro de um quadro é precedido por um preâmbulo necessário em toda transmissão por surto. Este preâmbulo consta de um tempo de guarda, uma palavra única (PU) para marcar início de surto e uma seqüência de dados que facilite a recuperação de portadora e de relógio de bit. A modulação é QPSK coerente e é mantido um sincronismo virtual em cada porta para compensar erros de freqüência, posição (que faz variar tempo de propagação das mensagens) e efeito Doppler introduzidos pelo satélite, que não é perfeitamente geoestacionário.

Com isto, demos aqui as principais características do sistema AMDT desenvolvido no CPqD. Vamos agora justificar a utilização de um código corretor de erros no sistema e um decodificador tão poderoso como o de Chase.

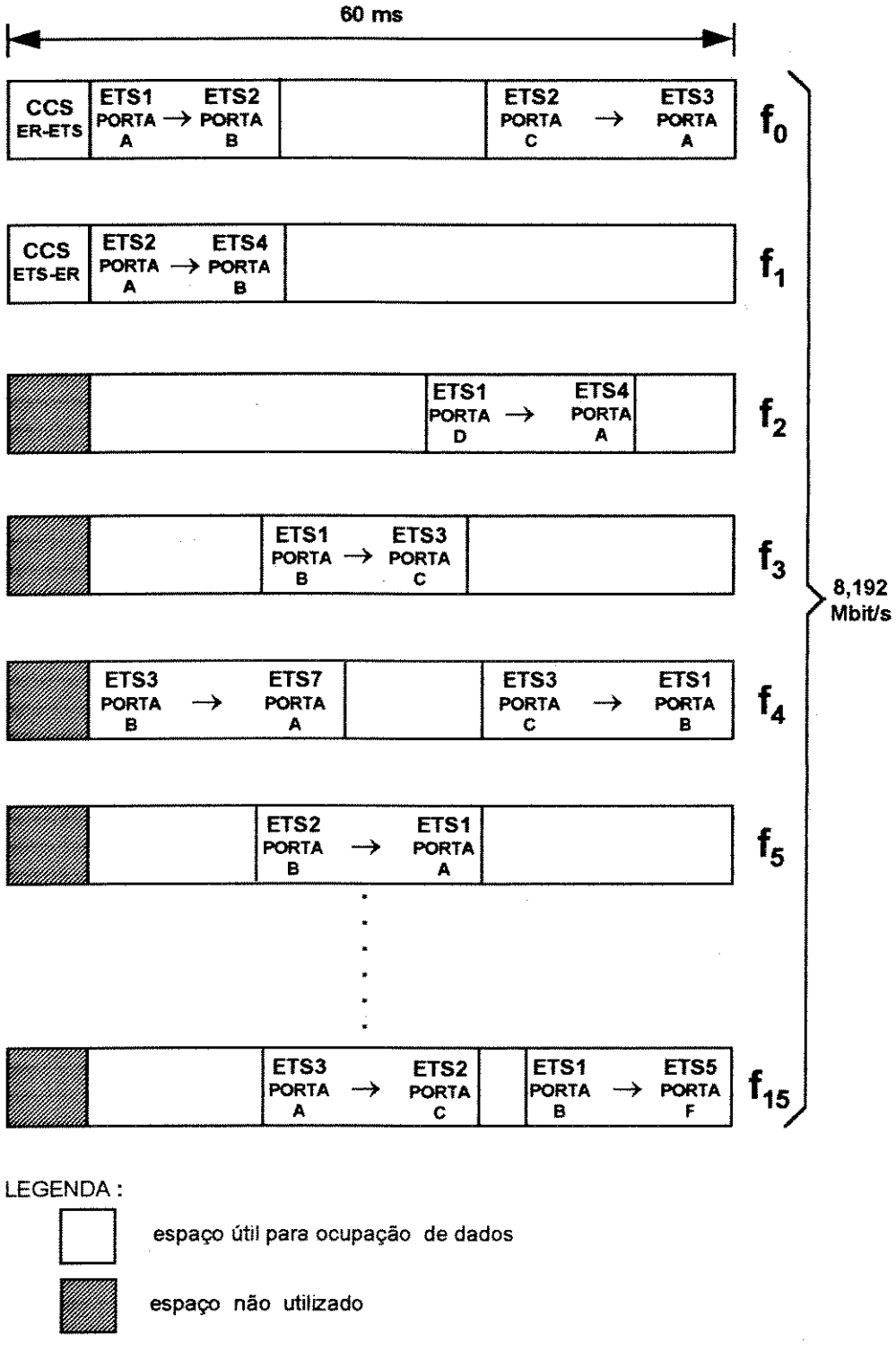


FIGURA 1.7. CAPACIDADE TOTAL DO SISTEMA AMDT (SAMSAT)

1.3.1. NECESSIDADE DE CÓDIGO CORRETOR DE ERRO NO AMDT

Para mostrar que é necessário o uso de códigos corretores de erro no sistema AMDT, vejamos primeiro alguns pontos conflitantes:

a) as estações precisam ser pequenas, relativamente baratas e flexíveis;

b) a potência disponível no satélite é limitada pelo recuo (back-off) necessário para não haver intermodulação entre as 16 portadoras (canal linear);

c) a atenuação de espaço livre no lance de subida e no lance de descida, somadas, ficam em torno de 200 dB;

d) a qualidade do sinal recebido deve ser boa o suficiente para garantir a confiabilidade exigida pelo usuário.

Esta qualidade exigida pelo usuário é tal que seja garantida a disponibilidade dos enlaces por mais de 99% do tempo. Para isso, a probabilidade de erro média de bit deve ser menor que 10^{-6} [42].

Por outro lado, para que as estações possam ser pequenas e baratas, o amplificador de potência de transmissão (HPA) tem que ser de potência de saída não muito alta (pois a relação preço/potência não é linear mas exponencial); o amplificador de baixo ruído (LNA) não pode ser de muito baixa temperatura de ruído (porque, novamente, para temperaturas de ruído abaixo de 100°K, o preço cresce assustadoramente) e a antena não pode ser de alto ganho porque, alto ganho em antenas, significa maiores diâmetros que, por sua vez, significam custos maiores (crescendo, também, exponencialmente em função do diâmetro).

Temos, então, que diminuir o desempenho do equipamento comum (antenas, HPA, LNA e os conversores de subida/descida), para que custe menos, e aumentar a complexidade do equipamento de processamento de sinais em FI e em banda básica, aumentando o seu desempenho, que depende mais da arquitetura escolhida e métodos de modulação do que do desempenho dos componentes em si. Por isto, os métodos usuais de modulação escolhidos para modems via satélite são o BPSK (2-PSK) ou o QPSK (4-PSK) coerentes, que são os de melhor desempenho conforme provado em [34], [44], [28], [12]. A tabela 1.3 dá os valores de P_b, probabilidade de erro de bit média, em função de E_b/N_o, obtida usando-se a eq. (1.27) abaixo, válida para um canal AWGN (caso de canais via satélite operando na região linear):

$$P_b = \frac{1}{2} erfc\left(\sqrt{\frac{E_b}{N_o}}\right) \quad \text{,para BPSK e QPSK} \tag{1.27}$$

Eb/No (dB)	Pb
6,9	10^{-3}
8,4	10^{-4}
9,7	10^{-5}
10,6	10^{-6}
11,2	10^{-7}
11,9	10^{-8}

TABELA 1.3. P_b X E_b/N_o PARA BPSK E QPSK NÃO-CODIFICADOS (CURVA TEÓRICA)

OBS.: Na tabela 1.3., $E_s/N_o = E_b/N_o$ para BPSK e $E_s/N_o = E_b/N_o + 3$ dB para QPSK pois o QPSK são 2 BPSKs ortogonais e cada símbolo representa 2 bits.

No entanto, devido às restrições sobre as antenas, LNAs e HPAs e também devido à máxima intermodulação aceitável (introduzida pelo satélite) e para conseguir maior capacidade de canal (canalização) do transponder, a especificação de sistema para o AMDT exige $P_b \leq 10^{-7}$ para $E_b/N_o = 8{,}5$ dB, conforme cálculos de sistema da EMBRATEL, feitos para viabilizar economicamente o sistema, para minimizar a intermodulação e para atender o critério de qualidade de sinal exigido pelos usuários potenciais [42].

Pela Tabela 1.3., vemos que, é impossível atingirmos esta especificação no caso sem codificação apropriada para a correção de erros no receptor.

Tal controle de erros pode ser realizado de duas maneiras: ou com a técnica de códigos corretores de erro (FEC) ou com a técnica de requisição automática de repetição de mensagens (ARQ).

A primeira técnica (FEC – forward error correction) usa a redundância introduzida na mensagem para alimentar um decodificador que faz parte do modelo "físico" de canal e transmissor/receptor. O código tem taxa $R = k/n$ e o sistema de codificação-decodificação opera com eficiência constante ao longo do tempo pois sempre descarta $(n-k)$ dígitos a cada n dígitos. A segunda técnica (ARQ - automatic repeat request) usa uma redundância de $(n-k)$ dígitos que não alimentarão um decodificador corretor mas um circuito de verificação de paridade que detecta os eventuais erros e, quando de suas ocorrências, envia um pedido de retransmissão do bloco errado, através de um canal reverso. Assim, esta técnica situa-se não no modelo "físico" mas na camada de protocolo, sendo necessária uma realimentação (canal reverso).

Em linhas gerais, temos as vantagens e desvantagens entre os dois métodos listadas a seguir:

a) **FEC:**

a.1.) **Vantagens:**

– não necessita canal reverso (realimentação);

– a eficiência de saída de informação é constante e igual à taxa $R = k/n$ do código;

– o atraso total pode ser constante para qualquer bloco;

a.2.) **Desvantagens:**

– a eficiência de saída que é, no caso, a taxa R de codificação cai com o aumento de ganho assintótico do código usado;

– a seleção do algoritmo de decodificação não é trivial, principalmente se se exige alta confiabilidade nos dados transmitidos;

– a confiabilidade dos dados decodificados é muito sensível a qualquer degradação nas condições de transmissão pelo canal (i.e., a curva P_b x E_b/N_o em um sistema codificado fica mais próxima da posição vertical (steeper) que a de um sistema não-codificado); no caso limite, Shannon provou que se E_b/N_o > -1,6 dB, $P_b \rightarrow 0$ e se E_b/N_o << -1,6 dB, $P_b \rightarrow 1/2$ ([44], cap. 2, sec. 2.5).

b) **ARQ:**

b.1.) **Vantagens:**

– baixíssima probabilidade de erros não-detectáveis ($P_\mu << 10^{-10}$);

– é efetivo em qualquer canal, com desempenho independente do tipo de ruído (AWGN ou canal com desvanecimento, etc.)

– simplicidade na implementação do codificador/decodificador, implicando em equipamentos mais baratos;

b.2.) **Desvantagens:**

– precisa de um canal reverso (realimentação por protocolo);

– atraso de decodificação variável para cada bloco;

– exige uma armazenagem, no lado de transmissão, dos blocos transmitidos cujo comprimento depende do dobro do tempo de propagação no canal mais o tempo de atraso do processo de codificação/decodificação; então, ou a fonte de dados precisa ser controlável ou **"buffers"** de memória precisam existir.

Pelo que foi exposto, e conhecendo as características de um enlace via satélite, que são: tempo de propagação de subida mais descida de aproximadamente 300 ms e ponto de operação de E_b/N_o situado entre 6 dB e 13 dB, a técnica de ARQ não oferece a solução ideal pois qualquer retransmissão de um bloco só aconteceria, no mínimo, 0,9 s após este bloco haver chegado, ou seja, seria necessário um buffer, no AMDT, de n=0,9 x 512 kbit/s ≡ 461 kbits no mínimo. Além disso, devido ao baixo E_b/N_o no ponto de operação, as retransmissões seriam muito freqüentes e inviabilizariam o sistema por causa da baixa eficiência de saída.

A solução, então, é introduzir um sistema que use FEC de forma a atender a especificação de $P_b \leq 10^{-7}$ para E_b/N_o = 8,5 dB e, caso o usuário deseje mais confiabilidade nas transmissões, usar então o ARQ (a cargo do usuário) para obter $P_b \leq 10^{-10}$ no final do processo híbrido FEC-ARQ.

Para o sistema AMDT, várias combinações de códigos e decodificadores foram consideradas.

Os códigos convolucionais pecam para transmissão por surtos devido à propagação da treliça após o final do surto, o que não acontece com códigos de bloco.

Entre todos os códigos de bloco considerados, o código de Golay estendido (24, 12) combinado com uma decodificação por decisão suave usando o algoritmo 2 de Chase foi o que mais se prestou às necessidades. Ele é prático, por ter taxa $R = 1/2$ (facilita os relógios a serem recuperados), por ser cíclico (facilita o codificador) e por ser linear e sistemático. Além disso, com a decisão suave em oito níveis, fornece um ganho suficiente para atender a especificação pois este ganho cobriria tanto a diferença teórica como a margem de implementação (degradação) do modem ([13], [31]).

A Fig. 1.8 mostra o desempenho do decodificador selecionado para o AMDT comparado a outros, obtidos por simulação [31]. Pessoa [32] simulou a sensibilidade do decodificador a variações de potência no CAG do AMDT e descobriu que a degradação para ± 6 dB de variação em relação ao nível ótimo é de apenas ± 0,3 dB.

1.3.2. RESTRIÇÕES DE IMPLEMENTAÇÃO DA ARQUITETURA DE UM DECODIFICADOR DE CHASE IMPOSTA PELO SISTEMA

1.3.2.1. SISTEMA DE MODULAÇÃO DO AMDT

O AMDT utiliza transmissão por surtos e modulação QPSK coerente. Do ponto de vista de desempenho da probabilidade de erro de bit, P_b, em função de E_b/N_o, sobre o canal AWGN, este sistema tem características idênticas ao do BPSK com o único cuidado de que a energia por símbolo é 3 dB superior à de um sistema BPSK para uma mesma energia por bit considerada em ambos os sistemas.

Do ponto de vista de implementação prática, o sistema exige uma série de processamentos de sinal após a demodulação e antes da decodificação propriamente dita. Estes processamentos são comentados a seguir.

1.3.2.2. SAÍDA DO DEMODULADOR QUANTIZADA

Assim como no modulador existem 2 canais P ("phase") e Q ("quadrature"), o demodulador tem como saídas 2 canais, chamados aqui de A e B. Cada canal é quantizado em 8 níveis com o auxílio de um conversor A/D de 3 bits, cujo mapeamento de níveis em palavras ou vetores de 3 bits é tal que o bit mais significativo dá a polaridade Y_i do dígito demodulado e os outros 2 dão uma medida do nível de confiabilidade (vetor α_i), a definir no Cap. 4, do dígito recebido.

A Fig. 1.9. fornece a tabela de conversão de níveis em palavras usada para o nosso caso (decodificador com algoritmo 2 de Chase).

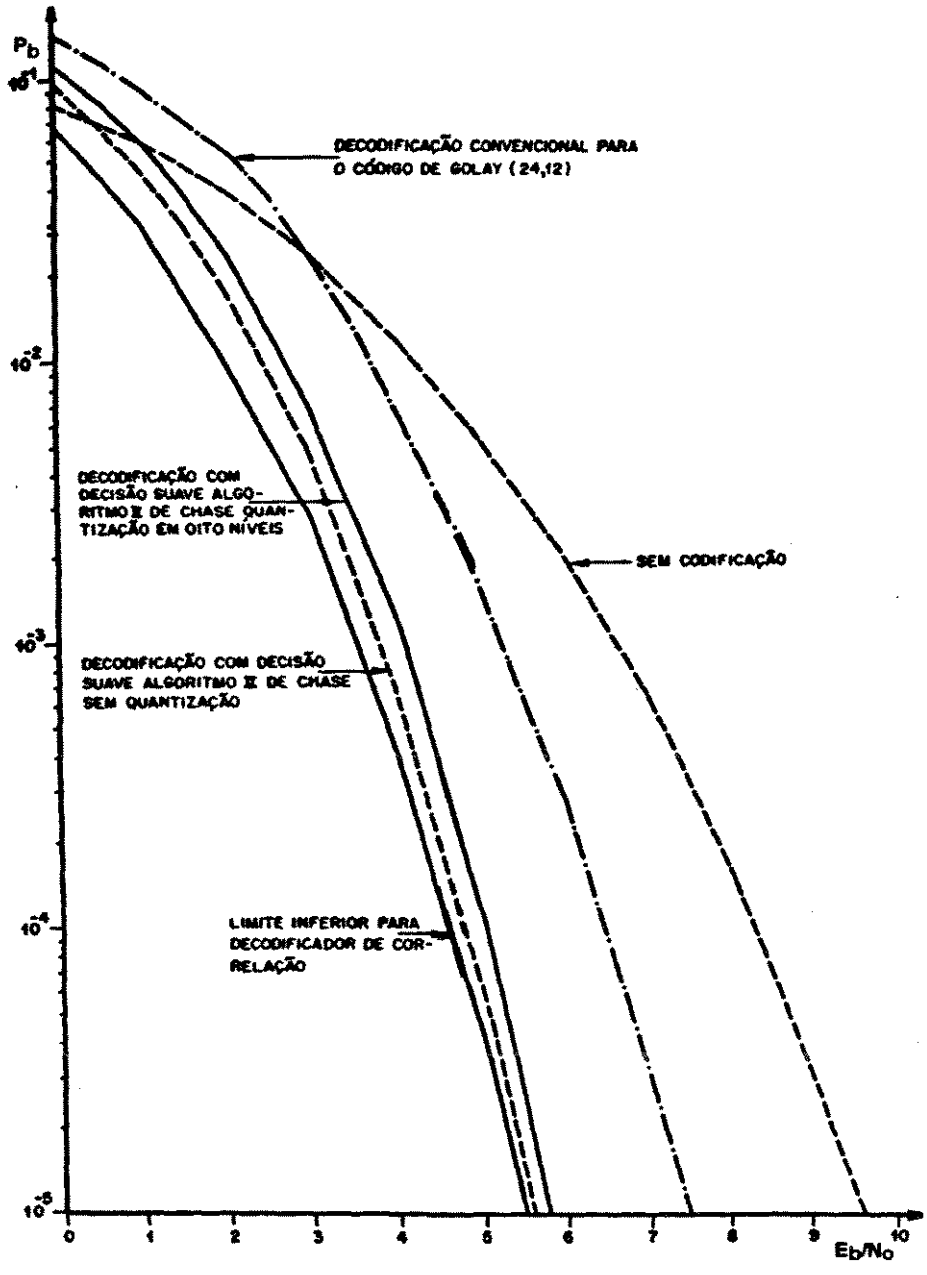
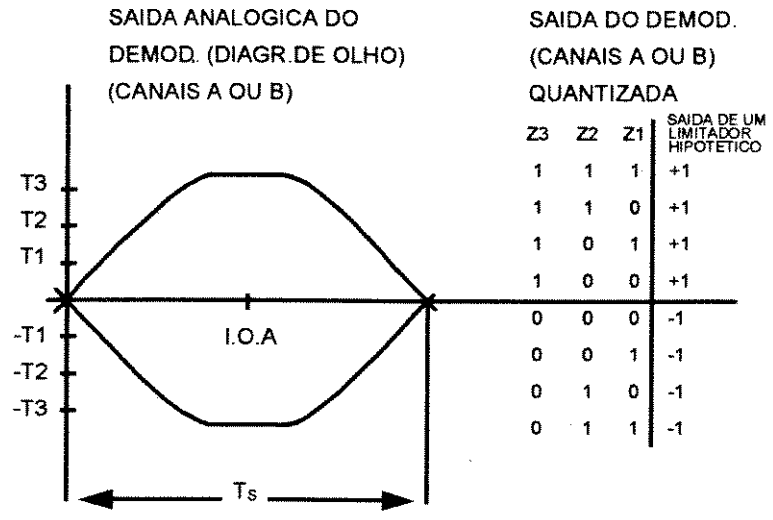


FIGURA 1.8. DESEMPENHO DO CÓDIGO DE GOLAY (24, 12) SOBRE O CANAL GAUSSIANO (MODULAÇÃO 2-PSK) [31]



T1, T2, T3 → Limiares de quantização
Ts → Período de um símbolo
I.O.A. → Instante ótimo de amostragem
Z3 → Valor de símbolo da seqüência recebida A ou B (Y_i)
Z2, Z1 → Níveis de confiabilidade do símbolo demodulado (α_i)

FIGURA 1.9. SAÍDA QUANTIZADA DO DEMODULADOR

O código usado para α_i é tal que, quanto maior a confiabilidade de Y_i (quanto mais afastado r_i estiver do limiar L), maior é o valor binário de α_i.

Para implementar esta saída, foi usada uma lógica combinacional de conversão após a saída de um conversor A/D de 3 bits do tipo **"flash"**, usando vários comparadores em paralelo. O resultado fornecido pela lógica combinacional é amostrado pelo relógio apenas quando já estiver estabilizado, para não resultar em erros de amostragem.

1.3.2.3. RESOLUÇÃO DAS AMBIGÜIDADES DE FASE DA DEMODULAÇÃO

O sinal QPSK coerente é demodulado após a recuperação de portadora na recepção, conseguida através da amarração de fase de um oscilador local a uma das quatro fases possíveis do sinal QPSK. Esta amarração é conseguida através de técnicas de malha fechada (PLLs, Costas Loop, Remodulação) ou através de sucessivas multiplicações em malha aberta. No caso de transmissões em surto, para minimizar o preâmbulo necessário, a multiplicação sucessiva em malha aberta é a melhor pois minimiza o tempo de aquisição. Para QPSK obtemos 4 x f_c através de

$o(t) = x^4(t)$, onde $o(t)$ é o sinal da portadora recuperada, $x(t)$ o sinal QPSK recebido e f_C a freqüência da portadora.

Entretanto, com qualquer uma das técnicas anteriores, devido ao fato da modulação ser coerente e não diferencial, surgem ambigüidades de fase nos canais A e B demodulados. Estas ambigüidades precisam ser removidas antes da decodificação. Tais ambigüidades são mostradas na Tabela 1.4. abaixo, onde:

γ (graus)	**A**	**B**
0°	P	Q
90°	$\overline{Q}$	P
180°	$\overline{P}$	$\overline{Q}$
270°	Q	$\overline{P}$

TABELA 1.4. RESOLUÇÃO DE AMBIGÜIDADES DE FASE EM FUNÇÃO DA FASE DE REFERÊNCIA DA PORTADORA RECUPERADA

$$x(t) = p(t)\cos\omega_c t + q(t)\text{sen}\,\omega_c t \qquad (1.28)$$

$$o(t) = \cos(\omega_c t + \gamma) \qquad (1.29)$$

$x(t)$ é o sinal QPSK e $o(t)$ a portadora recuperada através da potenciação de $x(t)$.

A resolução das ambigüidades de fase é conseguida pela monitoração das palavras únicas (conhecidas como PUs) presentes no início de cada surto, dentro do prêambulo. Se alguma delas estiver invertida em relação à que era esperada, o detector de palavra única a inverte e comuta os canais; se estiverem as duas palavras únicas invertidas, o detector de palavra única não comuta os canais mas apenas as inverte.

Com isto, resolve-se a ambigüidade introduzida na demodulação. Isto é feito nas 3 linhas de saída, correspondentes aos 3 bits para a quantização em oito níveis, para cada canal. Lembramos que as palavras únicas não são codificadas.

1.3.2.4. AMBIGÜIDADE DE DECOMUTAÇÃO DOS CANAIS A e B

Na transmissão, a seqüência de símbolos em forma serial, obtidos a partir do embaralhamento e codificação dos bytes de informação, é comutada em 2 linhas seriais P e Q que serão filtrados e entrarão no modulador.

O formato da comutação dentro da janela de tempo alocada à transmissão de um surto é o seguinte:

a) o 1° símbolo e todos os símbolos ímpares presentes à saída do codificador são enviados no canal P:

$P \leftarrow C\ (2k+1),\ k \in N$

b) o 2° símbolo e todos os símbolos pares são enviados no canal Q:

$Q \leftarrow C\ (2k),\ k \in N$

Assim, na entrada do decodificador, deverá haver um decomutador tal que, após a habilitação deste, serialize os símbolos recebidos nos canais A e B da recepção na seguinte forma:

$$Y = P_1\ Q_1\ P_2\ Q_2\ P_3\ Q_3\ \ldots.\ P_j\ Q_j \qquad (1.30)$$

O mesmo deve ser feito com os níveis de confiabilidade α, através de conversores paralelo-série.

1.3.2.5. HABILITAÇÃO DO DECODIFICADOR

Um surto consiste do preâmbulo e dos dados a serem transmitidos. Destas duas partes, o preâmbulo não é codificado e consiste do tempo de guarda, da seqüência para recuperação de portadora e relógio, e das palavras únicas. Apenas a parte de dados é codificada pelo codificador do código corretor de erros.

Assim, na recepção, o decodificador recebe um sinal de habilitação que indica início da parte de dados. Este sinal é enviado pelo detector de palavra única após a detecção da palavra única.

1.3.2.6. SINCRONISMO E RELÓGIOS DO DECODIFICADOR

Existem 2 sincronismos no sistema AMDT a serem observados, que são:

a) sincronismo de surto (ou local): é o sincronismo dado pelos relógios recuperados (de símbolo e de bit) e pelo pulso de detecção de qualquer palavra única constante do alfabeto de palavras únicas do sistema AMDT (são 4, definindo se o tipo de surto é de dados ou das referências). É o sincronismo de sistema perturbado pelas variações do satélite (efeito Doppler, variação da posição nominal do surto, etc);

b) sincronismo de sistema (ou de ER0); é o sincronismo fornecido pela estação de referência ER0 a todo o sistema AMDT. Este sincronismo é nominal e é mantido pela atualização da medida de diferença entre as posições nominais dos surtos e as posições reais destes dentro de um quadro. Esta diferença é compensada por uma FIFO que sincroniza os dados com o sincronismo de sistema. Assim sendo, como hipótese de trabalho, admitimos que todos os sinais após a regeneração (amostragem) e antes da decodificação, são corrigidos por FIFOs (first-in, first-out)

de forma a se enquadrarem dentro do surto nominal. Ou seja, o decodificador operará dentro do quadro nominal dado pela estação de referência, de forma que dados e relógios estarão totalmente sincronizados.

Um módulo de relógio mestre fornece qualquer relógio necessário a partir de 17,088 MHz. A taxa de símbolos é de 534 kbit/s nas saídas do demodulador. Após a conversão P/S a taxa é de 1068 kbit/s e na saída do decodificador a taxa será de 534 kbit/s.

Todos os relógios são síncronos na borda de subida, com a borda de descida indicando o meio do período dos símbolos.

1.3.2.7. ATRASO DE DECODIFICAÇÃO

O decodificador deve ter um atraso máximo de 2 a 3 blocos (i. e., 2 a 3 palavras-código): um bloco para calcular a síndrome e receber a seqüência rj, outro para decodificar rj, mais um para acomodar restrições práticas de "timing". Conforme veremos, no capítulo 7, o atraso da arquitetura proposta é de dois blocos mais dois bits, num total de 26 bits de informação.

1.4. DIVISÃO DA TESE

Dividimos o trabalho em sete capítulos, de forma a tentarmos facilitar ao máximo a compreensão dos diversos aspectos e condições de contorno que influenciaram nosso trabalho, bem como permitir a apresentação, clara e sistemática, dos resultados e conclusões advindos no decorrer deste trabalho.

Fornecemos abaixo o conteúdo de cada capítulo.

O presente capítulo 1 mostra a motivação da tese, fornecendo uma visão geral do problema específico que nos propusemos a resolver, uma revisão de conceitos básicos e a sistematização ou divisão da tese.

No capítulo 2 estudamos os decodificadores por decisão suave para códigos de bloco binários lineares sob um ponto de vista teórico e generalizado, comparando seu desempenho ótimo com os decodificadores por decisão abrupta. Analisamos também os efeitos da quantização sobre este desempenho. A referência básica deste capítulo foi [34], cap. 5.

O capítulo 3 é devotado ao estudo de diversos algoritmos disponíveis para decodificadores por decisão suave de códigos de bloco, exceto os algoritmos de Chase e de Hackett. Procuramos também realçar suas diferenças em desempenho e complexidade de implementação, para efeito de comparação. A referência básica deste capítulo foi [14], cap. 4.

No capítulo 4 apresentamos os algoritmos de Chase e a sua variante proposta por Hackett. Estes algoritmos não foram tratados no capítulo 3 junto com outros algoritmos não só porque um deles foi escolhido para a implementação da arquitetura mas principalmente porque são assintoticamente ótimos, pois são

algoritmos de decodificação de máxima verossimilhança. As principais referências deste capítulo são [13] e [16].

Nestes capítulos introdutórios, fazemos uma revisão geral do embasamento teórico necessário, e já disponível na literatura, para a consecução do trabalho aqui proposto. A partir do capítulo 5 até o final, incluindo apêndices, apresentamos as contribuições originais deste trabalho.

Dedicamo-nos, no capítulo 5, ao cerne de nosso trabalho: o desenvolvimento de uma arquitetura de hardware conveniente à implementação do algoritmo 2 de Chase para decodificação por decisão suave do CGE (24, 12), generalizando em seguida esta arquitetura a qualquer código corretor de erros de bloco, binário e linear. Detalhamos minuciosamente o método de análise e decomposição do algoritmo em processos paralelos, as técnicas de síntese dos blocos processadores e a análise de cada bloco e, finalmente, sua integração com o conjunto dos blocos que implementam os diversos processos, mostrando também uma idéia inicial da implementação dos blocos constituintes da arquitetura a nível de circuito, buscando a minimização do número de CIs a serem utilizados através de técnicas de maximização de funções de cada CI a ser utilizado no hardware.

No capítulo 6 apresentamos os resultados obtidos com o hardware. Iniciamos com a descrição do projeto lógico da arquitetura proposta no cap. 5, mostrando de forma detalhada a composição de cada bloco da arquitetura a nível de CIs.

Através de análise comparativa, procuramos demonstrar que o hardware mínimo para realizar o algoritmo 2 de Chase para o CGE (24, 12) é o que foi efetivamente implementado. Logo após isto fornecemos os resultados de desempenho de taxa de erro medidos em laboratório para uma das placas, comparando-os àqueles fornecidos por simulação ([13], [31]) e também dados sobre a reprodutibilidade das placas com a arquitetura proposta. Finalmente, junto às conclusões deste capítulo, fornecemos o resultado medido de taxa de erro do modem do sistema AMDT.

Por último, o capítulo 7 apresenta um resumo de cada capítulo anterior fornecendo as principais conclusões presentes em cada um. Sugerimos também algumas extensões da arquitetura, aplicada a outros sistemas, não incluídos neste trabalho, e que, por causa disso, necessitariam estudos mais aprofundados.

O apêndice A contém os métodos de teste da arquitetura. Para a caracterização completa do hardware em laboratório, são necessárias três categorias de testes. São elas: testes do hardware digital, para validar o projeto lógico; testes do decodificador inserido em canal AWGN, para caracterizar o desempenho do hardware; e testes no sistema AMDT, para validar o funcionamento do hardware integrado ao sistema.

O apêndice B fornece exemplos de seqüências que levam a resultados distintos ao decodificar por decisão suave e ao decodificar por decisão abrupta, ilustrando o princípio operativo do algoritmo de Chase.

CAPÍTULO 2

DECODIFICADORES POR DECISÃO SUAVE PARA CÓDIGOS DE BLOCO

2.1. INTRODUÇÃO

A finalidade deste capítulo é determinar o desempenho ótimo de um sistema digital que use códigos de bloco binários e lineares na transmissão e que utilize na recepção um decodificador com decisão suave de sua entrada sem quantização ou quantizada em Q níveis diferentes de amplitude, como mostra a Figura 2.1 abaixo:

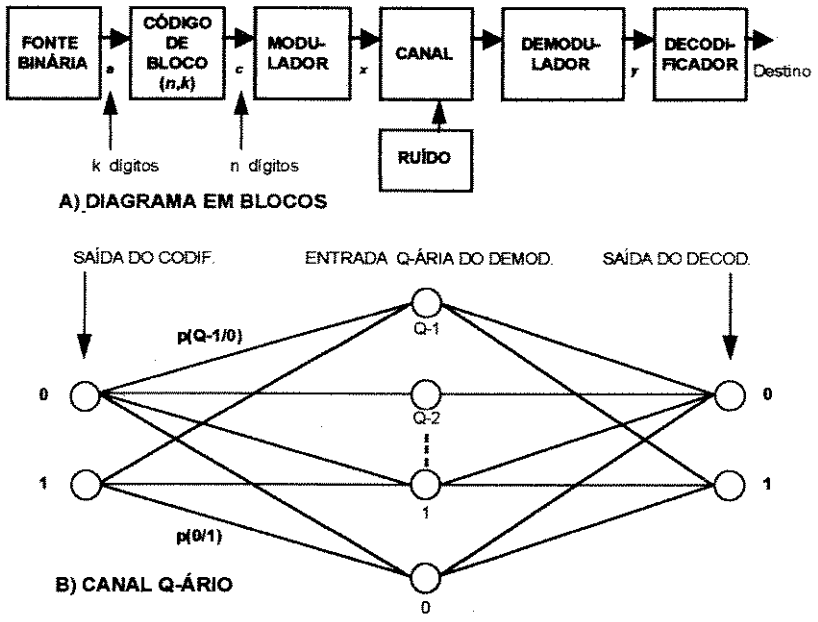


FIGURA 2.1. UM TIPO DE SISTEMA DE COMUNICAÇÃO DIGITAL

Para aquilo a que nos propusemos, consideramos apenas o canal AWGN (de ruído aditivo, branco e gaussiano), modulação PSK coerente antipodal (2 fases) e receptor casado (ou ótimo), que são os métodos mais eficientes em energia por bit necessária para um determinado desempenho em probabilidade de erro.

2.2. DESEMPENHO DO DECODIFICADOR ÓTIMO

Seja E_T a energia de transmissão de sinal para cada palavra-código **c** na saída do decodificador de um código de bloco (n,k) e seja E_S a energia necessária à transmissão de um único dígito da palavra-código. Como existem n dígitos em cada palavra-código, $E_T = n \cdot E_s$. Sabendo ainda que cada palavra-código carrega em seu bojo k bits de informação (i.e., k bits da fonte binária de informação), temos que a energia por bit de informação E_b valerá:

$$E_b = \frac{E_T}{k} = \frac{n}{k} \cdot E_S = \frac{E_S}{R} \tag{2.1}$$

onde:

$R = k/n$	taxa do código de bloco (n,k), binário
E_S	energia de transmissão por dígito de saída, que é 1 símbolo se o modulador for antipodal, i.e., só tiver 2 símbolos de saída
E_b	energia por bit de informação.

Vamos supor ainda que as palavras-código são equiprováveis a priori com probabilidade p_c :

$$p_c = \frac{1}{M} = \frac{1}{2^k} \tag{2.2}$$

Esta hipótese é razoável visto que, mesmo que a fonte primária não seja balanceada ou equiprovável, a condição é facilmente obtida através do uso de um embaralhador de dados entre a fonte binária e o codificador. Este embaralhador nada mais é que um FSR (feedback shift-register) configurado como um gerador de seqüências de máximo comprimento. Esta seqüência passa por todos os (2^J-1) padrões de comprimento J em um FSR de J estágios e pode ser somada (módulo 2) com a seqüência original da fonte sem que perca as propriedades de pseudo-aleatoriedade e equiprobabilidade ([33], cap. 7).

Suponhamos agora que os dígitos à saída do codificador sejam transmitidos um a um, i.e., cada um é um símbolo de saída do modulador, apenas para garantir a equação 2.1. Ou seja, o esquema de modulação é antipodal e aqui vamos supor

modulação 2-PSK (também conhecido por PSK antipodal, ou PSK binário) com as fases dadas abaixo na tabela 2.1:

ENTRADA DO MOD. X_i(i-ésimo símbolo)	FASE DA PORTADORA À SAÍDA DO MOD.
1	0° ou 0 rd
0	180° ou π rd

TABELA 2.1. ESTADOS DE UM MODULADOR 2-PSK

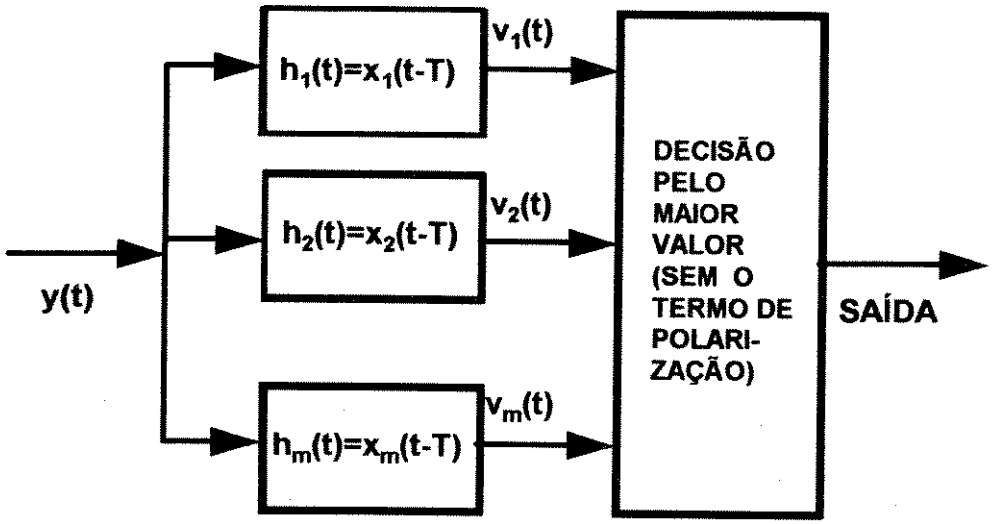
Voltamos a repetir: esta hipótese é feita apenas para não introduzir um fator L na equação 2.1 quando a transmissão é Q-ária (i.e., 1 símbolo de saída para cada L dígitos de entrada, $Q=2^L$). Para 2-PSK, cada palavra-código resulta em um entre $M=2^k$ sinais de transmissão possíveis sendo enviado ao receptor. Conforme se sabe ([34], cap. 4; [12]; [44], cap. 2), o receptor ótimo, que minimiza a probabilidade média de erro de uma entre M mensagens ou sinais para um canal AWGN, é constituído por um banco de M filtros casados cada um com um dos M possíveis sinais, em paralelo, como mostra a Figura 2.2. As saídas dos M filtros casados são comparadas ao final da mensagem e a saída de maior amplitude é tomada como sendo verdadeira. No nosso caso, cada mensagem tem n dígitos e é comparada ao final de cada bloco transmitido. Outra possibilidade é usarmos M correladores.

Para o caso analisado aqui, o receptor a ser implementado pode ser simplificado usando-se apenas um filtro casado (ou correlador) com o sinal PSK binário antipodal usado para transmitir cada símbolo (dígito) da palavra-código, seguido de um decodificador que forme *M VARIÁVEIS DE DECISÃO* correspondentes às M palavras-código $\mathbf{c_i}$ possíveis.

Precisamos agora achar resposta a 2 questões:

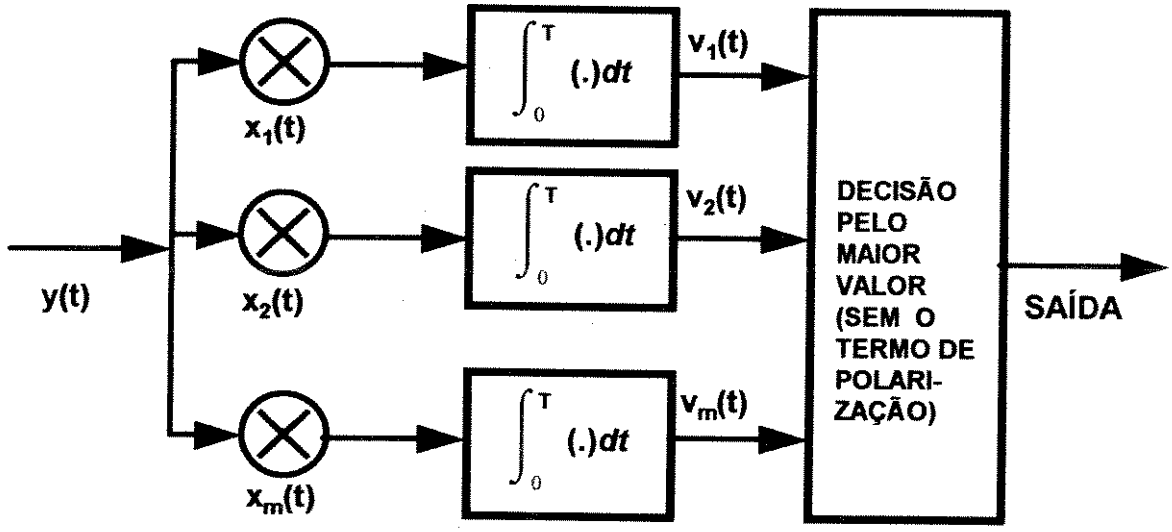
a) Qual deve ser a regra de formação das M VARIÁVEIS DE DECISÃO que garantem a optimalidade do receptor?

b) Quais são os limitantes (bounds) do desempenho de tal receptor?



— $x_i(t)$ é a forma de onda de transmissão associada à palavra-código $c_i(t)$
— T é o período de x_i
— $v_i(t)$ é variável de decisão

a) com filtros casados

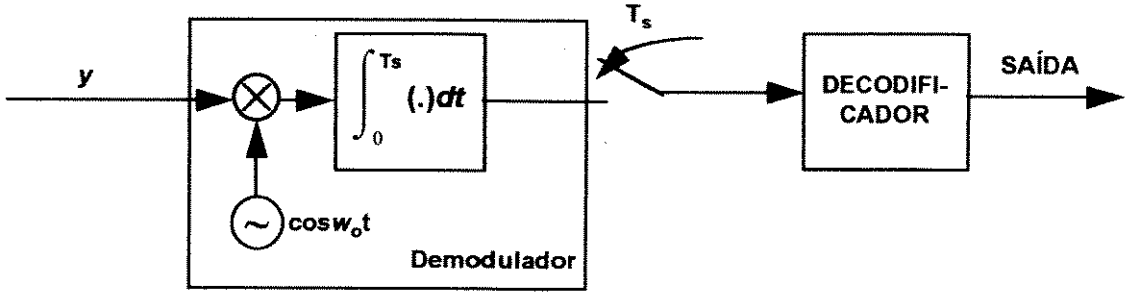


b) com correladores

FIGURA 2.2. RECEPTORES ÓTIMOS EM CANAIS AWGN

2.2.1. REGRA DE FORMAÇÃO DAS VARIÁVEIS DE DECISÃO

O receptor ótimo é representado abaixo, na fig. 2.3 :



LEGENDAS:

T_S = período de símbolo (dígito), $n(t)$ = ruído do canal

ω_o = freqüência da portadora; α = atenuação no canal

$x(t)$ – pulso ou forma de onda associado à saída do modulador

$y(t)$ – sinal à entrada do demodulador

$$y(t) = \alpha \sum_{i=-\infty}^{+\infty} x(t - iT_S) \cdot \cos\omega_o t + n(t)$$

FIGURA 2.3. RECEPTOR ÓTIMO SIMPLIFICADO

Sejam então y_j, $j \in \{ 1, 2,..., n \}$, os n valores amostrados à saída do integrador, a cada T_S segundos, para qualquer uma palavra-código $\mathbf{c_i}$. Como a modulação é 2-PSK coerente e antipodal, os y_j podem valer (ver tabela 2.1):

$$y_j = \begin{cases} 2\alpha E_s + \upsilon_j \text{ se } c_{ij} = x_{ij} = 1 \\ -2\alpha E_s + \upsilon_j \text{ se } c_{ij} = x_{ij} = 0 \end{cases} \qquad (2.3)$$

onde c_{ij} é o dígito da posição j da i-ésima palavra-código $\mathbf{c_i}$, x_{ij} é o símbolo correspondente que modula o pulso $x(t)$, α é a atenuação do canal AWGN sobre o sinal $(0 \le \alpha \le 1)$ e o conjunto de variáveis $\{\upsilon_j, j = 1,...,n\}$ representa o ruído branco aditivo gaussiano nos instantes de amostragem, sendo cada υ_j uma variável aleatória (V.A.) gaussiana de média zero e variância igual a $2E_S N_o$ (N_o é a densidade espectral bilateral de potência do ruído).

O receptor, sabendo de antemão quais são as M palavras-código que podem ser transmitidas (porque o código **C** é fixo por hipótese), ao receber o conjunto $\{y_j, j = 1,\ldots,n\}$ pode formar as seguintes variáveis de decisão U_i, representadas pelo sistema de equações (2.4) definido abaixo:

$$U_i = \sum_{j=1}^{n} (2c_{ij} - 1) \cdot y_j \quad \text{para todo } i \in 1,2,\ldots,M \text{ e } M = 2^k \tag{2.4}$$

Mas como $c_{ij} \in \{0,1\}$, teremos:

$$(2c_{ij} - 1) = \begin{cases} 1 \text{ se } c_{ij} = 1 \\ -1 \text{ se } c_{ij} = 0 \end{cases} \tag{2.5}$$

Então, levando (2.5) e (2.3) no sistema (2.4), podemos estabelecer o processo de decisão do receptor sobre qual palavra-código foi originalmente transmitida dado que ele conhece a forma de onda recebida. Desenvolvendo o sistema (2.4), teremos:

$$U_1 = \sum_{j=1}^{n} (2c_{1j} - 1) \cdot (2x_{1j} - 1) \cdot (2\alpha E_S) + (2c_{1j} - 1) \cdot \upsilon_j$$

$$\vdots$$

$$U_m = \sum_{j=1}^{n} (2c_{mj} - 1) \cdot (2x_{mj} - 1) \cdot (2\alpha E_S) + (2c_{mj} - 1) \cdot \upsilon_j$$

Onde o sinal de (2.3) no termo $2\alpha E_S$ foi substituído pelo termo $(2x_{lj} - 1)$, supondo que foi transmitida a palavra $\mathbf{c_l}$. Se calcularmos o valor máximo de U_i no sistema acima, observamos que U_i será máximo se e somente se os coeficientes $(2c_{ij} - 1)$ forem de mesmo sinal que os coeficientes $(2x_{lj} - 1)$, ou seja, se e somente se os termos c_{ij} se alinharem às variáveis x_{lj} para todo j, i.e., se e somente se $\mathbf{c_i}$ for igual a $\mathbf{x_i}$. Além disso, como U_i é uma variável aleatória, sua média valerá $2\alpha E_S N_o$ já que o conjunto $\{\upsilon_j\}$ tem média nula. Todos os outros valores, de U_1 a U_m com exceção de U_i, terão médias menores graças ao não-alinhamento de sinais dos termos que compõem o sistema de equações.

Ora, como U_i só é máximo se $c_{ij} = x_{ij}$, e como as palavras-código são diferentes entre si, o receptor que escolhe a palavra-código correspondente ao valor máximo de $U_1,...,U_m$ é ótimo no sentido de que ele sempre minimiza as chances de se cometer um erro no processo de decisão. Assim, o receptor proposto deve, a partir do cálculo das variáveis de decisão definidas em (2.4), escolher a função que assume o máximo valor dentro do conjunto calculado.

Isto é equivalente e proporcional ao cálculo de

$$\underset{M}{\text{MAX}}\left\{\text{Probabilidade de }\left[\text{sinal } x_i \text{ foi transmitido}/y(t)\right]\right\}$$

que é o processo de um receptor ótimo, também conhecido como receptor MAP (máximo a posteriori).

2.2.2. LIMITANTES DE DESEMPENHO DO RECEPTOR DE DECISÃO SUAVE (RECEPTOR ÓTIMO)

O desempenho do receptor pode ser avaliado através da probabilidade de erro em função da relação sinal-ruído na entrada do receptor.

Como o código de bloco binário C assumido aqui é linear e, além disso, todas as palavras-código são equiprováveis na transmissão, a probabilidade de erro na decodificação de um bloco formado por uma palavra-código $\mathbf{c_i}$ é igual à probabilidade de erro de decodificação de qualquer outra $\mathbf{c_j}$, $\forall\, i,j \;/\; \mathbf{c_i}, \mathbf{c_j} \in \mathbf{C}$. Então, a probabilidade de erro de bloco média (média sobre todas $\mathbf{c_i} \in$ C) é igual à probabilidade de erro de bloco de uma palavra-código qualquer. Para a nossa análise, consideramos aqui a palavra-código $\mathbf{c_1} = \mathbf{0}$ (vetor nulo), que pertence a C pois este é linear.

A probabilidade de se decodificar corretamente $\mathbf{c_1} = \mathbf{0}$ é dada pela probabilidade da variável de decisão U_1 ser maior que todas as outras $2^k - 1$ variáveis $\left(U_2, U_3, ..., U_{2^k-1}\right)$. Assim, calculando os valores médios e as variâncias das variáveis, obtém-se:

$$\text{a) } E[U_1] = 2\alpha E_S \cdot n; \qquad \sigma_{u_1}^2 = 2E_S N_o \tag{2.6}$$

$$\text{b) } E[Uj] = 2\alpha E_S \cdot n\left[1 - 2\left(W_j/n\right)\right];\ \sigma_{u_j}^2 = 2E_S N_o;\ j = 2,...,2^k - 1 \tag{2.7}$$

onde W_j é o peso de Hamming da palavra j.

Ainda existe a correlação cruzada entre $\mathbf{c_1}$ e $\mathbf{cj}$, que vale:

$$\rho_j = 1 - 2\left(W_j/n\right) \qquad j = 2,3,...,2^k - 1 \tag{2.8}$$

que complica a dedução do valor exato da probabilidade de decodificação correta de $\mathbf{c_1}$, já que U_1 não será uma V.A. independente de Uj.

É mais fácil e mais geral, valendo para qualquer código, calcular uma expressão (curva) limitante superior para o desempenho do decodificador (supondo U_1 e Uj independentes) ao invés de se tentar calcular exatamente a probabilidade de erro porque, neste caso, depender-se-ia da distribuição de pesos de código. Para isso, temos que calcular a probabilidade de que Uj exceda o valor de U_1 :

$$P_j = \Pr[U_j > U_1] = \Pr[U_j - U_1 > 0] = \Pr[U_1 - U_j < 0] \tag{2.9}$$

Definindo uma nova V.A. V :

$$V = U_1 - U_j = 2\alpha E_S n(1-\sigma_j) + N_1 - N_2 \tag{2.10}$$

Como N_1 e N_2 são gaussianos, $(N_1 - N_2)$ é gaussiano e então V é gaussiana de média dada abaixo (observar que $E_s\ n = E_T$ e que N_1 é a soma dos processos gaussianos V_j referentes a V_1 e N_2 é a soma dos V_j referentes a U_j):

$$\mathrm{E}[V] = 2\alpha E_T(1-\rho_j) \tag{2.11}$$

e variância

$$\sigma_V^2 = \mathrm{E}\left[(N_1 - N_2)^2\right] = \mathrm{E}\left[N_1{}^2\right] + \mathrm{E}\left[N_2{}^2\right] - 2\,\mathrm{E}[N_1 N_2] = 4E_T N_o(1-\rho_j) \tag{2.12}$$

Então:

$$\Pr[V<0] = \int_{-\infty}^{0} p_v dv = \frac{1}{\sqrt{2\pi\sigma_V}} \int_{-\infty}^{0} \exp\left[\frac{-v - 2\alpha E_T(1-\rho_j)}{2\sigma_V^2}\right] dv$$

$$\Pr[V<0] = \frac{1}{2}\mathrm{erfc}\left(\sqrt{\frac{\alpha^2 E_T}{2N_o}(1-\rho_j)}\right) \tag{2.13}$$

Mas, levando (2.8) em (2.13), temos:

$$P_j = \frac{1}{2}\mathrm{erfc}\left(\sqrt{\frac{\alpha^2 E_b}{N_o} \cdot R \cdot W_j}\right) \tag{2.14}$$

$$P_j = \Pr\left[U_j > U_1\right] = \frac{1}{2}\operatorname{erfc}\left(\sqrt{\frac{E_b'}{N_o}\cdot R\cdot W_j}\right) \tag{2.15}$$

onde:

$$\frac{E_b'}{N_o} = \frac{\alpha^2 E_b}{N_o}$$

Então, a probabilidade de erro de bloco média é limitada superiormente pela soma de 2^k-1 probabilidades do tipo P_j.

$$P_E \le \sum_{j=2}^{2^k-1} P_j$$

$$P_E \le \frac{1}{2}\sum_{j=2}^{2^k-1}\operatorname{erfc}\left[\sqrt{\frac{E_b'}{N_o}\cdot R\cdot W_j}\right] \tag{2.16}$$

Para avaliar a expressão (2.16) precisamos conhecer a distribuição de pesos das palavras-código, o que pode ser obtido, por exemplo, através das identidades de Mac Williams, desde que se conheça o código a ser utilizado ([33], cap. 3; [14], cap. 2).

Alternativamente, podemos modificar (2.15) de forma a dependermos somente da distância mínima (peso mínimo) do código, ao invés de termos que conhecer a distribuição de pesos do código. Para isso, observamos que:

$$\operatorname{erfc}\sqrt{\frac{E_b'}{N_o}\cdot R\cdot W_j} \le \operatorname{erfc}\sqrt{\frac{E_b'}{N_o}\cdot R\cdot d} < \exp\left(-\frac{E_b'}{N_o}\cdot R\cdot d\right) \tag{2.17}$$

Então, teremos:

$$P_E \le \frac{1}{2}\cdot\left(2^k-1\right)\cdot\operatorname{erfc}\sqrt{\frac{E_b'}{N_o}\cdot R\cdot d} < \frac{1}{2}\cdot\exp\left(-\frac{E_b'}{N_o}\cdot R\cdot d + k\ln 2\right) \tag{2.18}$$

Comparando (2.18) com o limitante superior da probabilidade de erro de um sistema 2-PSK não-codificado, que vale:

$$P_E = \frac{1}{2}\exp\left(-\frac{E_b'}{N_o}\right) \tag{2.19a}$$

Podemos definir um *GANHO DE CODIFICAÇÃO* do código como sendo a relação dos expoentes em (2.19) e (2.18) ([34], cap. 5):

$$G_{código} = \frac{-(E_b/N_o)\cdot R\cdot d + k\ln 2}{-(E_b/N_o)} = R\cdot d - k\frac{\ln 2}{(E_b/N_o)}$$

ou

$$G_{código_{dB}} = 10\log\left[R\cdot d - k\frac{\ln 2}{(E_b/N_o)}\right] \text{ dB} \tag{2.19b}$$

As expressões (2.16) e (2.18) fornecem os limites superiores para a PROBABILIDADE DE ERRO DE BLOCO MÉDIA para um sistema 2-PSK com código binário LINEAR E DE PALAVRAS EQUIPROVÁVEIS. A probabilidade de erro de BIT DE INFORMAÇÃO média P_b é muito mais complicada de se calcular a partir de P_E porque, quando há um erro no bloco, alguns dos k BITS DE INFORMAÇÃO podem estar errados e outros corretos. Saber quantos são corretos e quantos são errados é difícil. Entretanto, P_b é proporcional a P_E e a constante de proporcionalidade vale, no caso de *sinais (palavras-código) ortogonais* ([44], cap. 2):

$$\tau = \frac{2^{(k-1)}}{2^k - 1} \tag{2.20}$$

$$P_b = \tau\cdot P_E \tag{2.21}$$

A demonstração da equação (2.20) é dada em Viterbi e Omura ([44], cap. 2) e está resumida abaixo.

Para códigos ortogonais, todos os padrões de erro de bloco são equiprováveis, pois todas as 2^k palavras-código são equidistantes.

Desta maneira, temos as seguintes conclusões:

i - existem $\binom{k}{u}$ combinações ou formas pelas quais u bits em k podem estar errados;

ii- um erro de bloco causará qualquer padrão de erros no vetor de dados (informação), com probabilidades iguais de ocorrência valendo $\frac{P_E}{2^k-1}$, pois há 2^k-1 padrões de erro de bits.

Então, a probabilidade de erro em um bit será:

$$P_b = \frac{1}{k}\cdot\sum_{u=1}^{k} u\cdot\binom{k}{u}\cdot\frac{P_E}{2^k-1}$$

$$P_b = \left(\frac{1}{k}\cdot\sum_{u=1}^{k}\frac{u\cdot k!}{u!\cdot(k-u)!}\right)\cdot\frac{P_E}{2^k-1}$$

$$P_b = \frac{2^{(k-1)}}{2^k-1}\cdot P_E \qquad (2.22)$$

OBS.: o somatório é reduzido a $2^{(k-1)}$ de maneira similar àquela para derivação da fórmula de distribuição de Poisson, já bem conhecida dos textos de estatística; por isto, não fizemos todas as passagens aqui.

A partir de (2.20), observamos que:

$$\lim_{k\to 1}\tau = 1 \qquad (2.23)$$

e

$$\lim_{k\to\infty}\tau = \frac{2^{k-1}}{2^k-1} = \frac{1}{2} \qquad (2.24)$$

Então, para k não muito pequeno, temos, em média, para códigos ortogonais:

$$P_b = \frac{P_E}{2} \qquad (2.25)$$

Segundo ([34], cap. 5; [44], cap. 2], para códigos quaisquer, τ depende das propriedades de distância do código mas não é maior que 1/2, em média. Assim ambos afirmam que:

$$P_b \leq \frac{P_E}{2} \tag{2.26}$$

A referência ([44], cap. 2) vai além e diz que, como limitante inferior, teremos:

$$P_b \geq \frac{P_E}{k} \tag{2.27}$$

Como exemplo, eles fornecem o código de Golay estendido (24,12) sistemático, que tem d=8: para haver um erro de bloco, no mínimo 1/3=8/24 erros nos dígitos da palavra-código têm que ter ocorrido. Como o código considerado é sistemático e de taxa 1/2 (i.e., metade dos símbolos são bits de informação, a mesma razão de 1/3 ocorre, em média, nos bits de informação em erro). Assim, $P_b \cong P_E/3$.

2.3. COMPARAÇÃO DA DECODIFICAÇÃO POR DECISÃO SUAVE COM A DE DECISÃO ABRUPTA

A decodificação por decisão abrupta é aquela que considera a amostra de um símbolo com sendo quantizada em apenas dois níveis ("0" e "1") e passa a tratar estes símbolos quantizados como sendo os dígitos do vetor de entrada do decodificador (i.e., palavra-código mais ruído na entrada do decodificador).

A quantização em 2 níveis é facilmente realizada por um limitador ou comparador, que decide apenas a polaridade do símbolo demodulado, no instante ótimo de amostragem.

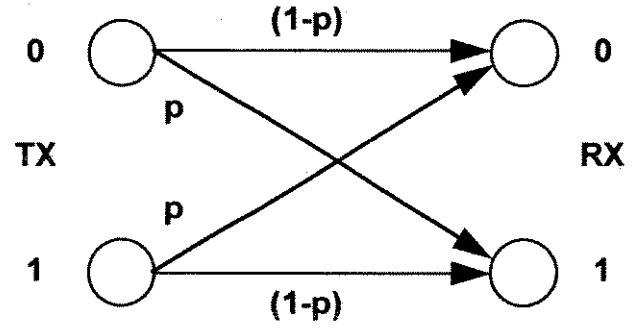
O decodificador usado então é o decodificador BINÁRIO DE DISTÂNCIA MÍNIMA, que usa, como critério de decodificação, a procura da palavra-código com menor distância de Hamming em relação ao vetor recebido. Este critério é o ótimo (do ponto de vista de minimização de P_E) para um canal binário simétrico (BSC) com probabilidades de transição p<1/2 e códigos lineares binários com fonte de bits equiprováveis (i.e. Pr{ transmitir 0 } = Pr{ transmitir 1 }), conforme ([34],[44],[33],[14],[27],[28],[12]).

Relembrando, temos que o BSC, dado na Figura 2.4, tem uma probabilidade de transição p que vale, para modulação 2-PSK ([34], cap. 4; [12], cap. 10]:

$$p = \frac{1}{2} \cdot \operatorname{erfc}\left(\sqrt{\frac{\alpha^2 E_T}{N_o}}\right)$$

$$p = \frac{1}{2} \cdot \operatorname{erfc}\left(\sqrt{\frac{E_b^{'}}{N_o} \cdot R}\right) \qquad (2.28)$$

onde $\frac{E_b^{'}}{N_o} \rightarrow$ relação sinal ruído por bit na entrada do receptor.



se p>1/2 há inversão de símbolos no canal

FIGURA 2.4 - CANAL BINÁRIO SIMÉTRICO (BSC)

De ([34], cap. 5; [44], cap. 2; [33], cap. 4; [14], cap. 1] sabemos que a probabilidade de erro de bloco P_E tem um limitante superior para códigos quase-perfeitos ou perfeitos dado por:

$$P_{E\,HARD} \leq \sum_{m=t+1}^{n} \binom{n}{m} \cdot p^m \cdot (1-p)^{n-m} \qquad (2.29)$$

com $t = \mathrm{INT}\left[(d-1)/2\right]$.

Ou ainda, usando a expressão obtida do fato de que P_E não pode ser maior que o total de palavras-código (exceto a originalmente transmitida) vezes a probabilidade de confundir a palavra transmitida com sua mais próxima ([34], [44], [33], [14]).

$$P_{E\,HARD} \le \left(2^k - 1\right) \cdot \sum_{m=\mathrm{INT}[d/2]+1}^{d} \binom{d}{m} \cdot p^m \cdot (1-p)^{d-m} \tag{2.30}$$

com $d = d_{\mathrm{MIN}}$.

Ou ainda usando o limitante de Chernoff, temos ([34], cap. 5):

$$P_E \le \left(2^k - 1\right)\left[4p(1-p)\right]^{d/2} \tag{2.31}$$

De (2.31) e (2.28):

$$P_E \le \left(2^k - 1\right) \cdot \left[2\,\mathrm{erfc}\sqrt{\frac{E_b'}{N_o} \cdot R} - \left(\mathrm{erfc}\sqrt{\frac{E_b'}{N_o} \cdot R} \right)^2 \right]^{d/2} \tag{2.32}$$

Para um código de Golay (23,12), que é código perfeito, a probabilidade P_E para decisão abrupta vale, exatamente:

$$P_{E_{HARD}} = \sum_{m=4}^{23} \binom{23}{m} p^m (1-p)^{23-m} = 1 - \sum_{m=0}^{3} \binom{23}{m} p^m (1-p)^{23-m} \tag{2.33}$$

Usando a expressão (2.33) e a (2.16), as referências ([34], cap. 5 e [44], cap. 2] avaliam e plotam o desempenho em decodificação por decisão abrupta e o limitante superior de desempenho em decodificação por decisão suave para o Código de Golay (23,12) e para o código estendido (24,12) (ver Figura 2.5 na próxima seção). No caso do código de Golay (23,12), a curva limitante superior para decisão suave apresenta um desempenho melhor em 2 dB (aproximadamente) em relação à curva exata do desempenho da decisão abrupta, para a região $10^{-2} > P_E > 10^{-6}$ (que é a faixa de probabilidades de erro usadas na prática no atual estágio tecnológico). Para o código estendido de Golay (24,12), a diferença entre as curvas também é de 2 dB, na região de interesse para a P_E, a favor da decisão suave. Esta diferença de 2 dB é aplicável não só a estes códigos, mas a todos os outros códigos de bloco lineares sobre um canal de ruído branco, aditivo e gaussiano (AWGN), sendo praticamente insensível ao tipo de código usado. A melhor performance da decodificação por decisão suave se deve a um aumento na capacidade de canal, conforme mostrado abaixo (veja equações (2.39) e (2.42)).

2.3.1. CAPACIDADES DE CANAL AWGN, COM CÓDIGO BINÁRIO, MODULAÇÃO 2-PSK COERENTE E DECODIFICAÇÃO POR DECISÃO SUAVE E ABRUPTA

Usando-se um decodificador por decisão abrupta, o modelo de canal utilizado é o BSC da Figura 2.4 e a capacidade de canal C valerá ([34], [44], [14], [12]), em bits/símbolo (ou bits/dígito):

$$C = 1 + p \cdot \log_2 p + (1-p) \cdot \log_2 (1-p) \tag{2.34}$$

com p dado por (2.28).

Supondo C=R (taxa k/n do código) e substituindo (2.28) em (2.34), chegamos à expressão abaixo:

$$R = 1 + \sqrt{\frac{E_b'}{N_o} R} \log_2 \sqrt{\frac{E_b'}{N_o} R} + \left(1 - \sqrt{\frac{E_b'}{N_o} R}\right) \log_2 \left(1 - \sqrt{\frac{E_b'}{N_o} R}\right) \tag{2.35}$$

Se R tende a ser nulo (i.e., palavra-código infinita), teremos que a equação (2.28) se transforma em:

$$\lim_{R \to 0} p \cong \frac{1}{2} - \sqrt{\frac{1}{\pi} \cdot \frac{E_b'}{N_o} \cdot R} \tag{2.36}$$

(aproximação para valores pequenos da função erro complementar).

Ainda para valores pequenos de x, sabemos que:

$$\log_2 (1+x) \cong \frac{\left(x - \frac{x^2}{2}\right)}{\ln 2} \tag{2.37}$$

(truncagem da expansão de Taylor do $\ln(1+a) = a - a^2/2 + \cdots$)

Então, levando (2.37) e (2.36) em (2.34), teremos o seguinte valor para a capacidade de canal C para o caso de decisão abrupta:

$$C = \frac{2}{\pi \ln 2} \cdot \frac{E_b'}{N_o} \cdot R \tag{2.38}$$

Se C=R e R tende a zero:

$$\lim_{R\to 0}\frac{E_b'}{N_o}=\frac{\pi}{2}\cdot\ln 2=0{,}37\ \text{dB} \tag{2.39}$$

Para o canal AWGN com decodificação por decisão suave, a capacidade C valerá (ver Figura 2.1a para notação) ([34], cap. 5; [44], cap. 2; [25]):

$$C=\frac{1}{2}\left\{\int_{-\infty}^{+\infty}p(y/x=0)\log_2\frac{p(y/x=0)}{p(y)}dy+\int_{-\infty}^{+\infty}p(y/x=1)\log_2\frac{p(y/x=1)}{p(y)}dy\right\} \tag{2.40a}$$

onde $p(y/x=0)$ e $p(y/x=1)$ representam as densidades de probabilidades condicionais da saída demodulada dado que foi transmitido um 0 ou um 1. Como o canal é AWGN:

$$p(y/x=0)=\frac{1}{\sqrt{2\pi}\cdot\sigma}\cdot e^{-(y-m_0)^2/2\sigma^2} \tag{2.40b}$$

$$p(y/x=1)=\frac{1}{\sqrt{2\pi}\cdot\sigma}\cdot e^{-(y-m_1)^2/2\sigma^2} \tag{2.40c}$$

onde $m_0=-2\alpha E_S$, $m_1=+2\alpha E_S$, energias associadas aos símbolos 0 e 1, $\sigma^2=2E_SN_o$, $E_S=R\cdot E_b$ e ainda:

$$P(y)=\frac{1}{2}\{p(y/x=0)+p(y/x=1)\}$$

Conforme se sabe, para um canal analógico (decisão suave sem quantização), quando R tende a ser nulo, de forma que a mensagem é infinita, a teoria de informação para canais contínuos prova que:

$$\lim_{R\to 0}C=\frac{1}{\ln 2}\cdot\frac{E_b'}{N_o}\cdot R \tag{2.41}$$

Se $C=R$:

$$\lim_{R\to 0}\frac{E_b'}{N_o}=\ln 2=-1{,}6\ \text{dB} \tag{2.42}$$

Ora, à medida que n aumenta, e R tende a ser nulo (mensagem infinita), o ganho obtido pela decisão suave em relação à abrupta é:

$$\Delta_{SH} = \frac{E_b/N_o\big|_{R\to 0}^{HARD}}{E_b/N_o\big|_{R\to 0}^{SOFT}} = \frac{(\pi/2)\ln 2}{\ln 2} = \frac{\pi}{2}$$

$$\Delta_{SH}\big|_{dB} = 10\log\frac{\pi}{2} \cong 2 \text{ dB} \qquad (2.43)$$

Para taxas $0 < R \leq 1$ de codificação, este ganho da decisão suave sobre a decisão abrupta pode ser calculado através de métodos numéricos e, à medida que $R \to 1$, $\Delta_{SH}\big|_{dB} \to 1$ dB.

As figuras 2.5 e 2.6, plotadas por Proakis em ([34], cap. 5), mostram, respectivamente, as curvas P_E x E_b/N_O e R x (E_b/N_O) mínimo para decisões suave e abrupta em um código de Golay (23,12).

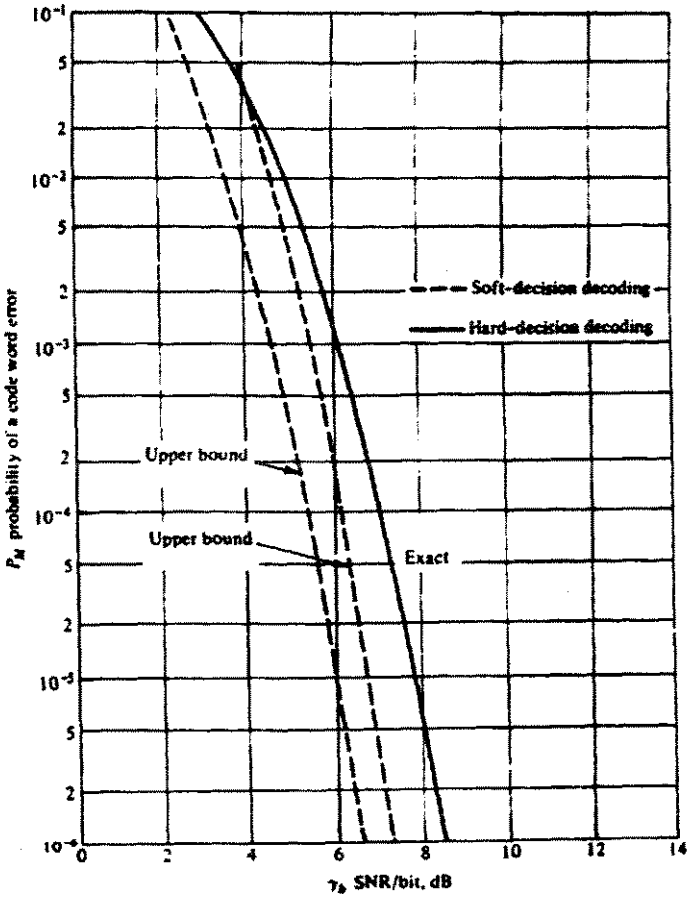


FIGURA 2.5. CURVAS E_b/N_O X P_E PARA CÓDIGO DE GOLAY (23,12) ([34], CAP.5)

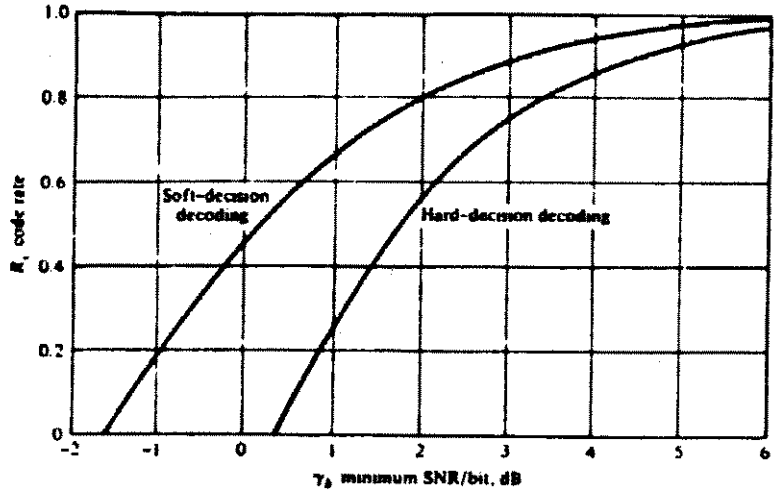


FIGURA 2.6. CURVAS *R* X E_b/N_O MÍNIMO PARA CÓDIGOS DE BLOCO LINEARES: RELATIVOS ÀS EQUAÇÕES (2.35) E (2.40) ([34], CAP.5)

Um outro meio de avaliarmos a diferença de ganhos, de forma mais aproximada que a anterior, baseia-se nos argumentos abaixo.

Observando a equação (2.19) notamos que, no limite, quando a relação sinal-ruído é muito alta, i.e., E_b/N_O tende a infinito, o ganho assintótico para a decisão suave valerá:

$$\lim_{E_b/N_o \to \infty} G_{cod}\big|_{dB} = 10\log[R \cdot d] \text{ dB para decisão suave.}$$

Similarmente, a partir da equação (2.32) obtemos para decisão abrupta:

$$\lim_{E_b/N_o \to \infty} G_{cod}\big|_{dB} = 10\log[R \cdot (t+1)] = 10\log\left[\frac{\mathrm{R \cdot d}}{2}\right] \text{ dB para decisão abrupta.}$$

Desta forma, a diferença entre os ganhos seria de 3 dB a favor da decisão suave, e no caso do CGE (24,12) os ganhos assintóticos de codificação seriam de 6 dB para a decisão suave e 3 dB para a decisão abrupta.

Os argumentos acima são encontrados em ([13];[14]). Entretanto, conforme provamos na equação (2.43) a diferença máxima entre os ganhos em decisão suave e abrupta é de 2 dB.

2.4. EFEITOS DA QUANTIZAÇÃO SOBRE O DESEMPENHO DE UM DECODIFICADOR POR DECISÃO SUAVE

Toda a análise de desempenho do decodificador por decisão suave para códigos de bloco lineares feita até aqui supôs, como modelo de canal, um receptor que usa amostras PAM na decodificação, que obviamente tem entrada de amostras não-quantizadas.

Na prática, todavia, o receptor de decisão suave (que é ótimo) tem limitações tecnológicas que impõem restrições sobre a implementação. Esta, via de regra, é feita como indica a Figura 2.1a e b.

Há um demodulador realizado com técnicas analógicas e um decodificador que é uma máquina digital para cálculos. Para interfacear o DEMOD e o DECOD coloca-se então um quantizador em Q níveis que fornece à entrada do DECOD um número igual a $\left[\mathrm{INT}(\log_2 Q)+1\right]$ entradas de seqüências binárias referentes à amostra regenerada e quantizada do "dígito analógico" demodulado.

A quantização impõe degradações ao desempenho do decodificador em relação àquele atingido por um receptor ótimo (não-quantizado). Comentamos aqui quão grande é esta degradação em função do número de níveis de quantização.

Uma primeira conclusão óbvia é que a degradação $D(Q)$ introduzida pela quantização de Q níveis vale necessariamente:

$$\lim_{R\to 0} D(Q) \le 2 \text{ dB} \quad \forall \mathrm{C}, \forall Q, R = k/n \tag{2.44}$$

porque a quantização mínima Q=2 níveis (0 e 1) equivale à decisão abrupta e 2 dB é a diferença de desempenho entre a decisão suave não-quantizada e a decisão abrupta na região de interesse $10^{-2} \le P_E \le 10^{-6}$.

A segunda conclusão, vinda de resultados de simulações feitas por diversos autores ([14], cap. 1; [26]) é que, na maioria dos casos, uma quantização em 8 níveis (quantizador de 3 bits de saída) tem uma performance da ordem de 0,2 dB a 0,5 dB pior que a do decodificador não-quantizado. Clark e Cain ([14], cap. 1) mostram curvas de E_S/N_0 x P_E considerando várias combinações de níveis de quantização (8 e 4 níveis, e 3 níveis com uma zona de apagamento) e formas de quantização (espaçamentos lineares, simétricos e uniformes, com Δ fixo entre as regiões de decisão; espaçamentos baseados em métrica ótima, com Δ = proporcional às métricas dos símbolos j, $m_j = -A - B\log(\Pr[j/0])$, e espaçamentos uniformes mas considerando regiões de não-decisão não-uniformes que indicariam "apagamento"), considerando um código de repetição simples (7,1). Os resultados obtidos por eles indicam que a quantização em 8 níveis, tanto com espaçamento linear ou ótimo,

degradam apenas 0,2 dB em relação à quantização infinita (i.e., amostras não-quantizadas), na região $10^{-7} \leq P_E \leq 10^{-3}$, sendo que para $P_E \leq 10^{-3}$ a curva para quantização linear diverge da quantização ótima, degradando o desempenho.

Os efeitos da quantização podem ser "encarados" de duas maneiras distintas:

a) ou consideramos a quantização em Q níveis introduzindo um *ruído de quantização* a ser somado ao ruído de canal ([28]; [35], cap. 5; [25], cap. 5; [30], cap. 9] e que degradará o desempenho;

b) ou consideramos a quantização em Q níveis como *alterando o modelo de canal* e influindo na capacidade de canal C;

Ambos os modelos levam a resultados finais idênticos para o desempenho pois, no fundo, são equivalentes. Mas o método b) tem algumas vantagens analíticas. Viterbi e Omura ([44], cap. 3] obtiveram algumas expressões analíticas para a capacidade de canal C em nats/símbolo para diversos tipos de canais a partir da equação (2.40a) (maior detalhamento ver ref. [44], cap. 3, seção 3.4):

a) para um canal AWGN contínuo (não-quantizado) de entrada binária:

$$C \cong \frac{E_S}{N_o} \text{ para } \frac{E_S}{N_o} << 1 \text{ (comparar com eq.(2.41))} \tag{2.45}$$

b) para um canal BSC (2 níveis de quantização), que é o pior caso para canais quantizados:

$$C \cong \frac{2}{\pi} \cdot \frac{E_S}{N_o} \text{ para } \frac{E_S}{N_o} << 1 \text{ (comparar com eq.(2.38))} \tag{2.46}$$

c) para um canal Q-ário com Q=8 níveis, considerando ruído AWGN, quantização uniforme com $\frac{\Delta}{2} = \frac{1}{2} \cdot \sqrt{\frac{N_o}{2}}$ e entradas binárias (Δ é o espaçamento de quantização):

$$C \cong 0{,}95 \frac{E_S}{N_o} \text{ para } \frac{E_S}{N_o} << 1 \left(\text{OBS.: } 10 \log_{10} \frac{1}{0{,}95} = 0{,}22 \text{ dB} \right) \tag{2.47}$$

Então, uma conclusão *importante* para uma engenharia de projeto de sistemas codificados que leve em consideração a relação custo-benefício é que com *Q=8 níveis de quantização, a degradação no desempenho é desprezível em relação à não-quantizada para decodificação por decisão suave, e, para qualquer código, vale aproximadamente 0,22 dB.*

CAPÍTULO 3

ALGORITMOS PARA DECODIFICADORES DE CÓDIGOS DE BLOCO POR DECISÃO SUAVE

3.1. INTRODUÇÃO

Pretendemos neste capítulo mostrar alguns dos algoritmos desenvolvidos para decodificação de códigos de bloco usando a técnica de decisão suave. É óbvio que tais algoritmos se aplicam a decodificadores de decisão abrupta, que daqui para a frente serão chamados DECODIFICADORES DE DISTÂNCIA MÍNIMA (de Hamming), mas que não são normalmente usados para tal, dada a implementação mais complexa do que a enormidade de decodificadores binários já desenvolvidos e estudados, sem nenhum ganho em desempenho ([14], [27], [33], [34]).

Apresentamos aqui os seguintes algoritmos:

a) algoritmo de Massey (APP ou decisão majoritária);
b) algoritmo de Hartmann-Rudolph;
c) algoritmo de Greenberger;
d) algoritmo de Weldon;
e) algoritmo de Omura ("conjunto de informação").

Existem ainda os algoritmos de Chase e suas variantes (Hackett, etc.) que serão estudados no próximo capítulo, visto que são os mais interessantes de se implementar na prática devido ao ganho que podem fornecer aliado à simplicidade de implementação dentro do atual espectro de tecnologias de CIs disponíveis e aplicações em sistemas de comunicação.

Os critérios de decisão, que todos os algoritmos citados acima usam, podem ser três:

— *máximo a posteriori (MAP)*, aplicável a qualquer canal;
— *máxima verossimilhança (MV)*, que é equivalente ao MAP só para fontes de símbolos equiprováveis à entrada do canal;

— *mínima distância (MD)*, que é equivalente à MV para canais Q-ários "simétricos" em relação à fonte binária equiprovável de entrada (i.e., $p(Q-1/0)=p(0/1)$, $p(Q-2/0)=p(1/1)$, etc; ver Figura 2.1.b).

3.2. ALGORITMO DE MASSEY

O algoritmo de Massey, também chamado de APP (*a posteriori* probability), decodificação por limiar (threshold), ou ainda decodificação por lógica majoritária "generalizada" é uma *extensão* da decodificação *binária* por lógica majoritária. Assim, é uma técnica que minimiza a probabilidade de erro média de *símbolo*, não de bloco, e que no limite para k infinito, dá mínima probabilidade de erro média de bit (assim como de bloco).

Sendo uma extensão da idéia de Massey para os decodificadores binários de lógica por decisão majoritária, podemos aplicar aqui todos os conceitos relativos a esta técnica: ortogonalização de códigos, decodificação em 1 ou mais passos, decodificadores por lógica majoritária tipo I ou tipo II, etc. ([27], cap. 7; [14], cap. 3; [33], cap. 10).

A idéia básica é estimar o valor do j-ésimo dígito c_j da palavra-código $\mathbf{c}$ de forma que:

$$c_j = V \Leftrightarrow \mathrm{MAX}\left\{\Pr\left[c_j = x / \{B_i\}\right]\right\} = \Pr\left[c_j = V / \{B_i\}\right] \tag{3.1}$$

onde $\{ B_i \}$ é um conjunto de dígitos B_is obtidos de somas de verificação de paridade excluído o dígito correspondente ao j-ésimo dígito do vetor-erro.

Pela regra de Bayes:

$$\Pr\left[c_j = V / \{B_i\}\right] = \frac{\Pr\left[\{B_i\} / c_j = V\right] \cdot \Pr\left[c_j = V\right]}{\Pr\left[\{B_i\}\right]} \tag{3.2}$$

podemos modificar a eq.(3.1) de forma que a regra de decodificação passa a ser:

$$c_j = 1 \Leftrightarrow \Pr\left[\{B_i\} / c_j = 0\right] \cdot \Pr\left[c_j = 0\right] < \Pr\left[\{B_i\} / c_j = 1\right] \cdot \Pr\left[c_j = 1\right] \tag{3.3}$$

Como o código é completamente ortogonalizável (por hipótese), teremos $J=d-1$ somas de verificação de paridade *ortogonais* sobre o j-ésimo dígito e então, devido à ortogonalidade, todas as probabilidades $\Pr\left[B_i / c_j = x\right]$ serão independentes e, por isto:

$$\Pr\left[\{B_i\}/c_j = x\right] = \prod_{i=0}^{J} \Pr\left[B_i / c_j = x\right]$$

A equação (3.3) fica então:

$$c_j = 1 \Leftrightarrow \Pr\left[c_j = 0\right] \cdot \prod_{i=0}^{J} \Pr\left[B_i / c_j = 0\right] < \Pr\left[c_j = 1\right] \cdot \prod_{i=0}^{J} \Pr\left[B_i / c_j = 1\right]$$

Se Pr[c_j=0]=Pr[c_j=1] (hipótese de símbolos equiprováveis), e transformando o produtório em somatório de logaritmos, vem que:

$$c_j = 1 \text{ se e somente se } \sum_{i=0}^{J} \log\left[\frac{\Pr\left[B_i / c_j = 0\right]}{\Pr\left[B_i / c_j = 1\right]}\right] < 0 \qquad (3.4)$$

Sabe-se ainda que:

$$B_i = c_j + \sum_{\substack{l=1 \\ l \neq j}}^{n_i} e_{il} \qquad (3.5)$$

onde e_{il} é o l-ésimo termo da i-ésima soma de verificação de paridade e n_i é o número de dígitos que compõem a soma ortogonal sobre c_j.

De (3.5) observamos que, se não houver erros, cada B_i é uma estimativa independente de c_j. Ainda vemos que:

a) B_i=0 se e somente se c_j=0 e n° par de erros ocorre ou
c_j=1 e n° ímpar de erros ocorre
b) B_i=1 se e somente se c_j=0 e n° ímpar de erros ocorre ou
c_j=1 e n° par de erros ocorre

Vamos assumir agora que p_i é a probabilidade de ocorrência de um número ímpar de erros na equação que define B_i e q_i=1–p_i a de ocorrência de um número par de erros. Então:

$$\Pr\left[B_i = 0 / c_j = 0\right] = \Pr\left[B_i = 1 / c_j = 1\right] = q_i = 1 - p_i \qquad (3.6.a)$$

$$\Pr\left[B_i = 1 / c_j = 0\right] = \Pr\left[B_i = 0 / c_j = 1\right] = p_i \qquad (3.6.b)$$

Então, se B_i=0:

$$\frac{\Pr\left[B_i / c_j = 0\right]}{\Pr\left[B_i / c_j = 1\right]} = \frac{q_i}{p_i} \tag{3.6.c}$$

Se B_i=1:

$$\frac{\Pr\left[B_i / c_j = 0\right]}{\Pr\left[B_i / c_j = 1\right]} = \frac{p_i}{q_i} = \left(\frac{q_i}{p_i}\right)^{-1} \tag{3.6.d}$$

Usando o artifício de que a expressão (1-2B_i) vale:

$$(1-2B_i) = \begin{cases} 1 \text{ se } B_i = 0 \\ -1 \text{ se } B_i = 1 \end{cases}$$

Podemos dizer que:

$$\frac{\Pr\left[B_i / c_j = 0\right]}{\Pr\left[B_i / c_j = 1\right]} = \left(\frac{q_i}{p_i}\right)^{(1-2B_i)}, \text{ para todo } B_i \in \{0,1\} \tag{3.6.e}$$

Então, levando (3.6.e) em (3.4), e usando propriedades de logaritmos, a regra de decodificação será:

$$c_j = 1 \text{ se e somente se } \sum_{i=0}^{J}(1-2B_i)\log\left(\frac{q_i}{p_i}\right) < 0 \tag{3.7}$$

Trabalhando a expressão (3.7), podemos modificar a regra de decisão para a dada a seguir:

$$c_j = 1 \text{ se e somente se } \sum_{i=0}^{J} B_i \log\left(\frac{q_i}{p_i}\right) > \frac{1}{2}\sum_{i=0}^{J}\log\left(\frac{q_i}{p_i}\right) \tag{3.8}$$

Esta expressão é similar àquela usada para o decodificador binário de lógica majoritária tipo II, que é dada por:

$$c_j = 1 \text{ se e somente se } \sum_{i=0}^{J} B_i w_i > \frac{1}{2} T \tag{3.9.a}$$

com ω_i sendo um fator peso proporcional à confiabilidade atribuída à i-ésima soma de verificação de paridade (no caso de lógica majoritária, todo ω_i=1) e

$$T = \sum_{i=0}^{J} \omega_i \tag{3.9.b}$$

Para a decisão suave, pode-se estabelecer este fator peso como sendo o logaritmo da relação de probabilidades, conforme se vê na equação (3.8).

Entretanto, p_i e q_i podem ser avaliados somente em função das saídas y_{il} e de um demodulador não-quantizado. Assim, de acordo com [14], se γ é a probabilidade de ocorrer um erro sobre o dígito c_j na equação B_i, teremos, para 2-PSK coerente e canal AWGN:

$$\gamma_{il} = \Pr[\text{erro} / y_{il}] = \frac{\exp\left[\dfrac{-4\sqrt{E_S}\,|y_{il}|}{N_o}\right]}{1 + \exp\left[\dfrac{-4\sqrt{E_S}\,|y_{il}|}{N_o}\right]} \tag{3.10}$$

E então, no caso de decisão suave, poderíamos, de acordo com Clark e Cain [14], estabelecer como fator peso ω_i uma relação que valerá:

$$\omega_i = \log\frac{q_i}{p_i} = \log\left[\frac{1 + \prod_{l=1}^{n_i}(1 - 2\gamma_{il})}{1 - \prod_{l=1}^{n_i}(1 - 2\gamma_{il})}\right] \tag{3.11}$$

Para a eq. (3.11) ser mais tratável em hardware, pode-se provar [14] que, se

$$\beta_{il} = -\ln(1 - 2\gamma_{il}) \tag{3.12}$$

vem que:

$$\omega_i = \log\left[\operatorname{cotgh}\left(\frac{1}{2}\sum_{l=1}^{n_i}\beta_{il}\right)\right] \qquad (3.13)$$

Assim, o cálculo dos ω_i's não depende de divisões das entradas r_{il}, *mas só de somas de funções em tabelas (ROMs)* das entradas r_{il} do decodificador (saídas do demodulador).

A implementação deste algoritmo APP de Massey para decodificadores que usam decisão suave é especialmente simples em dois casos, onde o hardware exigido é bastante simplificado:

a) canal do tipo apagamento, binário (i.e., 3 níveis de quantização);

b) só se usa a informação de confiabilidade (B_j) associada ao símbolo de menor confiabilidade (decodificador de distância mínima de Forney);

Para ambos os casos pode-se mostrar que o algoritmo APP de Massey é ótimo *assintoticamente.*

Maiores detalhes do algoritmo estão em Clark & Cain [14]. Lee & Weng [26] discutem a implementação do algoritmo (modificado por Goathal) no caso de canal AWGN com quantizador de 2 bits (4 níveis), usando só o símbolo menos confiável (caso b das simplificações), aplicado a um Código de Golay Estendido (24,12). Suas conclusões foram:

a) para uma faixa de probabilidade de erro média de bit $10^{-6} \leq Pb \leq 10^{-2}$, o decodificador implementado em decisão suave tem um desempenho de 1 dB (aproximadamente) melhor que o decodificador por decisão abrupta, no máximo;

b) o decodificador em decisão suave implementado usa 90 chips TTL-LS-MSI e 4 PROMs;

c) o decodificador em decisão suave implementado consegue ir a velocidades de entrada de 8 Mbit/s, (i.e., 4 Mbit/s de dados de informação na saída, já que o código tem R= 1/2).

Conforme veremos no capítulo 6, podemos, para este mesmo código, usar um decodificador com o algoritmo 2 de Chase, com desempenho melhor ainda (1,5 dB ou mais de ganho em relação ao de decisão abrupta), número comparável de chips (100 HCMOS, 3 RAMs, 3 EPROMs, 2 PROMs) e quantização em 3 bits (8 níveis) de canal AWGN, perdendo na velocidade máxima (que seria da ordem de 2,5 Mbit/s para o canal e 1,25 Mbit/s para bits de informação, se usarmos RAMs de 25 ns). Isto acontece porque, para aumentar o desempenho, precisamos aumentar a complexidade da lógica e, conseqüentemente, quase sempre a máxima velocidade de operação cai. Entretanto, o requisito de velocidade do sistema onde o nosso decodificador operará é menor, sendo a taxa de bits de informação igual a 534 kbit/s.

3.3. ALGORITMO DE HARTMANN-RUDOLPH (HR)

Também conhecido como decodificação ótima símbolo-a-símbolo, o algoritmo de Hartmann-Rudolph minimiza a probabilidade de erro média de um símbolo e não de bloco (assim como o de Massey). Apesar desta minimização levar, no limite, à mínima probabilidade de erro média de bloco (e também a de bit), é curioso notar que estes métodos podem levar a termos na saída do decodificador palavras que não são, necessariamente, palavras-código!

Este algoritmo é apropriado para códigos cíclicos e/ou códigos que tenham um número pequeno de dígitos de paridade (i.e., taxa R alta). Isto ocorre porque devemos calcular uma função de decisão complicada para todo e qualquer dígito de cheque de paridade (em contraste com o de Massey, onde só se computa aquelas ortogonais sobre um certo dígito). Para a compreensão da regra de decisão (decodificação) necessitamos de algumas definições, dadas a seguir:

a) **DEF. 1**: Sejam C e C' um código de bloco e seu dual. Seja c'_i a i-ésima palavra-código do código dual C' e c_{ij} o j-ésimo dígito. A razão de verossimilhança do j-ésimo dígito recebido r_j é dada por:

$$\Phi_j = \frac{\text{Prob}(r_j/1)}{\text{Prob}(r_j/0)} \tag{3.14}$$

b) **DEF. 2** : Seja a grandeza P_j

$$P_j = \frac{1-\Phi_j}{1+\Phi_j} \tag{3.15}$$

Então a regra de decisão será baseada em uma função de decisão f_d e é dada por:

$$c_m = 0 \Leftrightarrow f_d = \sum_{i=1}^{2n-k} \prod_{j=0}^{n-1} p_j^{c'_{ij} \oplus I_{jm}} \geq 0 \tag{3.16}$$

onde:

$I_{jm} = \begin{cases} 1 \text{ se } j = m \\ 0 \text{ se } j \neq m \end{cases}$, $\oplus$ = ou-exclusivo e c'_{ij} é o j-ésimo dígito da i-ésima palavra-código do código dual C' e c_m é o m-ésimo dígito da palavra decodificada.

Esta regra de decisão dada por (3.16) é melhor compreendida se soubermos que a função de decisão f_d vale:

$$f_d = \text{Prob}[c_m = 0/\mathbf{r}] \quad - \quad \text{Prob}[c_m = 1/\mathbf{r}] \tag{3.17}$$

Chegar de (3.17) a (3.16) é mostrado em ([14], cap. 4).

O problema agora é tornar (3.16) implementável. Para tanto, podemos observar que, levando (3.14) em (3.15), vem:

$$P_j = \frac{\text{Prob}(r_j/0) - \text{Prob}(r_j/1)}{\text{Prob}(r_j/0) + \text{Prob}(r_j/1)}$$

Seja r_j o dígito recebido e $\hat{r}_j$ o seu valor quantizado em 2 níveis. Então:

$$\text{Prob}\left[\hat{r}_j \text{ é errado}/r_j\right] = \frac{\text{MIN}\left[\text{Prob}(r_j/0), \text{Prob}(r_j/1)\right]}{\text{Prob}(r_j/0) + \text{Prob}(r_j/1)}$$

Com isto, teremos:

$$\left|P_j\right| = 1 - 2\,\text{Prob}\left[\hat{r}_j \text{ é errado}/r_j\right]$$

o que nos leva a

$$P_j = (-1)^{\hat{r}_j}\left[1 - 2\,\text{Prob}\left[\text{erro}/r_j\right]\right]$$

Desta forma, usando as definições de equações B_i de verificação de paridade, vem a seguinte "interpretação" para a função f_d.

Com B_i, q_i e p_i definidos como no algoritmo de Massey, sem a exigência de ortogonalidade sobre um determinado dígito de paridade (ver equações (3.5) e (3.6)).

$$f_d = \sum_{i=1}^{2^{n-k}} (1 - 2B_i)(q_i - p_i) \tag{3.18}$$

Então, a implementação envolve um processo subdividido em:

a) uso da matriz [H] de verificação de paridade do código C para calcular toda e qualquer estimativa em c_m (m-ésimo dígito) a partir das amostras dos dígitos recebidos decididos abruptamente (2 níveis), $\hat{r}_j$'s;

b) Ponderar cada estimativa com a diferença entre a probabilidade de que a estimativa é correta e a probabilidade de que ela é incorreta;

c) Somar todas as estimativas ponderadas e comparar com zero.

Neste sentido, o algoritmo HR é uma generalização do APP de Massey, envolvendo o cômputo de muitas equações de somas de verificação de paridade não necessariamente ortogonais. Daí o fato de ser eficiente apenas para códigos de taxa R alta.

Uma alternativa para diminuir o número de equações de verificação de paridade derivada do algoritmo HR é o algoritmo de Greenberger dado a seguir. Outro algoritmo derivado do HR é o de Hwang [20].

3.4. ALGORITMO DE GREENBERGER

Este algoritmo derivado do HR procura, obviamente, minimizar a probabilidade de erro média de símbolo (como no HR), mas descartando equações de paridade que contenham símbolos recebidos de baixa confiabilidade, i.e., que contêm pouca informação. O algoritmo consiste em 8 passos:

1) Ordenar os símbolos recebidos em ordem decrescente de confiabilidade (i.e., distância de seu valor quantizado em relação ao limiar de decisão abrupta (limiar de polaridade)).

2) Ordenar as colunas de [H] (matriz de verificação de paridade) tal que a coluna correspondente ao dígito mais confiável é a 1ª coluna (i.e., a mais à esquerda), etc, em ordem decrescente de confiabilidade, reordenando colunas da esquerda para a direita;

3) Fazer operações de linhas (soma, permutação, etc), elementares, tal que a matriz [H] reordenada seja nula acima da diagonal principal de uma sub-matriz quadrada que começa na parte superior e na parte mais à direita desta; a 1ª linha será nula nas $n-k-1$ posições menos confiáveis, a 2ª será nula em $n-k-2$ posições menos confiáveis, etc;

4) Usar a 1ª linha desta matriz [H] obtida em 3 aplicando o método HR para estimar cada símbolo recebido. Isto fará com que se use só duas palavras-código do código dual que são o vetor nulo $\mathbf{0}$ e a 1ª linha de [H], $\mathbf{h}_1$.

5) Repetir 4, usando 2 linhas de [H] obtida em 3, o que gerará 4 palavras do código dual: $\mathbf{0}$, $\mathbf{h}_1$ e outras duas que conterão a contribuição do $(k+2)$-ésimo símbolo mais confiável.

6) Continuar 5, adicionando novas linhas de [H] obtida em 3 até que as estimativas tenham variado pouco, estabilizando-se, ou até um certo número limite de iterações estabelecido a priori.

7) Baseado nas últimas estimativas dos símbolos recebidos, escolher os k mais confiáveis que formam um CONJUNTO DE INFORMAÇÃO (qualquer conjunto de k dígitos de palavra-código de um C(n,k) que possam ser especificados de forma independente, i.e., k dígitos obtidos através de colunas L.I. na matriz [G] do código C; ver [14]). Codificar novamente estes k dígitos, regenerando a palavra-código;

8) Reordenar os símbolos e transferir os k bits de informação ao destinatário.

Greenberger testou o algoritmo para o código de Golay (23,12) e precisou gerar 64 palavras do código dual, bem menos que as 1024 palavras do HR completo.

O algoritmo é orientado à aplicação de microprocessadores, sendo realizado por software.

Duas desvantagens de implementação se tornam evidentes:

a) Grande número de palavras testadas quando comparado com as 16 de um algoritmo 2 de Chase para o código de Golay (23,12) (tanto o de Greenberger como o HR completo), sem ganhar em desempenho (como veremos, o algoritmo de Chase é ótimo pois é equivalente a um receptor por correlação).

b) Ambos os algoritmos, tanto o de Greenberger como o HR completo hoje só são viáveis economicamente usando-se microprocessadores dado que os cálculos são extensos; conseqüentemente, as velocidades onde são aplicáveis têm que ser baixas (tipicamente menores que 64 kbit/s).

3.5. ALGORITMO DE WELDON

O procedimento imaginado por Weldon utiliza vários decodificadores binários de decisão abrupta para fornecer estimativas iniciais de símbolos e, a partir delas e de informações de confiabilidade dos dígitos, poder iniciar o cálculo de uma função de decisão.

Os seguintes passos devem ser efetuados:

1) O decodificador recebe uma palavra de N dígitos quantizados em Q níveis (geralmente, $Q = 2^m \Rightarrow$ quantizador de m bits), que serão ponderados (cada nível i, i=0 a Q–1) por uma função-peso σ_i tal que:

$$\sigma_0 = 0 \tag{3.19.a}$$

$$\sigma_0 \le \sigma_1 \le \sigma_2 \le \cdots \le \sigma_{Q-1} \tag{3.19.b}$$

$$\sigma_i + \sigma_{Q-1-i} = \sigma_{Q-1} \tag{3.19.c}$$

para $Q = 2^m \Rightarrow \{\sigma_i\} = \{0,1,2,\ldots,Q-1 \,/\, \sigma_i \in \mathbf{Z}\}$;

2) Cada σ_i recebido é representado por um número binário de r bits. Cria-se, então, r seqüências binárias diferentes de n dígitos. Como, por hipótese, vale (3.19), vem que:

$$\sigma_i = \sum_{j=1}^{r} a_{ij} V_j \qquad (3.20)$$

onde a_{ij} é um número binário e Vj é um número não-negativo, ponderador de a_{ij}, tal que:

$\sum_{j=1}^{r} V_j = \sigma_{Q-1}$.O número r é tal que nunca precisará ser maior que $\mathrm{INT}[(Q+1)/2]$, o que garantiria

$$\sigma_1 = V_1$$
$$\sigma_2 = V_1 + V_2$$
$$\sigma_{\mathrm{INT}[Q-1/2]} = V_1 + V_2 + \cdots + V_{\mathrm{INT}[(Q-1)/2]}$$
$$\sigma_{Q-1} = V_1 + V_2 + \cdots + V_{\mathrm{INT}[(Q+1)/2]} ;$$

3) Usando r decodificadores binários (decisão abrupta), decodificar cada uma das r seqüências e obter r palavras-código. Cada bit de cada palavra-código é, conseqüentemente, tratado como se fosse uma estimativa independente do símbolo transmitido;

4) Combinar as r estimativas bit a bit usando uma função de decisão:

$$c_j = 0 \Leftrightarrow f_d = \sum_{i=1}^{r} \left(1 - 2B_{ij}\right) \omega_i \qquad (3.21)$$

onde B_{ij} é o j-ésimo bit na i-ésima seqüência decodificada e ω_i é um "peso", proporcional à confiabilidade da estimativa, que vale:

$$\omega_i = V_i R_i \qquad (3.22)$$

com:

$V_i \rightarrow$ número fixo associado a cada decodificador ($V_i = 2^{i-1}$, i =1 para o bit menos significativo e assim por diante)

R_i= MAX [0, d_{MIN}–2 n_i]; n_i = nº de erros na decisão abrupta cometidos pelo i-ésimo decodificador.

Desta maneira, o algoritmo de Weldon permite que se corrija qualquer padrão de erros cujo peso suave é menor que $(Q-1).d_{MIN}/2$.

Este algoritmo, apesar de ser facilmente implementável não tem um desempenho muito bom quando comparado a outros cuja complexidade de hardware não é muito maior que a dele (por exemplo, o algoritmo de Massey, conforme ([14], [26]).

Tal como os anteriores, ele também procura minimizar a probabilidade de erro média de símbolo.

3.6. ALGORITMOS BASEADOS EM CONJUNTOS DE INFORMAÇÃO

Estes algoritmos baseados em conjuntos de informação, juntamente com o algoritmo de Chase (e suas variantes) minimizam a probabilidade de erro de bloco média, o que faz com que se minimize a probabilidade de erro média de bit. Assim, todos estes algoritmos sempre produzem, à saída do decodificador, palavras-código.

Os algoritmos baseados em conjuntos de informação para decodificadores por decisão suave são derivados daqueles desenvolvidos para decisão abrupta e divididos em dois tipos básicos: algoritmo de Omura e Decodificação parcial de síndrome.

3.6.1. CONCEITOS ÚTEIS

3.6.1.1. DEFINIÇÃO DE CONJUNTO DE INFORMAÇÃO

CONJUNTO DE INFORMAÇÃO para um código de grupo (n,k), é qualquer conjunto de k posições de uma palavra-código que podem ser (cada uma) especificadas de maneira independente em relação aos outros $(k-1)$ dígitos do conjunto. O restante dos dígitos $(n-k)$ são chamados de conjunto de paridade.

Portanto, dada a matriz [G] geradora do código, $\mathbf{c}= (c_1,c_2,...,c_n)$, $\mathbf{a}= (a_1,a_2,...,a_k)$ e $\mathbf{c}= \mathbf{a}[\mathbf{G}]$, um conjunto de informação é qualquer conjunto de k posições que podem ser obtidas através de colunas de [G] que sejam L.I.'s.

Exemplos:

a) Em um código sistemático, $[\mathbf{G}] = [\mathbf{I_k} \mid \mathbf{P}]$, as primeiras k posições da palavra-código formam um conjunto de informação;

b) Qualquer outro conjunto de posições do código sistemático pode formar um outro conjunto de informação desde que seja possível fazer a coluna correspondente (à posição) na matriz geradora [G] ter peso unitário (i.e., apenas 1 elemento "1" na coluna e o resto dos elementos valem "0"), através de operações elementares entre linhas da matriz [G].

3.6.1.2. ALGORITMO GERAL DA DECODIFICAÇÃO POR CONJUNTO DE INFORMAÇÃO

O fato das posições no conjunto de informação serem independentes faz com que este conjunto especifique completamente a palavra-código e com que os outros dígitos das $n-k$ posições possam ser reconstruídos a partir deles. O algoritmo então resume-se a:

1) Selecionar vários conjuntos de informação diferentes de acordo com alguma regra;

2) Construir uma palavra-código para cada conjunto assumindo que os dígitos das posições especificadas pelo conjunto de informação estão corretos.

3) Comparar cada palavra-código hipotética obtida em 2 com a seqüência recebida e selecionar a palavra-código mais próxima.

Existem dois métodos gerais para implementar o passo 1 : gerar conjuntos de informação a partir de conjuntos de posições previamente determinados ou então gerá-los a partir de procura aleatória.

Para implementar o passo 2, dois métodos são usados: ou se usa [G] para recodificar ou se usa [H] para calcular uma síndrome que, no caso de se assumir que as k posições do conjunto de informação são corretas, é o próprio padrão de erros nas $(n-k)$ posições que formam o conjunto de paridade.

Este algoritmo tanto pode ser usado em decisão abrupta como suave.

3.6.2. ALGORITMO DE OMURA

De uma maneira geral, os algoritmos baseados nos conjuntos de informação aplicados ao caso de decisão suave utilizam o algoritmo geral de 3.6.1.2 escolhendo como conjuntos de informação os k dígitos mais confiáveis da seqüência de n dígitos recebida e um número arbitrário j de dígitos menos confiáveis para o conjunto de paridade. Vários conjuntos de informação são então escolhidos entre o restante de $(n-j)$ posições para cobrir todas as combinações de até v erros.

O algoritmo de Omura consiste no seguinte:

1) Receber a seqüência demodulada **r**, escolher um conjunto arbitrário de paridade {P} e obter uma matriz [H] inicial na forma canônica reduzida e modificada através de operações elementares para contemplar o conjunto de paridade associado $\{k, k+1,...,n-1\}$ (como no algoritmo de Greenberger, visto na seção 3.4).

2) Calcular a síndrome **s** e achar um vetor-erro **e** inicial através de $\mathbf{s} = \mathbf{e}.[\mathbf{H}]^{\mathbf{T}}$. Para [H] em forma canônica reduzida , se $\mathbf{s}=(s_1, s_2,...,s_{n-k}) \Rightarrow \mathbf{e}=(0, 0,...0, s_1, s_2, ..., s_{n-k})$.

3) Achar um elemento no conjunto de informação tal que se o trocarmos com um elemento de {P}, obtendo {P'}, um novo conjunto de paridade, e gerarmos uma nova síndrome **s'** com {P'}, tenhamos **s'** com o mínimo peso possível.

4) Trocar realmente {P} por {P'} definido em 3, calcular nova matriz [H'] e calcular um novo vetor erro **e'** tal que **e'** seja igual à justaposição de um vetor **0** e **s'** no novo conjunto de paridade {P'}.

5) Se o peso de **s'** ou **e'** é menor que a capacidade corretora t ou igual a t , a procura termina. Caso contrário, volte em 2.

No caso de decisão abrupta $t = \mathrm{INT}\left[\frac{d-1}{2}\right]$ e qualquer {P} inicial serve.

No caso de decisão suave o algoritmo funciona para t valendo até $t=(d-1)$. O peso no caso de decisão suave deixa de ser o peso de Hamming e passa a ser um peso baseado na confiabilidade dos dígitos recebidos. Esta confiabilidade pode ser obtida através da quantização dos dígitos recebidos em Q > 2 níveis de forma a associar uma amplitude ao dígito, além da polaridade.

Para reduzir o número de vetores-erro testados (iterações) podemos ordenar os dígitos de recepção de acordo com a confiabilidade deles e escolher como conjunto inicial de paridade {P} sobre os $(n-k)$ dígitos menos confiáveis. Como t pode ser $t=(d-1)$, pode-se mostrar que o algoritmo é de máxima verossimilhança. Entretanto exige muitas iterações quando comparado com o algoritmo de Chase. Clark & Cain [14, cap. 4, seção 4.5.2] citam que o JPL (Jet Propulsion Laboratory da Califórnia) implementou o algoritmo de Omura em decisão suave para um código (128,64) que testava 10.000 padrões de erro aproximadamente (todos os de um e 2 erros e 25% dos padrões mais prováveis de 3 erros). O desempenho ficou 0,5 dB pior que o do decodificador de máxima verossimilhança no ponto de 2 dB de relação sinal-ruído. Se cada padrão de erro puder ser testado em 50 ns, a decodificação levaria 500 μs, o que equivaleria a uma taxa máxima de entrada de 256 kbit/s ou uma taxa de dados à saída de 128 kbit/s máxima. Conforme veremos, o desempenho de um decodificador de Chase também é 0,5 dB pior para $P_E=10^{-3}$ e só 0,3 dB pior para $P_E= 10^{-5}$ em relação ao decodificador por correlação, sendo necessárias muito menos seqüências de teste (16 apenas), sendo portanto mais simples (menos hardware) e mais veloz que este decodificador do JPL.

3.6.3. DECODIFICAÇÃO PARCIAL DE SÍNDROME

Este algoritmo visa diminuir o número de padrões de teste exigido para decodificadores baseados em conjunto de informação, através do artifício abaixo (caso seja possível):

1) Supor que, para um dado conjunto de paridade {P} escolhido e um conjunto de informação {S} associado, a matriz $[\mathbf{H}]^{\mathbf{T}}$ na forma canônica reduzida e reordenada tem uma sub-matriz identidade [I] na parte direita interior mais extrema. Então $[\mathbf{H}]^{\mathbf{T}}$ pode ser particionada em:

$$[\mathbf{H}]^T = \left[\frac{\mathbf{P1}\,|\,\mathbf{0}}{\mathbf{P2}\,|\,\mathbf{I}}\right] \tag{3.23}$$

com **P1** sendo r x$(k+r)$ com $1 \leq r \leq n$ arbitrário (depende do conjunto {**P**} escolhido).

2) Assim, a síndrome e o vetor-erro podem ser particionados em:

$$\mathbf{s} = \mathbf{e}[\mathbf{H}]^T$$
$$[\mathbf{s}_1|\mathbf{s}_2] = [\mathbf{e}_1|\mathbf{e}_2]\left[\frac{\mathbf{P1}\,|\,\mathbf{0}}{\mathbf{P2}\,|\,\mathbf{I}}\right] \tag{3.24}$$

onde $\mathbf{e_1}$ tem $(k+r)$ componentes e $\mathbf{s_1}$ tem r componentes.

3) a componente $\mathbf{s_1}$, a síndrome parcial, vale:

$$\mathbf{s}_1 = \mathbf{e}_1\mathbf{P1} \tag{3.25}$$
$$\mathbf{e}_2 = \mathbf{e}_1\mathbf{P2} + \mathbf{s}_2 \tag{3.26}$$

4) Como $\mathbf{e_1}$ corresponde ao padrão de erros **e** nas primeiras $k+r$ posições e $\mathbf{e_2}$ ao padrão **e** nas $n-k-r$ posições restantes, podemos usar (3.25) para achar vários padrões de erro a serem testados que tenham peso pequeno e sejam definidos sobre o conjunto de informação e então usar (3.26) para determinar o restante dos erros sobre o conjunto de paridade. Com isto podemos restringir o número de erros em $\mathbf{e_1}$ para algum número pequeno W e reduzir o número de cálculos necessários.

Não encontramos na literatura consultada nenhuma implementação. Clark e Cain [14] suspeitam que o tempo de decodificação seja maior que o do algoritmo de Omura, porque o número de conjuntos {**P**} aumentaria em relação ao de Omura, para um mesmo desempenho.

CAPÍTULO 4

ALGORITMOS DE CHASE E HACKETT

4.1. INTRODUÇÃO

No capítulo 2 estudamos qual o desempenho de decodificadores por decisão suave quando comparado com o de decisão abrupta para códigos de bloco lineares. No capítulo 3 descrevemos os algoritmos mais usados de decisão suave para códigos de bloco (APP, HR, Greenberger, Weldon, Omura e decodificação parcial de síndrome) exceto o algoritmo de Chase e suas variantes, que serão vistos aqui.

O fato de se descrever o algoritmo de Chase em um capítulo separado se deve a 2 fatores:

a) ele é um algoritmo de decisão de máxima verossimilhança (MV) em sua forma completa (algoritmo 1 a ser descrito adiante) e os outros são meras aproximações;

b) É simples e facilmente implementável;

Por isto merece ser estudado detalhadamente, dada a sua importância teórica e prática.

Neste capítulo serão descritos o algoritmo de Chase (nas 3 versões propostas por ele), o algoritmo de Hackett e o algoritmo de Chase generalizado. Provaremos também que o algoritmo de Chase é equivalente a um decodificador de máxima verossimilhança e, por isto, é ótimo.

Para este capítulo, as referências são ([14], [13], [16], [43], [31]).

4.2. IDÉIA BÁSICA DO ALGORITMO DE CHASE

Vamos imaginar o sistema Q-ário mostrado na Figura 2.1a e b.

A idéia básica do algoritmo de Chase é dada abaixo:

1) Supor um quantizador de J bits colocado entre a saída do demodulador e a entrada do decodificador seja tal que, para cada J bits que representam uma amostra de um dígito Y_i da seqüência de n dígitos **Y** recebida, tenhamos:

a) 1 bit (o mais significativo, MSB) → representa a polaridade (0 ou 1) do dígito Y_i

b) J-1 bits → representam um vetor de confiabilidade α_i de (J-1) componentes associados ao dígito Y_i

c) $Q = 2^J$ níveis

2) Introduzir *A PRIORI* inversões de polaridade em um número F arbitrário de dígitos que sejam os menos confiáveis da seqüência **Y**, a partir da geração de várias seqüências de teste **T** somadas ao vetor recebido **Y**, gerando palavras $\mathbf{Y'} = \mathbf{Y} \oplus \mathbf{T}$. O número de seqüências de teste a serem geradas é 2^F de forma a testar todas as combinações possíveis de inversões a priori nos F dígitos menos confiáveis.

OBS.: $\oplus$ indica soma módulo-2 (ou exclusivo).

3) Usando um decodificador *binário* (decisão abrupta), decodificar *BINARIAMENTE* a seqüência $\mathbf{Y'}=\mathbf{Y}\oplus\mathbf{T}$, para achar padrões de erro com erros nos dígitos mais confiáveis que os "F" dígitos menos confiáveis da seqüência Y, achando um vetor-erro **E**.

4) Somar **E** e **T** para obter $\mathbf{Z_T}=\mathbf{E}\oplus\mathbf{T}$, obtendo um padrão de erros que é a soma dos erros decodificados binariamente com os erros que já poderiam ter sido corrigidos previamente, quando da obtenção de $\mathbf{Y'}$.

5) Calcular o peso analógico associado a $\mathbf{Z_T}$ (este peso é a soma das confiabilidades α_i associadas à posição do i-ésimo dígito c_i).

6) Escolher o padrão $\mathbf{Z_T}$ correspondente ao menor peso analógico.

7) Decodificar $\mathbf{c} = \mathbf{X} = \mathbf{Y}\oplus\mathbf{Z_T}$

Para esclarecer o procedimento acima, vale fazermos alguns comentários:

a) No passo 2 o que se faz invertendo os dígitos de menor confiabilidade é equivalente a supor, antes mesmo de decodificar binariamente a seqüência recebida, que estes dígitos estão errados, o que é uma hipótese bastante razoável já que é pouco provável que o ruído que provoca o erro tenha uma amplitude tão alta que chegasse a valer o dobro dos níveis "analógicos" dos níveis lógicos 0 e 1, (levando, por exemplo, o nível transmitido com uma polaridade a chegar na recepção com a polaridade invertida mas com o mesmo nível).

b) Como supomos que os dígitos menos confiáveis estão errados, devemos em 3) aliviar a "carga de trabalho" do decodificador binário, que é capaz de corrigir só $t_1=\mathrm{INT}[(d-1)/2]$ erros, para que ele possa achar outros erros não tão prováveis. Com isto, o passo 2 retira eventuais "sobrecargas de erros" do decodificador binário.

c) O passo 4 simplesmente nos diz que o padrão de erros que ocorreram na seqüência é o dado pelo padrão de erros saindo do decodificador binário em conjunto com o padrão de erros testado sobre os dígitos menos confiáveis. É óbvio que, se invertemos algum dígito correto antes de decodificá-lo binariamente, na saída do decodificador binário o padrão de erros terá um dígito "1" nesta posição e então $\mathbf{Z_{Ti}}=\mathbf{T_i}\oplus\mathbf{E_i}=1\oplus1=0$, que é dígito do padrão de erros final, indicará que não houve erro naquela posição.

Conforme fica claro no procedimento, o algoritmo de Chase aumenta a capacidade corretora do sistema de codificação-decodificação de forma a podermos corrigir até t erros onde:

$$t = t_1 + F = \mathrm{INT}\left[\frac{d-1}{2}\right] + F \tag{4.1}$$

Qualquer decodificador por decisão suave pode sempre corrigir qualquer erro cujo peso analógico é menor que { [(Q-1) . d] / 2 }, para um código C de distância mínima d e quantizador de Q níveis.

Então, como a seqüência de polaridades **Y** recebida pode conter *até* (d–1) erros binários para não ser confundida com outra palavra-código, a pergunta mais natural é:

"Como escolher F de forma a maximizar a capacidade corretora do decodificador, t, dadas as 2 restrições acima? Como escolher F de forma que t= (d–1), valor máximo que t poderá assumir?"

Esta questão será respondida mais adiante. Primeiro devemos enunciar o algoritmo de Chase de uma maneira mais formal.

4.3. FORMALIZAÇÃO DO ALGORITMO DE CHASE

Seja o sistema de comunicação digital mostrado na Figura 2.1a.

Cada seqüência de k bits de informação **a**= (a_1,...a_k) é codificada em uma seqüência de n dígitos binários **c**= (c_1, c_2...c_n) que por sua vez é mapeada nos símbolos **X**= (X_1 X_2...X_n) (no caso de 2-PSK, X_i = c_i i=1,...n) transmitidos pelo canal e corrompidos por ruído de forma que à saída do demodulador temos uma seqüência **r** de n amostras do sinal recebido.

Seja agora essa seqüência **r** mapeada em uma seqüência **Y** de n *dígitos binários* (obtidos por um limitador, "hard-limiter"), **Y**= (y_1 y_2...y_n) e, além disso, em uma seqüência α de n *números positivos* denotada por $\alpha = \alpha_1\ \alpha_2...\alpha_n$ onde α_i está associado a y_i e r_i. Como *hipótese restritiva* suponhamos que:

$$\alpha_i > \alpha_j \Leftrightarrow \mathrm{Prob}[y_i \text{ estar correto}] > \mathrm{Prob}[y_j \text{ estar correto}] \tag{4.2}$$

Então, α_i é uma *INFORMAÇÃO MEDIDA NO CANAL* e dá uma medida relativa da confiabilidade dos dígitos binários y_i recebidos, $1 \le i \le n$.

A obtenção dos y_i e α_i é facilmente realizada por um quantizador de Q-níveis (conversor A/D de J bits, J arbitrário) onde o bit mais significativo representa a polaridade do sinal e, portanto, y_i e os J-1 bits menos significativos representam os α_i, onde $Q=2^J$.

Vamos agora definir alguns conceitos úteis:

1) Um *decodificador binário* é aquele que decodifica **Y** como sendo uma palavra-código $\mathbf{X_m}$ se $\mathbf{X_m}$ é aquela que difere no menor número de posições em relação a **Y**, entre todas as palavras do código **C**, *desde que* $W_H(\mathbf{X_m}+\mathbf{Y}) \leq \text{INT}\,[(d-1)/2]$ onde W_H é o peso de Hamming. Caso contrário nenhuma palavra-código é encontrada.

2) Um *decodificador binário completo* é aquele capaz de achar $\mathbf{X_m} \in \mathbf{C}$ tal que:

$$W_H(Y \oplus X_m) = \underset{1 \leq s \leq 2^k}{\text{MIN}} W_H(Y \oplus X_s)$$

Este decodificador *sempre* fornece à saída uma palavra-código, mesmo que errada (quando $W_H(Y \oplus X_m) \geq \text{INT}[(d-1)/2]$).

3) *Peso analógico* de uma seqüência $\mathbf{V}_j$ é definido por:

$$W_H(\mathbf{V}_j) = \sum_{i=1}^{n} \alpha_i V_{ji} \tag{4.3}$$

Chase [13] define um *DECODIFICADOR POR MEDIDA DE CANAL COMPLETO* como sendo aquele que decodifica **Y**, seqüência recebida, como sendo $\mathbf{X_m} \in \mathbf{C}$ desde que:

$$W_\alpha(Y \oplus X_m) = \underset{1 \leq s \leq 2^k}{\text{MIN}} W_\alpha(Y \oplus X_s) \tag{4.4}$$

Neste caso, queremos achar o padrão de erros $\mathbf{Z_m} = \mathbf{Y} \oplus \mathbf{X_m}$ de mínimo peso analógico.

Mais adiante mostraremos que um decodificador que satisfaça (4.4) é na verdade um receptor de máxima verossimilhança para o canal AWGN.

Queremos agora achar um algoritmo de decodificação com as seguintes restrições: satisfaça (4.4) (*restrição 1*) e que, por hipótese, use um decodificador binário para facilitar a implementação (*restrição 2*). Para tanto, Chase [13] propôs o que se segue:

O objetivo é usar as medidas de canal α e um decodificador binário de forma que este ajude a obter um pequeno conjunto de padrões de erro possíveis (ao invés de um único padrão de erros dado pelo decodificador binário).

Para tanto, perturbamos a seqüência recebida **Y** com um padrão de teste **T** de n dígitos que valem **0** nas posições que não queremos inverter e **1** naquelas onde

queremos inverter. A perturbação é conseguida através de soma módulo 2, obtendo uma nova seqüência, $\mathbf{Y}'$:

$$\mathbf{Y}' = \mathbf{Y} \oplus \mathbf{T} \tag{4.5}$$

Em seguida decodificamos $\mathbf{Y}'$ com um decodificador binário e obtemos um padrão de erros $\mathbf{Z}'$ relativo a $\mathbf{Y}'$. O padrão de erros real, relativo a $\mathbf{Y}$ é dado por $\mathbf{Z}_T$ e vale:

$$\mathbf{Z}_T = \mathbf{Z}' \oplus \mathbf{T} \tag{4.6}$$

que pode tanto ser como não ser diferente de $\mathbf{Z}'$, dependendo de $\mathbf{Y}'$ estar ou não mais próximo de outra palavra-código que não aquela que seria obtida caso fizéssemos só a decodificação binária de $\mathbf{Y}$. Assumindo um código $\mathbf{C}$ de bloco, linear com distância mínima d e empacotamento esférico das 2^k palavras-código dentro do espaço de 2^n seqüências de recepção possíveis, isto seria equivalente a perturbar $\mathbf{Y}$ de forma que $\mathbf{Y}'$ caia fora da esfera de raio INT[$(d-1)/2$] que envolve $\mathbf{Y}$ e está ilustrado na Figura 4.1.

O *algoritmo de Chase* é definido de forma a usar as medidas α para fazer com que a seqüência recebida $\mathbf{Y}$ seja sempre perturbada ao redor de uma esfera de raio $(d-1)$.

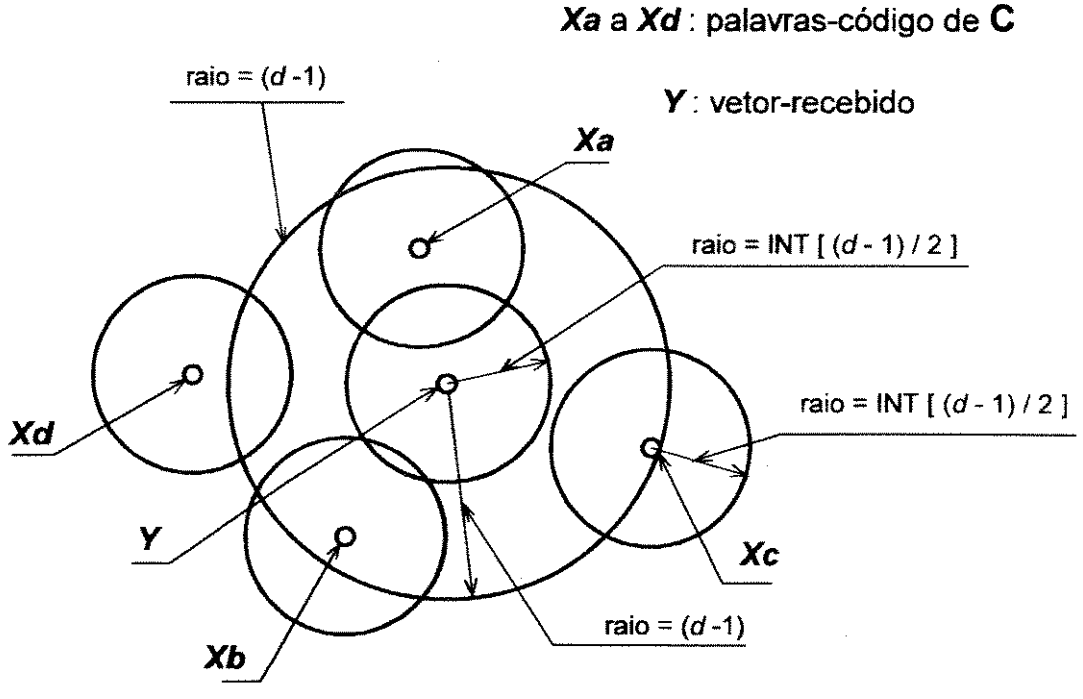


FIGURA 4.1. SITUAÇÃO NO ESPAÇO DE PALAVRAS

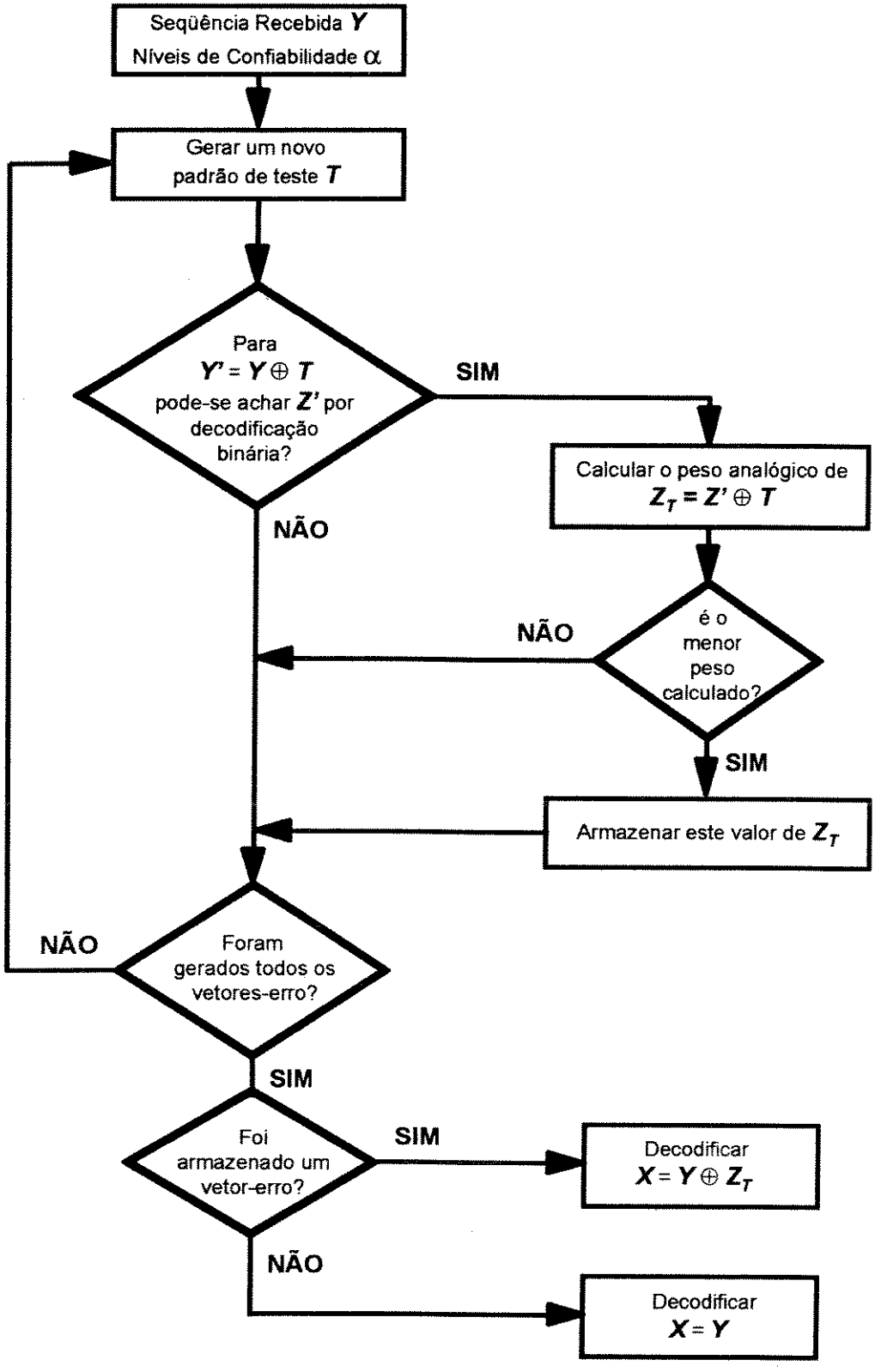


FIGURA 4.2. FLUXOGRAMA DO ALGORITMO DE CHASE

O algoritmo de Chase obedece então aos passos 1 a 7 vistos no item 4.2 e pode ser representado por um fluxograma, dado na Figura 4.2.

Devemos observar que o algoritmo de Chase não exige (como restrição) um decodificador binário completo mas apenas um decodificador binário, capaz de achar uma palavra-código apenas quando:

$$W_H(\mathbf{Z}') \leq \mathrm{INT}\left[(d-1)/2\right] \tag{4.7}$$

Assim, é perfeitamente possível que não se ache nenhum padrão de erro válido e consideramos como estimativa de $\mathbf{X_m}$ o próprio $\mathbf{Y}$ recebido, apesar de certamente estar em erro.

Vamos agora responder às perguntas feitas no final do item 4.2 ("como escolher F...."), descrevendo as 3 formas propostas por Chase para escolher as seqüências de teste perturbadoras da seqüência recebida, mas que usam, todas as 3 formas, o mesmo algoritmo da Figura 4.2.

4.4. ALGORITMO 1 DE CHASE

O algoritmo 1 de Chase considera como conjunto de padrões de teste aquele formado por *todos* os padrões $\mathbf{Tj}$ que levem a um $\mathbf{Z}_{\mathrm{Tj}}=\mathbf{E} \oplus \mathbf{Tj}$ de peso menor ou igual a $(d-1)$, para que as palavras modificadas $\mathbf{Yj}'$ a serem testadas distem de até $(d-1)$ da palavra recebida $\mathbf{Y}$. Assim:

$$\{\mathbf{T_j}\} = \left\{ \forall \mathbf{T_j} \Big/ W_H\left(\mathbf{Z}_{\mathbf{T_j}}\right) = \sum_{i=1}^{n} \mathbf{Z}_{\mathbf{T_{ji}}} \leq (d-1) \right\} \tag{4.8}$$

Como a decodificação decide com base no peso analógico e não no de Hamming, pode existir um vetor erro escolhido cujo peso de Hamming seja maior que INT [$(d$-1)/2], o que aumenta a capacidade corretora do sistema de codificação.

Um conjunto {Tj}, proposto por Chase, que respeita (4.8) mas é superdimensionado com certeza (i.e. é suficiente mas não necessário) é aquele que gera todos os $\mathbf{Z}_{\mathrm{Tj}}$ com peso binário menor que d. Tal conjunto seria o que contivesse todos os padrões de teste $\mathbf{Tj}$ de peso de Hamming igual a INT $[d/2]$ e o de peso igual a zero, pois o decodificador binário corrige até INT $[(d-1)/2]$ erros e por isso é capaz de gerar todos os padrões de erro $\mathbf{E}$ de peso menor ou igual a INT $[(d-1)/2]$ que combinados com $\mathbf{Tj}$, podem levar a qualquer $\mathbf{Z}_{\mathrm{Tj}}$ de peso até $(d-1)$. Ou seja:

$$\{\mathbf{T_j}\} = \left\{ \mathbf{T_j} \Big/ \mathbf{T_j} = \mathbf{0} \vee W_H\left(\mathbf{T_j}\right) = \sum_{i=1}^{n} \mathbf{T_{ji}} = \mathrm{INT}[d/2] \right\} \tag{4.9}$$

Devemos observar que o número de padrões de teste diferentes entre si será elevado e vale:

$$N_{T_j} = \binom{n}{\text{INT}[d/2]} = \frac{n!}{(n-\text{INT}[d/2])!(\text{INT}[d/2])!} \tag{4.10}$$

Portanto, o algoritmo 1 só é aplicável a códigos de distância mínima d pequena ou para prover a simulações em computador um limitante inferior para a probabilidade de erro de um decodificador por medidas no canal completo.

4.5. ALGORITMO 2 DE CHASE

No algoritmo 2 de Chase, considera-se como conjunto de padrões de erro possíveis somente os padrões $\mathbf{Z}_{Tj}$ com um número de erros menor ou igual a INT $[(d-1)/2]$ localizados em posições diferentes das que são dadas pelas INT $[d/2]$ posições de menor confiabilidade no vetor recebido **Y**. Por isto, todos os padrões $\mathbf{Z}_{Tj}$ testados têm no máximo $(d-1)$ erros mas o conjunto $\{\mathbf{Z}_{Tj}\}$ não contém mais *todos* os padrões de erro $\mathbf{Z}_{Tj}$ *possíveis* para serem testados, já que INT $[d/2]$ posições em erro possíveis estão fixadas dado um vetor **Y** recebido junto com seu vetor-confiabilidade $\alpha = (\alpha_1, \alpha_2, \alpha_3, \ldots \alpha_n)$.

O conjunto de padrões de teste, proposto por Chase [13], que gera todos os padrões de erro $\mathbf{Z}_{Tj}$ descritos acima é dado por:

$$\{\mathbf{T_j}\} = \begin{Bmatrix} \mathbf{T_j} \text{ tem qualquer combinação de dígitos 1 nas} \\ \text{INT}[\text{d}/2] \text{ posições de menor nível de confiabilidade} \end{Bmatrix} \tag{4.11}$$

Neste caso, existem $2^{\text{INT}[d/2]}$ combinações de valores dos dígitos cujas posições são as INT $[d/2]$ de menor confiabilidade e, portanto, existem $2^{\text{INT}[d/2]}$ padrões de teste **Tj** possíveis que valerão ou **"0"** ou **"1"** combinados de qualquer maneira nas INT $[d/2]$ posições de menor confiabilidade e **"0"** nas outras $[n-\text{INT}[d/2]]$ posições restantes.

Se o algoritmo 2 for implementado em software, $2^{\text{INT}[d/2]}$ é um número máximo de padrões **Tj**. Geralmente poderemos testar muito menos padrões porque, se o padrão de erros inicial $\mathbf{Z}'$, obtido pelo decodificador binário, tiver peso de Hamming unitário ($W_H(\mathbf{E})=1$), então todos os padrões de teste **Tj** com $W_H(\mathbf{Tj}) \leq$ INT $[(d-3)/2]$ levarão a $\mathbf{Z}_T=\mathbf{Z}'$. Por hardware obviamente devemos testar sempre as $2^{\text{INT}[d/2]}$ possibilidades para simplificar os circuitos.

Devemos observar que Chase mostra que o algoritmo 2 não tem um desempenho tão bom como o algoritmo 1 já que não se testa todas as posições de erro possíveis mas apenas as INT $[d/2]$ de menor confiabilidade. Mas se por um lado

perdemos em desempenho, ganhamos em número menor de padrões de teste a serem gerados e tempo de decodificação porque o algoritmo 1 gera um número astronômico de padrões de teste Tj (N_{Tj}, eq. 4.10), produzindo o padrão de teste desejado várias vezes (exatamente (INT [d/2]) vezes) e gerando padrões altamente improváveis um sem número de vezes.

Por exemplo, para o código de Golay estendido, CGE (24,12), como d=8, temos que o número de padrões de teste no algoritmo 2 é de:

$$N_{T_j}(2) = 2^{\mathrm{INT}[8/2]} = 16 \text{ padrões de teste}$$

Para o algoritmo 1, teríamos:

$$N_{T_j}(1) = \frac{24!}{20!\,4!} = 23 \times 22 \times 21 = 10.626 \text{ padrões de teste}$$

Quanto ao desempenho, Chase [13] e Pessoa & Arantes [31] fizeram simulações para os algoritmos que serão analisadas minuciosamente no capítulo 5, item 5.3.

Por enquanto, basta dizermos que a constatação principal é que os algoritmos 1 e 2 têm o mesmo desempenho para este código, em canal AWGN, na faixa de E_b/No de 0 a 6 dB.

O algoritmo 2 é de grande interesse prático, assim como o algoritmo 3, conforme será visto na próxima seção.

4.6. ALGORITMO 3 DE CHASE

É quase igual ao algoritmo 2 só que aqui o número de padrões de teste é bem mais reduzido, valendo apenas INT [(d/2)+1] ao invés de $2^{\mathrm{INT}(d/2)}$.

Neste caso, o conjunto {Tj} de padrões de teste é formado achando as (d–1) posições de menor confiabilidade do vetor Y recebido e fazendo com que os padrões Tj tenham dígitos **"1"** nas i posições de menor confiabilidade e sejam **"0"** nas (n–i) posições restantes, onde i valerá:

a) i= 1,3,5...d–1 e i=0 para d=d par;

b) i= 0,2,4...d–1 para d=d ímpar

É o algoritmo de menor conjunto {Tj} a se testar, sendo altamente desejável para a implementação de decodificadores para códigos grande distância mínima d, já que aqui o número de Tj's cresce linearmente com d, enquanto nos outros cresce combinatoriamente (algoritmo 1) e exponencialmente (algoritmo 2).

Entretanto, Chase mostra, por simulação, que o desempenho é inferior em relação aos outros 2 algoritmos (como veremos na seção 5.3).

4.7. OPTIMALIDADE DO ALGORITMO DE CHASE

Mostramos na seção 2.2.1 que um receptor é ótimo no sentido de que tem mínima probabilidade de erro caso ele use como regra de decisão um processo MAP (máximo a posteriori) que escolhe como saída a forma de onda x tal que:

$$x = x_i \Leftrightarrow \underset{x_j}{\text{MAX}}\ \text{Prob}\left[x_j \middle| y(t)\right] = \text{Prob}\left[x_i \middle| y(t)\right] \tag{4.12}$$

Mostraremos aqui o que Chase prova para o decodificador baseado em seu algoritmo: que ele é equivalente a um receptor MAP para um canal AWGN.

Vamos inicialmente supor o receptor de sinal da Figura 4.3, recebendo um sinal binário antipodal (i.e., modulação 2-PSK) e um canal de ruído branco aditivo e gaussiano (AWGN).

A onda y(t) recebida entra em n filtros de decisão. A onda transmitida foi $\mathbf{X}=\mathbf{c}=(x_1\ x_2\ x_3 \ldots x_n)$. A saída dos filtros é uma estatística de decisão ponderada e a regra de decisão, sobre os dígitos x_i é, por hipótese:

se $v_i \geq 0 \quad \Rightarrow x_i=1=y_i$

se $v_i < 0 \quad \Rightarrow x_i=0=y_i$

onde v_i é uma estatística de decisão (sem ponderação).

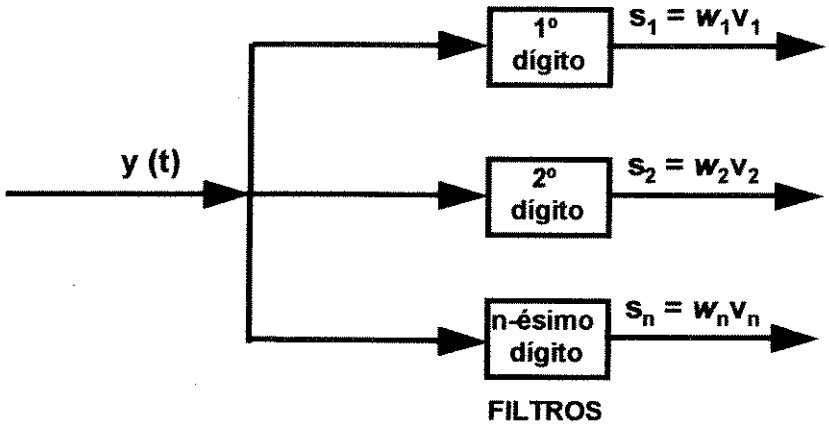


FIGURA 4.3. MODELO DO RECEPTOR DE CHASE

A estatística ponderada, que é na verdade a saída dos filtros, é a estatística de decisão multiplicada por um fator-peso W_i positivo. A medida de canal ou valor de confiabilidade α_i do dígito Y_i será denotada e definida por:

$$\begin{aligned} \alpha_i &= |W_i\, v_i| \\ s_i &= W_i\, v_i \end{aligned} \tag{4.13}$$

Para o canal AWGN vamos assumir que os N filtros de decisão são casados com as formas de onda de forma que a estimativa v_i do dígito x_i, dada pelo i-ésimo filtro casado, será (v_i aqui assumida como tensão, não energia como na seção 2.2.1).

$$\begin{cases} v_i = \sqrt{E} + N_i & \text{se } x_i = 0 \\ v_i = -\left(\sqrt{E} + N_i\right) & \text{se } x_i = 1 \\ W_i = 1 \end{cases} \tag{4.14}$$

OBS.: Reparar que aqui assumimos mapeamento de fases $0 \rightarrow 0^o$ e $1 \rightarrow 180^o$.

onde E é a energia do sinal que representa x_i e N_i é o ruído AWGN de densidade de potência (unilateral) No, média zero e variância N_i^2= No/2.

Então, como em 2.2.1, o receptor casado é MAP pois é de máxima verossimilhança (MV) e os símbolos x_i são equiprováveis (Prob (0)= Prob (1)= 1/2) e o ruído é AWGN.

Mas a regra de decisão de máxima verossimilhança pode ser modificada:

$$\underset{m}{\text{MAX}}\ \text{Prob}\left[s|x_m\right] \Leftrightarrow \underset{m}{\text{MAX}} \sum_{i=1}^{n} W_i V_i (-1)^{x_{mi}},\ 1 \le m \le 2k \tag{4.15}$$

onde $\mathbf{s}= (s_1, s_2, \ldots s_n)$ e $\mathbf{X_m}= (x_{m1}\, x_{m2} \ldots x_{mn})$

O somatório em (4.15) pode ser dividido em 2 parcelas: um somatório sobre todas as posições i dos dígitos onde $x_{mi} \neq Y_i$, que formariam um conjunto de posições $\mathbf{s_m}$, e um somatório sobre todas as posições i dos dígitos onde $x_{mi} = Y_i$. Então o membro à direita em (4.15) é equivalente a:

$$\underset{m}{\text{MAX}} \sum_{i=1}^{n} w_i v_i (-1)^{x_{mi}} = \underset{m}{\text{MAX}} \left\{ \sum_{i \notin S_m} |w_i v_i| - \sum_{i \in S_m} |w_i v_i| \right\} \tag{4.16}$$

levando (4.13) em (4.16) teremos:

$$\underset{m}{\text{MAX}}\left\{\sum_{i\notin S_m}|w_i v_i| - \sum_{i\in S_m}|w_i v_i|\right\} =$$

$$\underset{m}{\text{MAX}}\left\{\sum_{i\notin S_m}\alpha_i - \sum_{i\in S_m}\alpha_i + \sum_{i\in S_m}\alpha_i - \sum_{i\notin S_m}\alpha_i\right\} = \underset{m}{\text{MAX}}\left\{\sum_{i=1}^{n}\alpha_i - 2\sum_{i\in S_m}\alpha_i\right\} \tag{4.17}$$

Como o primeiro termo é igual para todas as palavras-código $\mathbf{X_m}\in\mathbf{C}$, teremos:

$$\underset{m}{\text{MAX}}\left\{\sum_{i=1}^{n}\alpha_i - 2\sum_{i\in S_m}\alpha_i\right\} \Leftrightarrow \underset{m}{\text{MIN}}\sum_{i\in S_m}\alpha_i$$

Mas:

$$\underset{m}{\text{MIN}}\sum_{i\in S_m}\alpha_i = \underset{m}{\text{MIN}}\sum_{i=1}^{n}\alpha_i Z_{mi} \overset{\Delta}{=} \underset{m}{\text{MIN}}\, W_\alpha(Z_m) \tag{4.18}$$

com $\mathbf{Z_m}= \mathbf{Y}\oplus\mathbf{X_m}$ e $\mathbf{S_m}$, voltamos a repetir, é o conjunto de posições onde os dígitos de $\mathbf{X_m}$ e $\mathbf{Y}$ são diferentes.

Então, a regra de decisão MV é equivalente à de mínimo peso analógico do padrão de erros para um canal AWGN.

C.Q.D.

Como um último comentário, o algoritmo de Chase praticamente dobra a capacidade corretora de qualquer código já que por decodificação binária (decisão abrupta), temos $t_{HARD} = \text{INT}\left[(d-1)/2\right]$ e por decodificação usando o algoritmo de Chase temos $t_{SOFT} = (d-1)$.

4.8. ALGORITMO DE HACKETT

Hackett [16] particularizou o algoritmo 2 de Chase para a decodificação de um código de Golay estendido (24,12) propondo uma variante de algoritmo que é mais eficiente quando implementada em software que a versão original de Chase porque diminui o número de padrões de teste à metade do normalmente exigido, graças a

certas propriedades que o código de Golay estendido (CGE) (24,12) apresenta e que podem estar presentes em outros códigos de importância prática.

O decodificador binário usado por ele foi o de técnica de "error-traping" ([33], [14], [27]) e os resultados foram:

a) implementado em um microprocessador, os resultados práticos coincidem com a teoria dentro de 0,1 dB;

b) simulado em um Cyber 176 usando FORTRAN, o tempo de decodificação médio, por bit de informação, foi de 175 μs +/-10%.

O algoritmo de Hackett é mostrado na Figura 4.4. e é interessante enfatizarmos que ele se aplica não só ao CGE (24,12), mas *a todo código de bloco* (n,k) que seja *estendido*, i.e., seja obtido de um código $(n-1,k)$ pela adição de um dígito de paridade total. É isto que permite com que o número de padrões de teste seja reduzido à metade, ou seja, passe de $2^{\text{INT}\,[d/2]}$ para $2^{\text{INT}\,[d/2-1]}$ pois a idéia é sempre complementar o dígito menos confiável a priori, toda vez que a paridade total recebida somada MOD 2 com a paridade regenerada no CGE, isto é:

$$r = \sum_{i=1}^{24} y_i = y_{24} \oplus \sum_{i=1}^{23} y_i$$

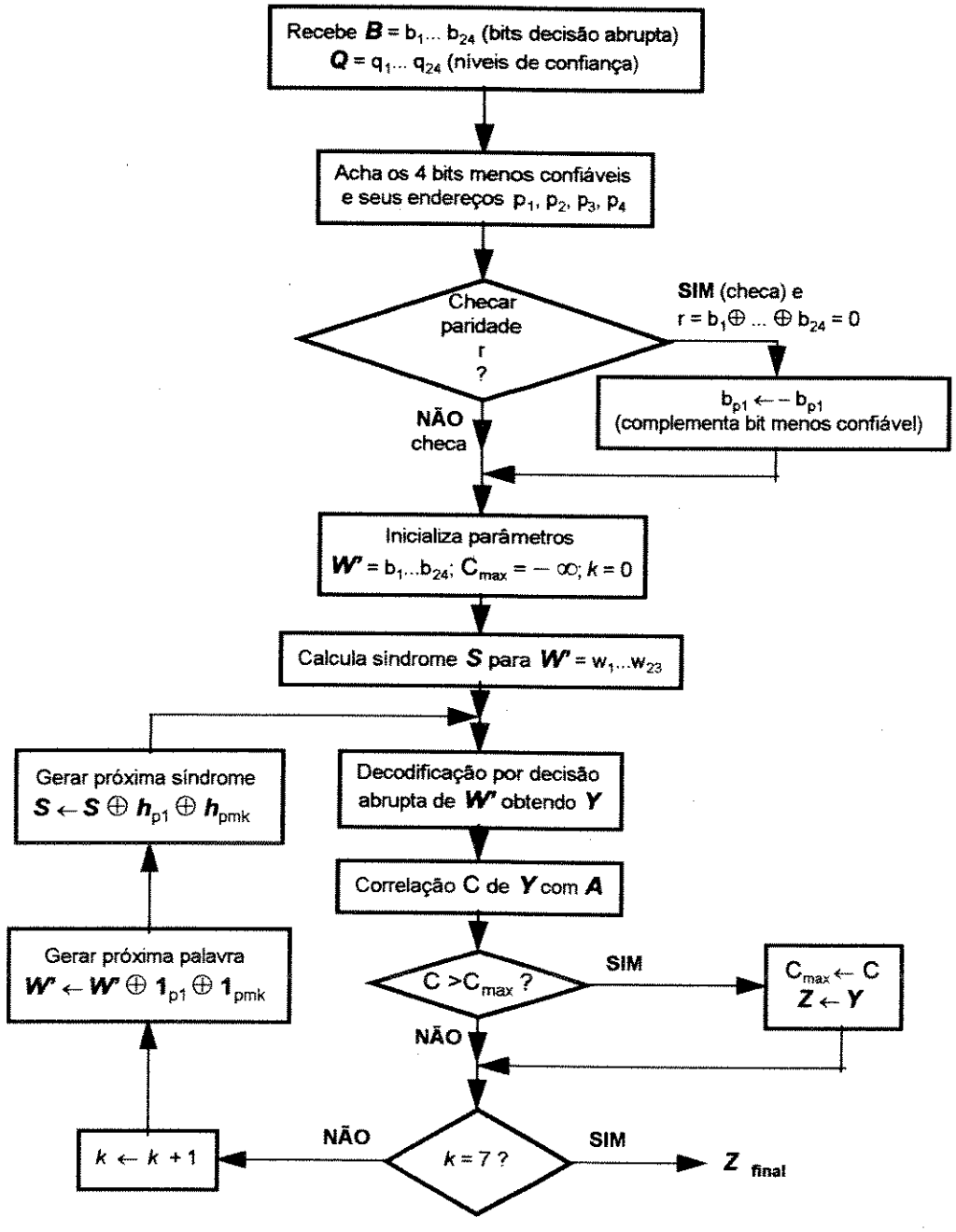
valer 0 (i.e., quando elas coincidirem). Esta inversão é feita a priori, toda vez que **r=0**, para termos sempre um número ímpar de erros na saída do decodificador binário já que uma decodificação binária de um número par de erros raramente produz uma palavra-código com melhor correlação com a entrada analógica do que aquela palavra produzida por decodificação binária de um número ímpar de erros segundo Hackett [16].

4.9. GENERALIZAÇÃO DO ALGORITMO DE CHASE

Tendolkar e Hartmann propuseram em 1984 uma generalização do algoritmo de Chase para a decodificação por decisão suave de códigos binários de bloco e lineares [43].

Esta generalização tem importância à medida em que permite uma redução de hardware, já que não exige um decodificador binário de capacidade corretora $t = \text{INT}\left[(d-1)/2\right]$ mas apenas um com capacidade corretora $t \le t^*$.

Para apresentar o algoritmo generalizado, vamos recorrer a algumas definições adotadas em [43]:



SIMBOLOGIA:
A - Vetor analógico que foi recebido antes da quantização
B - Vetor de polaridade (bits da decisão abrupta)
Q - Vetor dos níveis de confiabilidade
W - Vetor de teste
C_{max} - Variável que armazena o maior valor que foi achado para a correlação de **Y** com **A**
C - Correlação entre **Y** e **A**
Y - Palavra decodificada pelo decodificador binário
k - Variável auxiliar de contagem
$\mathbf{h}_{p_l}$ - Linha da matriz [H] endereçada por p_l
$\{M_k\}$- seqüência que causa todas as combinações de trocas em um número par de bits que possam ocorrer
Z - Variável que guarda o padrão de máxima correlação

FIGURA 4.4. FLUXOGRAMA DO ALGORITMO DE HACKETT PARA O CGE(24,12)

1) Distância Euclidiana: Sejam uma palavra-código $\mathbf{x_m} = (x_{m1}...x_{mn})$ mapeada em uma seqüência de símbolos ou sinais $S(\mathbf{x_m}) = S(x_{m1})....S(x_{mn})$ e uma seqüência recebida $\mathbf{r} = (r_1....r_n)$. Denota-se a distância euclidiana entre $S(\mathbf{x_m})$ e $\mathbf{r}$ por $d_E(S(\mathbf{X_m}), r)$; uma regra de decisão de máxima verossimilhança em canais AWGN com

sinalização antipodal (2-PSK) é decodificar **r** com $\mathbf{X_m}$ se d_E (S ($\mathbf{X_m}$), **r**)≤ d_E (S (**X**v),**r**) para todo m diferente v. Tal regra é equivalente a:

$$d_E\left(S(X_m),r\right) \le d_E\left(S(X_v),r\right) \Leftrightarrow \left[S(X_m)\cdot r\right] \ge \left[S(X_v)\cdot r\right]$$

onde [a.b] é o produto interno no espaço de sinais ([12] apêndice A; [25], capítulo 1). É equivalente ao cômputo do peso analógico, proposto por Chase.

2) $t^* = \text{INT}\,[(d-1)/2]$ e $t << t^*$ e **Y**= Vetor obtido pela decisão abrupta de **r**.

3) Um padrão de teste de n elementos é denotado por **Tj** e um conjunto de padrões de teste denotado por {**Tj**}.

Assim, o algoritmo de Chase generalizado pode ser enunciado nos seguintes passos:

a) Para cada **Tj** ∈ {**Tj**}, decodificar o vetor **Y**⊕**Tj** através de um decodificador binário de capacidade corretora $t << t^*$ e construir um conjunto **C'** que contenha todas as palavras-código X tais que, para algum **Tj** ∈ {**Tj**}, tenhamos d_H(**X**, **Y**⊕**Tj**)≤ t.

b) Se **C'** não for conjunto vazio, selecionar $\mathbf{X_m}$ como estimativa da palavra-código originalmente transmitida, onde $\mathbf{X_m}$ deve satisfazer às seguintes condições:

b1) $\mathbf{X_m} \in \mathbf{C'}$

b2) $d_E\left(S(\mathbf{X_m}),\mathbf{r}\right) \le d_E\left(S\left(\mathbf{X'_m}\right),\mathbf{r}\right), \forall \mathbf{X'_m} \ne \mathbf{X_m}, \in \mathbf{C'}$

Se **C'** for vazio, o algoritmo não consegue decodificar a seqüência **Y**.

O algoritmo de Chase generalizado pode ser completamente especificado pela capacidade corretora t do decodificador binário utilizado e pelo conjunto de padrões de teste {**Tj**}. Para um dado t, a complexidade e performance do algoritmo serão funções de {**Tj**}.

Tendolkar e Hartmann fornecem então os conjuntos {**Tj**} necessários para se implementar os Algoritmos 1, 2, e 3 de Chase na forma generalizada, que apresentamos sem as correspondentes demonstrações, mostradas em [43].

4.9.1. ALGORITMO 1 GENERALIZADO

Considera-se como conjunto de padrões de teste o seguinte:

a) CASO DE d ÍMPAR:

Definem-se $m = \text{INT}\left[d/(2t+1)\right]$ e conjuntos S_i cujos elementos são todos os vetores binários de n elementos e peso de Hamming $[t + (i\text{-}1)\,(2t+1)]$ com $1 \le i \le m$. Então teremos:

$$\{\mathbf{T_j}\} = \bigcup_{i=1}^{m} S_i$$

onde $\bigcup$ representa a união de conjuntos.

b) CASO DE d PAR:

Definem-se $m' = \mathrm{INT}[(d-1)/(2t+1)]$ e conjuntos S_i' de todos os vetores binários de n elementos e peso de Hamming $[1+t+(i-1)\ (2t+1)]$ com $1 \le i \le m'$ e {To}, o vetor nulo **0** de n elementos. Então teremos:

$$\{\mathbf{T_j}\} = \{\mathbf{T}_0\} \bigcup \left[\bigcup_{i=1}^{m'} S_i' \right]$$

Com esta relação de {Tj} asseguramos que todas as palavras-código, de distância de Hamming menor ou igual a $(d-1)$ do vetor **Y** recebido, serão identificadas e consideradas para teste.

4.9.2. ALGORITMO 2 GENERALIZADO

Considerando t tal que $0 \le t \le t^* = \mathrm{INT}\ [(d-1)/2]$, {Tj} será então o conjunto de vetores binários de comprimento n que tenham quaisquer número de dígitos **"1"** nas ω posições menos confiáveis, sendo os outros dígitos **"0"**, onde ω vale, em função de t:

$$\omega = \begin{cases} n-t & \text{, para } 0 \le t \le (d/4)-1 \\ t^* & \text{, para } d \text{ ímpar e } t = t^* \\ \mathrm{MIN}\left\{(n-t), \mathrm{INT}\left[(t+1) \times \dfrac{d}{4(t+1)-d}\right]\right\} & \text{, para os outros valores} \end{cases}$$

onde ω tem a propriedade de que, para toda palavra-código **X** tal que $d_{E^2}(\mathbf{S(X)}, \mathbf{r}) \le d \cdot E_s$, não mais que t erros estarão localizados fora das ω posições menos confiáveis.

Tendolkar e Hartmann ainda propõem um algoritmo 3 generalizado, com uma regra de construção bastante complicada para o conjunto de padrões {Tj} e com um

número de padrões muito pouco menor que o exigido para o algoritmo 2 generalizado, ao contrário do que ocorre entre os algoritmos 2 e 3 de Chase "puros"; além disso, como ocorre nos algoritmos "puros", a perfomance do algoritmo 3 generalizado é também pior que aquela do algoritmo 2 generalizado.

De qualquer forma, os algoritmos generalizados propostos por Tendolkar e Hartmann são, segundo eles mesmos [43], vantajosos quando a distância mínima do código, d, cresce de forma a tornar anti-econômica a utilização de um decodificador binário (decisão abrupta) de capacidade corretora de t^* erros. Dão como exemplo, o uso de seus algoritmos generalizados para o código de resíduo quadrático (47,23) com d igual a 12.

No caso de um código de Golay estendido (24,12), de d igual a 8, o uso do algoritmo 2 de Chase, "puro", leva a apenas 16 padrões de teste e o decodificador binário (decisão abrupta) a ser utilizado pode ser feito por uma EPROM de 4096 posições, que é suficientemente econômica atualmente. Por isto, em nosso caso, não usaremos os algoritmos generalizados.

Encerramos aqui a discussão teórica sobre o algoritmo de Chase. Vamos discutir aspectos de implementação do algoritmo no capítulo 5.

CAPÍTULO 5

A ARQUITETURA DO DECODIFICADOR DE CHASE

5.1. ARQUITETURA DO HARDWARE: CONSIDERAÇÕES GERAIS

Neste capítulo, estudamos algumas arquiteturas de hardware para a implementação do decodificador por decisão suave com algoritmo 2 de Chase para o CGE (24, 12) e comentamos a seguir algumas conclusões que nos levaram a optar pela arquitetura apresentada nos próximos itens.

Pelo item 1.3.2.7., sabemos que o tempo máximo para decodificação e decisão do melhor padrão de teste é de 24 dígitos a 1068 kbit/s:

$$T_{DECOD} = \frac{n}{velocidade} = \frac{24}{1068 \cdot 10^3} \equiv 24{,}47\ \mu s \qquad (5.1)$$

Supusemos aqui que os vetores $\mathbf{Y}$ e α_i, definidos como anteriormente (cap. 4), já haviam sido recebidos e a síndrome $\mathbf{S}$, referente ao sinal recebido $\mathbf{Y}$, foi calculada à medida que os dígitos entravam no decodificador, assim como suas posições menos confiáveis.

Então, temos disponível para processar cada seqüência de teste (são 16 Tj's no algoritmo 2 de Chase para o CGE (24, 12) com d=8) um tempo que vale:

$$t_{proc.\mathbf{T}_j} = \frac{T_{DECOD}}{16} \approx 1{,}4\ \mu s \qquad (5.2)$$

Isto significa que, em um processo serial, disporíamos de 1,4 μs para montar um padrão de teste Tj (especificando os endereços, até 4, das posições de Tj que teriam dígitos 1's), decodificar binariamente, calcular o peso analógico e compará-lo com o registro de peso analógico.

Tal processamento serial realizado em software com um microprocessador é inviável dentro de 1,4 μs com os atuais microprocessadores disponíveis no mercado atualmente. Os microprocessadores de 8 bits monolíticos mais rápidos (8051, p. ex.) têm ciclo de instrução de 1 μs o que é insuficiente (uma só manipulação de 24 bits levaria 3 μs, dobro dos 1,4 μs totais disponíveis). Com um microprocessador monolítico mais poderoso, como o 80386 de 32 bits, clock de 16 MHz e 3,64 MIPS (mega instruções/seg.), em 1,4 μs poderíamos fazer apenas 5 instruções manipulando 32 bits, o que ainda não é suficiente para uma rotina de decodificação.

Podemos, então, considerar processadores do tipo bit-slice, configuráveis a nível de CPU, bipolares e mais rápidos. Com eles, associados em uma arquitetura tipo RISC (reduced instruction set), poderíamos realizar o decodificador em software para 1,068 Mbit/s, usando o decodificador de Hackett (8 Tj's ao invés de 16). Todavia, este decodificador seria muito mais caro, com maior consumo de energia, de depuração (debug) muito mais difícil e com maiores propensões a problemas de EMI (electromagnetic interference) do que o hardware que será descrito por nós.

Outra arquitetura possível, valendo-se ainda do processamento serial, seria implementar o hardware específico com CIs MSI TTL-S, TTL-AS ou mesmo ECL. As desvantagens seriam as mesmas do processador bit-slice, usando muito mais chips que este.

Outra arquitetura que mereceu atenção foi a utilização de um DSP (digital signal processor) monolítico, que é um microprocessador dotado de arquitetura em pipeline, processador vetorial e muita velocidade na manipulação de strings de dados. Foi descartada após análise de mercado que indicava ser caro e de difícil fornecimento no Brasil. Contudo, a idéia ainda é promissora.

Por último, passamos a comentar a arquitetura final escolhida. A arquitetura escolhida utiliza como elementos de hardware chips baratos da série MSI TTL-LS ou da nova série 74HC/HCT (high-speed C-MOS, compatíveis pino a pino com TTL) e memórias C-MOS ou N-MOS (PROM, RAM e EPROM). Estes chips promovem uma queda brutal nos níveis de consumo, têm menor emissão de interferências e menor susceptibilidade a EMI, maiores margens de ruído, são muito fáceis de achar no mercado e têm baixo custo. Sua única desvantagem seria a velocidade de processamento, que é totalmente contornada pelo uso de uma arquitetura de processamento pipeline, com processos sendo realizados em paralelo por processadores vetoriais específicos para uma determinada tarefa. Maiores detalhes são encontrados em [8] no que tange a definições mais formais de "processamento pipeline", "processadores vetoriais", "processos paralelos", etc.

Para nós, o que interessa aqui é mostrarmos como identificar os processos paralelos dentro do macro-processo de decodificação e quais as cadências e entrelaçamentos dos diversos processos paralelos. É isto que faremos a seguir.

5.2. DIVISÃO DO PROCESSO DE DECODIFICAÇÃO EM PROCESSOS PARALELOS

Revendo o algoritmo 2 de Chase, seções 4.3 e 4.5, para o CGE (24, 12), encontramos os seguintes processos que podem ser definidos:

a) Armazenar a seqüência recebida (decidida abruptamente) **Y**;

b) Calcular a síndrome primária $\mathbf{S} = \mathbf{Y}\,[\mathbf{H}]^T$;

c) Procurar e armazenar os endereços POSi, i=1,...,*n* das posições dos INT [*d*/2] = 4 dígitos menos confiáveis a partir da recepção dos vetores α_i de confiabilidade dos dígitos;

d) Armazenar o peso analógico α_i (níveis de confiabilidade) de cada dígito Yi do bloco **Y** (i∈ 1,2....,24);

e) Gerar os $2^4 = 16$ padrões de teste **Tj** a partir das 4 POSi;

f) Gerar as 16 síndromes de teste $\mathbf{S_{Tj}}$ (j∈0,1,....15) que serão calculadas a partir de $\mathbf{S_{Tj}} = \mathbf{T_j}\,[\mathbf{H}]^T$;

g) Obter as 16 síndromes modificadas $\mathbf{Sj'} = \mathbf{S} \oplus \mathbf{S_{Tj}}$;

h) Obter os 16 padrões de erro **E'j** a partir das síndromes **Sj'**, por tabela decodificadora binária (look-up table);

i) Calcular $\mathbf{Ej} = \mathbf{E'j} \oplus \mathbf{Tj}$;

j) Calcular os 16 pesos analógicos $W\alpha$ (**Ej**) a partir da soma dos vetores-confiabilidade α_i associados às posições que valem **"1"** em **Ej**;

k) Comparar os 16 W_α (**Ej**) e achar o **Ej** de menor peso analógico; armazenar este vetor $\mathbf{E_{jMIN}}$;

l) Somar o padrão de erros **Ej** à seqüência **Y**, obtendo a estimativa da palavra-código originalmente transmitida, $\mathbf{X} = \mathbf{Y} \oplus \mathbf{E_{jMIN}}$;

m) Transferir os primeiros *k* dígitos de **X** como bits de informação ao usuário.

A passagem do fluxograma do algoritmo 2 de Chase da Fig. 4.2. para a divisão dos processos dada acima não é direta e precisa ser esclarecida em vários pontos, o que faremos no que se segue para provar que é adequada.

Primeiramente, precisamos provar que o cálculo de $W\alpha$ (**Ej**) como mostrado no item j é válido e vantajoso. Outro fato a ser provado é que o fato de trabalharmos com os *endereços das posições no vetor* é mais vantajoso que trabalhar com a palavra toda.

5.2.1. PROVA DO ITEM j

Precisamos provar que o cálculo do peso analógico a partir da soma dos vetores-confiabilidade endereçados pelas posições cujos dígitos valem **"1"** no vetor **Ej** é equivalente ao cálculo do peso analógico da diferença entre a palavra recebida e a decodificada. Queremos achar

$$\underset{m}{\text{MIN}}\, W_\alpha(\mathbf{Y} \oplus \mathbf{X_m}) \tag{5.3}$$

como exige o algoritmo de Chase na eq. (4.4). Sabemos ainda que:

$$W_\alpha(K) = \sum_{i=1}^{n} \alpha_i k_i$$

Como $\mathbf{X}_\text{m} = \mathbf{Y} \oplus \mathbf{E}_\text{m}$ (5.4)
onde $\mathbf{E}_\text{m}$ é o padrão de erros, levando (5.4) em (5.3), teremos:

$$\underset{m}{\text{MIN}}\, W_\alpha(\mathbf{Y} \oplus \mathbf{X_m}) = \underset{m}{\text{MIN}}\, W_\alpha(\mathbf{Y} \oplus \mathbf{Y} \oplus \mathbf{E_m}) = \underset{m}{\text{MIN}}\, W_\alpha(\mathbf{E_m})$$

Mas:

$$W_\alpha(\mathbf{E_m}) = \sum_{i=1}^{n} \alpha_i \mathbf{e}_i \tag{5.5}$$

A grande vantagem de (5.5) sobre (5.3) é que só devemos somar até (d-1) termos dos α_i endereçados pelos dígitos que valem **"1"** em $\mathbf{E}_\text{m}$ ao invés da soma de n termos $\sum_{i=1}^{n} \alpha_i (\mathbf{Y}_i \oplus \mathbf{X}_{mi})$, reduzindo portanto o número de chips necessários.

5.2.2. PROVA DA VANTAGEM DE SE TRABALHAR COM ENDEREÇOS DAS POSIÇÕES EM ERRO

Vamos provar aqui que em todo o processo de decodificação é vantajoso trabalharmos com os endereços das posições em erro ao invés de trabalharmos com os vetores completos, desde que n seja suficientemente grande.

Seja um código binário linear de bloco de comprimento n e dimensão k, denotado por $\mathbf{C}$ (n,k), e que tenha distância mínima d.

No caso de uma decodificação binária convencional (decisão abrupta), o decodificador tem capacidade corretora:

$$t = \text{INT}\left[\frac{d-1}{2}\right]$$

a) DECODIFICAÇÃO CONVENCIONAL:

A decodificação convencional determina um vetor-erro **E** de n elementos e realiza a soma de **E** com o vetor recebido **r** para achar a estimativa da palavra-código originalmente transmitida **c:**

$$\mathbf{c} = \mathbf{r} \oplus \mathbf{E} \tag{5.6}$$

Como **r**, **E** e **c** têm n dígitos, o número de linhas físicas (trilhas de circuito impresso) a serem manipuladas é:

$$L_{f1} = n \tag{5.7}$$

b) DECODIFICAÇÃO COM A MANIPULAÇÃO DOS ENDEREÇOS DAS POSIÇÕES EM ERRO:

Para endereçar binariamente um número inteiro entre 1 e n precisamos de um número de bits de endereço que vale:

$$NBA = \begin{cases} \log_2 n = \Gamma & \text{para } n = 2^\Gamma, \ \Gamma \text{ inteiro} \\ \text{INT}[\log_2 n] + 1 & \text{para quaisquer } n \text{ inteiros, } n \neq 2^\Gamma \end{cases} \tag{5.8}$$

Como o decodificador binário pode corrigir até t erros, o número máximo de linhas físicas a manipular seria:

$$L_{f2} = t \cdot \{\text{INT}[\log_2 n] + 1\} \tag{5.9}$$

Comparando (5.9) com (5.7), dada uma capacidade corretora t, a figura 5.1. apresenta os gráficos de L_{f1} e L_{f2} versus n, para $t = 3$ e $t = 7$.

O CGE (24, 12) tem $t = 3$ e $n = 24$. Calculamos os pontos de intersecção entre L_{f1} e L_{f2} para $t = 3$ e $t = 7$ e vemos que, com $t = 3$, e para qualquer código com n >>10, passa a ser vantajoso o uso das posições em erro ao invés da palavra toda. Devemos também salientar que, com o algoritmo 2 de Chase, a capacidade corretora aumenta para $(d-1) = 7$ de um CGE (24, 12), embora continuemos a usar um decodificador binário com $t = 3$.

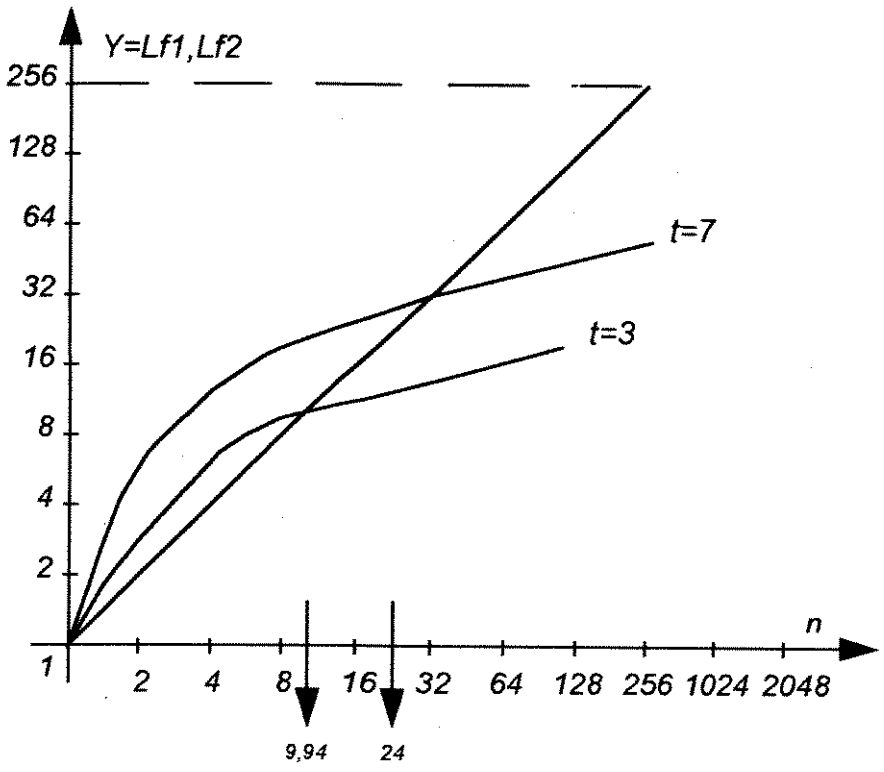


FIGURA 5.1. CURVAS L_{f1}, L_{f2} VERSUS n

5.2.3. FLUXOGRAMA DOS PROCESSOS QUE IMPLEMENTAM O ALGORITMO 2 DE CHASE

A fig. 5.2. apresenta o fluxograma dos processos apresentados nos itens a a m do início desta seção.

As linhas cheias indicam o fluxo de dados de um processo para outro e as pontilhadas representam a dependência (do processo) de dados calculados previamente (em outro processo anterior). O sinal ENDEC é um sinal que habilita o decodificador quando o nível é baixo (**"0"**); PA é um registrador de peso analógico que no início do processo recebe o valor associado à máxima confiabilidade, i.e., é carregado com o valor $\alpha_{MÁX}.d = 3.8 = 24$. J é um contador módulo 16, incrementado ao final de cada processo associado a um padrão de teste **Tj**, j $\in$ 0,...,15. M é o registro que recebe os dígitos correspondentes aos bits de informação para transferi-los posteriormente ao usuário. Uma função adicional incorporada ao decodificador é um medidor de BER (bit error rate), taxa de erros média na linha, que dá um alarme quando esta for maior que 10^{-2}, pois neste caso, a taxa de erro de bit à saída do decodificador não atende mais aos requisitos de alto desempenho garantido ao usuário, quando o canal é AWGN (tipicamente, enlaces satélite podem ser modelados assim).

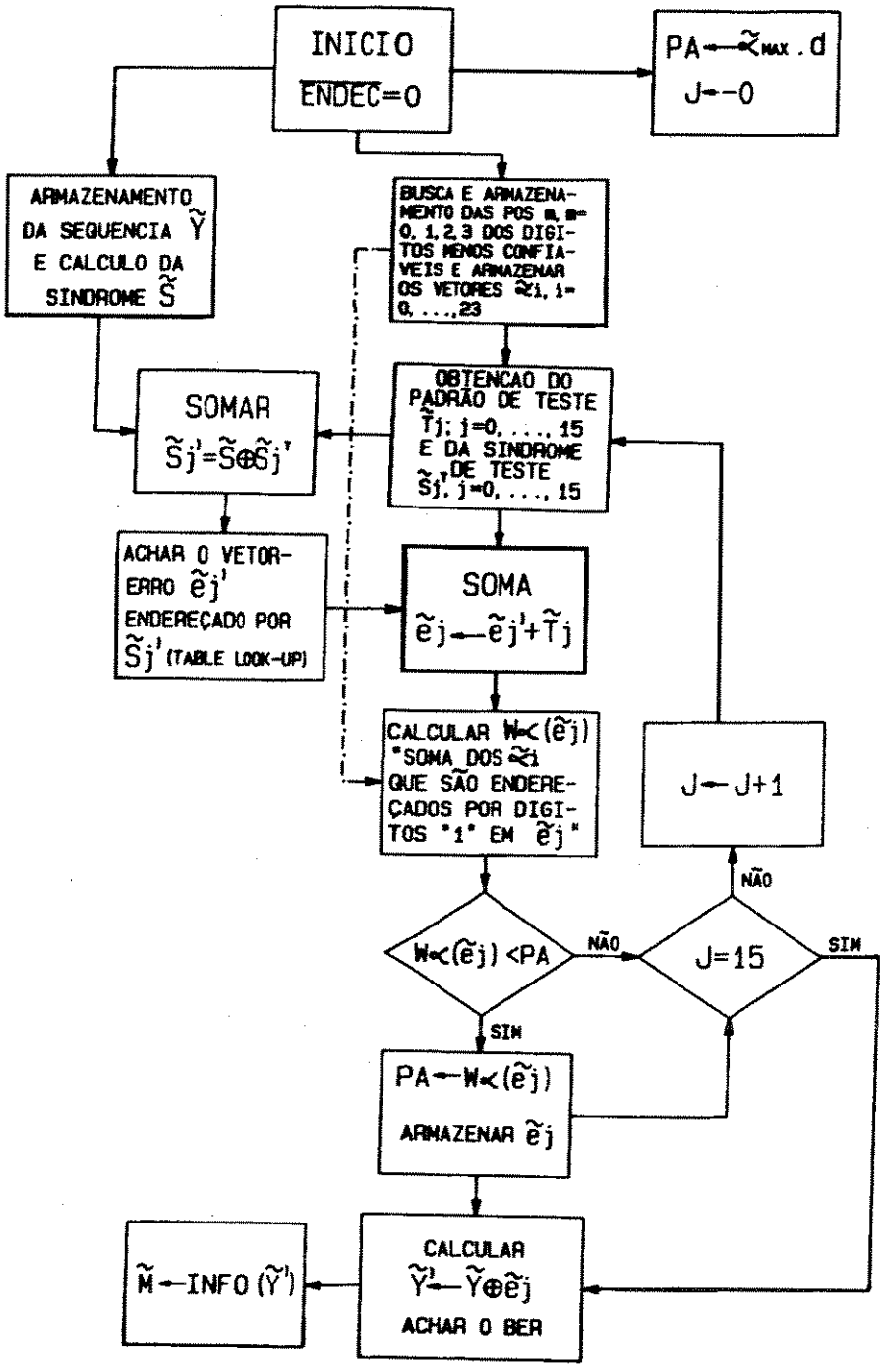


FIGURA 5.2. FLUXOGRAMA DA SEQÜÊNCIA DOS PROCESSOS QUE IMPLEMENTAM O ALGORITMO 2 DE CHASE

Abaixo fazemos alguns comentários importantes para o entendimento dos processos que serão detalhados logo mais:

a) Para o CGE (24, 12), precisamos de 5 linhas ou bits de endereço para a posição de qualquer dígito entre 1 e 24.

b) A síndrome primária **S** pode ser calculada com um registrador de síndrome convencional.

c) A busca das 4 posições menos confiáveis é feita sucessivamente à medida que os dígitos Y_i e os vetores associados α_i vão chegando; não há, portanto, **"perda"** de tempo para executar este processo.

d) Para calcular a síndrome-teste S_{Tj} fazemos o ou-exclusivo (X-OR) bit-a-bit das linhas da matriz $[\mathbf{H}]^T$ (gravada em PROM), cujos endereços são dados pelos endereços dos dígitos que estão supostamente em erro, i.e., pelos endereços dos dígitos que são **"1"** no vetor-teste **Tj**.

PROVA (item d)

Seja um vetor-erro de um código (n, k), $\mathbf{E} = (e_0, e_1, \ldots e_{n-1})$ e a matriz $[\mathbf{H}]^T$, $(n-k)$ x n:

$$[\mathbf{H}]^T = \begin{bmatrix} h_{00} & h_{01} & \cdots & h_{0,n-k-1} \\ h_{10} & h_{11} & \cdots & h_{1,n-k-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n-1,0} & h_{n-1,1} & \cdots & h_{(n-1),(n-k-1)} \end{bmatrix} = \begin{bmatrix} \mathbf{h_0} \\ \mathbf{h_1} \\ \vdots \\ \mathbf{h}_{n-1} \end{bmatrix}$$

Então, a síndrome **S** será:

$$\mathbf{S} = \mathbf{E} \cdot [\mathbf{H}]^T = \sum_i \mathbf{h}_i \quad , \forall\, i \setminus (\mathbf{E}_i = 1) \wedge 0 \le i \le n-1 \qquad (5.10)$$

Para o nosso caso, devemos fazer até 4 somas já que **Tj** tem até 4 dígitos **"1"**.

e) Como temos que calcular o peso $W\alpha$ (**Ej**) no menor tempo possível e economizar ao máximo o número de chips, a tabela decodificadora **S'j** → **E'j** pode ser modificada para uma tabela **S'j** → (*endereços das posições cujos dígitos são* **"1"** *em* **E'j**) que, usados em conjunto com os endereços POSm, m= {0, 1, 2, 3}, suprimem a soma **Ej= E'j** ⊕ **Tj** e passam a endereçar diretamente os α_i necessários ao cálculo do peso analógico. As vantagens são:

– eliminação de um somador binário de 24 bits (reparar aqui que não é X-OR bit-a-bit mas soma ordinária);

– ao invés de 24 linhas de saída, a tabela fornecerá 20 linhas apenas (4 endereços no máximo, de 5 bits cada um), economizando espaço de lay-out;

– ao invés de termos 24 X-ORs para calcular **Ej** = **E'j**⊕ Tj, usamos um circuito que analisa a coincidência de endereços entre as quatro POSm ($1 \leq m \leq 4$) de menor confiabilidade e os 4 endereços de saída da tabela decodificadora.

f) A correção, ao invés de ser feita pelo X-OR, **c** = **Y** ⊕ **Ej**, passa a ser feita pela inversão dos dígitos no instante em que este estiver saindo para o usuário (por meio de um contador que fornece o endereço do dígito que está saindo, quando este coincide ou com qualquer POSm ou qualquer endereço de Ej).

5.3. DETALHAMENTO DO MACRO-PROCESSO DE DECODIFICAÇÃO

Após toda a discussão anterior, estamos prontos, agora, para apresentar o macro-processo de decodificação, os circuitos (em diagramas de bloco) que implementam os processos mais complexos e mais importantes, e o diagrama em blocos do decodificador do AMDT.

A Fig. 5.3. mostra o MACRO-PROCESSO DE DECODIFICAÇÃO.

A Fig. 5.3. é essencialmente a mesma da Fig. 5.2. com o detalhe de distribuir no tempo a seqüência de processos. Estes tempos estão dentro dos círculos à esquerda e os processos que ocorrem entre eles são separados por linhas pontilhadas.

A convenção de tempos é dada abaixo:

t_i → tempo de duração do dígito (i) da palavra-código (k)

$$i = \begin{cases} 0 \rightarrow \text{instante inicial} \\ 1 \text{ a } 24 \rightarrow \text{duração dos bits} \end{cases}$$

$t_{i'}$ → idem, da próxima palavra-código (k+1)

$t_{i''}$ → idem, da próxima palavra-código (k+2)

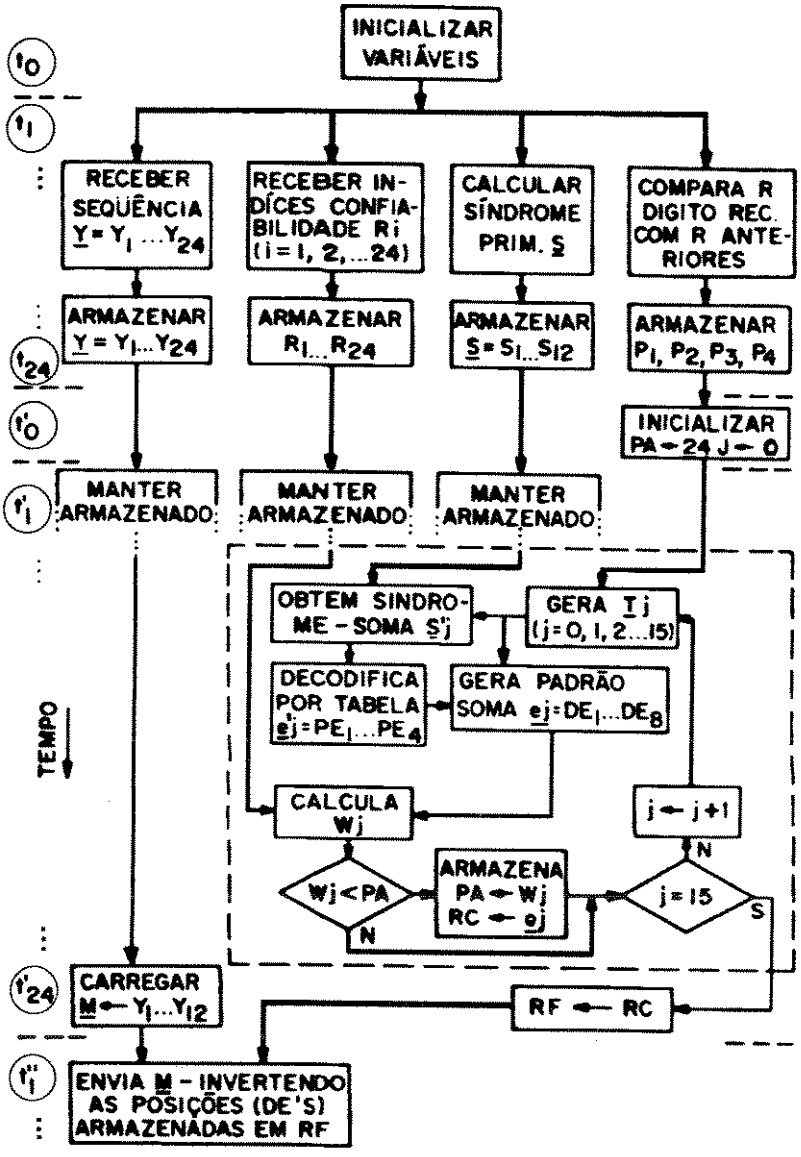
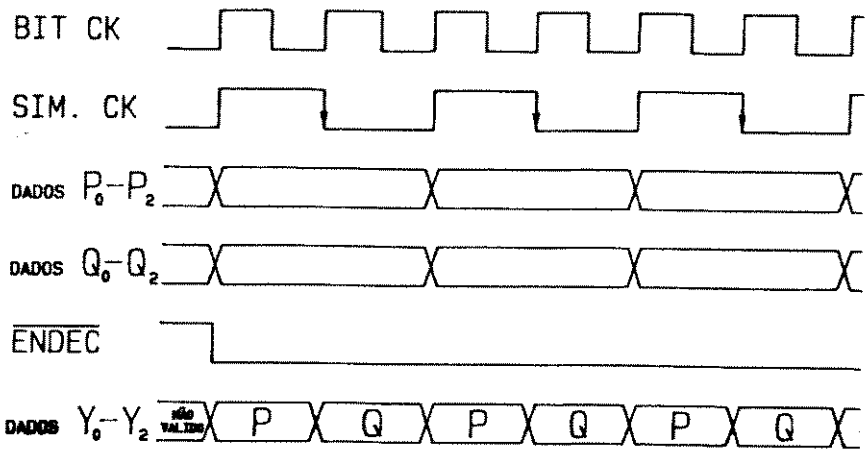


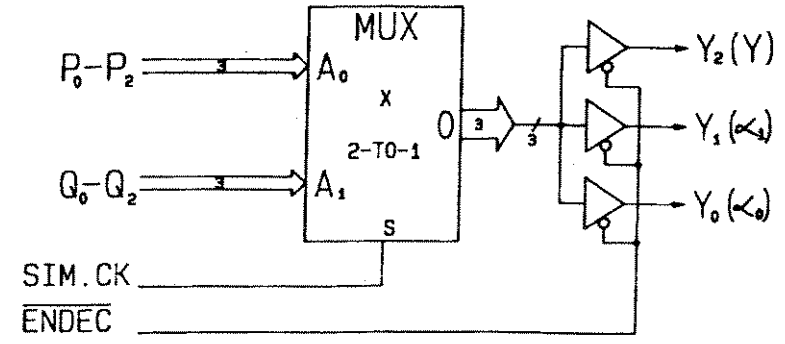
FIGURA 5.3. MACRO-PROCESSO DE DECODIFICAÇÃO

5.3.1. DECOMUTAÇÃO DOS CANAIS P e Q

Conforme salientamos no capítulo 1, seção 1.3.2.4., a recepção de uma palavra de 24 bits exige uma recomposição dos sinais dos canais P e Q, que, apesar de não fazer parte do processo de decodificação apresentado na Fig. 5.3., deve ser feita pelo hardware decodificador de forma a decomutar corretamente os canais P e Q. A Fig. 5.4. mostra o circuito necessário para realizar esta decomutação e seu "timing" (diagrama de sinais no tempo) associado.



a) " TIMING " DO CONVERSOR PARALELO-SÉRIE



b) CIRCUITO DO CONVERSOR PARALELO-SÉRIE

FIGURA 5.4. DECOMUTAÇÃO DOS CANAIS P E Q

5.4. CÁLCULO DA SÍNDROME PRIMÁRIA

O cálculo da síndrome (primária) **S** é feito com um registrador de síndrome convencional, como mostra a fig. 5.5., sem **"perda"** de tempo, pois o cálculo é feito à medida que os dígitos a decodificar chegam serialmente. Como se sabe,

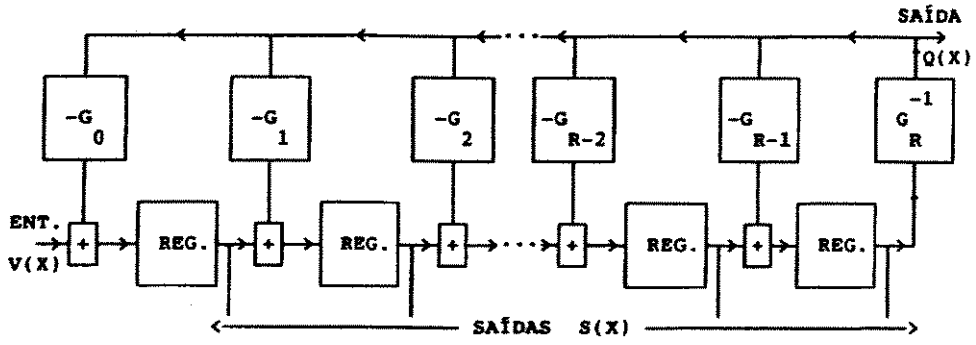
$$\mathbf{S} = \mathbf{v}\,[\mathbf{H}]^{\mathrm{T}} \qquad (5.11.a)$$

Ou seja, em códigos cíclicos:

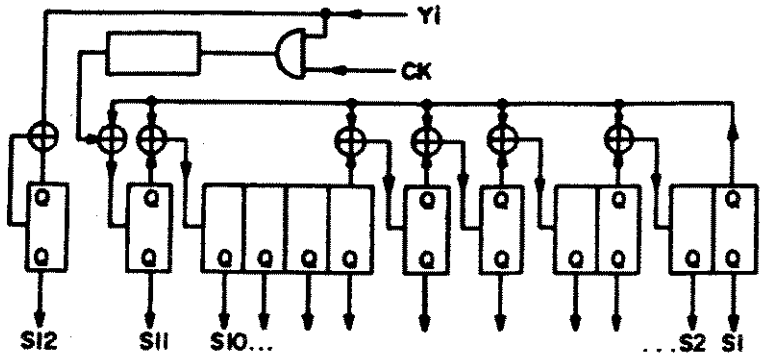
$$S(x) = \mathrm{R}\left[\frac{v(x)}{g(x)}\right] \qquad \mathrm{R} = \text{resto} \qquad (5.11.b)$$

A fig. 5.5. a) apresenta o circuito para $g(x)$ genérico onde:

$$g(x) = g_r X^r + g_{r-1} X^{r-1} + \cdots + g_1 X + g_0 \qquad (5.11.c)$$



a) genérico



b) registro de síndrome para o CGE (24,12)

FIGURA 5.5. CÁLCULO DA SÍNDROME

Como para o CG (23, 12)

$$g_1(x) = X^{11} \oplus X^9 \oplus X^7 \oplus X^6 \oplus X^5 \oplus X \oplus 1$$

E, no CGE (24, 12), se C = $(C_0,...,C_{23})$, $C_{23} = \sum_{i=0}^{22} C_i$ é o 12° bit da síndrome, o gerador de síndrome será conforme mostrado na fig. 5.5.b)

5.5. PROCESSO 01: PROCURA DAS 4 POSIÇÕES DE MENOR CONFIABILIDADE

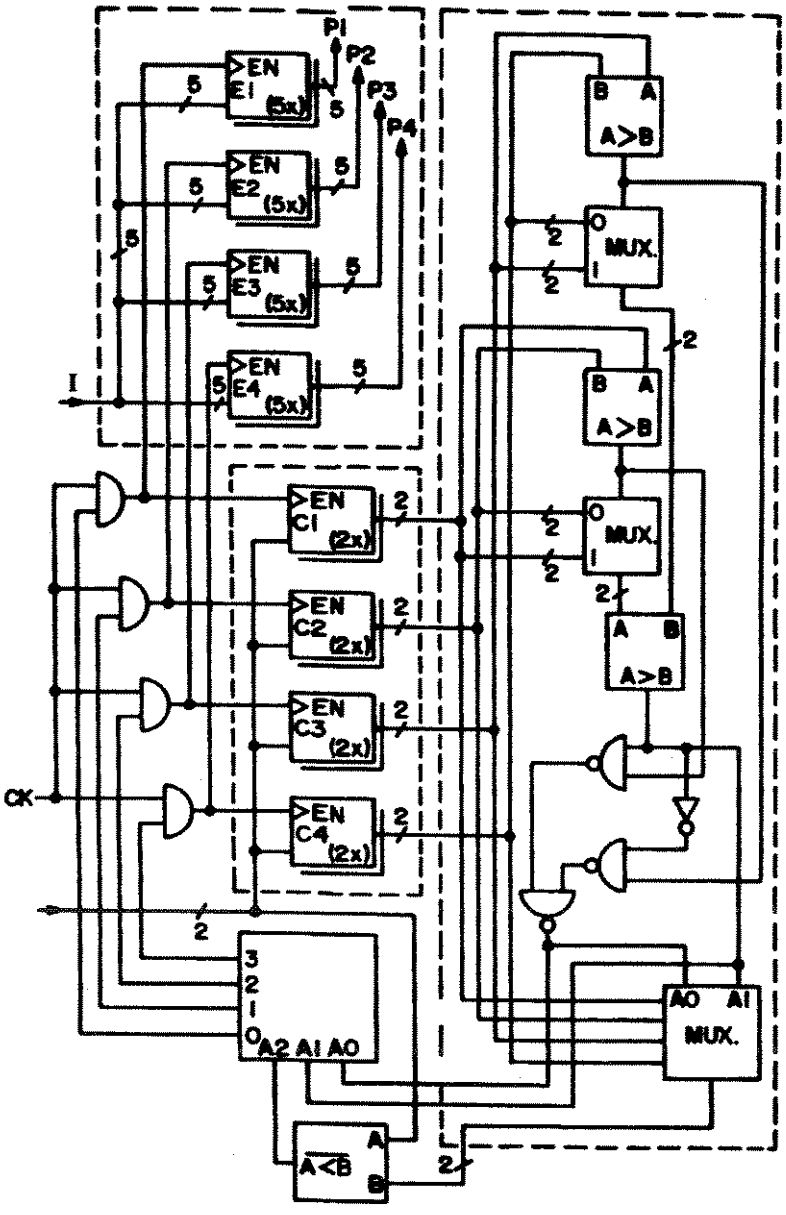
Este processo pode ser implementado com o circuito da Figura 5.6.

Neste processo, P1, P2, P3 e P4 são os endereços das posições (de 1 a 24) dos 4 dígitos menos confiáveis, cujos pesos α_i estarão armazenados em W1, W2, W3 e W4. I é um contador de endereços com excursão de 1 a 24. Deste contador I depende, portanto, toda a estratégia de se trabalhar com os endereços das posições dos dígitos

nos vetores processados pela arquitetura: vetor de recepção, **Y**, vetores padrões de teste, **Tj**, vetor-erro, etc.

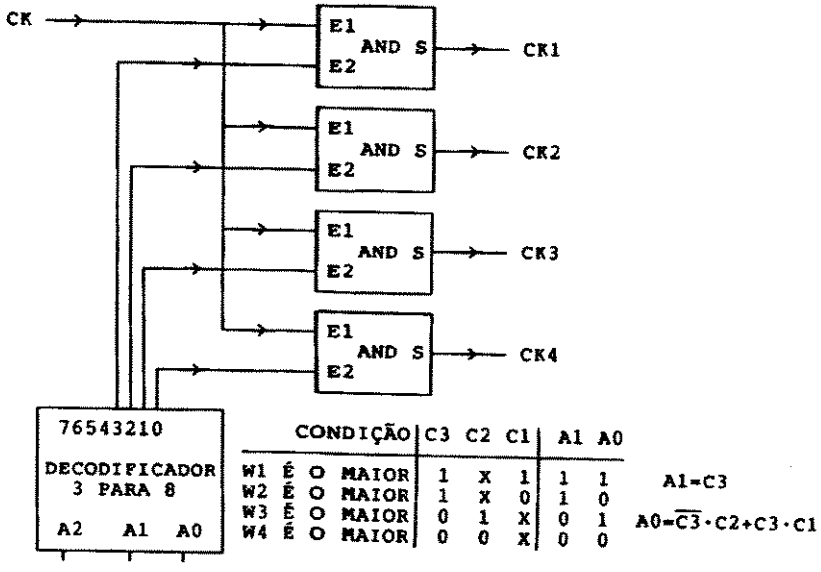
O processo funciona da forma descrita a seguir. À medida que os dígitos de um vetor de recepção vão chegando, cada confiabilidade associada a cada dígito é comparada com o maior valor de confiabilidade armazenado nos registradores W1 a W4, que possuem as 4 menores confiabilidades que chegaram até o momento da comparação. Este maior valor dentre W1 a W4 é obtido através da lógica de comparação formada pelo conjunto de comparadores, multiplexadores e portas NAND da figura 5.6 a). Caso o valor da confiabilidade que chega seja maior que o maior valor de confiabilidade armazenado até este momento, ele é desprezado. Caso o valor da confiabilidade que chega seja menor que o maior valor de confiabilidade armazenado (i.e., o maior entre os 4 menores até este instante), ocorre uma troca: o valor que chega é armazenado no registro Wi que guarda o maior valor até o instante da comparação, sendo este maior valor descartado após a comparação. A mesma lógica de seleção, que atualiza o registro Wi que possui o maior valor de confiabilidade, faz também a atualização do registro de posição Pi correspondente a Wi ($1 \leq i \leq 4$). Na condição de igualdade entre o valor da confiabilidade que chega e o maior valor de confiabilidade em Wi, é indiferente, do ponto de vista do algoritmo de Chase, a ocorrência ou não-ocorrência de atualização de registros.

Para garantir o funcionamento deste processo, é necessária a inicialização dos registros W1 a W4 com o valor máximo de confiabilidade que o quantizador fornece, no início de cada bloco de 24 dígitos que entram no decodificador. No nosso caso, este valor é 3, representado por convenção como sendo 11 na base 2.



A) CIRCUITO GLOBAL

FIGURA 5.6. CIRCUITO DE PROCURA DAS QUATRO POSIÇÕES MENOS CONFIÁVEIS



B) DETALHAMENTO DO SUB-BLOCO SELETOR 3-PARA-8

FIGURA 5.6. CIRCUITO DE PROCURA DAS QUATRO POSIÇÕES MENOS CONFIÁVEIS

5.6. PROCESSO 02: OBTENÇÃO DOS 16 PADRÕES DE TESTE Tj

Tendo em código binário as 4 posições (endereços) dos dígitos menos confiáveis, precisamos gerar todas as 16 combinações possíveis destes. A solução é dada por um contador módulo 16 "modulando" ou mascarando P1, P2, P3 e P4.

Seja **J** = J3 J2 J1 J0 $\in$ { 0, 1, 2 .... F_H } um contador módulo 16 e P1, P2, P3, P4 os endereços dos 4 dígitos menos confiáveis da palavra recebida **Y**. Para gerar Tj, basta fornecermos os 4 endereços que eventualmente podem ser "1" em Tj, chamados aqui DT1, DT2, DT3, DT4. Então:

$$DT1 = J0.(P1) = J0.(P11\ P12\ P13\ P14\ P15)$$

$$DT2 = J1.(P2) = J1.(P21\ P22\ P23\ P24\ P25)$$

$$DT3 = J2.(P3) = J2.(P31\ P32\ P33\ P34\ P35)$$

$$DT4 = J3.(P4) = J3.(P41\ P42\ P43\ P44\ P45) \qquad (5.12)$$

Precisamos ainda observar qual a velocidade de mascaramento necessária para efeito de cálculo da velocidade de relógio do contador módulo 16. Este relógio, diferentemente do caso do contador I (contador mestre de posições) que recebe

diretamente o relógio de dígitos (no caso, 1068 kHz), é função das premissas da arquitetura.

Como queremos minimizar o hardware usado e também o tempo de decodificação, dado que os endereços das posições de menor confiabilidade só estão disponíveis ao fim do processo de entrada, ou seja, após o primeiro bloco do tempo de decodificação, e dado que queremos decodificar em, no máximo, dois blocos, dispomos então de um bloco para gerar todos os padrões de teste. Portanto, se R_d é a taxa de dígitos, N o tamanho do bloco em dígitos e d a distância mínima do código usado, a freqüência f_M do relógio do contador de mascaramento será, genericamente:

$$f_M = \frac{2^{\mathrm{INT}\left[d/2\right]}}{N} \cdot R_d \qquad (5.13)$$

Em nosso caso específico:

$$f_M = \frac{16}{24} \cdot R_d = \frac{2}{3} \cdot R_d \qquad (5.14)$$

Portanto, o contador de mascaramento deve receber um relógio de 2/3 da taxa de dígitos (1068 kbit/s), ou seja, 712 kHz.

A figura 5.7. mostra o diagrama em blocos do circuito necessário para implementar este processo.

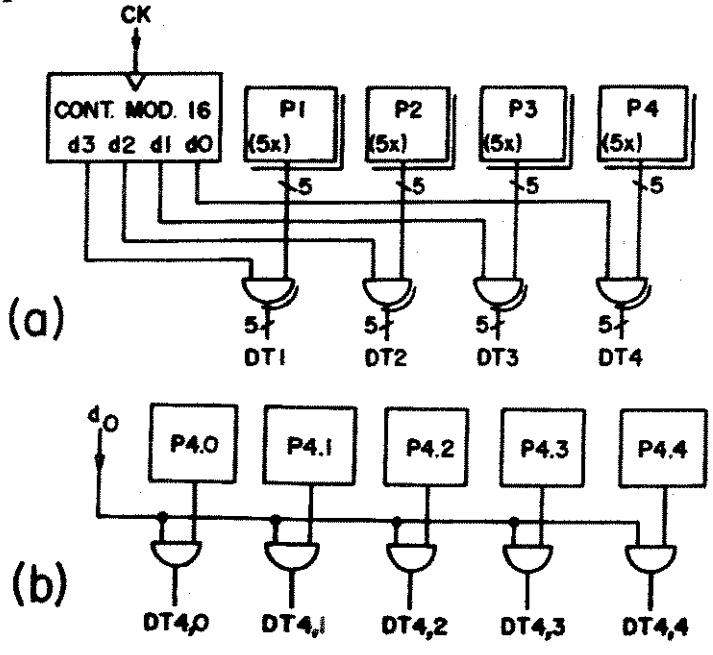


FIGURA 5.7. CIRCUITO DE GERAÇÃO DOS PADRÕES DE TESTE

5.7. PROCESSO 03: CÁLCULO DAS SÍNDROMES-TESTE S_{Tj}

Conforme o procedimento, dado no item 5.2.3. (comentário d), fazemos aqui 4 somas bit-a-bit de linhas da matriz $[H]^T$ (gravada em PROM) endereçadas pelos DT_i de **Tj** (usando a eq. (5.10)).

Observamos que:

$$DT_i = \begin{cases} 0 & \text{se o dígito for 0} \\ 1\ a\ 24_{10} = 1_H\ a\ 18_H & \text{se o dígito for 1} \end{cases}$$

A fig. 5.8. mostra o circuito utilizado e a fig. 5.9 mostra o conteúdo da PROM.

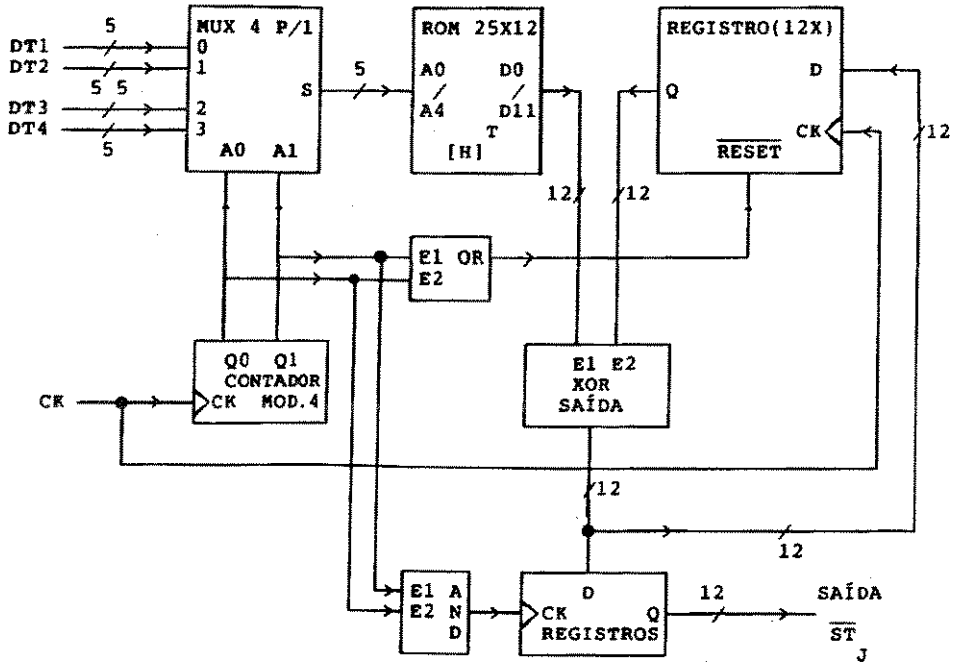


FIGURA 5.8. CIRCUITO PARA CALCULAR S_{Tj}

ENDEREÇO HEXADEC.	S(0-7) HEXAD.	S(8-11) HEXAD.	BITS MATRIZ PAR. TRANSP. 0	1	2	3	4	5	6	7	8	9	10	11
00	00	00	0	0	0	0	0	0	0	0	0	0	0	0
01	75	0C	1	0	1	0	1	1	1	0	0	0	1	1
02	9F	0C	1	1	1	1	1	0	0	1	0	0	1	1
03	4B	0D	1	1	0	1	0	0	1	0	1	0	1	1
04	E3	0E	1	1	0	0	0	1	1	1	0	1	1	1
05	B3	09	1	1	0	0	1	1	0	1	1	0	0	1
06	66	0B	0	1	1	0	0	1	1	0	1	1	0	1
07	CC	0E	0	0	1	1	0	0	1	1	0	1	1	1
08	ED	09	1	0	1	1	0	1	1	1	1	0	0	1
09	DA	0B	0	1	0	1	1	0	1	1	1	1	0	1
0A	84	0E	0	0	1	0	1	1	0	1	1	1	1	1
0B	1D	08	1	0	1	1	1	0	0	0	1	1	0	1
0C	3A	0E	0	1	0	1	1	1	0	0	0	1	1	1
0D	01	0B	1	0	0	0	0	0	0	0	0	0	0	1
0E	02	08	0	1	0	0	0	0	0	0	0	0	0	1
0F	04	08	0	0	1	0	0	0	0	0	0	0	0	1
10	08	08	0	0	0	1	0	0	0	0	0	0	0	1
11	10	08	0	0	0	0	1	0	0	0	0	0	0	1
12	20	08	0	0	0	0	0	1	0	0	0	0	0	1
13	40	08	0	0	0	0	0	0	1	0	0	0	0	1
14	80	08	0	0	0	0	0	0	0	1	0	0	0	1
15	00	09	0	0	0	0	0	0	0	0	1	0	0	1
16	00	0A	0	0	0	0	0	0	0	0	0	1	0	1
17	00	0C	0	0	0	0	0	0	0	0	0	0	1	1
18	00	08	0	0	0	0	0	0	0	0	0	0	0	1

FIGURA 5.9. TABELA DE PROGRAMA DAS PROMS DE CONVERSÃO DE T_J EM S_J

O relógio do contador módulo 4 da figura 5.8. é obviamente quatro vezes mais rápido que o relógio do contador de mascaramento para que o valor final da soma (i.e., a síndrome-teste $\mathbf{S_{Tj}}$) esteja disponível no início do próximo padrão de teste, ou seja, ao final do período disponível para o padrão de teste vigente. De forma geral, temos então que a freqüência deste relógio, fST, deve ser:

$$f_{ST} = \mathrm{INT}\left[\frac{d}{2}\right] \cdot f_M \qquad (5.15)$$

Com fM dado na equação (5.13). Novamente, o relógio só depende dos parâmetros do código usado e da taxa de dígitos.

Uma variante do circuito, de forma a incluir também a soma com a síndrome primária, **S**, obtida pelo registrador de síndrome, economizando então um somador

bit-a-bit adicional, pode ser implementada caso estejam disponíveis registradores e PROMs do tipo "tri-state". Assim, apenas elaborando o circuito de controle para realizar cinco somas sucessivas, temos um circuito menos volumoso para realizar duas etapas do macro-processo de decodificação: cálculo da síndrome de teste e soma desta com a síndrome primária. O preço a pagar é a utilização de um contador módulo 8 no lugar de módulo 4 (da figura 5.8) e dobrar a freqüência de operação do contador, agora chamada *fSOMA*:

$$f_{SOMA} = 2 \cdot \mathrm{INT}\left[\frac{d}{2}\right] \cdot f_M \qquad (5.16)$$

Por economizar somadores (cada somador deve ser um conjunto de X-ORs de mesma dimensão que a síndrome, i.e., $(n-k)$), adotamos esta última solução e temos que *fSOMA* = [(2.4.2)/3] Rd = (16/3) Rd = 5.696 kHz. Este é o valor máximo de freqüência usado pela arquitetura aqui apresentada. Não é um valor alto de freqüência, se comparado às necessárias para a operação dos decodificadores de Viterbi disponíveis em CIs dedicados, que usam freqüências de operação mínima em torno de 80 a 140 vezes o valor da taxa de bits (este valor depende do tipo de treliça), operando então com freqüência da ordem de 5 MHz para decodificar uma linha de apenas 50 kbit/s. Não é também um valor alto do ponto de vista de interferência nos circuitos de RF. Entretanto, é um valor que já exige certas restrições e cuidados no projeto físico de placas (disposição de lay-out, uso de pistas blindadas, aterramentos convenientes, etc) muito densas que usem CIs com tempos de propagação já relevantes (como é o caso da placa que contém esta arquitetura).

5.8. PROCESSO 04: ACHAR O VETOR-ERRO E'j (TABLE LOOK-UP)

Após calcularmos $\mathbf{S'j} = \mathbf{S} \oplus \mathbf{S}_{Tj}$, queremos achar os endereços das posições do vetor-erro **E'j** cujos dígitos valem "1", através de um decodificador binário (tabela decodificadora). O código tem t=3 e n=24 e é quase perfeito. Então a tabela terá 4x5 linhas de saída, como mostra a fig. 5.10.

Vale a pena lembrarmos que o procedimento de confecção da ROM ou EPROM que implementa o decodificador de decisão abrupta (i.e., o decodificador binário) é muito trabalhoso. Envolve o uso de um microcomputador que prepara a tabela a partir de um programa com o seguinte algoritmo exaustivo (válido para códigos de bloco lineares perfeitos ou quase-perfeitos):

a) geração de todos os padrões de erro de peso mínimo até o peso t=INT [$(d-1)/2$], com t sendo a capacidade corretora do código;

b) Escolha de alguns padrões de erro de peso igual a $(t+1)$, quando o código é quase perfeito, de forma a completar o conjunto de $2^{(n-k)}$ padrões;

c) Calcular as síndromes correspondentes aos padrões, multiplicando-los por $[H]^T$, matriz de verificação de paridade transposta;

d) ordenar a tabela formada, em ordem crescente de síndromes, e separar os bits em grupos de 8 (para facilitar EPROMs).

Tal programa, codificado em TURBO PASCAL, executado em uma máquina do tipo PC-XT com relógio de 8 MHz, consumiu cerca de 3 horas de CPU, mesmo após otimização dos algoritmos de geração (de padrões e síndromes) e das estruturas de dados usadas, no caso do CGE (24, 12).

Os padrões de erro utilizados foram todos os de 0, 1, 2 e 3 erros mais 1/6 de todos os padrões de 4 erros (para completar os 4096 padrões corrigíveis). Como padrões de 4 erros selecionados, usamos todos os que tinham erro no primeiro dígito do bloco.

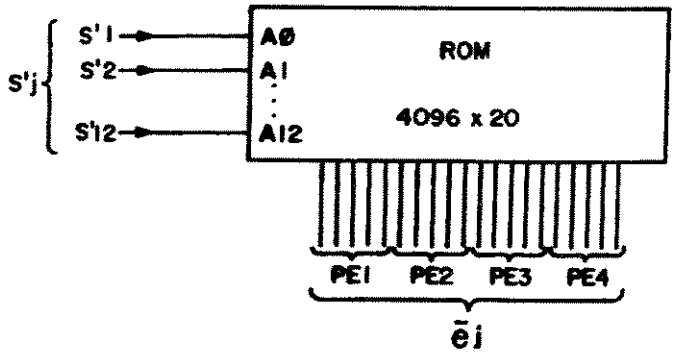


FIGURA 5.10. DECODIFICADOR BINÁRIO

5.9. PROCESSO 05: CÁLCULO DO PESO ANALÓGICO

A fig. 5.11. mostra o hardware que permite calcular o peso analógico do vetor-erro. Os multiplexadores fazem uma análise da coincidência de quaisquer PEi com quaisquer DTj. Caso haja DTj = PEi, $\forall$i, j, não há erro nesta posição. Neste caso, acessamos a posição 0 da RAM de pesos (α_i) onde estará gravado previamente α_i = 0.

Para realizar o cálculo do peso analógico dos 16 padrões de erro, dados pela soma bit-a-bit do padrão de erro dado pelo decodificador binário mais o vetor de teste associado, o relógio do registrador de pesos analógicos necessita de uma freqüência muito mais alta que a taxa de dígitos de entrada.

Em nosso caso específico, o decodificador binário fornece até quatro endereços de posições em erro e o vetor de teste outros quatro endereços de posições de baixa confiabilidade do vetor recebido (bloco de 24 dígitos na entrada do decodificador). Há, portanto, a necessidade de se somar até oito valores de confiabilidades (associadas a essas posições) dentro do intervalo em que um dos padrões de teste

está valido. Lembrando que os padrões de teste são 16, há, portanto, que se achar 16 pesos analógicos (por bloco de 24 dígitos), num total de 16x8 = 128 somas (e acessos de leitura à RAM de confiabilidade). A freqüência de relógio do registro acumulador do somador deve, então, ser igual a:

$$f_{PA} = \frac{128}{24} \cdot R_d = \frac{16}{3} \cdot R_d \qquad (5.17)$$

onde fPA é a freqüência de relógio do acumulador do somador de cálculo do peso analógico do padrão de erros sob teste e também a freqüência de acessos à RAM de confiabilidades e R_d a taxa de dígitos à entrada do decodificador. Em nosso caso, obtemos fPA = 5.696 kHz, mesmo valor de $fSOMA$ dada em (5.16), podendo-se, então, utilizar o mesmo relógio para os dois circuitos.

De forma geral, o relógio necessário para o acumulador será dado por:

$$f_{PA} = \left\{ \mathrm{INT}\left[\frac{d}{2}\right] + t^* \right\} \cdot f_M \qquad (5.18)$$

com

$$t^* = \begin{cases} t & \text{para códigos perfeitos} \\ (t+1) & \text{para códigos quase-perfeitos} \end{cases}$$

onde fPA é a freqüência necessária para o relógio do acumulador do somador de pesos analógicos, fM dado segundo a eq. (5.13) e t^* é o número máximo de erros que o decodificador binário consegue corrigir.

É interessante observarmos que fPA será igual a $fSOMA$ sempre que o código utilizado for quase-perfeito (o que é verdade para quase todos os códigos de bloco usados na prática).

Outro aspecto interessante a ser observado é que, pela teoria, o decodificador de Chase só corrige até $(d-1)$ erros mas o hardware trabalha com d endereços de posições em erro no vetor de erro sob teste, sempre! Esta incongruência é aparente porque o analisador de coincidência de endereços impede, para um padrão final selecionado, na prática, que as d posições em erro tenham endereços ativos. No mínimo, um deles estará com o endereço fantasma 0 da RAM de pesos (e confiabilidades) e o somador somará uma confiabilidade 0 no peso analógico do padrão sob teste. Entretanto, estas d posições em erro são necessárias porque, devido à implementação em hardware ser fixa, não se sabe a priori quais endereços de posições em erro fornecidos pelo decodificador binário são coincidentes aos endereços de posições menos confiáveis fornecidos pelo gerador de padrões de teste.

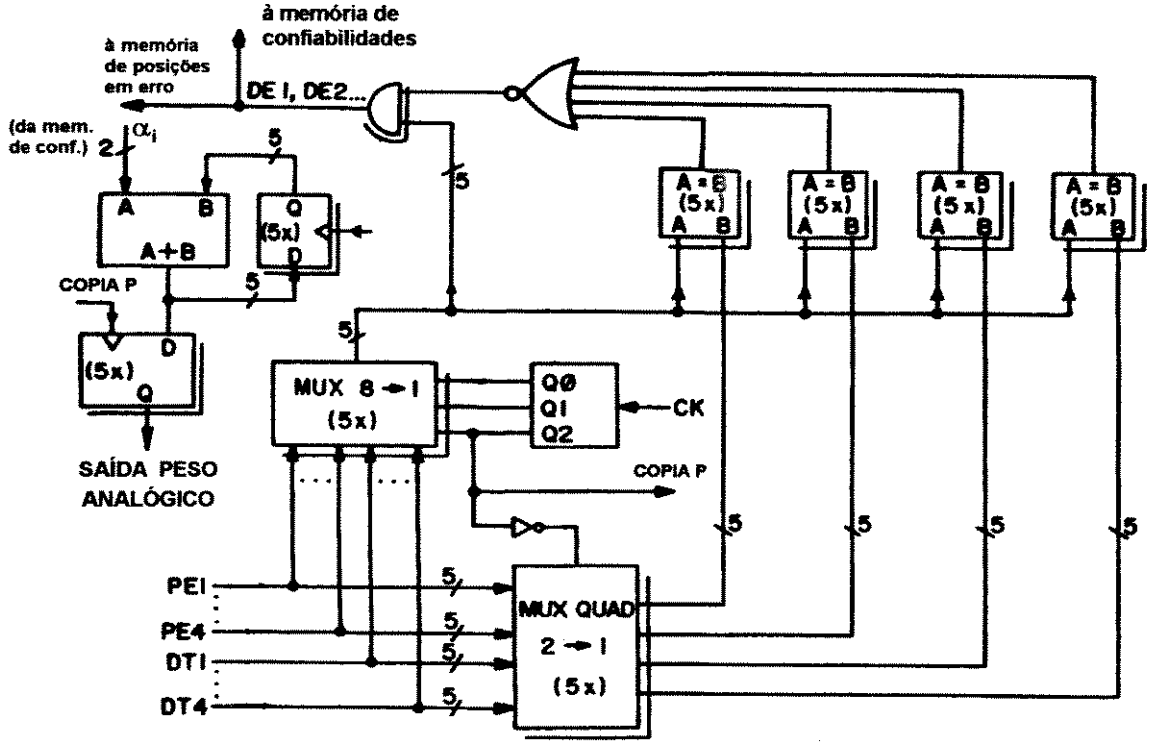


FIGURA 5.11. CIRCUITO PARA CÁLCULO DO PESO ANALÓGICO

Voltando à fig. 5.11., que mostra o diagrama em blocos do circuito que implementa o processo de cálculo do peso analógico de cada padrão de teste, ressaltamos que omitimos as linhas de "reset" dos registros, que são forçados a ir a "0" no início de cada cálculo de peso analógico dos padrões sob teste. Além disso, o circuito implementado pode dispensar os multiplexadores caso os registros de padrões de erro do decodificador binário (PE1 a PE4) e os dos dígitos sob teste (DT1 a DT4) sejam do tipo "tri-state" e estejam ligados em um barramento comum. A multiplexação ocorre então naturalmente, com um decodificador de 3 para 8 linhas realizando a habilitação de cada registro no barramento em instantes de tempo distintos e de forma seqüencial. Como os padrões PEi e DTj estão presentes em tempos distintos, temos ainda que usar quatro flip-flops que indicam o status de ter havido ou não uma coincidência de valor entre um determinado DTj com algum dos quatro PEi. Este bit de status é usado para "apagar" o DTj coincidente quando ocorre sua liberação no barramento.

Finalmente, se o tempo de comparação, necessário para o comparador de métricas (descrito na próxima seção) atuar, for suficientemente pequeno, o valor do peso analógico SF do padrão de erros sob teste não precisa ser armazenado em um registrador separado do acumulador como ocorre na fig. 5.11. Para não utilizarmos o registrador de peso analógico do somador, basta que o tempo de comparação e atualização do comparador de métricas seja menor que o tempo disponível entre a obtenção do peso analógico e o "reset" do acumulador. Deve-se observar que esta

restrição é apenas para economia de CIs e não uma restrição necessária ao projeto da arquitetura proposta.

5.10. PROCESSOS 06 e 07: CORREÇÃO DAS POSIÇÕES EM ERRO E CÁLCULO DO BER

A fig. 5.12. mostra o hardware necessário. Caso o peso analógico do vetor erro em questão seja o menor de todos, os DTi, i∈{1,...,4} e os PEj, j∈{1,...4} são copiados nos latches de acesso ao "corretor", conjunto de 8 comparadores de DT's e PE's com o valor da contagem módulo 12.

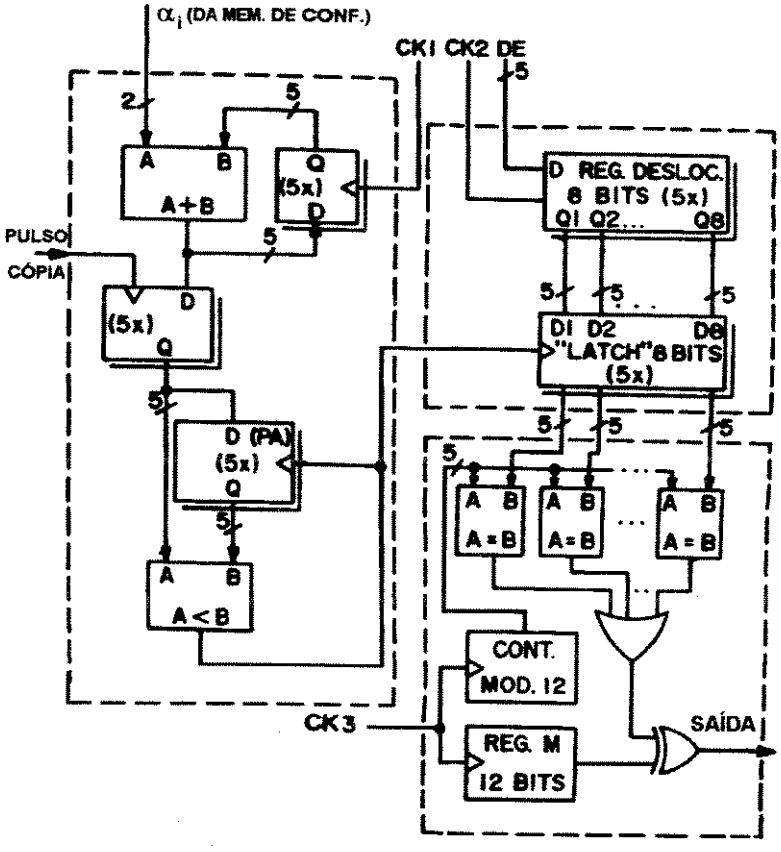


FIGURA 5.12. CORRETOR E CÁLCULO DE BER

O circuito que implementa estes processos, ilustrado na fig. 5.12, funciona da seguinte forma: à medida que os pesos analógicos (SF) são calculados, para cada padrão de erro sob teste, eles são então comparados ao menor peso analógico encontrado até o momento, previamente armazenado em um registrador. Ao mesmo tempo que o peso analógico é calculado, os endereços das posições em erro do padrão de erro sob teste são armazenados em outro registro, representados na fig. 5.12. pelo

conjunto de oito registradores de deslocamento de cinco bits. Este conjunto de endereços de posições em erro é obtido pela justaposição dos endereços de posições em erro fornecida pelo decodificador binário (decisão abrupta) com os endereços dos dígitos de teste, após a análise de coincidência de endereços.

Se o resultado da comparação indicar que o novo peso analógico é menor que o armazenado no registrador de menor peso analógico, o novo peso é armazenado neste registrador e o conjunto de endereços das posições em erro deste padrão de teste é copiado em um conjunto de registradores, que guardam o padrão de erros de menor peso analógico.

Ao final da varredura dos 16 padrões de teste, teremos então o padrão de erros mais provável, dado pelo conjunto de endereços das posições em erro de menor peso analógico.

Podemos agora, de posse do padrão de erros final, proceder à correção e ao cálculo da taxa de erro de entrada (i.e., taxa de erro do canal). O processo de correção é do tipo clássico: à medida que os bits de informação vão saindo do decodificador eles são invertidos através de um ou-exclusivo que recebe a linha de correção (que fornece nível lógico se não houver erro e 1, se houver erro). O processo de obtenção deste sinal de correção é que difere um pouco do usual: ao invés de usar os k dígitos (do padrão de erros) correspondentes aos k bits de informação, usamos um conjunto de comparadores onde cada um compara cada endereço de posição em erro com o endereço do bit que está saindo, dado por um contador módulo k sincronizado à cadência da seqüência de bits de informação que saem. Os comparadores fornecem 0 se não houver coincidência entre os endereços das posições em erro e 1 se houver coincidência. O contador de número de erros estimado no canal, usado para fornecer um alarme que indica a qualidade deste canal, também recebe como relógio um sinal que é resultado da comparação dos endereços de posições em erro com o endereço 0, que é, conforme convencionamos antes, na seção 5.7., um endereço "fantasma" ou "dummy" usado para indicar ausência de erros.

Uma restrição de projeto dada pela arquitetura neste circuito é o valor de carregamento inicial do registro de menor peso analógico. Este valor inicial deve ser maior que o maior peso analógico possível para um padrão de erros qualquer. Como trabalhamos com o padrão de erros, e não com o vetor de saída, e lembrando que o decodificador de Chase corrige até $(d-1)$ erros, há, no máximo, $(d-1)$ dígitos 1 em qualquer padrão de erros sob teste.

Portanto, esta restrição é:

$$\mathrm{PAI} > \alpha_{MAX} \cdot (d-1) \tag{5.19}$$

onde PAI é o peso analógico inicial do registro de menor peso analógico. Observamos que há necessidade de PAI ser estritamente maior e não maior ou igual ao produto da máxima confiabilidade de um dígito por $(d-1)$ porque o comparador de métricas que seleciona o menor peso precisa pulsar ao menos uma vez durante os 16

padrões de teste para que seja copiado no mínimo um conjunto de endereços de posições em erro.

Como restrição de projeto, usamos, conforme já comentado na seção 5.2.3., a restrição:

$$\mathrm{PAI} = \alpha_{MAX} \cdot d \tag{5.20}$$

No caso específico aqui descrito, este valor é 3x8 = 24. Este valor é carregado no registrador de menor peso analógico toda vez que começa o processamento de um novo bloco que entre no decodificador, e é feito de forma independente do sinal de atualização resultante da comparação de métricas.

O conjunto de registradores de deslocamento e registros de armazenamento dos endereços das posições em erro, do padrão de erros sob teste, envolve um número de CIs razoavelmente grande e foi usado aqui para ilustrar a execução do processo. Um número muito menor de CIs pode ser usado caso substituamos este conjunto de registradores por uma memória RAM integrada e excitadores de barramento, com um esquema de paginação adequado. O circuito final que implementa a arquitetura proposta lança mão desta técnica, que é descrita a seguir, na seção 5.12. Antes, porém, é útil apresentarmos o diagrama em blocos do decodificador como um todo, mostrando as interconexões entre os diversos blocos que realizam os processos do macro-processo de decodificação da fig. 5.3.

5.11. DIAGRAMA DE BLOCOS DO DECODIFICADOR

O hardware do decodificador, com as interconexões entre os diversos blocos que o compõem, está representado, sob a forma de diagrama de blocos, na fig. 5.13.

Este diagrama mostra, além dos blocos já analisados anteriormente, que são responsáveis pela execução dos processos mais importantes da decodificação, outro bloco – RAM 50x3 – que nada mais é que uma memória de dados e confiabilidades, responsável pelos processos restantes do macro-processo de decodificação: recepção e armazenamento das confiabilidades e dígitos da seqüência de recepção, e conversão de velocidade e transmissão dos bits de informação à saída. Esta memória de dígitos e confiabilidades também é paginada, para permitir um compartilhamento de processos.

A fig. 5.13. mostra o hardware específico da arquitetura proposta para o decodificador de Chase, usando o código de Golay estendido CGE (24,12), conforme se vê pelas dimensões das memórias. Entretanto, este diagrama em blocos serve para qualquer código de bloco binário, linear e sistemático; o que muda é apenas a dimensão das memórias e o número de "fios" nas interconexões entre blocos.

Os processos do macro-processo de decodificação estão divididos entre os diversos blocos de hardware de acordo com a descrição a seguir:

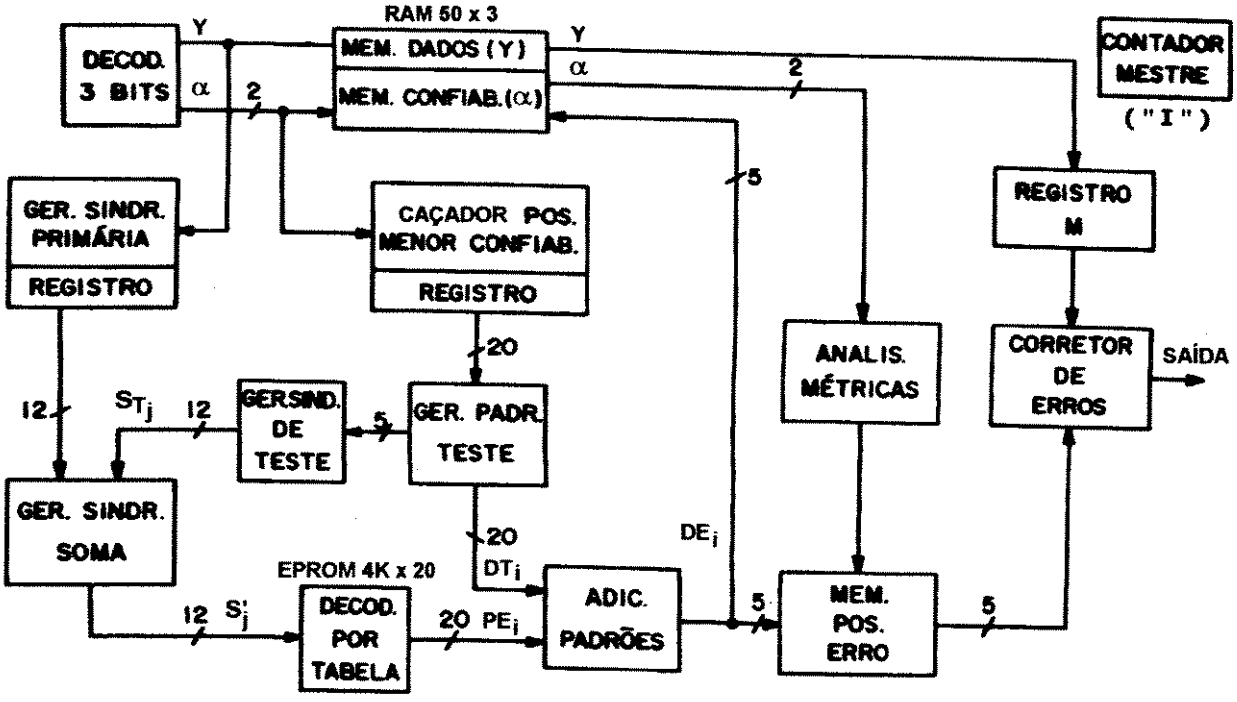


FIGURA 5.13. DIAGRAMA DE BLOCOS DO DECODIFICADOR PROPOSTO

a) memória de dados e confiabilidades - RAM 50x3: recepção e armazenamento das seqüências de dados e confiabilidades, e conversão de velocidade e transmissão dos bits de informação à saída; participa também do cálculo do peso analógico;

b) gerador de síndrome primária: cálculo da síndrome primária e armazenamento da mesma;

c) caçador de posições menos confiáveis: procura e armazenamento das 4 posições menos confiáveis do bloco a ser decodificado; existe, neste bloco, o contador mestre de posições (contador I), responsável pela geração de endereços (de 1 a 24) das posições de dígitos dentro do bloco, endereços estes que são usados por todos os outros blocos;

d) gerador de padrões de teste: obtenção e armazenamento dos 16 padrões de teste, a partir das 4 posições menos confiáveis;

e) gerador de síndromes de teste: processo de cálculo da síndrome de teste associada ao padrão de teste vigente, através da transformação linear de multiplicação pela matriz de verificação de paridade transposta;

f) somador de síndromes: processo de cálculo da síndrome de teste associada ao padrão de teste vigente e soma da síndrome primária com síndrome de teste;

g) decodificador binário (ou de decisão abrupta): tabela decodificadora gravada em EPROM, realiza o processo de decodificação por decisão abrupta através do fornecimento de um padrão de erros (com até 4 erros) endereçado pela síndrome-soma;

h) analisador de coincidência de endereços: "soma bit-a-bit" do padrão de erros decodificado binariamente com o padrão de teste vigente, através do apagamento de endereços de dígitos "1" comuns aos dois padrões; fornece o padrão de erros que deve ter seu peso analógico calculado;

i) analisador de métricas: cálculo do peso analógico e seleção do menor peso analógico entre os 16 calculados a cada bloco;

j) memória de posições em erro: armazenamento dos endereços das posições em erro do padrão de menor peso analógico;

l) corretor de erros: processo de enviar os bits de informação à saída, invertendo os que estão em erro, segundo os endereços das posições em erro do padrão de menor peso analógico;

m) medidor de taxa de erro: sinalizar, através de alarme, estimativas de taxas de erro altas no canal, ou seja, na entrada do decodificador; o limiar de taxa de erro é independente dos parâmetros do código e do decodificador, sendo variável de acordo com a qualidade de sinal desejada pelo usuário-receptor; já a taxa de erro no canal só depende da relação sinal/(ruído mais interferência) disponível;

n) conversor paralelo-série: interface entre o decodificador e o quantizador do demodulador QPSK.

5.12. TIMING E PAGINAÇÃO DE MEMÓRIAS

Conforme havíamos comentado nas seções 5.10 e 5.11, o processo de armazenamento dos endereços das posições em erro do padrão de erros de menor peso analógico é feito de forma mais econômica se usarmos uma memória RAM. Similarmente, o uso de memória RAM para armazenar dados e confiabilidades é mais racional e econômico que outras configurações, principalmente ao se trabalhar com os endereços das posições dos dígitos.

Esta seção mostra em detalhes estas soluções adotadas na arquitetura para o compartilhamento de memória entre os processos e os cuidados e restrições necessários ao bom funcionamento do decodificador.

Analisando o macro-processo de decodificação, fig. 5.3, notamos que os processos de armazenamento são necessários para evitar duplicação de circuitos, no caso de palavras-código em seqüência transmitidas através do canal, que é o que ocorre em cem por cento das aplicações práticas (seja essa seqüência de blocos finita, no caso de transmissão por surtos como o nosso, ou infinita, no caso de transmissão contínua). Entretanto, em alguns dos processos – aqueles que envolvem recepção e armazenamento ou armazenamento e transmissão – a duplicação é inevitável porque a cadência dos dados obedece a taxas de dígitos (ou de bits) fixas.

No caso da arquitetura proposta, o uso de técnicas de paginação emula a duplicação de circuitos através da divisão da área de memória disponível em partes iguais mas com funções distintas alternantes: enquanto uma está disponível para escrita de dados, a outra fica disponível para leitura de dados, invertendo-se as funções no bloco seguinte a ser decodificado. Além disso, o acesso para escrita ou leitura é controlado pelo nível lógico do relógio mais rápido do sistema, i.e., o controle do acesso depende se o relógio está ON = 1 ou OFF = 0, numa multiplexação temporal de funções.

Tanto na memória de posições em erro quanto na memória de dados e confiabilidades, a paginação principal é feita por intermédio do contador mestre de posições, que, ao indicar o final de um bloco (24° dígito em nosso caso), pulsa para um divisor por 2, de forma a fazer a saída deste divisor ficar em nível lógico estável (0 ou 1) durante o período de um bloco, alternando este nível no bloco seguinte e assim, alternando-se a cada bloco, durante todo o surto. Esta saída é ligada então a um endereço maior que $32 = 1F_H$ nas linhas de endereço da memória.

5.12.1. MEMÓRIA DE POSIÇÕES EM ERRO

A memória de posições em erro usa as duas páginas para compartimentalizar, em regiões distintas, os endereços das posições em erro que estão sendo gravados na memória – para serem usados na correção dos bits que sairão do decodificador, no bloco seguinte – e os endereços das posições em erro que estão sendo lidos da memória – para corrigir os bits que estão saindo do decodificador.

Além destas duas páginas, cada uma delas possui dois segmentos – ou sub-páginas – que são usadas para armazenar, de forma indistinta, os endereços das posições em erro que têm o menor peso analógico. O motivo da existência dos dois segmentos é dado abaixo, para o caso específico do CGE (24, 12), que foi implementado, mas o raciocínio vale e é facilmente estendido a outros casos.

Existem 16 padrões de teste diferentes, cada um podendo levar a até 16 padrões de erro associados diferentes. Para cada padrão de erros, o peso analógico ou métrica associada só é conhecido após 8 somas e acessos à memória de dados e confiabilidades. Portanto, a métrica do padrão de erros vigente só é conhecida ao final de cada seqüência de 8 endereços de posições de dígitos em erro e, para evitar registradores de deslocamento ou outros elementos de memória, exceto a RAM de erros, no bloco **"memória de posições em erro"**, à medida que cada acesso à memória de dados e confiabilidades é feito, para leitura da confiabilidade endereçada por um endereço de posição de dígito em erro, a ser usada no cálculo de métrica, este endereço é gravado como dado na memória de posições em erro, em um dos segmentos. Ao final de 8 acessos, o somador já tem o valor da métrica. Caso este valor seja maior ou igual àquele armazenado no registro de menor peso analógico, o comparador de métricas não pulsa e mantemos o mesmo segmento, de forma que os próximos 8 endereços de posições de dígitos em erro, associados a um novo padrão de teste, sejam gravados sobre os 8 endereços anteriores. Entretanto, caso o valor da

métrica calculada seja menor que o valor armazenado no registro de menor peso analógico, o comparador de métricas pulsa, trocando o valor do registro de menor peso analógico, e, simultaneamente, comandando uma troca de segmento de forma que, supondo que não haja mais métricas menores que esta até o fim dos 16 padrões de teste, teremos preservado um segmento com os oito endereços das posições em erro do padrão de menor peso, onde não houve acessos para escrita posteriores. Assim, toda vez que o comparador de métricas pulsar, ocorrerá uma troca de segmentos de forma a preservar, gravados, os endereços das posições em erro do padrão de menor peso.

Ao final dos testes sobre os 16 padrões, precisamos fazer a leitura destes endereços para o corretor atuar. Isto é facilmente feito, endereçando a memória na página anterior, no complemento do valor do último segmento onde houve escrita de dados.

Para que a troca de segmentos ocorra de maneira ordenada, apesar dos segmentos serem indistintos em seus formatos, todo o primeiro padrão de teste tem os endereços das posições de dígitos em erro associados gravados no primeiro segmento. Isto é feito forçando, através de um "reset", um flip-flop tipo T, que gera o segmento, a ir para nível lógico **0** toda vez que um novo bloco começa a ser decodificado. Antes de passarmos a um exemplo de funcionamento desta memória, vamos mostrar sua divisão estrutural, dada na fig. 5.14.

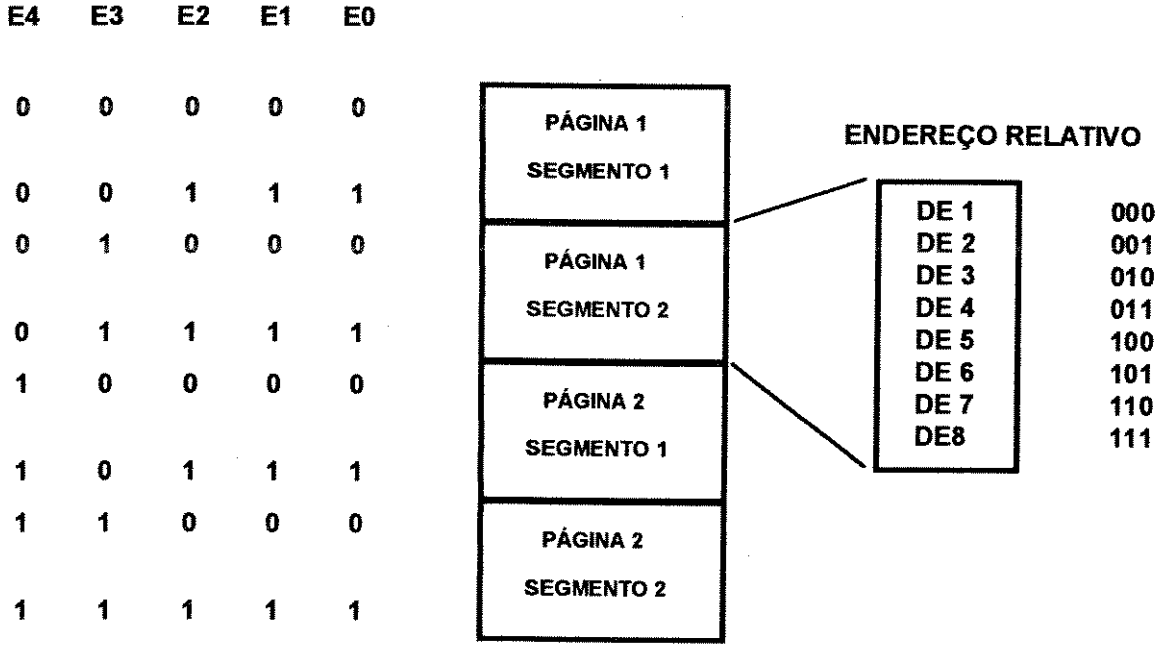


FIGURA 5.14 MAPEAMENTO DA MEMÓRIA DE POSIÇÕES EM ERRO

Vejamos alguns exemplos, supondo sempre que estamos analisando um bloco cuja página seja a página 1 para gravação.

Suponhamos a seguinte seqüência de pesos analógicos para os 16 padrões de teste: 4, 8, 4, 8, 10, 12, 10, 8, 10, 10, 12, 10, 8, 8, 10, 12. O comparador pulsaria apenas uma vez, no primeiro padrão, trocando o peso inicial (24) por 4. Os endereços do padrão de erros de menor peso ficariam gravados na PÁGINA 1 SEGMENTO 1 e o indicador de segmento estaria indicando SEGMENTO 2. No próximo bloco, a página de gravação será a 2 mas para a saída de dados do bloco anterior, que passa a ocorrer neste instante, usamos, na leitura dos endereços dos erros, a PÁGINA 1 e o complemento do bit que indica o último segmento onde houve gravação nesta página. Como o indicador de segmentos indica SEGMENTO 2, para a leitura dos endereços do padrão de menor peso, usamos SEGMENTO 1.

Um caso mais complexo é dado a seguir. Sendo a seqüência de pesos: 12, 8, 10, 10, 8, 4, 0, 4, 4, 8, 10, 10, 8, 12, 12, 16, o comparador pulsaria nos padrões de teste 1, 2, 6, 7, mudando de segmento sucessivamente, nestes padrões, até trocar de novo de SEGMENTO 2 para SEGMENTO 1 no final do padrão 7. Então, o SEGMENTO 2 tem armazenado os endereços das posições de dígitos em erro do padrão de menor peso.

A velocidade exigida de gravação na memória de posições em erro é a mesma que a leitura da memória de dados e confiabilidades para o processo de cálculo de pesos: 128 ciclos de escrita por bloco ou 5.696.000 ciclos de escrita por segundo. Como o compartilhamento entre escrita e leitura é multiplexado no tempo pelo relógio de 5.696 kHz (nível baixo indica ciclo de escrita, nível alto indica ciclo de leitura), para efeito de correção dos bits, o que se faz é uma varredura cíclica do segmento que tem o padrão de menor peso durante a saída dos doze bits de informação (que saem à taxa de 534 kbit/s). Assim, por bit de informação, realizamos a comparação de sua posição com as 8 posições do segmento de menor métrica, num total variável de 10 ou 11 comparações por bit (só 8 são efetivas; as outras 2 ou 3 são repetições), gerando um pulso de correção quando há coincidência entre a posição do bit e um dos endereços de posições em erro lidos do segmento de "menor métrica" da memória de posições em erro.

5.12.2. MEMÓRIA DE DADOS E CONFIABILIDADES

Esta memória, RAM (50x3), apesar de possuir uma paginação simples, tem um compartilhamento entre processos bastante complexo, já que estes possuem cadências diferentes.

O mapeamento de memória é dado na fig. 5.15.

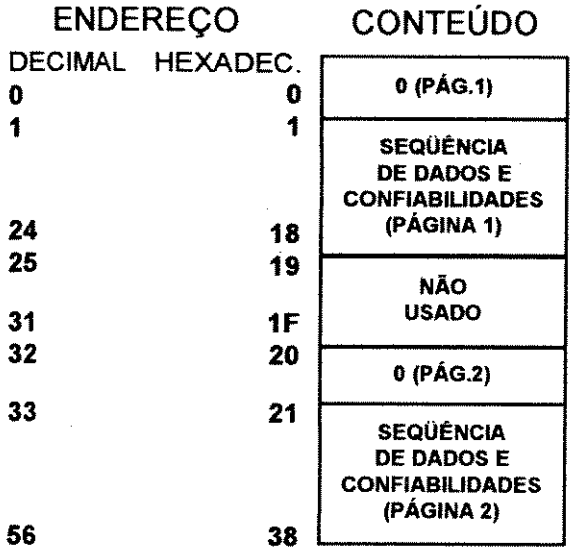


FIGURA 5.15. MAPEAMENTO DA MEMÓRIA DE DADOS E CONFIABILIDADES

O mapeamento acima reflete exatamente a descrição de hardware já feita: as posições dos dígitos da seqüência recebida assumem valores de 1 a 24, dadas por um contador mestre de posições (contador I da fig. 5.6.a), com saídas ligadas às linhas de endereço menos significativas da memória (através de isoladores "tri-state"); a página é dada por um flip-flop tipo T (divisor por 2), que alterna seu estado a cada bloco, e corresponde à linha de endereço mais significativa da memória e os endereços 0 de cada página têm gravada a confiabilidade "dummy" 0.

Esta constante "dummy", por razões de garantia de manutenção de seu valor, decidimos gravá-la a todo dígito que entre no decodificador. Esta gravação é, na realidade, um processo adicional àqueles que a memória deve suportar.

Então, a lista de processos que compartilham a memória de dados e confiabilidades é a seguinte:

a) recepção e armazenamento da seqüência de dados a ser decodificada e confiabilidades associadas, num total de 24 ciclos de escrita por bloco, a 1068 kbit/s;

b) gravação de 0 no endereço 0 da página usada, com 24 ciclos de escrita por bloco, a 1068 kbit/s;

c) leitura dos bits de informação a enviar à saída, com 12 ciclos de leitura por bloco, a 534 kbit/s;

d) leitura das confiabilidades para cálculo de peso analógico, com 128 ciclos de leitura por bloco, com relógio de 5.696 kHz.

Lembramos que todos estes processos são síncronos e devem estar sincronizados entre si, para garantir o não aparecimento de colisões ou "overlaps" de processos nos barramentos da memória. Isto é bastante difícil de se fazer, por causa da relação fracionária de fases entre os processos a), b), c) com o processo d).

Entretanto, é possível realizar estes sincronismos mútuos graças a uma multiplexação conveniente, conforme a descrita abaixo.

Em primeiro lugar, estabelecemos, arbitrariamente, que a leitura das confiabilidades seria realizada durante os intervalos de tempo em que o relógio de $(16/3)R_d$, 5.696 MHz, estivesse em nível lógico 1. O restante dos processos, que, em conjunto, realizam menos acessos à memória que o processo de leitura de confiabilidades, são, todos, realizados durante os intervalos de tempo em que o relógio de $(16/3)R_d$ estiver em nível lógico 0.

Para garantir a execução dos outros três processos, sem ocorrência de colisões, o período de um dígito do bloco a ser decodificado é dividido em quatro intervalos de tempo iguais, gerados por um contador módulo 4, excitando um decodificador, e seu relógio deve ser $4xR_d$.

A ocupação dos processos é dada a seguir:

a) intervalo 1 (contador no estado 00): escrita de 0 na posição 0 da página em uso;

b) intervalo 2 (contador no estado 01): leitura do bit de informação a ser enviado à saída, endereçado pelas posições de 1 a 12;

c) intervalo 3 (contador no estado 10): escrita dos dígitos e confiabilidades do bloco que vai ser decodificado, nos endereços relativos de 1 a 24 da página em uso;

d) intervalo 4 (contador no estado 11): não utilizado.

E estes processos só são ativados quando estão dentro dos intervalos respectivos acima e o relógio de 5.696 kHz está em nível lógico 0.

Esta técnica de multiplexação no tempo destes processos está perfeitamente casada com o "duty cycle" disponível para o relógio de $(16/3)R_d$, que é de 1/3, e permite, então, o uso de memórias RAM com tempos de acesso otimizados para maximizar suas eficiências de ocupação de tempo (ver ref. [2] application note AP-74) resultando então no menor custo possível para estas memórias, já que os tempos de acesso requeridos são os máximos possíveis para o sistema.

Devemos reparar que o "duty cycle" de 1/3, para o relógio de 5.696 kHz é muito conveniente para o sistema, porque esta freqüência é 16/3 da taxa básica Rd (1068 kbit/s). Pode então ser obtida por um PLL ("phase locked-loop") que gere 16 vezes a taxa (i.e., 17,088 MHz), passando esta saída por um divisor por 3. Como divisores por 3 têm normalmente "duty cycle" de 1/3 ou 2/3, escolhemos o de 1/3.

Para calcularmos o máximo tempo de acesso das memórias a serem utilizadas, sabemos que este deve ser igual ao t_{ON} do relógio de 5.696 kHz. Como t_{ON}/t_{OFF} = 1/3, teremos:

$$T_{\substack{ACESSO \\ RAM}} \leq 58{,}5 \text{ ns} \qquad (5.21)$$

Na prática, há ainda tempos de propagação nas lógicas de controle, capacitâncias e indutâncias de carga, tremor das formas de onda ("jitter" e/ou "skew") e até mesmo o tempo de trânsito nas trilhas de circuito impresso, de forma que o tempo de acesso não pode ser igual ou muito próximo ao valor máximo

disponível. Tipicamente, uma trilha de 13 cm de comprimento com carga de 40 pF, em placa de epoxy, dá um tempo de propagação de 4,5 ns. Lógicas de controle com apenas um nível de profundidade lógica, feitas com dispositivos TTL-LS ou HCMOS, como queremos, introduzem tempos de propagação entre 8 e 10 ns (ref. [2], application note AP-74).

Assim, por segurança de projeto, decidimos usar memórias com tempo de acesso garantido de 45 ns (RAMs funcionalmente equivalentes entre si são normalmente oferecidas em versões de 100, 70, 55, 45, 35 e 25 ns). Assim:

$$T_{\substack{ACESSO \\ RAM}} = 45 \text{ ns} \tag{5.22}$$

Desta análise feita para a memória de dados e confiabilidades, podemos concluir que também a memória de posições em erro necessita de um tempo de acesso de 45 ns.

A eficiência de ocupação de tempo por acesso nas memórias é de:

$$eficiência/_{acesso} = \frac{45}{58{,}5} = 77\% \tag{5.23}$$

Já a eficiência de ocupação de tempo total por bloco cai, para as duas memórias, por causa das restrições de sincronismo entre processos, mesmo tendo sido otimizadas.

$$eficiência_{\substack{RAM \\ de erros}} = \frac{T_{acesso} \cdot \dfrac{n^o \, de acessos}{bloco}}{T(1\,bloco)} = \frac{45\text{ns} \times 256}{24/1068\text{kHz}} = 52\% \tag{5.24}$$

$$eficiência_{\substack{RAM \\ de erros \\ e confiabilidades}} = \frac{45\text{ns} \times (12 + 24 + 24 + 128)}{24/1068\text{kHz}} = 38\% \tag{5.25}$$

Todas as idéias desta seção podem ser facilmente estendidas a outros códigos de bloco lineares e sistemáticos, exceto estes últimos valores associados a tempos de acesso. Basta que, no lugar do relógio de $(16/3)R_d$, raciocinemos com o relógio dado pela equação (5.18).

Maiores detalhes sobre o conceito de eficiência de ocupação de tempo de RAMs são dados na ref. [2], no application note AP-74, "High Speed Memory System Design Using 2147H", de Joe Altnether.

CAPÍTULO 6

RESULTADOS

6.1. INTRODUÇÃO

Abordaremos neste capítulo os resultados obtidos com a arquitetura proposta para o decodificador por decisão suave usando o algoritmo dois de Chase para o código de Golay estendido (24, 12).

Iniciamos com a descrição detalhada do projeto de cada bloco do hardware que implementa a arquitetura. Em seguida, forneceremos os dados referentes a número total de componentes, e análise de complexidade do hardware. Finalmente, apresentaremos os resultados dos testes obtidos em diversas placas com esse mesmo hardware.

O projeto do hardware obedeceu às seguintes fases:

a) concepção da arquitetura para execução do algoritmo, descrita no capítulo 5;

b) projeto (lógico e elétrico) do hardware, que implementa a arquitetura, voltado à minimização do número de CIs;

c) construção de um protótipo em placa com conexões elétricas do tipo "wire-wrap", para depuração e validação do projeto;

d) projeto de placa de circuito impresso, com "lay-out" visando minimizar os problemas decorrentes de interferências, diafonias e reflexões elétricas potencialmente perigosas;

e) construção de um número razoável de placas de circuito impresso, para verificação de reprodutibilidade e outros problemas associados à fabricação, bem como os testes do sistema AMDT. Em nosso caso, foram construídas dez placas reproduzindo o hardware da placa "wire-wrap" (protótipo).

6.2. DESCRIÇÃO DO PROJETO DE HARDWARE

Como já foi dito antes, a tecnologia que escolhemos para a implementação do hardware é a lógica HCMOS, excetuando alguns poucos dispositivos, que são os

seguintes: RAMs (N-MOS de alta velocidade), PROMs, alguns comparadores de magnitude (TTL Schottky) e alguns drivers de memória (TTL Low Schottky). A razão de lógica diferente para as RAMs e PROMs é a inexistência, até o momento, de dispositivos C-MOS equivalentes com o desempenho ou confiabilidade requeridos pela aplicação. Nos outros casos, a mudança da lógica foi necessária por restrições de velocidade e tempos de propagação.

Entretanto, precisamos realçar que esta mistura de famílias lógicas não implica em translações de níveis porque todas as séries utilizadas são compatíveis entre si e com a família TTL em termos dos níveis elétricos correspondentes aos níveis lógicos 0 e 1. A única exceção é no caso de CIs TTL alimentando entradas HCMOS e TTL simultaneamente. Neste caso, temos para saída em nível lógico 1, V_{OH} (TTL) $\geq$ 2,4V e V_{IH} (HCMOS) $\geq$ 3,15V.

Portanto, dependendo do número de entradas TTL, é necessário o uso de um resistor de "pull-up" após a saída "totem-pole" TTL, de forma a manter V_{OH} (TTL) >> V_{IH} (HCMOS). Todavia, se a saída TTL só alimenta entradas N-MOS e HCMOS, este resistor é perfeitamente dispensável porque sempre teremos, na prática, V_{OH} (TTL) $\geq$ 3,15V (a corrente de entrada de um dispositivo HCMOS é tão ínfima, que o uso deste resistor só seria mandatório para mais de 50 entradas HCMOS ligadas a uma saída TTL). Maiores detalhes encontram-se na ref. [5].

Outra restrição de projeto, ao usarmos dispositivos HCMOS, é a necessidade de redes resistivas de "pull-up" em barramentos "tri-state" que alimentem entradas HCMOS, porque, dada a altíssima impedância de entrada dos HCMOS, nos intervalos de tempo em que o barramento estivesse em estado de alta impedância, os dispositivos HCMOS com entradas em aberto tenderiam a oscilar, levando os circuitos influenciados por eles a estados indesejáveis ou mesmo a situações de "latch-up".

O projeto lógico está intimamente ligado à concepção da arquitetura (já descrita no cap. 5), e, para minimizar o número de CIs, adotamos o procedimento de projetar cada bloco da arquitetura separadamente e, em seguida, analisar as possibilidades de redução do número de CIs na interface entre os blocos, de forma a integrá-los o mais fortemente possível, aumentando as interdependências mútuas.

6.2.1. DESCRIÇÃO DO PROJETO LÓGICO

Analisamos aqui a composição lógica de cada bloco constituinte da arquitetura proposta, a saber:

a) interfaces de entrada e saída do decodificador;
b) contador mestre de posições;
c) conversor paralelo-série;
d) gerador de síndrome primária;
e) caçador de posições menos confiáveis;
f) memória de dados e confiabilidades;
g) gerador de padrões de teste;
h) gerador de síndromes de teste;
i) somador de síndromes;
j) tabela decodificadora;
k) analisador de coincidência de endereços;
l) analisador de métricas;
m) memória de posições em erro;
n) corretor de erros;
o) medidor de taxa de erro.

Na divisão acima, incluímos as interfaces de entrada e saída, que não estavam no diagrama em blocos do cap. 5, e separamos o contador mestre de posições do bloco caçador de posições menos confiáveis, por causa de sua importância na geração de endereços. É o contador I do bloco caçador de posições menos confiáveis apresentado no cap. 5.

Separamos também o medidor de taxa de erros do corretor de erros para facilitar a descrição a seguir. De forma geral, todos estes blocos já foram apresentados no capítulo 5, onde se encontram figuras mostrando suas composições em sub-blocos.

6.2.1.1. INTERFACES DE ENTRADA E SAÍDA DO DECODIFICADOR

Realizam a troca de sinais de entrada e saída do "hardware" do decodificador com outras partes do sistema, localizadas em outras placas.

A interface de entrada recebe todos os relógios necessários (5696 kHz, 4272 kHz, 1068 kHz, 534 kHz), habilitação do decodificador e os canais P e Q quantizados em 3 bits e com ambigüidade de fase de demodulação já resolvida (num total de 6 linhas: 2 de polaridade e 4 de confiabilidade). É composta por 2 CIs receptores de linha, cada um contendo 6 portas inversoras com Schmitt-trigger na entrada, tendo tripla função: isoladora, para não carregar cada sinal enviado ao decodificador com mais de uma entrada, regeneradora de sinais, para melhorar as formas de onda e filtrar reflexões, picos induzidos ("spikes") e diafonia nos sinais, e inversora. A cada porta de entrada é associado um resistor de "pull-up" ou "pull-down" de 1 kΩ, com as funções de garantir níveis lógicos adequados nas entradas das portas HCMOS

quando não existir excitação das outras placas e também diminuir a impedância de carga que as linhas de entrada "enxergam". Além disso, foi necessária a inclusão de um pequeno capacitor acelerador na linha de 5696 kHz, para melhorar a forma de onda.

A interface de saída é formada por um conjunto de 3 portas inversoras com alta capacidade de corrente ("drivers"), que enviam os sinais de bits de informação decodificados (invertidos) e alarme (não-invertido). Estas portas fazem parte de um CI com 6 drivers inversores.

6.2.1.2. CONTADOR MESTRE DE POSIÇÕES

Este bloco, originalmente representado como contador I do caçador de posições menos confiáveis (fig. 5.6), é um contador síncrono de 5 bits, que muda de estado na borda de subida do relógio de dígitos (1068 kHz), e que varia sua contagem entre os valores 1 (00001) e 24 (11000), ciclicamente a partir da habilitação do decodificador. Enumera (ou endereça) as posições dos dígitos de um bloco, à taxa de 1068 kbit/s, através de suas saídas e gera, adicionalmente, pulso indicando o 24° dígito (final de um bloco), pulso de cópia de síndrome primária, pulso de apagamento dos registradores de síndrome e endereços de posições menos confiáveis e onda quadrada indicativa de bloco par (nível 1) ou ímpar (nível 0) entrando no decodificador. Esta onda quadrada serve para a geração de paginação nos blocos de memórias de dados e confiabilidades e de endereços das posições em erro.

Dois CIs, contadores síncronos de 4 bits, em cascata, são o núcleo deste bloco, gerando, em suas saídas, contagens de 1 a 24. O estado "24" é detectado por uma porta NAND, cuja saída, NDIG24, força carregamento síncrono, nos contadores, do valor 1. Entretanto, a inicialização do circuito é feita quando o sinal de habilitação muda de nível, através de uma porta ou-exclusivo cujas entradas estão ligadas às saídas de integradores RC, com constantes de tempo diferentes, por sua vez ligados ao sinal de habilitação. Assim, quando o sinal de habilitação muda de nível 0 para 1, o ou-exclusivo produz um pulso estreito e de nível 1; passando este pulso por uma porta inversora, geramos o pulso de carregamento inicial. Este pulso é então combinado com a saída do detector de estado "24" através de um AND, formando o sinal de carregamento do valor 1 nos contadores. Este mesmo sinal, combinado com um pulso de nível 0 bastante estreito, gerado pelo relógio de 4272 MHz, dá origem ao pulso de apagamento, NLIMR, do registrador de síndrome e dos registradores de posições menos confiáveis, que também serve para inicializar os registros de confiabilidades associadas às posições menos confiáveis com o valor de confiabilidade máxima de um dígito, i.e., 3 (11).

Finalmente, o pulso de detecção do estado "24" comanda um flip-flop D (ligado como flip-flop T) para a geração da onda quadrada, indicativa de bloco, que fica em nível 0 durante os 1°, 3°, 5°,... blocos e em nível 1 para os 2°, 4°, 6°,... blocos de dígitos que entram para ser decodificados. Através de outra combinação de portas, é gerado um sinal estreito para a cópia de síndromes e endereços de posições menos

confiáveis, após a 24° borda de subida de relógio de 1068 kHz mas antes do pulso de apagamento NLIMR. Chamaremos este sinal de NCOPIA.

Às 5 saídas dos contadores e à saída do flip-flop T damos os nomes PEMP (0-5), endereços de escrita na memória de pesos.

6.2.1.3. CONVERSOR PARALELO-SÉRIE

É meramente um CI, multiplexador quádruplo de 2 linhas para 1 linha, onde 3 multiplexadores realizam a conversão paralelo-série dos canais P e Q, previamente demodulados, regenerados, quantizados e sem ambigüidades de fase, em 3 linhas seriais: de dígitos de polaridade, PY, e 2 linhas de confiabilidade, PPA0 e PPA1, de forma a restaurar a correta ordenação dos dígitos dentro do bloco, conforme explicado no item 1.3.2.4. A velocidade de entrada é 534 kbit/s e de saída 1068 kbit/s.

6.2.1.4. GERADOR DE SÍNDROME PRIMÁRIA

Este bloco consiste em um registro de síndrome comum, ([33], [14]), idêntico àquele que seria usado em uma decodificação por decisão abrupta de um código de bloco cíclico (i.e., um circuito que realiza a divisão pelo polinômio gerador $g(x)$). Veja fig. 5.5.b.

É composto por um registrador de deslocamento com realimentações via ou-exclusivos, formado por 2 CIs do tipo flip-flop D sêxtuplo e 6 portas ou-exclusivo, onde uma das células flip-flop é apenas um atrasador de um dígito e as outras 11 compõem o registro que divide pelo $g(x)$ do CG (23, 12). Há ainda uma porta AND, que barra a entrada do 24° dígito, e um flip-flop D realimentado através de um ou-exclusivo para calcular o dígito de paridade total, i.e., o último dígito da síndrome. As 12 linhas que compõem a síndrome são copiadas, antes do pulso de apagamento (NLIMR) do registro de síndrome, em um registro de armazenamento formado por 2 flip-flops D tri-state sêxtuplos, ao final do 24° dígito, através do pulso de cópia (NCOPIA). Evitamos assim, o uso de 2 registradores de síndrome.

6.2.1.5. CAÇADOR DE POSIÇÕES MENOS CONFIÁVEIS

Este bloco foi implementado como na fig. 5.6, exceto pelo contador I, já descrito no item 6.2.1.2.

É constituído por 4 flip-flops D sêxtuplos, registros dos 4 endereços menos confiáveis selecionados até um instante qualquer; 4 flip-flops D duplos, para armazenar as 4 menores confiabilidades, associadas aos endereços, selecionadas até um instante qualquer; 3 multiplexadores, para o roteamento da maior confiabilidade entre as 4 menores, selecionadas até um instante qualquer, que será comparada à confiabilidade que está chegando; 4 comparadores de 4 bits, para realizar as comparações que comandam os multiplexadores; 1 demultiplexador de 3 para 8 linhas, para seleção do relógio de atualização, operando em conjunto com 2 CIs OR de 4 portas, de 2 entradas cada; lógica de seleção composta por 3 portas NAND e 1

inversora, para gerar corretamente o endereço fornecido ao demultiplexador; por último, um conjunto de 3 flip-flops D octais, para armazenar, após o 24° dígito, os 4 endereços de posições menos confiáveis, liberando o resto do circuito para atuar no bloco seguinte.

6.2.1.6. MEMÓRIA DE DADOS E CONFIABILIDADES

Devido às técnicas de paginação e de compartilhamento dos processos, já descritas no capítulo 5, o circuito deste bloco ficou razoavelmente pequeno, mas com operação bastante complexa.

É composto por:

a) um CI de memória RAM 1Kx4, de 45 ns, em tecnologia N-MOS;

b) um excitador/isolador 3-estados, sêxtuplo, de endereços de escrita de dados e confiabilidades na memória;

c) um excitador/isolador 3-estados, sêxtuplo, de dados e confiabilidades a serem escritos na memória, enviados pelo conversor paralelo-série;

d) um excitador/isolador 3-estados, octal, que força escrita de 0 no endereço 0 ou 32 (20_H),em função da página disponível para escrita;

e) um excitador/isolador 3-estados, sêxtuplo, de endereços de leitura de bits de informação que carregarão o registrador de saída;

f) um excitador/isolador 3-estados, sêxtuplo, de endereços de leitura de confiabilidades a serem somadas para um padrão de teste, em tecnologia TTL-LS por razões de tempos de propagação associados.

Possui ainda um conjunto de portas AND, OR, NAND e inversores que fornecem os sinais de comando ao conjunto de excitadores de barramentos e memória, a partir de um contador de 4 bits cujo relógio é 4272 kHz. Redes resistivas de "pull-up", em filme espesso, são associados aos barramentos (usamos 3 delas).

6.2.1.7. GERADOR DE PADRÕES DE TESTE

Gera seqüencialmente os 16 padrões de teste, a partir dos 4 endereços de posições menos confiáveis armazenados nos "latches" octais do caçador de posições menos confiáveis. O circuito é basicamente o mesmo da fig. 5.7, constituído por: um CI contador módulo 16 (4 bits), que recebe um relógio de 712 kHz, para gerar os bits de "mascaramento" das posições menos confiáveis; um conjunto de 5 CIs de 4 portas AND de 2 entradas; um conjunto de registradores de armazenamento dos vetores de teste gerados, até o fim do processo de cálculo do peso analógico associado a cada um, composto por 2 CIs flip-flops D octais e 1 CI flip-flop D sêxtuplo.

6.2.1.8. GERADOR DE SÍNDROMES DE TESTE E SOMADOR DE SÍNDROMES

Estes dois blocos são implementados em um circuito que utiliza um barramento 3-estados em comum (para os dois blocos), conforme já foi exposto na seção 5.7.

O circuito consiste em:

a) uma memória composta por 2 PROMs 32x8, de 25 ns, em tecnologia TTL Schottky, que contém a matriz de verificação de paridade transposta acrescida de uma linha toda nula, de acordo com a tabela da fig. 5.9;

b) três multiplexadores duplos de 4-para-1 linha, responsáveis pelo seqüenciamento dos 4 dígitos de teste que serão usados como endereço das PROMs;

c) um contador de 4 bits associado a uma lógica de portas que, a partir do relógio de 5696 kHz, geram sinais de comando vitais para o sincronismo deste bloco e dos blocos restantes. Estes sinais são: um sinal de relógio intermitente (PSSP) de 5 bordas de subida e periódico a cada 8 períodos do relógio de 5.696 kHz e um sinal de habilitação das PROMs (PCOST) por 4 das 5 bordas de PSSP. Há ainda outros sinais de controle de menor importância.

d) um registro formado por 2 CIs, com 6 flip-flops D cada um, usados para guardar o resultado da soma bit-a-bit a cada borda de PSSP;

e) doze portas ou-exclusivo que formam o somador bit-a-bit das colunas da matriz guardada nas PROMs ou da síndrome primária com o resultado acumulado no registro descrito no item anterior.

A soma é feita em 5 passos. Ao final das primeiras 4 bordas de subida de PSSP obtém-se a síndrome de teste nas saídas do registro acumulador, que é somada na quinta borda de subida de PSSP à síndrome primária. Este valor final será usado como endereço da tabela decodificadora.

6.2.1.9. TABELA DECODIFICADORA

É o bloco que realiza o processo de decodificação do padrão de erros associado à síndrome-soma, i.e., é o decodificador binário (decisão abrupta) exigido pelo algoritmo de Chase. É composto por 3 CIs de memória EPROM 8Kx8 de 200 ns, em tecnologia N-MOS, que guardam uma tabela de 4096 posições com 20 saídas. Têm como entrada os endereços dados pela síndrome-soma e como saída os (até) 4 endereços das posições em erro no vetor recebido perturbado pelo vetor-teste. Seis redes resistivas de "pull-up" em filme espesso são associadas ao barramento de saída. As saídas das EPROMs estão válidas 200 ns após PCOST = 1 e haver ocorrido a quinta borda de subida de PSSP, até PCOST = 0.

6.2.1.10. ANALISADOR DE COINCIDÊNCIA DE ENDEREÇOS

Este bloco realiza a soma bit-a-bit do padrão de erros decodificado com o padrão de teste vigente através da justaposição das posições em erro decodificadas com as posições dos dígitos de teste, suprimindo-se aqueles endereços que apareçam simultaneamente nos dois conjuntos. Assim, os endereços resultantes desta

justaposição indicarão quais confiabilidades armazenadas serão usadas para o cálculo de peso analógico associado ao padrão vigente.

O circuito implementado baseia-se em um barramento e compõe-se de: 4 registros octais 3-estados para armazenar as (até) 4 posições em erro decodificadas que serão multiplexadas no tempo no barramento de justaposição; 4 isoladores sêxtuplos 3-estados para multiplexar, junto com as posições em erro, os (até) 4 dígitos de menor confiabilidade do padrão de teste vigente; um contador módulo 8 e um decodificador 3-para-8 linhas para liberar, seqüencialmente, no barramento de justaposição, as 4 posições em erro e os 4 dígitos de teste; 4 comparadores de igualdade de 8 bits, para fazerem a análise de coincidência entre posições em erro e dígitos de teste; 2 flip-flops duplos, tipo D, e 4 portas OR, para apagar os dígitos de teste coincidentes com qualquer posição em erro, e, finalmente, uma lógica de 11 portas AND responsáveis pelo apagamento das posições em erro e dígitos de teste coincidentes. O barramento de justaposição também possui uma rede resistiva de "pull-up" em filme espesso.

6.2.1.11. ANALISADOR DE MÉTRICAS

O analisador de métricas é o bloco responsável pelo cálculo dos 16 pesos analógicos (correspondentes aos 16 padrões de erro obtidos) e pela seleção do menor destes pesos.

Compõe-se de:

a) um somador binário completo de 4 bits e uma porta ou-exclusivo, responsáveis pela soma da parcela de peso analógico acumulada (em somas anteriores) com o valor de confiabilidade lido da memória de dados e confiabilidades, quando esta tiver como endereço as posições em erro ou dígitos de teste que não coincidiram no barramento de justaposição do analisador de coincidência de endereços;

b) um registrador sêxtuplo que funciona como acumulador de parcelas da soma de confiabilidades, e que recebe pulso de apagamento ("reset") ao final de cada vetor teste;

c) dois isoladores sêxtuplos, sendo um responsável pela inicialização forçada do comparador de métricas, carregando o valor 24 (11000 binário) como métrica inicial de cada bloco no registrador de armazenamento de menor métrica, e o outro sendo responsável pela liberação das saídas do somador para comparação somente nos instantes adequados, i.e., após o término de cada soma que fornece o peso analógico associado a cada vetor-teste;

d) dois comparadores de 4 bits ligados em cascata para realizar as comparações sucessivas dos pesos analógicos (métricas) calculadas até que se encontre o menor entre eles;

e) um registrador sêxtuplo de armazenamento da menor métrica entre as 16 associadas aos padrões de teste de um bloco;

f) lógica de seleção composta por uma porta NAND de 2 entradas e uma porta ou-exclusivo de 2 entradas, que gera o pulso de cópia do valor de uma métrica

(associada a um vetor-teste) no registrador de armazenamento, caso os comparadores indiquem que esta métrica é menor que aquela já armazenada no registrador. Resistores de "pull-up" em filme espesso são usados nos barramentos de comparação e de soma e ajudam nas interfaces entre lógica TTL-Schottky e HCMOS, pois os comparadores e o somador precisam ser TTL-Schottky para atender aos requisitos de velocidade.

6.2.1.12. MEMÓRIA DE POSIÇÕES EM ERRO

Este bloco armazena as posições em erro do padrão de menor peso analógico (métrica), selecionado no analisador de métricas.

É constituído por:

a) dois CIs de memória RAM 1Kx4, de 45 ns, em tecnologia NMOS, compondo uma memória de 32 palavras de 5 bits, responsáveis pelo armazenamento dos 4 padrões de erro e 4 dígitos de teste associados a cada um dos 16 padrões de teste. O armazenamento é realizado em 2 páginas divididas em 2 segmentos cada, que se alternam a cada bloco recebido, conforme mecanismo já explicado na seção 5.12.1;

b) um isolador sêxtuplo 3-estados e um contador de 4 bits que, juntos, fornecem o endereço de escrita de cada uma das posições em erro na memória de erros;

c) um isolador sêxtuplo 3-estados e um contador de 4 bits que, juntos fornecem o endereço de leitura das posições em erro correspondentes ao padrão de menor métrica previamente selecionado;

d) um isolador sêxtuplo 3-estados para fornecer os conjuntos de posições em erro à memória, via barramento de dados, na fase de escrita;

e) dois flip-flops tipo D, duplos, responsáveis por dar a paginação e segmentação corretas da memória de erros, tanto na fase de escrita quanto na de leitura.

Três redes resistivas de "pull-up" em filme espesso estão ligadas aos barramentos de dados e endereços da memória de posições em erro.

6.2.1.13. CORRETOR DE ERROS

Além da função óbvia de corrigir erros, este bloco também realiza a transferência dos bits de informação (armazenados na memória de confiabilidades e dados) para a saída, convertendo a velocidade de transferência de entrada (1068 kbit/s) para a de saída (534 kbit/s).

É composto por:

a) um registrador de deslocamento de 12 bits, composto por dois CIs de 8 bits em cascata, de entrada série e saídas paralelas, operando a 1068 kbit/s, que recebe os 12 bits de informação da memória de dados e confiabilidades;

b) um registrador de deslocamento de 12 bits, também composto por dois CIs de 8 bits em cascata, de entradas paralelas e saída série, operando a 534 kbit/s, que transfere os bits de informação à saída;

c) um conformador de pulsos de relógio para gerar bordas de subida, a 1068 kHz, em fase com os dados lidos da memória de dados e confiabilidades;

d) um contador auxiliar de posições, de 4 bits, que gera endereços de "1" a "12", à taxa de 534 kbit/s e em fase com o processo de deslocamento do bloco para a saída.

e) um comparador de igualdade de 8 bits para comparar as 8 posições em erro, lidas no segmento da memória de posições em erro correspondente à menor métrica, com o endereço do bit que está sendo transferido para a saída. Este comparador é habilitado só na fase de leitura da memória de posições em erro;

f) um flip-flop tipo D que funciona como registrador de erro. Este flip-flop recebe um pulso de "reset" a cada bit que sai do decodificador, indo a "1" e registrando o erro toda vez que o comparador pulsa;

g) uma porta ou-exclusivo, que realiza efetivamente a correção nos bits;

h) um flip-flop tipo D atuando como "latch", para copiar o bit já corrigido antes de transferi-lo para a saída;

i) um registrador sêxtuplo para colocar nas fases corretas os endereços de bits e o bit a corrigir.

6.2.1.14. MEDIDOR DE TAXA DE ERRO

O medidor de taxa de erro calcula a taxa de erro média de bit (bit error rate ou BER), à entrada do decodificador, sinalizando, por meio de um alarme, se este BER de entrada excedeu ou não um determinado limiar previamente estabelecido de acordo com o objetivo de desempenho do sistema onde o decodificador irá funcionar. Este objetivo de desempenho é normalmente o BER médio de saída, que tem que ser menor que um limitante máximo aceito pelo usuário do sistema.

Como este decodificador opera em surtos – ver cap. 1 – o medidor de taxa de erro é muito mais complexo do que em uma transmissão contínua. Isto decorre do fato de que, para se fazer uma medida de taxa de erro com um grau razoável de precisão, é necessário se observar uma população muito grande de bits, conforme Kerczewski, Daugherty e Kramarchuk [22] mostram em seu trabalho. Baseado em seus resultados, mostraremos ao final desta seção qual é o intervalo de confiança da medida feita por este medidor, cuja composição passamos a descrever:

a) um contador módulo 256 síncrono, composto por 2 CIs contadores de 4 bits, cuja função é contar o número de erros de bits de informação, detectados pelo decodificador, dentro de um determinado intervalo de tempo igual ao número de bits que constituem a população total observada vezes o período de um bit. O relógio deste contador de erros recebe, portanto, os próprios pulsos de correção de bits, vindos do corretor de erros. Este contador de erros é apagado (i.e., recebe um "reset") ao final de cada período de observação. Ele tem ainda uma auto-desabilitação dada pelo pulso de transbordo ("vai-um" ou "ripple carry") que o inibe ao contar 256 erros. Além disso, recebe uma linha de habilitação que só o deixa

contar passado o tempo de atraso de decodificação após a habilitação do decodificador (26 bits de informação);

b) um temporizador, composto por 4 contadores assíncronos de 4 bits, que conta um número inteiro (potência de 2) de períodos de relógio de bit, somente durante a parte da duração dos surtos que contenha dados válidos, ou seja, compensando o tempo de atraso de decodificação. Assim, este temporizador recebe como relógio o próprio relógio de bit (534 kHz em nosso caso) convenientemente controlado por um sinal que fornece a janela temporal onde os dados decodificados de um surto estão válidos. Além disso, este temporizador, que atua como um contador da população total de bits observados, recebe também um "auto-reset" vindo da saída do último estágio de contagem;

c) um temporizador de atraso, composto por um contador assíncrono de 4 bits, uma lógica OR de diodos e uma porta AND para "modular" o relógio de bit com a janela temporal de dados válidos, fornece o relógio para o temporizador de população total de bits (descrito em b) acima). A janela de dados válidos é gerada através do contador de forma a se iniciar após os primeiros 36 bits depois que o decodificador foi habilitado, terminando quando o decodificador é desabilitado. Assim, a janela começa 10 bits válidos após o atraso de decodificação e termina junto com a desabilitação de entrada do decodificador, ou seja, 26 bits de saída antes do surto terminar. Esta janela foi gerada desta forma para economizar hardware, já que a geração de uma janela de observação que cobrisse todos os bits válidos exigiria circuitos mais complexos que o implementado. Entretanto, para um canal bem comportado, com ruído aditivo branco e gaussiano apenas, o BER médio não depende da posição (ou fase) das janelas observadas, mas apenas do tamanho da população de bits observados, porque o sistema pode ser considerado estacionário no sentido amplo [25];

d) Um registrador, flip-flop tipo D, da condição de alarme medida na última temporização da população de bits observados. Este flip-flop recebe em sua entrada D a linha de transbordo ("ripple carry") do contador de erros e, como relógio, o sinal de saída do último estágio de contagem do temporizador de população de bits observados;

e) Um indicador visual de alarme formado por um diodo emissor de luz em série com um resistor limitador de corrente ligados entre a saída Q do registro de alarme e terra de sinal;

f) Uma rede de controle das condições de reset de alarme formada por 4 diodos e um resistor de "pull-up" fazendo lógica AND, onde cada diodo está ligado respectivamente a: um sinal de limpa ao ligar ("power-on reset") fornecido por uma rede RC; um sinal de reset assíncrono fornecido por um dispositivo de E/S (entrada/saída, ou I/O) da unidade processadora que contém a CPU controladora da estação onde o decodificador está inserido, para prover ao software de controle do sistema um meio de reset assíncrono deste alarme; um sinal temporizado por uma rede RC que desative o alarme, caso o sinal de habilitação do decodificador fique inativo por um período maior que um quadro do sistema AMDT, para dar cobertura

nos casos em que uma estação saia de serviço estando o alarme ativado e, finalmente, um quarto sinal que fornece um "reset" de alarme após 8.192 (8K) bits de informação. Este último sinal é necessário apenas porque a condição de alarme a ser fornecida ao processador da estação deve ser dada na forma presença de pulso (alarme ativo) / ausência de pulso (alarme inativo) e não na forma de níveis lógicos somente (p. ex., 1 para alarme ativo e 0 para alarme inativo).

No medidor em questão, o número total de bits observados pode ser programado, por meio de um "jumper", em 16.384 bits ou 32.768 bits permitindo, então, dois limiares para o alarme:

a) Período 16.384 bits: o alarme pulsa se BER for maior ou igual a 256/16.384, i.e., 1,5625E(-2);

b) Período 32.768 bits: o alarme pulsa se BER for maior ou igual a 256/32.768, i.e., 7,8125E(-3).

Finalmente, dado o princípio de funcionamento do medidor de BER, podemos agora analisar as imprecisões inerentes a esta técnica de medida da taxa de erro de entrada do decodificador, i.e., da taxa de erro do canal.

A primeira imprecisão diz respeito ao não-conhecimento da seqüência de bits originalmente transmitida que, a rigor, levaria à medida exata. Entretanto, para isto ser factível, deveríamos ter uma fonte de bits que produzisse uma seqüência de bits totalmente conhecida e dois canais com atrasos idênticos , sendo um deles o canal real (que produz erros) e o outro livre de erros. O resultado da comparação bit-a-bit entre as recepções dos dois canais forneceria o número de erros. Na prática, o canal sem erros pode ser substituído por um receptor auto-sincronizável em uma seqüência binária pseudo-aleatória utilizada como seqüência de entrada na transmissão, como é feito nos equipamentos de teste do tipo "medidor de BER" sugeridos no apêndice A. Obviamente, este tipo de medida exige que a fonte real de bits não esteja ligada ao codificador, inviabilizando, portanto, medidas em ambiente de operação (on-line) como a que fazemos com o medidor de BER interno ao decodificador. Assim, o uso dos pulsos de correção como sendo os erros reais que aconteceram no canal durante a transmissão é, na verdade, uma estimativa dos erros reais e pressupõe, portanto, que desprezemos a taxa de erro de saída do decodificador frente à taxa de erro de entrada. Indo um pouco além, os erros à saída do decodificador são de duas espécies: erros devidos a ultrapassagem da capacidade corretora mas passíveis de detecção e erros devidos à não-detecção. Assim, como a probabilidade de não-detecção é menor que a probabilidade de erro à saída do decodificador (porque, para qualquer código, uma não-detecção só ocorre se o número de erros for maior ou igual a d e a capacidade corretora é sempre menor que d), podemos assumir como limitante superior de não-detecção a taxa de erro à saída do decodificador. Desta forma, para este código CGE(24,12) e o decodificador baseado no algoritmo 2 de Chase, dada uma taxa de erro de 1E(-2) à entrada do decodificador, teremos uma taxa de erro menor que 1E(-6) à saída, se o canal for AWGN (conforme veremos a seguir nos resultados práticos obtidos). Então, a

imprecisão cometida por usarmos os pulsos de correção ao invés dos erros reais no medidor, seria de, no máximo:

$$ERRO(1) = 1 - \frac{1 \cdot 10^{2} - 1 \cdot 10^{-6}}{1 \cdot 10^{-4}} = 1 \cdot 10^{-4} = 0{,}01\% \qquad (6.1)$$

Esta imprecisão é, portanto, desprezível para todos os efeitos práticos.

A segunda imprecisão diz respeito ao fato de que, mesmo dispondo de uma contagem real dos erros, i.e., mesmo que os testes sejam feitos com uma fonte conhecida de bits e a contagem de erros seja baseada na comparação bit-a-bit das seqüências transmitida e recebida, a medida da taxa de erro seria apenas um estimador estatístico da taxa de erro verdadeira dado que a medida é feita dentro de um intervalo de tempo finito. Em termos estatísticos podemos dizer que a população de bits observada para a medida não é o espaço amostral total (este é infinito e a população finita). Assim, Kerczewski, Daugherty e Kramarchuk [22] mostram que para BER em torno de 1E(-2) ou menores, e para populações maiores que 1000 bits, a natureza binomial da contagem de erros pode ser aproximada por uma distribuição de Poisson e, levando o desvio padrão desta aproximação na desigualdade de Tchebychev, obtém-se a fórmula abaixo:

$$\text{Prob}\left[M < \frac{1}{(1-P)\cdot(BERV)\cdot\left(E^{2}\right)} \right] \geq P \qquad (6.2)$$

onde

M: tamanho da amostra, i.e., população de bits observados

$BERV$: BER verdadeiro

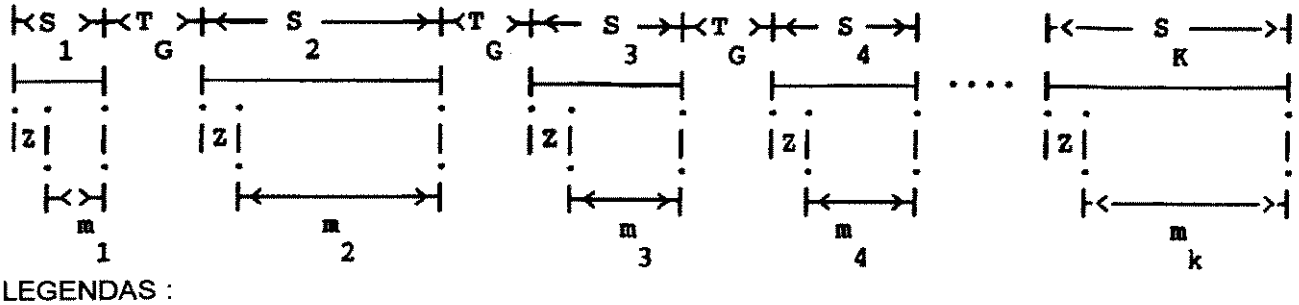
P: probabilidade do BER medido ($BERM$) estar com tolerância de $\pm E$ em relação ao $BERV$, para M bits observados

E: limitante do erro de medida valendo:

$$E = \frac{|BERV - BERM|}{BERV} \qquad (6.3)$$

A partir da equação (6.2), observamos que o medidor de taxa de erro fornece o alarme para a taxa de erro desejada com tolerância de 20% e com uma probabilidade de acerto maior que 90% em ambas as programações. Para tanto, basta resolvermos a desigualdade entre colchetes na equação (6.2) para P, fixando o parâmetro E em 20% e notando que o produto $M.BERV$ é, na verdade, o número de bits em erro medidos, que em nosso caso é fixo e vale 256.

A terceira imprecisão diz respeito ao fato do medidor acumular os erros somente dentro de subintervalos dentro dos surtos, sendo estes subintervalos não contíguos, por força de simplicidade exigida pelo hardware. A figura 6.1. a seguir esclarece o esquema de medida:



LEGENDAS :
tg - tempo de guarda entre os surtos do sistema AMDT
S1, S2, S3, S4 - tamanho dos surtos 1, 2, 3, 4, (em bits)
M1, M2, M3, M4 - tamanho dos subintervalos de medição de BER dentro dos surtos 1, 2, 3, 4, em bits, com mi < Si
Z - número de bits dentro de um surto, descartados da população observada; por construção do medidor, $Z = S_i - M_i$ vale 36 bits.

FIGURA 6.1. DISTRIBUIÇÃO DOS SUBINTERVALOS DE MEDIÇÃO DE BER DENTRO DOS SURTOS

M_k é tal que $M_1 + M_2 + ... + M_k = M$ bits com $M = 16.384$ ou 32.768 bits.

Supondo que, dentro de M_i ocorreram n_i erros, o medidor estima BER como sendo:

$$BERM = \frac{n_1 + n_2 + \cdots + n_k}{m_1 + m_2 + \cdots + m_k} = \frac{\sum_{i=1}^{k} n_i}{\sum_{i=1}^{k} m_i} \qquad (6.4)$$

desde que:

$$\sum_{i=1}^{k} m_i = M = 16.384 \text{ ou } 32.768 \text{ bits} \qquad (6.5)$$

Claramente, uma medida que tivesse observado a totalidade de bits em cada surto, se dentro de cada S_i ocorreram r_i erros, seria diferente da dada em (6.4) e valeria, na expressão genérica:

$$BER' = \frac{\sum_{1}^{k} r_i}{\sum_{1}^{k} S_i} \tag{6.6}$$

com

$$\sum_{i=1}^{k} S_i = M \tag{6.7}$$

Todavia, se o ruído que provoca os erros for um processo aleatório estacionário no sentido amplo, a sua média estatística não depende do instante de tempo onde foi observada [25]. Como numa distribuição binomial B (n,p) com parâmetros n (número de experiências) e p (probabilidade de sucessos) a média é [23]:

$$E\,[B\,(n, p)] = n.p \tag{6.8}$$

Se fixarmos n=M, sabendo que E [B (n,p)] é constante no tempo porque o processo é estacionário no sentido amplo, teremos que:

$$BER' = BERM = p \tag{6.9}$$

Então, se o ruído for aditivo, gaussiano e branco, o processo é estacionário [25], cap. 3) porque sua média é constante no tempo e é nula. Além disso, também a sua autocorrelação não depende dos instantes de tempo mas somente de sua diferença.

Assim, a terceira imprecisão do medidor é desprezível em um sistema via satélite operando em banda C (6 GHz/4 GHz) onde não há desvanecimentos e o ruído térmico predomina inclusive sobre interferências de outros sinais dentro e fora do transponder. Quão desprezível pode ser esta imprecisão é algo que só depende do grau de fidelidade do ruído real de sistema a um processo aleatório estacionário no sentido amplo, o que é extremamente difícil de se avaliar na prática.

Finalmente, ainda há uma quarta e última imprecisão associada à medida em si e que também pode ser desprezada. Ela se refere ao método de medida, que observa a taxa de erro de bit à entrada, quando o processo aleatório (que provoca os erros) atua sobre os dígitos demodulados. É esta taxa de erros de dígitos à entrada

do decodificador, junto com a distribuição de erros que limita o desempenho de sistema (taxa de erro de bit à saída do decodificador). Portanto, a rigor, deveríamos estar medindo a taxa de erro de dígitos à entrada do decodificador e não a taxa de erro de bit. Entretanto, lembrando que o código CGE (24, 12) utilizado é sistemático e lançando mão de um raciocínio idêntico ao usado na análise da terceira imprecisão, podemos deduzir que a taxa de erro de bit à entrada do decodificador tem o mesmo valor que a taxa de erro de dígito à entrada do decodificador no caso de ruídos que sejam processos aleatórios estacionários no sentido amplo e de demoduladores que, para cada erro de símbolo, produzam à saída apenas um erro de dígito (o que é trivial para BPSK e que, para QPSK, basta usar código de Gray ([12],[28] no modulador).

Como comentário final, gostaríamos de esclarecer que esta descrição e análise detalhadas do funcionamento do alarme foram feitas nesta seção de forma mais extensa que para outros blocos porque estes últimos já haviam sido analisados e descritos no capítulo 5 e aqui preferimos dar apenas suas composições em termos de CIs. Já no caso do medidor de taxa de erro, esta análise e descrição do bloco de forma detalhada eram pouco relevantes à compreensão da arquitetura proposta no capítulo 5. Entretanto, são relevantes na discussão da estrutura específica do hardware implementado para este bloco e, por isto, resolvemos incluí-las aqui.

6.3. COMPARAÇÃO ENTRE O HARDWARE IMPLEMENTADO E OUTRAS ALTERNATIVAS

Na presente seção, mostraremos que o hardware implementado, apesar de complexo, é o que otimiza a solução de espaço através da minimização do número de componentes, para um decodificador por decisão suave baseado no algoritmo 2 de Chase para o CGE (24, 12). Claro está que, ao assumirmos o algoritmo como sendo fixo, e ao mantermos os níveis de quantização fixos, estamos supondo um objetivo de desempenho quanto à taxa de erro que não variará, i.e., não admitimos, para efeito da análise feita aqui, uma degradação deste desempenho, já analisado nos capítulos 1 e 4.

Entretanto, esta análise é obrigatória para demonstrar que a placa seria inviável em termos de espaço para acomodar outras soluções, daí a necessidade dela ser complexa.

No capítulo 5, seção 5.2. provamos algumas relações que ajudam a minimizar o hardware, seja através da redução do número de parcelas envolvidas no cômputo do peso analógico, seja através da diminuição do número de linhas de sinal necessárias para a representação de vetores internamente à placa.

Entretanto, apesar destes resultados indicarem uma tendência de diminuição do hardware, não ficou provado que o número de CIs seria maior com qualquer outra solução. É isto o que tentaremos fazer aqui nesta seção, realçando novamente que esta não é uma prova que possa ser encarada como definitiva e estática no

tempo, porque o mercado de componentes é muito dinâmico e situações hoje vantajosas podem, no futuro, tornarem-se desvantajosas (em tamanho ou custo).

Inicialmente, analisemos a situação de uma implementação direta do algoritmo 2 de Chase, sem as simplificações adotadas que estão comentadas no capítulo 5, em particular nas seções 5.2. e 5.12.

Vamos também assumir que algumas implementações de alguns dos blocos mais triviais não serão alteradas. Estes blocos seriam o registrador de síndrome, o decodificador binário, as interfaces de entrada/saída e o conversor paralelo-série, para os quais é trivial que quaisquer outras tentativas de implementação redundam em um número maior de circuitos e CIs ([33], [27], [14]). Cada bloco acima foi implementado com 8, 3, 3, e 1 CIs respectivamente, totalizando 15 CIs.

Após análise criteriosa da implementação direta, concluímos que teríamos um acréscimo de:

$$11 + 1 + 11 + 1 + 30 + 1 + 2 = 57 \text{ CIs} \qquad (6.10)$$

em relação à implementação original, que usa 124 CIs.

As parcelas da soma acima são correspondentes, respectivamente, aos seguintes conjuntos:

a) caçador de posições menos confiáveis;

b) gerador de padrões de teste;

c) gerador de síndromes de teste, somador de síndromes e registro de síndrome;

d) tabela decodificadora;

e) analisador de coincidência de endereços, somador de confiabilidades e memória de dados e confiabilidades;

f) corretor de erros (etapa de saída), comparador de métricas e memória de posições em erro;

g) medidor de taxa de erro.

Portanto, definitivamente, uma implementação direta do algoritmo 2 de Chase demandaria um número excessivamente maior de CIs do que a implementação feita. Este número, 46% maior que o número final a que chegamos, não poderia ser acomodado dentro do espaço físico disponível para o hardware do decodificador, mesmo usando placas filhas, pois o número de interconexões entre os CIs aumentaria de forma ainda maior, já que é função da combinação dos pinos dos CIs e não do número de CIs em si mesmo.

Agora que já temos a estimativa do número de CIs em uma implementação direta do algoritmo 2 de Chase, vamos analisar outras alternativas de implementação.

Suponhamos que façamos uma mescla de implementação dos blocos de forma que alguns trabalhem com os vetores de 24 dígitos diretamente e outros com os endereços dos dígitos que valem 1, para tentar aproveitar as vantagens de número de CIs necessários de uma e outra forma. Seriam necessários então conversores de forma direta para forma endereçada e vice-versa.

O conversor da forma de representação direta para a representação por endereços exigiria um dentre os seguintes 2 circuitos:

a) uso de um contador de endereços de 1 a 24, composto por 2 CIs contadores de 4 bits síncronos, 1 CI com 4 portas NAND, 1 CI com 4 portas OR e 4 CIs flip-flops D sêxtuplos, num total de 8 CIs; tem o inconveniente de impor atrasos iguais ao tempo de contagem, exigindo compensações nos outros blocos, que irão ter registros de atraso para os processos entrarem em fase;

b) uso direto de 24 CIs isoladores sêxtuplos, com entradas de sinal iguais aos endereços de 1 a 24 e entradas de habilitação dadas por cada dígito do vetor de 24 posições a ser convertido, cujas saídas são multiplexadas no tempo por mais alguns outros CIs contadores ou, dependendo da parte afetada, pelo contador mestre de posições. Portanto, a solução com mínimo de CIs é a a).

O conversor da forma de representação por endereços para a forma direta poderia ser implementado por um conjunto de 12 CIs flip-flops D duplos com set e reset, um conjunto de 5 CIs decodificadores 3-para-8 linhas, atuando sobre os sets dos flip-flops, um gerador de reset único composto por um CI AND de 4 portas, um CI contador de 4 bits e um conjunto de 4 CIs isoladores sêxtuplos para multiplexação temporal dos 4 endereços que serão entradas do decodificador, num total de 23 CIs.

Então, o conjunto de conversão e reconversão, a ser colocado como interface entre os blocos, exigiria um total de 31 CIs, o que elimina qualquer possibilidade de economia através do uso dos blocos mais econômicos em número de CIs.

Conseqüentemente, provamos que a implementação feita por nós, baseada em uso de endereços de dígitos 1 de vetores e em somas simplificadas (ver item 5.2) é bem mais econômica em termos do número de CIs exigidos do que uma implementação direta ou mesclada que use CIs LSI e MSI disponíveis no mercado, em tecnologias C-MOS de alta velocidade (série 74HC/HCT) e N-MOS. Em termos de custo, o hardware implementado custa pouco menos de US$ 300 entre placa de circuito impresso, componentes e mão de obra para montagem.

Finalmente, na seção 5.1. realçamos que uma arquitetura promissora pode se valer de DSPs (digital signal processors). Entretanto, mesmo usando o DSP mais rápido disponível hoje (segundo semestre de 1990), o TMS320C30, com ciclo de instrução de 60 ns e custo de US$ 300 por unidade, teríamos 23 ciclos de instrução (apenas) para processamento de cada padrão de teste, insuficientes para fazer uma rotina que envolveria desde a montagem do padrão de teste, mais as 8 somas do cálculo do peso analógico, mais o processo de decisão. Se usássemos mais DSPs, a placa ficaria anti-econômica, pois além dos DSPs (sejam um ou mais), ainda é necessário todo um hardware de suporte do processador, envolvendo memória de programa, memória de dados, isoladores de barramento, etc. Outros processadores do tipo DSP, mais baratos, como o TMS320C25, que custa US$ 40 e tem ciclo de máquina de 100 ns, têm além da inconveniência de tempos de processamento maiores e, portanto, insuficientes, a inconveniência de terem arquitetura de 16 bits, ao invés de 32 bits, e os vetores a serem manipulados são de 24 dígitos. Então, o uso

de DSPs para processamento do algoritmo 2 de Chase aplicado ao CGE (24, 12) a taxas de 1 Mbit/s é uma idéia somente promissora. É necessário esperarmos uma redução maior de custo conjugada a um aumento da velocidade de processamento.

Por causa de todos os fatores analisados anteriormente, concluímos que o hardware implementado é o que fornece a melhor razão custo/benefício em relação a custo, espaço, confiabilidade ou consumo mas que, por outro lado, é razoavelmente complexo e que, por isto mesmo, exige um procedimento de teste mais apurado que outros hardwares mais convencionais usando CIs LSI e MSI comercialmente disponíveis.

Após a análise dos resultados obtidos com a implementação do hardware de decodificação, baseado na arquitetura proposta no capítulo 5, quanto ao projeto do hardware, apresentaremos nas duas próximas seções os resultados de desempenho medido do decodificador implementado bem como medidas de reprodutibilidade deste desempenho para outras placas, atestando a robustez do projeto e sua viabilidade como produto fabricável e não mero protótipo de laboratório.

6.4. MEDIDAS DE DESEMPENHO - RESULTADOS

6.4.1. RESULTADOS DE LABORATÓRIO

Foram montadas, depuradas e testadas dez placas de circuito impresso contendo o hardware projetado, já em versão industrializável. Estas placas receberam rótulos de um a dez e em seguida escolhemos uma delas, a de número 7, para ser caracterizada exaustivamente em termos de seu desempenho quando inserida no canal. As restantes foram caracterizadas apenas em alguns pontos, para reduzir o tempo de teste necessário.

Nesta seção 6.4., mostramos e discutimos os resultados de desempenho desta placa Unidade Decodificadora UDC # 7.

Estes resultados foram obtidos a partir do teste do decodificador inserido no canal, descrito no apêndice A.

Entretanto, vale a pena recordarmos aqui as principais características do teste e do set-up de medição de laboratório:

a) o teste é realizado em banda básica;

b) o teste usa operação em surtos semi-infinitos para cada medida feita, ou seja, dentro do intervalo de tempo de cada medição, há um único surto;

c) o teste usa, como palavra-código transmitida, o vetor nulo de 24 dígitos **0**, ao qual é somado o ruído; ao se valer da linearidade do sistema de codificação-decodificação, é necessário que o hardware sob teste tenha passado sem problemas em um teste de tráfego de dados (sem correção ativada e sem ruído injetado);

d) o ruído gerado é do tipo AWGN, na faixa de 12 kHz a 1052 kHz;

e) o único elemento de degradação, introduzido no sistema de medição, em relação ao comportamento teórico do decodificador 2 de Chase com quantização uniforme de 3 bits, é o comportamento não ideal do conversor A/D de 3 bits utilizado;

f) o medidor de taxa de erro utilizado no teste é o HP3763 error-detector, programado para recepção de palavra nula (i.e., tudo zero) e no modo de medição contínua "SLOW BER", onde o resultado mostrado no display, e enviado a uma impressora térmica, é obtido como sendo o quociente de 100 erros medidos sobre o número de bits observados necessários para se acumular 100 erros. Apenas para medidas muito longas usamos o modo contagem de erros.

Para uma caracterização completa da placa # 7 procuramos medir as taxas de erro de saída, tanto em decisão suave quanto em decisão abrupta (já que isto é possível apenas com a mudança de um estrape), para uma faixa de taxas de erro de entrada (i.e., de canal) que vai de 6×10^{-2} a 3×10^{-3}, cobrindo toda a região de operação que o decodificador terá que suportar ao operar com o modem do sistema AMDT.

Na maioria das vezes, o número de medidas feitas para um determinado ponto de operação varia de ponto para ponto de forma que a população de medidas não foi uniforme para todos os pontos medidos. Isto ocorre porque o tempo de duração de uma medida e o intervalo entre medidas são extremamente rápidos, fazendo com que não tenhamos velocidade suficiente para controlar as chaves dos instrumentos de maneira a ter uma uniformidade do número de medidas. Entretanto, procuramos garantir um número mínimo de 4 valores medidos em cada condição de cada ponto de operação, exceto no ponto de taxa de erro mais baixa (devido ao tempo muito longo).

Os valores de ruído foram ajustados de forma que pudéssemos medir o desempenho do decodificador nos seguintes pontos (aproximados) de taxa de erro à entrada do decodificador:

$6{,}0 \times 10^{-2}$; $5{,}0 \times 10^{-2}$; $4{,}0 \times 10^{-2}$; $3{,}0 \times 10^{-2}$; $2{,}0 \times 10^{-2}$; $1{,}7 \times 10^{-2}$; $1{,}5 \times 10^{-2}$; $1{,}2 \times 10^{-2}$; $1{,}0 \times 10^{-2}$; $8{,}0 \times 10^{-3}$; $6{,}0 \times 10^{-3}$; $4{,}0 \times 10^{-3}$; $3{,}0 \times 10^{-3}$.

A partir dos dados obtidos em laboratório calculamos a média estatística dos valores, juntamente com a variância, através de fórmulas usuais de estatísticas [23].

$$\bar{x} = \mathrm{E}[X] = \frac{1}{N} \cdot \sum_{i=1}^{N} x_i \tag{6.11}$$

$$\sigma_x = \sqrt{\mathrm{E}\left[(X - \bar{x})^2\right]} = \left[\frac{1}{(N-1)} \cdot \sum_{i=1}^{N} (x_i - \bar{x})^2 \right]^{1/2} \tag{6.12}$$

onde:

$\bar{x}$ – média estatística dos valores amostrais da V.A. X
σ_x – desvio padrão de X
x_i – i-ésimo valor amostral de X
N – tamanho da amostra de X (número de pontos medidos)
E [.] – esperança estatística.

Estas médias estão mostradas na tabela 6.1., a seguir.

BERENT Estimado	BERENT (Média)	Nº de PTOS. Medidos BERENT	Desvio-Padrão BERENT	BERSDA (Média)	Nº de PTOS. Medidos BERSDA	Desvio-Padrão BERSDA	BERSDS (Média)	Nº de PTOS. Medidos BERSDS	Desvio Padrão BERSDS	Observação
6,0 E -2	5,90 E -2	18	0,52 E -2	1,18 E -2	17	0,21 E -2	7,50 E -4	22	1,23 E -4	
5,0 E -2	5,02 E -2	23	0,37 E -2	6,70 E -3	18	1,26 E -3	3,16 E -4	14	0,40 E -4	
4,0 E -2	3,93 E -2	18	0,49 E -2	3,15 E -3	13	0,55 E -3	9,48 E -5	15	1,07 E -5	
3,0 E -2	2,87 E -2	21	0,23 E -2	9,46 E -4	19	2,25 E -4	2,53 E -5	13	0,45 E -5	
2,0 E -2	1,99 E -2	24	0,20 E -2	1,84 E -4	15	0,45 E -4	3,63 E -6	11	0,55 E -6	
1,7 E -2	1,64 E -2	21	0,18 E -2	1,12 E -4	19	0,36 E -4	2,27 E -6	07	0,31 E -6	
1,5 E -2	1,53 E -2	20	0,15 E -2	7,00 E -5	15	1,76 E -5	1,32 E -6	05	0,15 E -6	
1,2 E -2	1,18 E -2	12	0,11 E -2	2,52 E -5	12	0,35 E -5	3,65 E -7	04	0,28 E -7	
1,0 E -2	1,02 E -2	36	0,99 E -2	1,48 E -5	10	0,38 E -5	1,73 E -7	10	0,19 E -7	
8,0 E -3	7,61 E -3	27	0,79 E -3	3,66 E -6	07	0,63 E -6	3,78 E -8	10	1,18 E -8	
6,0 E -3	6,40 E -3	32	0,68 E -3	1,97 E -6	10	0,45 E -6	1,83 E -8	11	0,91 E -8	
4,0 E -3	4,01 E -3	26	0,32 E -3	2,53 E -7	22	0,56 E -7	1,39 E -9*	12	1,35 E -9	* em dec. suave, há várias medidas com menos que 100 erros acumulados
3,0 E -3	3,24 E -3	25	0,31 E -3	1,11 E -7	11	0,17 E -7	1,33 E -10 *	01	1,15 E -10	* idem acima

TABELA 6.1. MÉDIA DOS VALORES DE TAXAS DE ERRO DE DIVERSOS PONTOS DE OPERAÇÃO DO DECODIFICADOR OBTIDOS EM LABORATÓRIO (UDC # 7) – DESEMPENHO MÉDIO

NOTAÇÃO: Para maior clareza dos algarismos, substituímos na tabela a notação científica convencional X, Y. 10 (Z) pela notação abreviada X,Y E (Z) .

SIGLAS:

BERENT – taxa de erro de bit (BER) à entrada do decodificador

BERSDA – taxa de erro de bit (BER) à saída do decodificador, em decisão abrupta

BERSDS – taxa de erro de bit (BER) à saída do decodificador, em decisão suave.

BERENT ESTIMADO – valor estimado "a olho" pela mera observação em laboratório da excursão da BERENT , ao estarmos em um ponto de observação.

Agora, de posse dos valores médios das taxas de erro de entrada e de saída (para decisões suave e abrupta) correspondentes à UDC # 7, podemos traçar as curvas de desempenho médio desta placa.

Estas curvas podem assumir diversas formas, cada uma adequada a um fim específico:

a) curva do $\log_{10}$ $\overline{\text{BERENT}}$ versus $\log_{10}$ $\overline{\text{BERSDS}}$ ou $\log_{10}$ $\overline{\text{BERSDA}}$;

b) curva $\overline{\text{BERENT}}$ versus $\overline{\text{BERSDS}}$ ou $\overline{\text{BERSDA}}$ em escala DILOG;

c) curva E_{bi}/No em decibéis versus $\overline{\text{BERENT}}$ ou $\overline{\text{BERSDS}}$ ou $\overline{\text{BERSDA}}$ em escala MONOLOG;

Estão apresentadas nas subseções a seguir, onde também discutimos a finalidade de cada uma delas.

6.4.2. CURVA DE TENDÊNCIA DO DESEMPENHO

Apresentamos a curva $\log_{10}\overline{\text{BERENT}}$ versus $\log_{10}\overline{\text{BERSDS}}$ e a curva $\log_{10}\overline{\text{BERENT}}$ versus $\log_{10}\overline{\text{BERSDA}}$, plotadas em escala linear na figura 6.2.

Conforme se sabe teoricamente ([44] ,[14]), para relações sinal-ruído relativamente altas, que levem a taxas de erro inferiores a $1x10^{-2}$, em um canal AWGN, a teoria de informação nos diz que os ganhos de codificação podem ser comparados através de uma relação entre expoentes de uma mesma base.

Conseqüentemente, $\log_{10}\overline{\text{BERENT}}$ e $\log_{10}\overline{\text{BERSDA}}$ ou $\log_{10}\overline{\text{BERSDS}}$ estão relacionados de forma linear e os coeficientes angular e de intersecção da reta podem ser obtidos através de uma regressão linear sobre os pontos medidos experimentalmente, usando-se o método dos mínimos quadráticos.

Esta é a finalidade desta curva: obter duas retas que forneçam a tendência das taxas de erro em decisão abrupta e decisão suave, para que possamos, através dos coeficientes angulares e através dos antilogaritmos dos coeficientes lineares (intersecção da reta com o eixo das ordenadas), traçá-las em escala DILOG e termos, então, curvas que forneçam os valores das taxas de erro diretamente (conforme veremos na próxima subseção).

A tabela 6.2. a seguir relaciona as taxas de erro médias e seus logaritmos decimais.

$\overline{BERENT}$	$\log_{10}\overline{BERENT}$	$\overline{BERSDA}$	$\log_{10}\overline{BERSDA}$	$\overline{BERSDS}$	$\log_{10}\overline{BERSDS}$
5,90 E -2	-1,229	1,18 E -2	-1,928	7,50 E -4	-3,125
5,02 E -2	-1,299	6,70 E -3	-2,174	3,16 E -4	-3,500
3,93 E -2	-1,406	3,15 E -3	-2,502	9,48 E -5	-4,023
2,87 E -2	-1,542	9,46 E -4	-3,024	2,53 E -5	-4,597
1,99 E -2	-1,701	1,84 E -4	-3,735	3,63 E -6	-5,436
1,64 E -2	-1,785	1,12 E -4	-3,951	2,27 E -6	-5,644
1,53 E -2	-1,815	7,00 E -5	-4,155	1,32 E -6	-5,879
1,18 E -2	-1,928	2,52 E -5	-4,599	3,65 E -7	-6,438
1,02 E -2	-1,991	1,48 E -5	-4,830	1,73 E -7	-6,762
7,61 E -3	-2,119	3,66 E -6	-5,437	3,78 E -8	-7,423
6,40 E -3	-2,194	1,97 E -6	-5,706	1,83 E -8	-7,738
4,01 E -3	-2,397	2,53 E -7	-6,597	1,39 E -9	-8,857
3,24 E -3	-2,489	1,11 E -7	-6,955	1,33 E -10	-9,876

TABELA 6.2. LOGARITMOS DECIMAIS DAS TAXAS DE ERRO MÉDIAS OBTIDAS EXPERIMENTALMENTE PARA A UDC # 7

Lançamos estes pontos em gráfico na figura 6.2., que também mostra as retas obtidas através de regressão linear destes pontos para os casos de decisão abrupta e de decisão suave, usando os valores dos logaritmos decimais das taxas de erro médias medidas experimentalmente.

As regressões lineares foram realizadas através de funções internas de uma calculadora TI-58C e forneceram os seguintes resultados, supondo que na abcissa temos $\log_{10}\overline{BERENT}$ e nas ordenadas, $\log_{10}\overline{BERSDA}$ e $\log_{10}\overline{BERSDS}$:

a) caso da decisão abrupta ($X = \log_{10}\overline{BERENT}$ e $Y = \log_{10}\overline{BERSDA}$):

– reta:

$$Y = m_{da} X + b_{da} \tag{6.13}$$

– ponto de intersecção com eixo das ordenadas (coeficiente linear):

$$b_{da} = 3{,}133 \tag{6.14}$$

– coeficiente angular:

$$m_{da} = 4{,}031 \tag{6.15}$$

– coeficiente de correlação:

$$\rho_{da} = m_{da} \cdot \frac{\sigma_x}{\sigma_y} = 0{,}999 \tag{6.16}$$

com σx e σy sendo os erros quadráticos médios dos pontos nas direções X e Y respectivamente.

b) caso da decisão suave ($X = \log_{10}\overline{BERENT}$ e $Y = \log_{10}\overline{BERSDS}$):

$Y = m_{ds} X + b_{ds}$ (6.17)

– coeficiente linear:

$b_{ds} = 3,191$ (6.18)

– coeficiente angular:

$m_{ds} = 5,054$ (6.19)

– coeficiente de correlação:

$$\rho_{ds} = m_{ds} \cdot \frac{\sigma_x}{\sigma_y} = 0,996 \quad (6.20)$$

com σx e σy erros quadráticos médios dos pontos nas direções X e Y respectivamente:

Além disso, as regressões servem para fornecer pontos específicos, úteis no traçado de retas em escala DILOG, tabelados abaixo, nas tabelas 6.3. e 6.4.:

$\overline{BERSDA}$	$\log_{10}\overline{BERSDA}$	$\log_{10}\overline{BERENT}$	$\overline{BERENT}$
1 E -2	-2	-1,273	5,328 E -2
1 E -3	-3	-1,522	3,009 E -2
1 E -4	-4	-1,770	1,700 E -2
1 E -5	-5	-2,018	9,600 E -3
1 E -6	-6	-2,266	5,422 E -3
1 E -7	-7	-2,514	3,062 E -3
1 E -8	-8	-2,762	1,730 E -3
1 E -9	-9	-3,010	9,770 E -4
1 E -10	-10	-3,258	5,518 E -4

TABELA 6.3. PONTOS DA CURVA DE TENDÊNCIA DA DECISÃO ABRUPTA - UDC # 7

$\overline{\text{BERSDS}}$	$\log_{10}\overline{\text{BERSDS}}$	$\log_{10}\overline{\text{BERENT}}$	$\overline{\text{BERENT}}$
1 E -2	-2	-1,027	9,400 E -2
1 E -3	-3	-1,225	5,958 E -2
1 E -4	-4	-1,423	3,778 E -2
1 E -5	-5	-1,621	2,400 E -2
1 E -6	-6	-1,818	1,519 E -2
1 E -7	-7	-2,016	9,633 E -3
1 E -8	-8	-2,214	6,108 E -3
1 E -9	-9	-2,412	3,873 E -3
1 E -10	-10	-2,610	2,456 E -3

TABELA 6.4. PONTOS ESPECÍFICOS DA CURVA DE TENDÊNCIA DA DECISÃO SUAVE (UDC # 7)

Finalmente, realçamos que os resultados medidos têm excelente aderência em relação às retas, conforme atestam os altos valores dos coeficientes de correlação dados em (6.16) e (6.17).

6.4.3. CURVAS DE DESEMPENHO DA TAXA DE ERRO

Chamamos de curvas de desempenho da taxa de erro as curvas $\overline{\text{BERENT}}$ versus $\overline{\text{BERSDA}}$ e $\overline{\text{BERENT}}$ versus $\overline{\text{BERSDS}}$ traçadas em escala DILOG (com escalas logarítmicas nos dois eixos), obtidos para a UDC # 7.

Estas curvas, traçadas de forma direta a partir da plotagem dos pontos experimentais da tabela 6.1. e de suas regressões lineares (com pontos das tabelas 6.3. e 6.4), estão nas figuras 6.3. e 6.4., onde segmentamos a escala por motivo de facilidade de desenho.

Nestas mesmas figuras, apresentamos as curvas teóricas esperadas para o caso de decisão abrupta e o caso de decisão suave sem quantização usando o algoritmo 2 de Chase. Estas curvas foram obtidas de forma indireta, usando os pontos da tabela 6.6. Estes pontos foram deduzidos das curvas Ebi/No versus BER encontradas em [31] e a partir da tabela 6.5. abaixo, que fornece os valores teóricos de Pb versus Ebi/No sem codificação, baseada na fórmula:

$$P_b = \frac{1}{2}\operatorname{erfc}\sqrt{\frac{E_{bi}}{N_o}} \tag{6.21}$$

onde:

P_b – probabilidade de erro de bit

E_{bi}/N_O – energia por bit de informação por densidade espectral de potência de ruído, à entrada do receptor.

erfc – função erro-complementar.

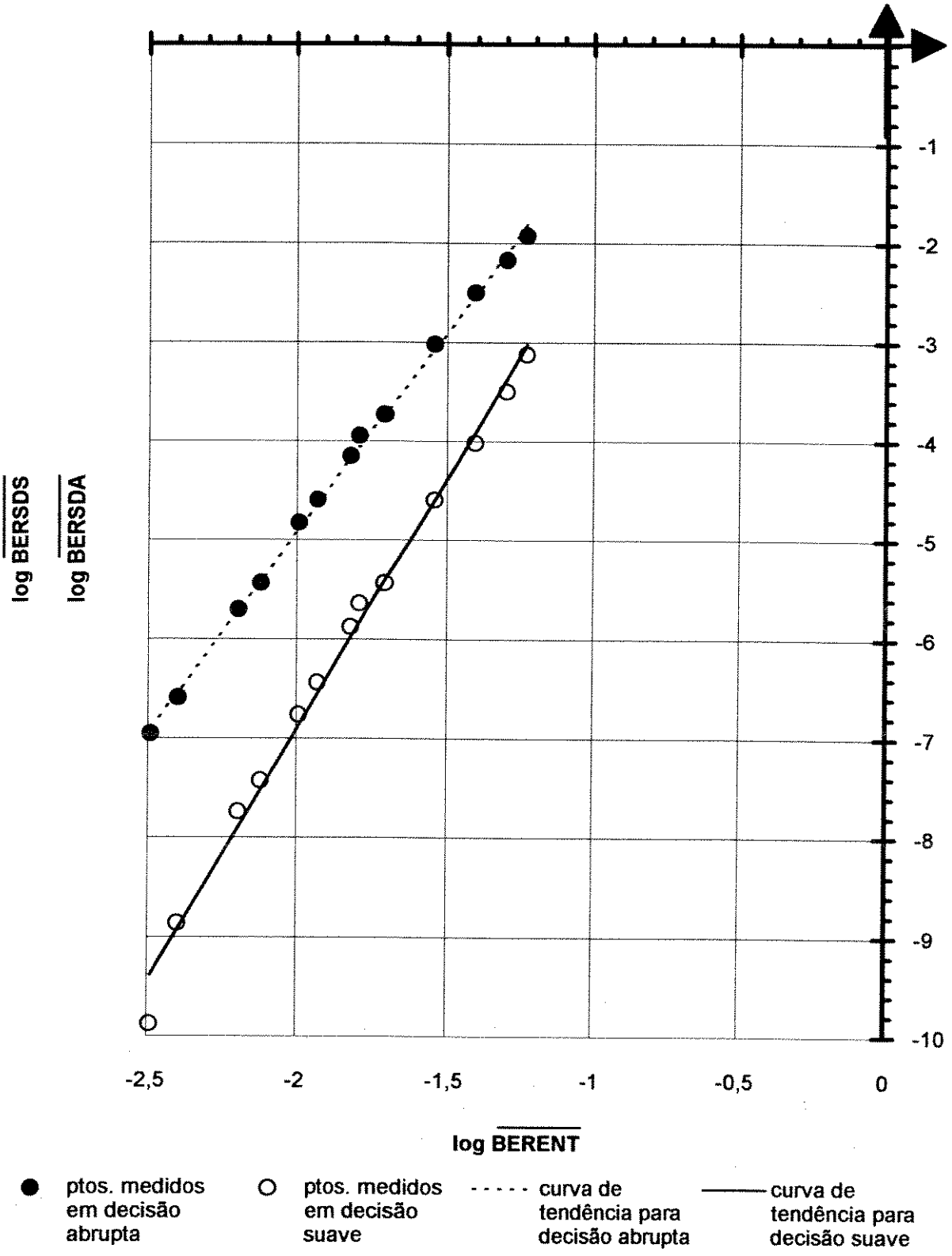


FIGURA 6.2. CURVAS DE TENDÊNCIA DO DESEMPENHO DA UDC #7, PARA 0 CGE (24,12)

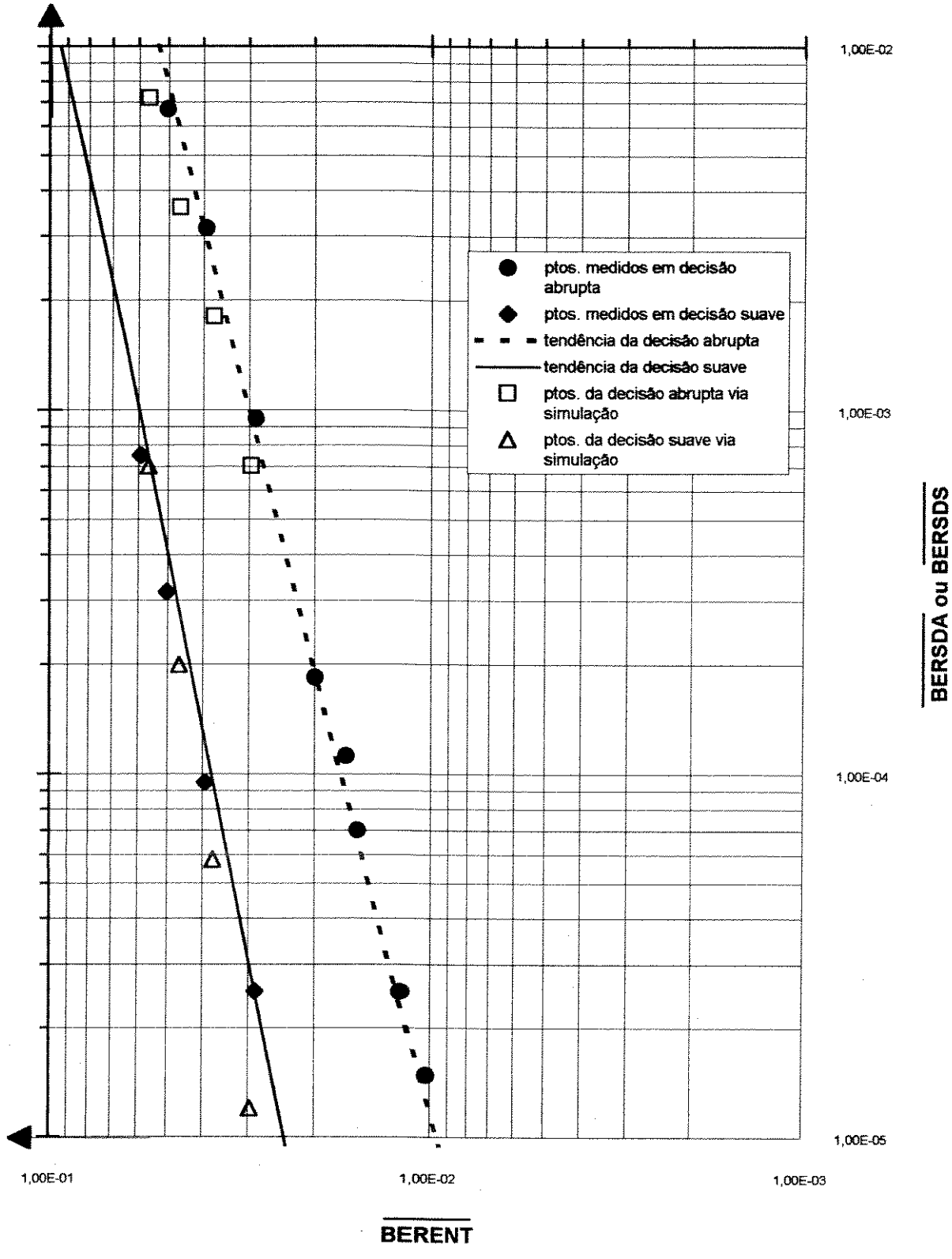


FIGURA 6.3. CURVAS DE DESEMPENHO DE TAXA DE ERRO DA UDC #7 - (BER ALTO)

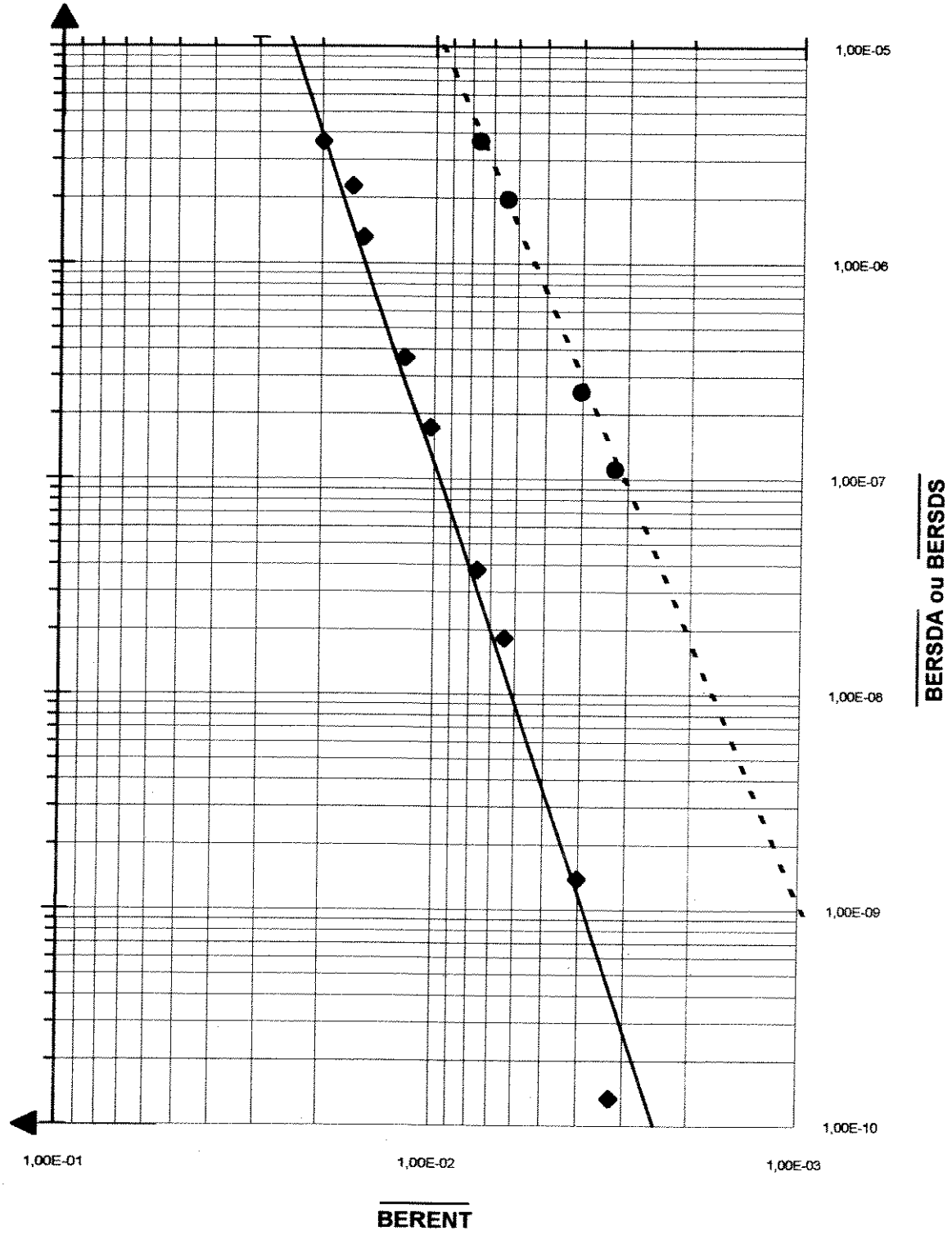


FIGURA 6.4. CURVAS DE DESEMPENHO DE TAXA DE ERRO DA UDC #7 - CONT. (BER BAIXO)

E_{bi}/N_o (dB)	P_b	E_{bi}/N_o (dB)	P_b
1,0	5,628 E -2	7,5	3,989 E -4
1,5	4,640 E -2	8,0	1,909 E -4
2,0	3,751 E -2	8,5	8,403 E -5
2,5	2,966 E -2	9,0	3,364 E -5
3,0	2,228 E -2	9,5	1,212 E -5
3,5	1,717 E -2	10,0	3,876 E -6
4,0	1,250 E -2	10,5	1,085 E -6
4,5	8,794 E -3	11,0	2,617 E -7
5,0	5,954 E -3	11,5	5,340 E -8
5,5	3,862 E -3	12,0	9,031 E -9
6,0	2,388 E -3	12,5	1,237 E -9
6,5	1,400 E -3	13,0	1,339 E -10
7,0	7,727 E -4		

TABELA 6.5. PROBABILIDADE DE ERRO VERSUS RELAÇÃO SINAL/RUÍDO PARA QPSK COERENTE

Para construir uma tabela $\overline{\text{BERENT}}$ por $\overline{\text{BERSDA}}$ ou por $\overline{\text{BERSDS}}$, assumimos primeiro uma igualdade entre $\overline{\text{BERENT}}$ e P_b, somamos 3dB ao E_{bi}/N_o correspondente e, com o resultado desta soma, achamos nas curvas plotadas em [31] o valor de $\overline{\text{BERSDA}}$ ou $\overline{\text{BERSDS}}$ (conforme a curva usada). Esta tabela está dada abaixo.

$\overline{\text{BERENT}}$	Ebi/No (dB)	Ebi/No + 3 (dB)	$\overline{\text{BERSDA}}$	$\overline{\text{BERSDS}}$
5,628 E -2	1,0	4,0	$\approx$ 7,2 E -3	$\approx$ 7,0 E -4
4,640 E -2	1,5	4,5	$\approx$ 3,6 E -3	$\approx$ 2,0 E -4
3,751 E -2	2,0	5,0	$\approx$ 1,8 E -3	$\approx$ 5,8 E -5
2,966 E -2	2,5	5,5	$\approx$ 7,0 E -4	$\approx$ 1,2 E -5

TABELA 6.6. PONTOS TEÓRICOS DE DESEMPENHO (OBTIDOS POR SIMULAÇÃO COMPUTACIONAL)

Estas curvas de desempenho das figuras 6.2. e 6.3. servem, então, para se obter, de forma muito rápida, os valores das taxas de erro de bit à saída do decodificador dada a taxa de erro de bit à entrada, para ambos os casos: de decisão abrupta e de decisão suave da UDC # 7.

Também, conforme podemos observar na figura 6.2., os resultados obtidos estão muito próximos daqueles que são encontrados por simulação computacional, i. e., os valores teóricos ([31], [13]).

6.4.4. CURVAS DE RELAÇÃO SINAL-RUÍDO

As curvas de desempenho obtidas na sub-seção anterior originam-se de forma natural das medidas realizadas em laboratório, onde uma medição do E_{bi}/N_o em banda-básica fica dificultada pelo esquema de teste adotado (transmissão da palavra nula, etc) e por falta de instrumental adequado (p. ex. medidor de potência ou um voltímetro RMS verdadeiro com a resposta de freqüência adequada).

Apresentamos aqui as curvas de $\overline{\text{BERENT}}$, $\overline{\text{BERSDA}}$ e $\overline{\text{BERSDS}}$ em função de E_{bi}/N_o, que é a relação entre a energia por bit de informação e a densidade espectral de potência do ruído.

Estas curvas são construídas de forma indireta a partir do procedimento fornecido no apêndice A.

Inicialmente, supomos $\overline{\text{BERENT}}$ igual a P_b e traçamos a curva E_{bi}/N_o x $\overline{\text{BERENT}}$. Depois, associamos $\overline{\text{BERSDA}}$ ou $\overline{\text{BERSDS}}$ ao valor indireto de E_{bi}/N_o, através dos seguintes cálculos:

a) dado $\overline{\text{BERSDA}}$ ou $\overline{\text{BERSDS}}$ achamos o $\overline{\text{BERENT}}$ correspondente;

b) com este $\overline{\text{BERENT}}$ obtemos o E_{bi}/N_o sem codificação correspondente;

c) somamos 3 dB a este E_{bi}/N_o sem codificação e associamos este valor a $\overline{\text{BERSDA}}$ ou $\overline{\text{BERSDS}}$, conforme o caso.

Apresentamos na tabela 6.7. os resultados destes cálculos para decisão abrupta e na tabela 6.8. os resultados para decisão suave.

A partir destes dados tabelados acima (nas tabelas 6.7. e 6.8.) e dos dados da tabela 6.5., plotamos as seguintes curvas:

a) curva $\overline{\text{BERENT}}$ por E_{bi}/N_o, baseada na tabela 6.5., que representa o caso da recepção de sinal QPSK coerente;

b) curva $\overline{\text{BERSDA}}$ por $(E_{bi}/N_o)'$, baseada na tabela 6.7., que representa o caso da recepção de sinal QPSK coerente, com decodificação por decisão abrupta do CGE (24, 12) usando a UDC # 7;

c) curva $\overline{\text{BERSDS}}$ por $(E_{bi}/N_o)'$, baseada na tabela 6.8., que representa o caso da recepção de sinal QPSK coerente, com decodificação por decisão suave com 8 níveis de quantização, através do algoritmo 2 de Chase para o CGE (24, 12), usando a UDC # 7;

d) pontos da simulação do algoritmo 2 de Chase ([31], [13]) para o CGE (24, 12) para o caso sem quantização.

Estas curvas e pontos, plotados em escala MONOLOG, podem ser vistas nas figuras 6.5., 6.6. e 6.7., de forma segmentada no eixo das ordenadas, para melhor visualização dos gráficos.

As curvas de relação sinal-ruído prestam-se a vários fins, sendo especialmente úteis na determinação dos ganhos dos sistemas de codificação-decodificação e na comparação entre vários desempenhos distintos.

Deixaremos a análise de desempenho e as conclusões que podem ser obtidas a partir destas medidas da UDC # 7 para uma seção posterior, onde será feita uma análise global do desempenho do hardware implementado, abrangendo as diversas placas construídas.

Por ora, basta notarmos que o desempenho medido para a decisão suave com a UDC # 7 é muitíssimo próximo ao desempenho obtido por simulação, conforme se vê nas figuras 6.5. e 6.6.

Passamos agora a analisar, na próxima seção, o desempenho medido para outras placas montadas com o mesmo hardware baseado na arquitetura proposta no capítulo 5, para verificar a reprodutibilidade de desempenho, que viabilizará a arquitetura e o hardware implementado como estruturas fabricáveis.

		(da curva sem codific.)	$(E_{bi}/N_O)'=$	
$\overline{BERSDA}$	$\overline{BERENT}$	Ebi/No (dB)	Ebi/No + 3 (dB)	Observações
1,18 E -2	5,90 E -2	≈0,90	3,90	Pontos Medidos
6,70 E -3	5,02 E -2	≈1,30	4,30	Pontos Medidos
3,15 E -3	3,93 E -2	≈2,40	5,40	Pontos Medidos
9,46 E -4	2,87 E -2	≈2,60	5,60	Pontos Medidos
1,84 E -4	1,99 E -2	≈3,20	6,20	Pontos Medidos
1,12 E -4	1,64 E -2	≈3,55	6,55	Pontos Medidos
7,00 E -5	1,53 E -2	≈3,70	6,70	Pontos Medidos
2,52 E -5	1,18 E -2	≈4,10	7,10	Pontos Medidos
1,48 E -5	1,02 E -2	≈4,30	7,30	Pontos Medidos
3,66 E -6	7,61 E -3	≈4,70	7,70	Pontos Medidos
1,97 E -6	6,40 E -3	≈4,90	7,90	Pontos Medidos
2,53 E -7	4,01 E -3	≈5,45	8,45	Pontos Medidos
1,11 E -7	3,24 E -3	≈5,70	8,70	Pontos Medidos
1 E -2	5,33 E -2	≈1,15	4,15	* Pontos de Regressão
1 E -3	3,00 E -2	≈2,45	5,45	* Pontos de Regressão
1 E -4	1,70 E -2	≈3,50	6,50	* Pontos de Regressão
1 E -5	9,60 E -3	≈4,40	7,40	* Pontos de Regressão
1 E -6	5,42 E -3	≈5,10	8,10	* Pontos de Regressão
1 E -7	3,06 E -3	≈5,75	8,75	* Pontos de Regressão
1 E -8	1,73 E -3	≈6,30	9,30	* Pontos de Regressão
1 E -9	9,77 E -4	≈6,85	9,85	* Pontos de Regressão
1 E -10	5,52 E -4	≈7,25	10,25	* Pontos de Regressão

TABELA 6.7. CONSTRUÇÃO DA CURVA $\overline{BERSDA}$ POR E_{bi}/N_O

		(da curva sem codific.)	$(E_{bi}/N_o)' =$	
$\overline{BERSDS}$	$\overline{BERENT}$	Ebi/No (dB)	Ebi/No + 3 (dB)	Observações
7,50 E -4	5,90 E -2	≈0,90	3,90	Pontos Medidos
3,16 E -4	5,02 E -2	≈1,30	4,30	Pontos Medidos
9,48 E -5	3,93 E -2	≈2,40	5,40	Pontos Medidos
2,53 E -5	2,87 E -2	≈2,60	5,60	Pontos Medidos
3,63 E -6	1,99 E -2	≈3,20	6,20	Pontos Medidos
2,27 E -6	1,64 E -2	≈3,55	6,55	Pontos Medidos
1,32 E -6	1,53 E -2	≈3,70	6,70	Pontos Medidos
3,65 E -7	1,18 E -2	≈4,10	7,10	Pontos Medidos
1,73 E -7	1,02 E -2	≈4,30	7,30	Pontos Medidos
3,78 E -8	7,61 E -3	≈4,70	7,70	Pontos Medidos
1,83 E -8	6,40 E -3	≈4,90	7,90	Pontos Medidos
1,39 E -9	4,01 E -3	≈5,45	8,45	Pontos Medidos
1,33 E -10	3,24 E -3	≈5,70	8,70	Pontos Medidos
1 E -2	9,40 E -2	—	—	* Pontos da Regressão
1 E -3	5,96 E -2	≈0,85	3,85	* Pontos da Regressão
1 E -4	3,78 E -2	≈1,95	4,95	* Pontos da Regressão
1 E -5	2,40 E -2	≈2,90	5,90	* Pontos da Regressão
1 E -6	1,52 E -2	≈3,65	6,65	* Pontos da Regressão
1 E -7	9,63 E -3	≈4,35	7,35	* Pontos da Regressão
1 E -8	6,11 E -3	≈5,00	8,00	* Pontos da Regressão
1 E -9	3,87 E -3	≈5,50	8,50	* Pontos da Regressão
1 E -10	2,46 E -3	≈5,95	8,95	* Pontos da Regressão

TABELA 6.8. CONSTRUÇÃO DA CURVA $\overline{BERSDS}$ POR E_{bi}/N_o

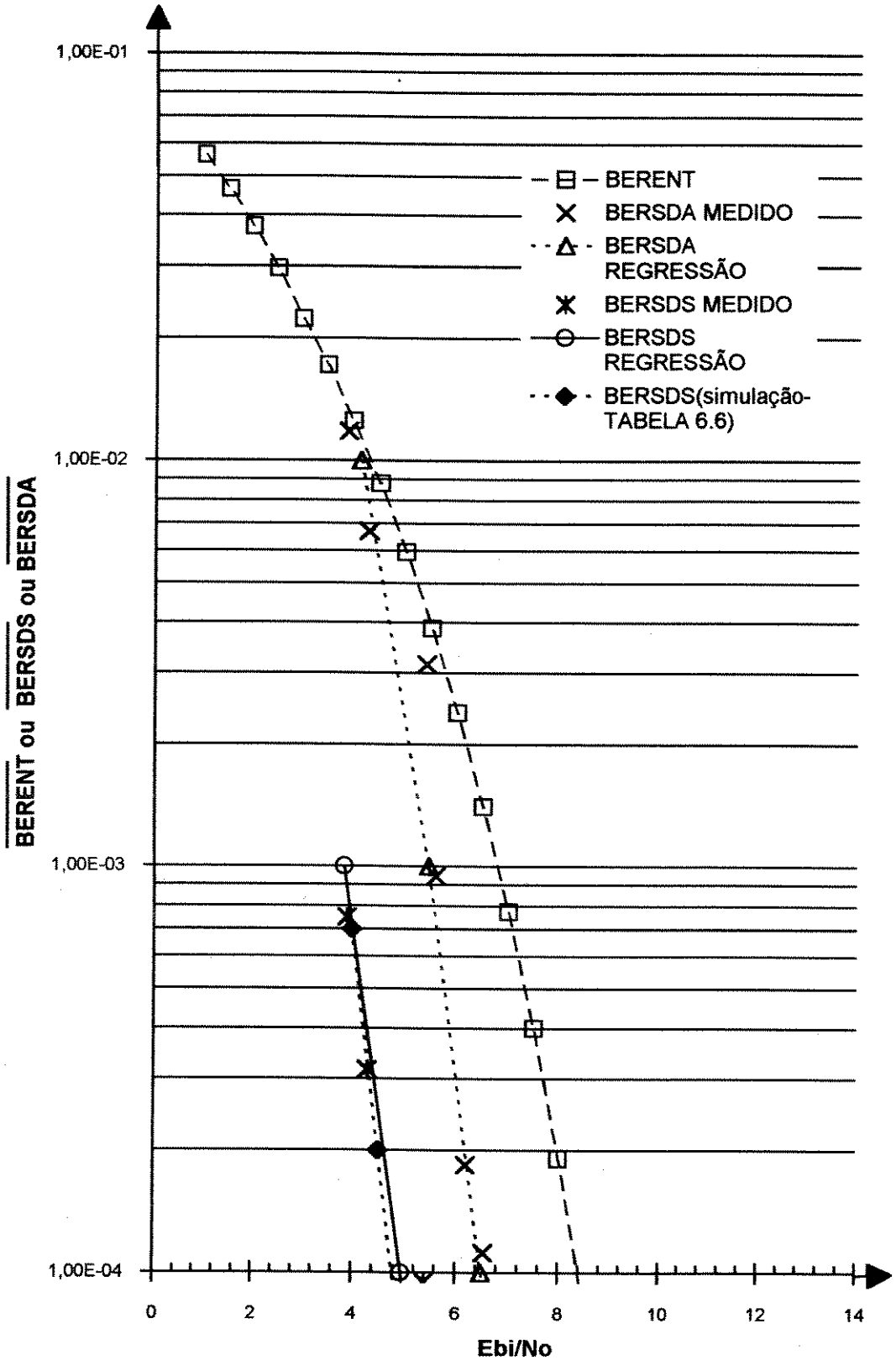


FIGURA 6.5. CURVAS DE BER VERSUS E_{bi}/N_O PARA A UDC #7

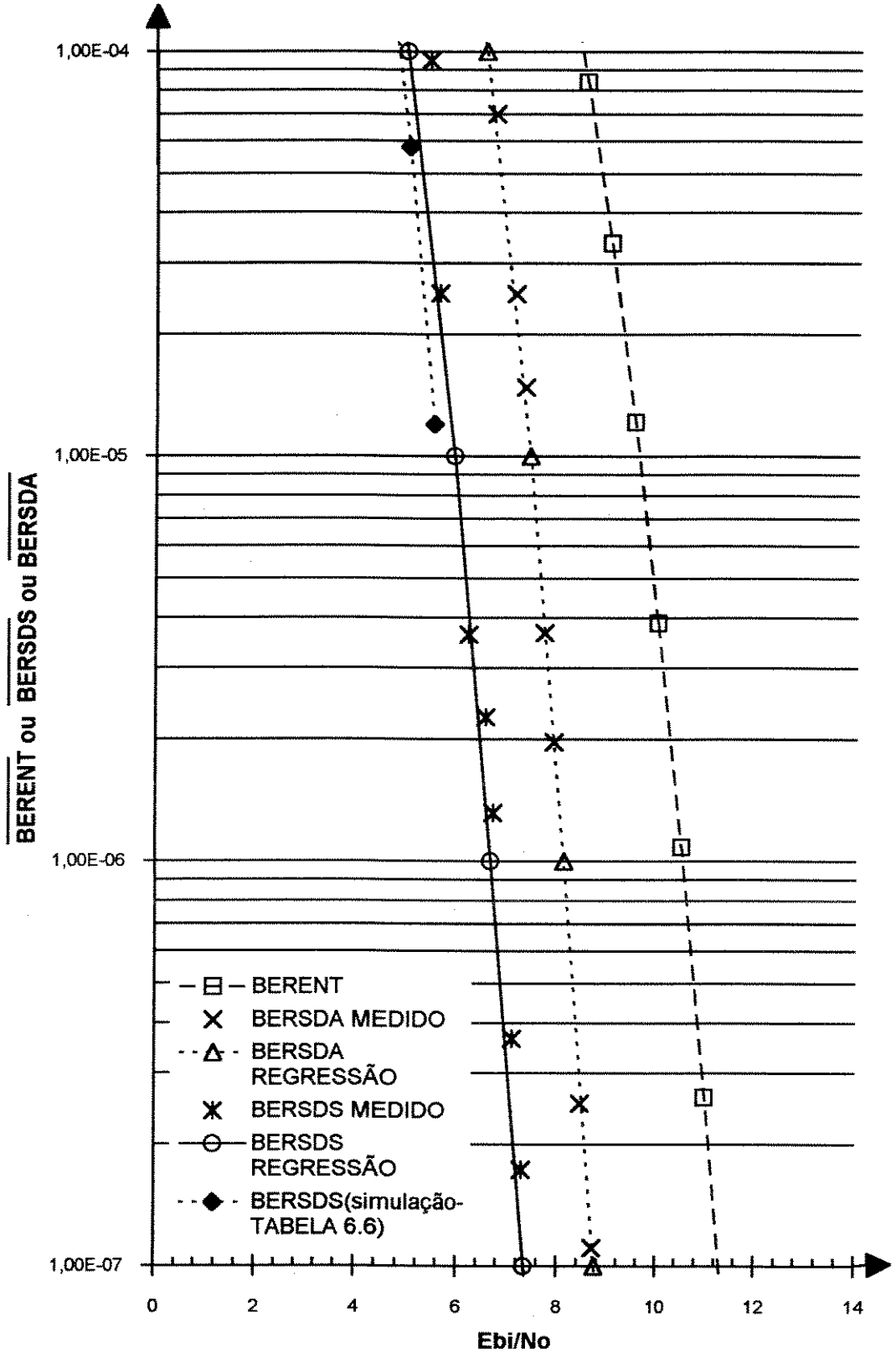


FIGURA 6.6. CURVAS DE BER VERSUS E_{bi}/N_o PARA A UDC #7 (CONTINUAÇÃO)

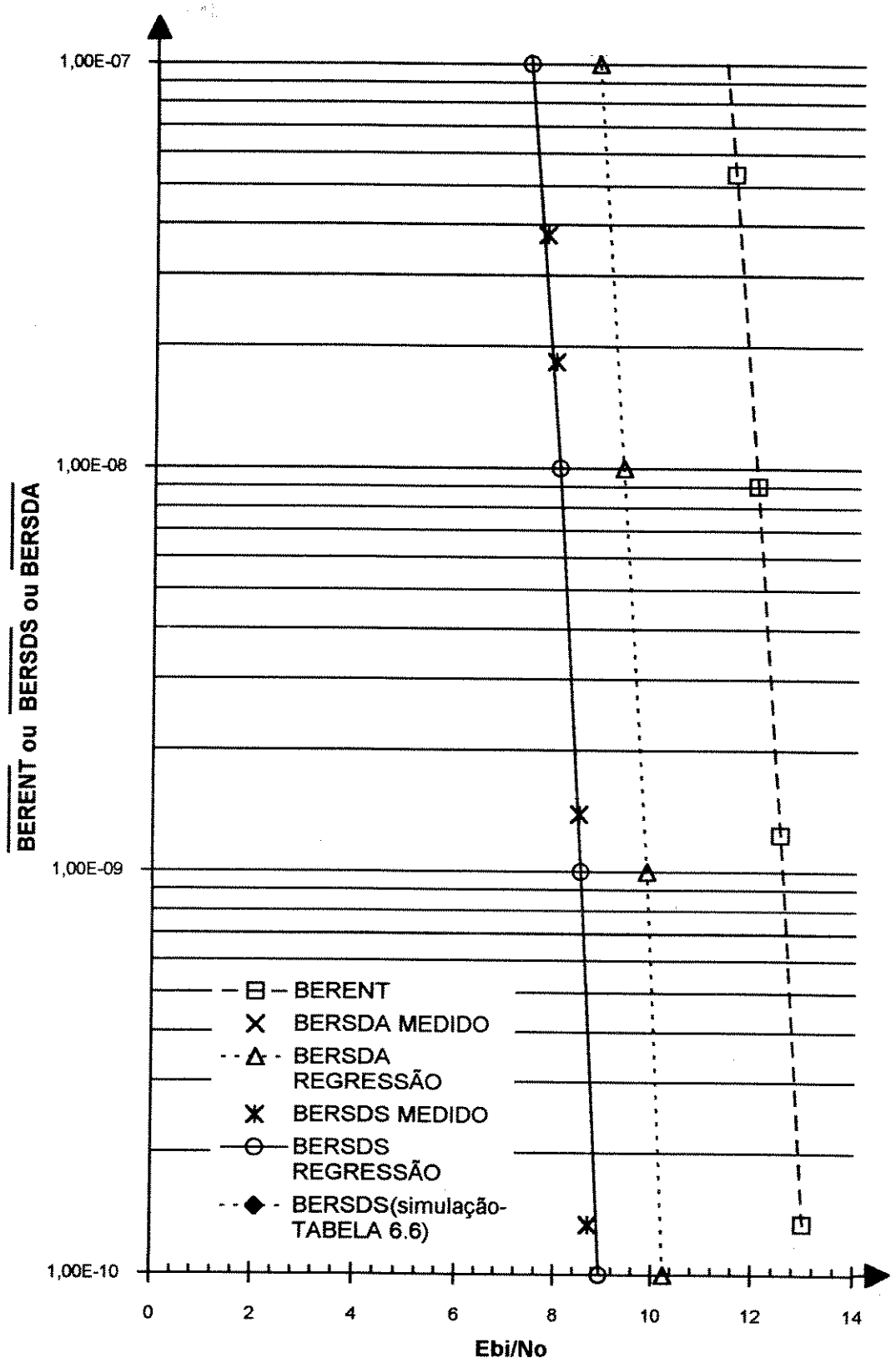


FIGURA 6.7. CURVAS DE BER VERSUS E_{bi}/N_o PARA A UDC #7 (CONTINUAÇÃO)

6.5. REPRODUTIBILIDADE DO HARDWARE RESULTADOS

Após a caracterização exaustiva da UDC # 7, passamos a caracterizar as nove placas restantes, utilizando o mesmo set-up de testes anterior, porém medindo apenas o desempenho em um ponto específico de operação.

Este ponto específico de operação tem que ter dois atributos desejáveis:

a) rapidez de medida do desempenho;

b) capacidade de detecção de pequenas falhas do hardware, não encontradas durante a depuração, através da observação de anomalias nas taxas de erro medidas.

Como esta capacidade de detecção de falhas é função direta da sensibilidade a pequenas variações na taxa de erro e essa sensibilidade aumenta para taxas de erro pequenas, os dois atributos são conflitantes entre si.

Empiricamente, no laboratório, o ponto de operação que nos deu o melhor compromisso entre os dois atributos gira ao redor da taxa de erro de entrada de $2{,}0 \times 10^{-2}$.

Com isto, como a taxa de erro de saída (em decisão suave) esperada é em volta de 3×10^{-6} a 4×10^{-6}, o tempo de uma medição de 100 erros a 534 kbit/s dura 47 a 62 segundos (1 minuto, aproximadamente) e, neste valor de taxa de erro de saída, erros provocados pelo hardware e não pelo ruído AWGN já não são mais mascarados pelos erros vindos do ruído, havendo tempo de observação no analisador lógico com durações ao redor de 4 a 5 segundos.

Então, tomando como ponto de operação de referência o ponto de taxa de erro média à entrada de $2{,}0 \times 10^{-2}$, e usando como valor de referência a taxa de erro média à saída da UDC # 7, fizemos uma série de medidas de taxa de erro à saída das outras UDC's.

Os resultados estão apresentados na tabela 6.9, dada a seguir:

Rótulo da UDC	Resultados Medidos @ BERENT = 2 E -2 **BERSDS**	**$\overline{\text{BERSDS}}$**
# 7	Ver Figura 6.4	3,6 E -6
# 1	1,4 E -6; 1,3 E -6; 1,2 E -6; 1,3 E -6	1,30 E -6 (σ = 0,08 E -6)
# 2	3,5 E -7; 5,5 E -7; 6,7 E -7; 4,7 E -7	5,10 E -7 (σ = 1,35 E -7)
# 3	6,7 E -7; 1,8 E -6; 7,1 E -7; 5,8 E -7; 1,0 E -6; 6,8 E -7; 6,0 E -7	8,63 E -7 (σ = 4,36 E -7)
# 4	1,1 E -6; 1,5 E -6; 9,7 E -7; 9,4 E -7	1,13 E -6 (σ = 0,26 E -6)
# 5	9,0 E -7; 9,0 E -7	9,0 E -7 (σ = 0)
# 6	2,5 E -6; 3,1 E -6; 2,3 E -6; 1,8 E -6	2,4 E -6 (σ = 0,54 E -6)
# 8	4,1 E -7; 5,3 E -7; 3,1 E -7	4,1 E -7 (σ = 1,1 E -7)
# 9	1,3 E -6; 1,2 E -6; 1,2 E -6; 1,3 E -6	1,25 E -6 (σ = 0,06 E -6)
# 10	2,0 E -6; 2,2 E -6; 1,8 E -6; 1,5 E -6; 2,0 E -6; 3,3 E -6; 9,6 E -7	1,97 E -6 (σ = 0,72 E -6)

TABELA 6.9. RESULTADOS DE DESEMPENHO DAS OUTRAS PLACAS OPERANDO EM DECISÃO SUAVE

O ponto de referência de taxa de erro média à entrada de 2 x 10^{-2} foi ajustado para a mesma potência de ruído anteriormente ajustada para a medida da UDC # 7, e a taxa de erro de entrada averiguada no início das medidas de cada placa.

O número de medidas realizadas por placa variou em função do critério de aprovação usado, que foi o seguinte:

a) se a placa estivesse com taxas de erro de saída estáveis e menores ou iguais à da UDC # 7, bastavam de 2 a 4 medidas;

b) se a placa estivesse com taxas de erro de saída instáveis, oscilando um pouco, 4 medidas era o mínimo necessário;

c) se a placa apresentasse taxas de erro de saída maiores que o dobro da UDC # 7, voltava à fase de depuração e, ao retornar, um mínimo de 7 medidas eram feitas, para garantir que a depuração havia sido efetiva.

Para inferir que este método de avaliação em um único ponto seria efetivo, medimos, para a segunda placa analisada no laboratório, a UDC # 9, o desempenho em um segundo ponto, referente a uma taxa média de erro à entrada de 1,0 x 10^{-2}. Os resultados obtidos foram tabelados abaixo:

UDC #	$\overline{\text{BERENT}}$	Resultados Medidos - BERSDS	$\overline{\text{BERSDS}}$
7	1,0 E -2	Ver Figura 6.4	1,73 E -7
9	1,0 E -2	2,2 E -7; 2,0 E -7; 2,2 E -7; 2,1 E -7; 1,5 E -7; 1,5 E -7; 1,7 E -7	1,89 E -7 (σ = 0,31 E -7)

TABELA 6.10 RESULTADOS DA UDC # 9 NO PONTO DE 1,0 E -2 PARA DECISÃO SUAVE

Comparando as taxas de erro médias à saída do decodificador operando em decisão suave, para as UDC # 7 (referência) e UDC # 9, em dois pontos de operação distintos, podemos assumir que o método de avaliação por ponto único é mais que suficiente para se aceitar uma placa como operacional. Isto é óbvio do ponto de vista teórico, mas não era garantido na prática, onde há muitíssimas variáveis desconhecidas que poderiam influir neste aspecto, mas que, diante destas medidas, parecem não existir ou, ao menos, não influir na região de interesse para a operação.

Do ponto de vista da reprodutibilidade do desempenho, a tabela 6.9. nos indica que este se manteve dentro de limites que permitem afirmar que tanto a arquitetura proposta para o decodificador quanto o hardware implementado são adequados para a manufatura industrial porque mostram-se reprodutíveis, além do fato do hardware ser robusto e barato.

6.6. CONSIDERAÇÕES SOBRE O DESEMPENHO MEDIDO

Conforme apresentamos nas seções 6.4 e 6.5, os resultados de desempenho medidos apresentam alta correlação com os resultados previstos teoricamente (por simulação ou analiticamente) e também alta correlação entre si, quando medidos sobre um conjunto de placas. Todavia, não quantificamos estas correlações.

Para encerrar este capítulo, forneceremos nesta seção algumas conclusões importantes baseadas nas seções 6.4 e 6.5 precedentes e nas seções 2.2, 2.3, 1.2, 1.3 e 6.2.

6.6.1. GANHO DO SISTEMA DE CODIFICAÇÃO-DECODIFICAÇÃO

Definimos ganho do sistema de codificação-decodificação como sendo a diferença em dB entre a relação sinal/ruído por bit de informação do sistema sem codificação e a relação sinal/ruído por bit de informação do sistema com codificação necessárias para produzir à saída uma mesma taxa de erro de bit de informação.

O ganho sempre é especificado a uma determinada taxa de erro de bit de informação porque as curvas de desempenho são não-lineares.

Então, referindo-se às figuras 6.5, 6.6 e 6.7, o ganho é obtido diretamente observando-se as diferenças na direção E_{bi}/N_o para um mesmo BER especificado.

A tabela 6.11 fornece os diversos ganhos (ganho em decisão suave com quantização de 8 níveis, ganho em decisão abrupta, etc) em função das taxas de erro, para a UDC # 7.

BER	G_{DS} (dB)	G_{DA} (dB)	Δ DSA (dB)	G_{COD} (dB)	Δ SH (dB)
1 E -3	2,90	2,25	0,65	(-2,424)	N/D
1 E -4	3,50	1,95	1,55	1,269	1,6
1 E -5	3,65	2,15	1,5	2,700	2,0
1 E -6	3,90	2,40	1,5	3,426	N/D
1 E -7	N/D	N/D	N/D	3,925	N/D
1 E -8	4,00	2,70	1,3	4,284	N/D
1 E -9	4,05	2,70	1,35	4,510	N/D
1 E -10	4,15	2,85	1,3	4,685	N/D

TABELA 6.11. GANHOS DA UDC # 7

Na tabela acima, a notação usada é a seguinte:

G_{DS} – ganho do sistema de codificação-decodificação em decisão suave usando 8 níveis de quantização e algoritmo 2 de Chase para o CGE (24, 12), em relação a um sistema sem codificação (2-PSK ou 4-PSK), obtido através de medidas de laboratório sobre a UDC # 7;

G_{DA} – ganho do sistema de codificação-decodificação em decisão abrupta (decodificação binária) para o CGE (24, 12), em relação a um sistema sem codificação (2-PSK ou 4- PSK), obtido através de medidas de laboratório sobre a UDC # 7;

ΔDSA – ganho do sistema operando em decisão suave sobre o sistema operando em decisão abrupta, medido como sendo a diferença entre G_{DS} e G_{DA}, em dB;

G_{COD} – limitante superior para o ganho do sistema de codificação-decodificação, operando em decisão suave na capacidade do canal sem quantização, fornecido pela equação (2.19) (ver capítulo 2) aplicada ao CGE (24, 12), que tem:

$R = 1/2 \qquad d = 8 \qquad k = 12$

O E_{bi}/N_o correspondente às taxas de erro tabeladas foi extraído da tabela 6.6., na coluna de $(E_{bi}/N_o)'$.

ΔSH – ganho do sistema operando em decisão suave, sem quantização, sobre o sistema operando em decisão abrupta, obtido por simulação (ver capítulo 1, seção 1.2.1.3.).

N D – significa valor não-disponível ou insuficiência de dados.

Comparando esta tabela e os resultados das seções 1.2.1.3., 1.3.1., 2.2. e 2.3. anteriores, muitas conclusões importantes advêm e estão listadas abaixo:

1) As simulações feitas por Chase [13] e Pessoa & Arantes [31] indicavam (cap. 1, seção 1.2.1.3.), para P_b = 1,0 x 10^{-4}, no caso de decodificação por decisão suave usando o algoritmo 2 de Chase, com quantização de 8 níveis, um ganho de 1,6 dB em relação à decodificação binária e um ganho de 3,0 dB sobre uma transmissão não-codificada, supondo que o código utilizado era o CGE (24, 12). O hardware implementado para este mesmo decodificador, usando a UDC # 7, produziu,

conforme a tabela 6.11., ganhos de 1,55 dB e 3,50 dB em relação à decodificação binária e à transmissão não-codificada, respectivamente, para uma taxa de erro de bit à saída de $1,0 \times 10^{-4}$.

PORTANTO, NESTA TAXA, O HARDWARE PROJETADO PRODUZ OS RESULTADOS ESPERADOS, OBTIDOS POR SIMULAÇÃO, COM ERROS DE 0,05 dB (DECISÃO ABRUPTA) E 0,50 dB (DECISÃO SUAVE).

2) Pelas mesmas simulações, para $P_b = 1,0 \times 10^{-5}$, o ganho do decodificador binário é de 2 dB sobre a transmissão não-codificada e o ganho do decodificador por correlação, sem quantização, é de 4 dB sobre o esquema não-codificado (seção 1.2.1.3). Com a UDC # 7 obtivemos em laboratório ganhos de 2,15 dB e de 3,65 dB, respectivamente, para a decisão abrupta e para a suave. Como o algoritmo 2 de Chase está a apenas 0,1 dB do desempenho do decodificador por correlação e a quantização em 8 níveis degrada o desempenho em cerca de 0,22 dB, o ganho teórico do sistema com quantização em 8 níveis usando o algoritmo 2 de Chase seria de 3,68 dB em relação à transmissão não-codificada nesta taxa de erro de bit.

PORTANTO, NESTA TAXA, O HARDWARE PROJETADO PRODUZ OS RESULTADOS ESPERADOS, OBTIDOS POR SIMULAÇÃO, COM ERROS DE 0,15 dB (DECISÃO ABRUPTA) E 0,03 dB (DECISÃO SUAVE).

3) O ganho assintótico sobre a transmissão não-codificada é de 3 dB para o decodificador binário e de 6 dB para o decodificador por correlação, quando o código é o CGE (24, 12) – ver seção 1.2.1.3. e cap. 2, equação (2.19) – e, para a UDC # 7, em taxa de erro de bit à saída suficientemente baixa, como $1,0 \times 10^{-10}$, as curvas de tendência das medidas fornecem ganhos de 2,85 dB e 4,15 dB respectivamente. Para esta mesma taxa, a eq. (2.19) fornece um limitante superior para o ganho em decisão suave, usando o decodificador por correlação sem quantização, de 4,69 dB, ou 4,47 dB supondo quantização de 8 níveis.

ENTÃO, NESTA TAXA DE $1,0 \times 10^{-10}$, O HARDWARE PROJETADO FORNECE OS GANHOS ESPERADOS, OBTIDOS POR MEIOS ANALÍTICOS, DENTRO DE VARIAÇÕES DE 0,15 dB (DECISÃO ABRUPTA) E DE 0,32 dB (DECISÃO SUAVE).

6.6.2. RELACIONAMENTO DAS TAXAS DE ERRO

Uma outra forma, a nosso ver mais racional, de fornecer o desempenho do hardware projetado é através das relações entre as taxas de erro de bit de informação à saída do decodificador, em ambos os casos, de decisão suave e de decisão abrupta, e a taxa de erro de bit à entrada do decodificador, que é a taxa de erros correspondente àquela que se obteria em uma transmissão não-codificada.

Achamos que esta forma é mais racional porque, do ponto de vista do decodificador (e não mais do canal), é exatamente o que ele enxerga: erros ocorrendo em uma seqüência de símbolos recebidos, que serão removidos através de um algoritmo matemático (manipulação algébrica, procedimentos de estimação

estatística, etc.) de forma a reduzir a ocorrência destes erros, na saída, sobre os bits de informação.

Assim, esta caracterização, baseada na função do decodificador de reduzir erros, seria feita através da explicitação de uma FUNÇÃO DE TRANSFERÊNCIA DE TAXA DE ERRO DO DECODIFICADOR, definida como uma relação matemática biunívoca entre as taxas de erro de entrada e saída.

Além disso, esta função de transferência é, via de regra, para códigos usuais (de bloco ou convolucionais, lineares) uma relação simples e elegante, como uma reta, por exemplo.

Nas figuras 6.3 e 6.4 já foram plotadas as relações entre as taxas de erro de entrada e saída do hardware implementado.

Entretanto, é na figura 6.2 que nos baseamos para obter uma relação simples para o desempenho medido da UDC # 7, usando regressão linear.

A partir das equações 6.13. a 6.20 podemos obter as funções de transferência da UDC # 7 e concluir o seguinte:

1) PARA O DECODIFICADOR OPERANDO EM DECISÃO ABRUPTA, A FUNÇÃO DE TRANSFERÊNCIA DE TAXA DE ERRO DA UDC # 7 VALE:

$$\log_{10} \overline{\text{BERSDA}} = (4{,}031 \log_{10} \overline{\text{BERENT}}) + 3.133 \qquad (6.22)$$

2) PARA O DECODIFICADOR OPERANDO EM DECISÃO SUAVE, A FUNÇÃO DE TRANSFERÊNCIA DE TAXA DE ERRO DA UDC # 7 VALE:

$$\log 10\ \overline{\text{BERSDS}} = (5{,}054 \log 10\ \overline{\text{BERENT}}) + 3{,}191 \qquad (6.23)$$

6.6.3. DESEMPENHO DO CONJUNTO DE PLACAS

A exposição dos resultados de desempenho obtidos para as dez placas UDC, feita na seção 6.6., mostra que o resultado obtido pelo conjunto converge para o resultado esperado e que os resultados, de cada UDC, são similares entre si.

Para uma taxa de erro à entrada de $2{,}0 \times 10^{-2}$, a média de conjunto das taxas de erro de bit médias das dez placas vale, usando a tabela 6.9.:

$$E\,[\overline{\text{BERSDS}}] = 1{,}44 \cdot 10^{-6} \qquad (6.24)$$

onde E [·] representa média estatística.

O desvio padrão do conjunto vale:

$$\sigma\text{CON} = 0{,}99 \cdot 10^{-6} \qquad (6.25)$$

E estes resultados nos permitem concluir que, em média, o desempenho médio de cada UDC é sempre um pouco melhor que o da UDC # 7, o que, por mera

coincidência, é um resultado muito bom para a garantia de que todas as placas estão dentro dos parâmetros desejáveis de desempenho.

6.6.4. OBJETIVO DE DESEMPENHO

Na seção 1.3.1. do capítulo 1 apresentamos a única restrição de desempenho imposta ao decodificador, que é fazer com que a taxa de erro de bit à saída do decodificador, média, seja menor que 1,0 x 10^{-7}, para uma relação sinal/ruído E_{bi}/N_o de 8,5 dB.

A partir da figura 6.6. obtemos os seguintes resultados de desempenho da UDC # 7 operando com decisão suave:

a) para E_{bi}/N_o = 8,5 dB, a taxa de erro de bit à saída é de 1,0 x 10^{-9};

b) a taxa de erro de bit à saída é de 1,0 x 10^{-7} quando E_{bi}/N_o valer aproximadamente 7,35 dB, permitindo ao restante do modem (exceto o quantizador, já incluído na medida) ter uma margem de implementação de 1,15 dB para acomodar degradações de recuperação de portadora, recuperação de relógio, controles automáticos de ganho e de freqüência (CAG e CAF), ruídos de fase e espúrios de osciladores, intermodulações, filtragens, etc.

Portanto, a conclusão final sobre o desempenho do hardware projetado é:

1) O HARDWARE PROJETADO ATENDE AO REQUISITO DE DESEMPENHO EM TAXA DE ERRO DO SISTEMA E, JUNTO COM O CONVERSOR A/D DE 3 BITS, AINDA DEIXA UMA MARGEM DE IMPLEMENTAÇÃO DE 1,15 dB AO RESTANTE DOS EQUIPAMENTOS DAS ESTAÇÕES QUE COMPÕEM O SISTEMA AMDT VIA SATÉLITE.

Para complementar esta conclusão, fornecemos abaixo os resultados medidos do sistema, com o modem realizando um loop a nível de FI de 70 MHz e com introdução de ruído branco, aditivo e gaussiano, no ponto de operação de C/No de 66 dB/Hz, onde C é a potência total do espectro modulado. A UDC utilizada neste teste foi a UDC # 7 e a medida feita foi o número de erros em 1 x 10^{+8} bits à saída.

C/N_0 (dB/Hz)	E_{bi}/N_0 (dB)	Faixa de BERENT	Número Mínimo de Erros	Número Máximo de Erros	Número de Medições	$\overline{BERSDS}$
66,0	8,725	8,0 E -3 até 1,0 E -2	3	55	285	2,35 E -7 (σ = 1,98.10-7)

TABELA 6.12. RESULTADOS DO TESTE DO MODEM COMPLETO

6.7. COMPARAÇÕES COM OUTRAS ARQUITETURAS

Quando iniciamos este trabalho, em 1986, a única arquitetura de hardware, proposta e implementada, para um decodificador por decisão suave de códigos de bloco, publicada na literatura, era aquela concebida por Lee & Weng [26], e que comentamos no capítulo 3.

Recentemente, porém, o interesse em decodificação por decisão suave de códigos de bloco vem aumentando. Por causa disto, Abbaszadeh & Rushforth [7] e Koga [24] propuseram, em 1987 e 1989, respectivamente, duas novas arquiteturas para a decodificação por decisão suave de códigos de bloco.

Assim, para finalizarmos este trabalho, compararemos nesta seção a arquitetura que propusemos com as outras três, sob os pontos de vista de aplicação, desempenho e máxima velocidade de operação.

Lee & Weng [26] propuseram e implementaram uma arquitetura de baixa complexidade para decodificar o CGE (24, 12) por decisão suave com quantização de 2 bits (4 níveis), usando o algoritmo chamado "weighted erasure decoding". Esta arquitetura utiliza dois decodificadores por decisão abrupta, sendo um para o bit mais significativo de saída do quantizador e o outro para o bit menos significativo. Um comparador associado a uma lógica de chaveamento seleciona o decodificador que fornece a seqüência de saída, em função do número de erros apresentados por cada decodificador e do peso dado a cada decodificador. Devemos ressaltar que estes pesos variam em função do tipo de canal (AWGN, Rayleigh, etc), do valor de Eb/No e dos limiares de quantização. No algoritmo de Chase, a confiabilidade é função somente dos limiares de quantização e do tipo de canal. A arquitetura escolhida levou aos seguintes resultados:

a) o hardware para o CGE (24, 12) usou 90 CIs TTL MSI e 4 PROMs;

b) a velocidade máxima deste hardware é de 4 Mbit/s para bits de informação e 8 Mbit/s para os dígitos no canal;

c) os ganhos de codificação, operando em decisão suave, foram de 3,0 dB para taxas de erro de bit de $1{,}0 \times 10^{-5}$ e 2,5 dB para $1{,}0 \times 10^{-3}$; as curvas plotadas em [26] indicam uma melhoria da decisão suave sobre a decisão abrupta em cerca de 1 dB para taxas de erro de bit à saída variando entre 1×10^{-2} a 1×10^{-6}.

Lembrando que a arquitetura que propusemos para o CGE (24,12) tem:

a) hardware com 124 CIs (116 CIs MSI, 3 EPROM, 2 PROM e 3 RAM), incluindo o medidor de taxa de erro (7 CIs) e as interfaces de entrada/saída (3 CIs), funções estas não existentes no hardware de Lee & Weng;

b) ganhos de codificação, em decisão suave, de 3,65 dB para taxa de erro de bit de 1×10^{-5} e 2,9 dB para 1×10^{-3} e uma melhoria da decisão suave sobre a decisão abrupta de 1,5 dB para taxas de erro de bit menores que 1×10^{-3} (ver tabela 6.11);

c) taxa de transmissão máxima da ordem de 1,25 Mbit/s para bits de informação e 2,5 Mbit/s para o canal, no caso de usarmos RAMs com 25 ns de tempo

de acesso, mantendo-se a tecnologia HCMOS no restante do hardware, concluímos que a arquitetura proposta neste trabalho tem um desempenho 0,5 dB superior ao da proposta por Lee & Weng, na faixa de interesse de taxas de erro de operação, a um custo ligeiramente maior, porque o número de CIs adicionais é de apenas 20 CIs (em torno de US$ 50, se cada unidade custar US$ 2,5). Como a aplicação é a mesma, comunicação digital via satélite, se tivéssemos que ganhar estes 0,5 dB na antena, HPA ou LNA, o custo seria muitíssimo mais alto. Todavia, nossa arquitetura é tecnologicamente inviável para operar em taxas de informação na faixa de 4 Mbit/s pois, apesar da lógica poder ser transformada de tecnologia HCMOS para TTL-LS ou TTL-S, memórias RAMs com tempos de acesso menores que 25 ns só são disponíveis em tecnologia ECL; no caso desta troca de tecnologia de memórias ser feita, as implicações seriam, no mínimo, o uso de conversores de nível (já que a tecnologia ECL é não compatível com a TTL em termos de características elétricas) e, no máximo, a troca de toda a lógica para tecnologia ECL. De qualquer forma, a opção que se fizesse acarretaria em profundas mudanças no hardware por nós concebido, fosse a nível de número de CIs ou a nível de consumo, dissipação, lay-out e mesmo projeto lógico.

Outra arquitetura, proposta em 1987 por Abbaszadeh e Rushforth [7], foi concebida para ser implementada por CIs dedicados VLSI, em tecnologia N-MOS de 1,3 μm, para o CGE (24, 12). Esta arquitetura usa como algoritmo de decodificação o algoritmo de Conway-Sloane [15], proposto em 1986, e não analisado no capítulo 3 deste trabalho. Este algoritmo é de máxima verossimilhança, porém só é aplicável a códigos corretores de erro que contenham sub-códigos (i.e., subgrupos) cujos arranjos-padrões tenham poucas classes laterais e que possam ser decodificados de forma simples e rápida. No caso do CGE (24, 12), existe um sub-código (24, 5) com 128 classes laterais. O algoritmo exige decodificação por mínima distância euclidiana (cálculo dos produtos internos ou correlação) para cada uma das classes laterais, e consegue realizar a decodificação por máxima verossimilhança muito rapidamente (com cerca de 1/30 das operações requeridas por um decodificador por correlação). A arquitetura foi dividida em 2 CIs dedicados com operação paralela e em "pipeline". Ela assume que o vetor recebido foi quantizado em 16 níveis (4 bits) e representado em complemento de 2 (i.e., a quantização não produz à saída um dígito mais significativo de polaridade e os outros dígitos não são simétricos ao redor do limiar "0", como nos algoritmos que envolvem confiabilidades). A organização do hardware possui as seguintes características:

a) um primeiro chip, chamado pré-processador (PPC), calcula 6 valores associados à decodificação do sub-código;

b) um segundo chip, chamado calculador de produto interno (IPC), decodifica o sub-código (24, 5) para 128 classes laterais e seleciona a palavra final decodificada para o CGE (24, 12);

c) o IPC é baseado em 8 processadores internos seqüenciais, cada um processando 16 classes laterais e formando, no conjunto, um "pipeline";

d) o tempo de atraso de decodificação é de 576 ciclos do relógio principal dos processadores e cada bloco dura 144 ciclos deste mesmo relógio; portanto, o atraso de decodificação é de 4 blocos e a freqüência de símbolos é de (12/144) vezes o relógio principal;

e) para a tecnologia empregada, os autores [7] acreditam que o relógio principal será de 5 MHz a 7 MHz, o que fornece uma taxa de transmissão de bits de informação de 416 kbit/s a 625 kbit/s, no máximo.

Os autores não forneceram no trabalho nenhum dado referente a desempenho medido, nem a possíveis aplicações.

Desta forma, comparada à nossa proposta, podemos concluir que a arquitetura descrita em [7] tem aplicação limitada a códigos que contenham subgrupos de fácil decodificação por correlação e, ainda assim, tem atraso de decodificação maior e velocidade de operação máxima menor que em nossa arquitetura, além de exigir relógios de freqüências mais elevadas que no nosso hardware aplicado ao CGE (24, 12). Todas estas desvantagens decorrem do fato de que esta arquitetura exige 144 ciclos de máquina e a que propomos exige 128 ciclos de máquina. Além disso, esta arquitetura [7] só é implementável usando-se CIs dedicados pois sua implementação com CIs comerciais é inviável dado o alto número de funções que seriam requeridas. Nossa arquitetura foi implementada com CIs comerciais porém também é integrável, e um CI dedicado não é viável no Brasil dada a baixíssima demanda, que não proporcionaria economia de escala.

Por último, há uma arquitetura, proposta em 1989 por Koga [24], voltada a aplicações de transmissão digital via satélite em sistemas AMDT de altíssima velocidade (120 Mbit/s) do INTELSAT. Tal arquitetura é, necessariamente, em "pipeline" e usa um algoritmo de Chase modificado pelo autor, para operar com o código BCH (127, 112) corretor de erros duplos/detector de erros triplos, adotado pelo INTELSAT no sistema AMDT de alta capacidade (120 Mbit/s). Para conseguir operar em alta velocidade, Koga procura alterar o número de padrões de teste exigido pelo algoritmo, pois conforme observa , o algoritmo 1 é sempre impraticável e os algoritmos 2 e 3 só dão bons resultados quando há somente um pequeno número de posições menos confiáveis (da ordem de $d/2$) na palavra recebida, como ocorre no CGE (24, 12) e outros códigos (n,k,d) com relação n/d pequena. Quando a relação n/d é grande, como no código BCH (127, 112), outros critérios para seleção do conjunto de padrões de teste são desejáveis. Assim, Koga procura, no seu trabalho, avaliar outros métodos de geração de padrões de teste, propondo um método onde os pesos de Hamming dos padrões de teste são fixados em função do número "aparente" de erros (decodificação binária), e o número de padrões de teste e os níveis de quantização e seus espaçamentos são variados. Propõe, então, o uso de quantização de 4 bits (16 níveis) uniforme, com espaçamento 0,12, que fornece uma melhoria de 1 dB sobre a decodificação abrupta na taxa de erro de bit de 1×10^{-4} e com apenas 4 padrões de teste, ou um ganho sobre os sistemas não codificados de 4 dB na taxa de erro de bit de 1×10^{-6}, para 7 padrões de teste ou mais. Feita esta análise, propõe uma arquitetura similar à nossa, e que possui um bloco que estima o número

aparente de erros a partir da síndrome obtida do vetor recebido. Então, analisa esta arquitetura sob os aspectos relacionados às dificuldades de implementação (tempos de acesso de memórias, tecnologia a ser usada, etc). Conclui que não haveria dificuldades substanciais para a implementação da arquitetura operando a 120 Mbit/s.

Comparada à nossa arquitetura, concluímos que, apesar de não ter sido implementada por hardware específico, o diagrama de blocos das duas é bastante similar, sendo que Koga propõe tecnologia ECL para operar a 120 Mbit/s; o atraso de decodificação na arquitetura dele é de 3 blocos ao invés de 2, porém o número de padrões de teste pode cair para quatro ao invés dos 8 necessários ao algoritmo 2 de Chase (porque, neste código BCH, d vale 6), com um desempenho, obtido por simulação, em torno de 0,7 dB pior que a decodificação por correlação.

CAPÍTULO 7

CONCLUSÕES E SUGESTÕES PARA FUTURAS PESQUISAS

Este capítulo final está dividido em 2 seções: a primeira reapresenta as principais conclusões deste trabalho, já fornecidas ao final de cada capítulo ou das seções dos capítulos. A segunda seção apresenta algumas idéias e sugestões nossas para a extensão deste trabalho em outras aplicações.

7.1. CONCLUSÕES

Este trabalho apresentou uma arquitetura digital para decodificadores por decisão suave de códigos corretores de erros de bloco, lineares, sistemáticos e binários, usando o algoritmo 2 de Chase. Especificamente, a arquitetura proposta foi implementada com sucesso para o código de Golay estendido (24, 12), usando-se CIs SSI e MSI comercialmente disponíveis, para uso no modem do sistema SAMSAT desenvolvido no CPqD-TELEBRÁS para comunicações de dados via satélite, funcionando a 534 kbit/s de taxa de informação e 1068 kbit/s de dígitos.

Para o desenvolvimento deste trabalho, iniciamos, no capítulo 2, revendo que o uso de decisão suave na decodificação de códigos corretores de erro de bloco, lineares e binários, com transmissão antipodal, resulta em um ganho assintótico de 2 dB sobre sistemas que usem decodificação com decisão abrupta, assumindo que a decisão suave é não quantizada. No caso de quantização linear de 8 níveis (3 bits), a degradação no desempenho e, conseqüentemente, no ganho assintótico vale 0,22 dB, independentemente do código utilizado.

No capítulo 3, revimos diversos algoritmos para a decodificação por decisão suave de códigos de bloco, analisando-os como possíveis alternativas ao algoritmo de Chase tanto do ponto de vista de desempenho bem como do ponto de vista da implementação. Os algoritmos revistos foram os de Massey, de Hartmann-Rudolph, de Greenberger, de Weldon e os de Omura e decodificação por síndrome parcial. Todos estes algoritmos são sub-ótimos ou ótimos assintoticamente, o que já é uma desvantagem destes algoritmos em relação ao algoritmo de Chase, que é ótimo,

porque o desempenho deles é pior do que o desempenho oferecido pelo algoritmo de Chase. Assim, eles só seriam vantajosos caso apresentassem grandes simplificações de hardware nas suas implementações e ainda assim atingissem a velocidade de transmissão requerida. As principais conclusões sobre estes algoritmos foram as seguintes:

a) *de Massey* – a implementação é simples para canais binários do tipo apagamento ou quando só se usa um único valor de confiabilidade, associada ao dígito de menor confiabilidade do bloco. Mesmo assim, ela leva a circuitos com um número comparável de CIs em relação à arquitetura do algoritmo de Chase e o desempenho fica cerca de 0,5 dB pior que o obtido via algoritmo de Chase; como vantagem, o circuito pode atingir velocidades mais altas que o nosso;

b) *de Hartmann-Rudolph (HR)* – o processo utilizado para a implementação é muito mais complexo que o de Chase, porque estão envolvidas muitas somas de verificação de paridade; conseqüentemente, o circuito para o HR é bem maior que o exigido pelo de Chase e só passa a ser vantajoso para taxas de código (k/n) bem alta. Como exemplo, o HR precisa computar 1024 palavras no caso de CG (23, 12), o que inviabiliza sua implementação em hardware, exceto os baseados em microprocessadores, para velocidades baixíssimas.

c) *de Greenberger* – reduz as 1024 palavras do HR para 64, no caso de CG (23, 12). Pode ser implementado em microprocessadores, mas devido à limitação de relógio destes microprocessadores e à extensão de cálculos, só são viáveis para taxas de transmissão baixas, menores que 64 kbit/s;

d) *de Weldon* – é de fácil implementação em hardware, viabilizando circuitos simples e alta velocidade de operação, porém seu desempenho é bem pior que o dos outros algoritmos;

e) *de Omura* – apesar de ser teoricamente um algoritmo ótimo, sua implementação em hardware é dificílima porque, em primeiro lugar, os conjuntos de paridade exigem muitas operações para serem construídos e têm tamanhos variáveis e depois o atraso de decodificação é variável. Assim, a implementação só é viável usando-se computadores de porte razoável rodando o algoritmo em software. Mesmo assim, a taxa de bits não pode ser muito alta para não comprometer o atraso de decodificação. Pelo mesmo motivo, o número de padrões a testar deve ser limitado e, por causa disso, o desempenho cai muito rapidamente. Ao contrário, o algoritmo de Chase, que também é ótimo, tem um desempenho que cai lentamente em função de redução do número de padrões de erro testados e os circuitos para sua implementação são muito mais baratos que um computador. Pelos mesmos motivos expostos acima, a decodificação parcial de síndrome, similar ao algoritmo de Omura porque ambos se baseiam em "Conjuntos de informação" [14], não é alternativa viável para substituir eventualmente o algoritmo de Chase.

O algoritmo de Chase e suas variantes generalizadas e a proposta por Hackett foram revistos no capítulo 4. Primeiramente, apresentamos o algoritmo de Chase de forma mais intuitiva que a descrita por ele no trabalho original [13], para realçarmos a idéia básica do algoritmo baseados mais em conceitos físicos (que

regem o canal e seu ruído) do que em expressões matemáticas. Em seguida, descrevemos formalmente o algoritmo de Chase tal como proposto por ele e revimos as três formas do algoritmo. Revimos, então, que o algoritmo de Chase é ótimo pois equivale a um receptor de máxima verossimilhança. Em função do conjunto de padrões de teste escolhido, pode ocorrer uma degradação ínfima de desempenho. Apresentamos então o algoritmo de Hackett, que é uma variação do algoritmo 2 de Chase, reduzindo o número de padrões de teste exigidos. Este algoritmo de Hackett apresenta uma pequena degradação em relação ao algoritmo 2, mas simplifica a decodificação quando implementado em microprocessadores, para taxas baixas de transmissão. Para o caso de hardware dedicado, a arquitetura não ficaria muito menos complexa que a arquitetura necessária para o algoritmo 2 de Chase. Foi feita então uma revisão da generalização do algoritmo de Chase proposta por Tendolkar e Hartmann para simplificar a implementação nos casos em que o código usado tenha distância mínima muito grande e o decodificador por decisão abrupta, "interno" ao decodificador por decisão suave, tenha que ser complexo.

O capítulo 5 apresenta o que consideramos o cerne deste trabalho: a arquitetura que concebemos para o decodificador usando o algoritmo dois de Chase, e que foi implementada para o CGE (24, 12) usado no sistema SAMSAT, respeitando-se as restrições de implementação impostas por este sistema – apresentadas no cap. 1, seção 1.3.2., que são:

a) transmissão por surtos e sistema de modulação QPSK coerente;

b) quantização das 2 saídas do demodulador em 3 bits (8 níveis), sendo o bit mais significativo correspondente ao dígito de polaridade e os outros dois fornecem uma medida do nível de confiabilidade do símbolo recebido segundo a tabela da fig. 1.9;

c) a resolução das 4 ambigüidades de fase da demodulação é realizada previamente à decodificação, no circuito detector de palavras únicas;

d) a decomutação das 2 linhas demoduladas deve ser feita de forma a remontar o bloco comutado em 2 linhas pelo codificador;

e) o decodificador recebe um sinal de habilitação indicador do início da porção de dados codificados de um surto (o preâmbulo não é codificado);

f) o relógio de símbolos é de 534 kHz e o de dígitos de 1068 kHz; a borda de descida dos relógios indica o meio do período do símbolo ou do dígito;

g) o atraso de decodificação deve ser o menor possível, em torno de dois a três blocos (palavras-código).

Iniciamos este capítulo 5 mostrando a inviabilidade ou a inconveniência de arquiteturas que usassem microprocessadores ou processadores digitais de sinais (DSPs).

Em seguida, apresentamos como concebemos a arquitetura a partir da divisão do algoritmo 2 de Chase em processos paralelos, que manipulam os dados de forma ligeiramente diferente da proposta por Chase, porém de forma mais vantajosa em termos práticos: o peso analógico computado é o peso analógico dos vetores erro e não o da palavra decodificada; não usamos vetores de comprimento n, mas só os

endereços de posições cujos dígitos valham 1; finalmente, não usamos cálculo de síndromes na sua forma convencional para calcular a síndrome associada a cada um dos vetores-teste exigidos pelo algoritmo de Chase, mas sim o resultado da soma da síndrome primária com uma síndrome de teste, sendo esta última obtida pela soma das até 4 linhas da matriz **[H]** endereçadas pelos até 4 dígitos **"1"** do padrão de teste vigente, no caso do CGE (24, 12). O macro-processo de decodificação é apresentado na fig. 5.3 e ocorre em duas etapas.

Na primeira etapa, que dura todo o período de um bloco, ocorrem a recepção e armazenamento dos símbolos demodulados, quantizados e sem ambigüidade de fase (dígito Y de polaridade e par de dígitos α de confiabilidade associada a Y); o cálculo e armazenamento da síndrome primária, associada ao vetor **Y**; e a procura e armazenamento dos endereços das 4 posições com as menores confiabilidades dentro do bloco recebido.

Na segunda etapa, que dura todo o período do bloco subseqüente, ocorrem a obtenção e armazenamento dos 16 padrões de teste; cálculo de uma síndrome de teste associada ao padrão de teste vigente; soma bit-a-bit das síndromes primária e de teste; decodificação binária, obtendo um padrão de erros associado à síndrome-soma; **"soma"**, através de multiplexação, seguida de análise de coincidência de endereços, do padrão de erros com o padrão de teste vigente; cálculo do peso analógico associado; seleção do menor peso analógico através de comparações sucessivas; armazenamento das posições em erro (**"soma"** do padrão de erros e padrão de teste) correspondentes ao padrão de menor peso analógico; cálculo da taxa de erro de bit; e a transferência dos bits de informação a um registrador de saída. A partir desta etapa, a decodificação está completa, restando apenas a transferência dos bits de informação à saída, com conversão de velocidade e correção dos bits errados através de inversão lógica nas posições em erro armazenadas anteriormente. Todo este processo ocorre durante o segundo bloco subseqüente, mais um atraso de 2 bits de informação.

Após termos montado o macro-processo de decodificação baseado no algoritmo 2 de Chase, mostramos, através de diagrama de blocos, a concepção do hardware que implementa cada processo principal da decodificação para o CGE (24, 12). Esta concepção é posteriormente generalizada para cada processo, obtendo-se os parâmetros principais dos circuitos (freqüências de relógio, número de registros, valores máximos de variáveis) em função dos parâmetros do código (n, k, d) e da taxa de transmissão de dígitos Rd. Desta forma, analisamos: a decomutação dos canais demodulados; o cálculo da síndrome primária; a procura das 4 posições de menor confiabilidade, em um processo que não exige ordenação de posições; a obtenção dos 16 padrões de teste; o cálculo das síndromes de teste; a tabela decodificadora para decodificação binária; o cálculo do peso analógico com análise de coincidência de endereços; e finalmente, a correção das posições em erro e cálculo da taxa de erro de bit.

De posse destes componentes, pudemos então apresentar o diagrama de blocos do decodificador por decisão suave usando o algoritmo 2 de Chase para o CGE (24, 12), com os seguintes blocos:

a) contador mestre de posições;
b) conversor paralelo-série de entrada;
c) gerador de síndrome primária;
d) caçador de posições menos confiáveis;
e) memória de dados e confiabilidades;
f) gerador de padrões de teste;
g) gerador de síndromes de teste;
h) somador de síndromes;
i) tabela decodificadora;
j) analisador de coincidência de endereços;
k) analisador de métricas;
l) memória de posições em erro;
m) corretor de erros;
n) medidor de taxa de erro;
o) interfaces de entrada e saída do decodificador.

Finalmente, ainda no capítulo 5, mostramos como racionalizamos o uso de memórias na arquitetura, usando técnicas de paginação, para minimizar o número de CIs envolvidos no armazenamento de dados, confiabilidades e posições em erro, e ainda assim permitir o compartilhamento de memória entre os processos. Cabe aqui evidenciarmos que a arquitetura concebida permitiu a sua implementação através de CIs "standard", disponíveis comercialmente, e que a mesma se vale de técnicas modernas de projeto digital tais como processamento paralelo, "pipelining", e paginação de memórias.

Por último, o capítulo 6 apresentou os resultados que obtivemos com a arquitetura que propusemos, implementada para o caso do CGE (24, 12). Demos a descrição detalhada do projeto lógico e elétrico de cada bloco que compõe o hardware, cujo empacotamento eletromecânico e lay-out dos componentes (124 CIs, 1 conector, capacitores, resistores, diodos e LED), em uma placa de 13,05 x 8,87 pol^2, resultou em densidade de 0,63 pol^2/CI equivalente e cujo consumo final, foi, na média, de 1,25 A com um "ripple" de fonte acarretado pelo "lay-out" de no máximo 0,55 V pico a pico.

Mostramos em seguida que, para o CGE (24, 12), o hardware proposto é aquele que minimiza o número de CIs comerciais (disponíveis atualmente, sem serem microprocessadores) exigidos para a implementação do algoritmo 2 de Chase no sistema AMDT.

Finalmente, apresentamos os resultados de desempenho medidos em laboratório para 10 placas com o hardware proposto para decodificar por decisão suave o CGE (24, 12). Estes resultados mostraram excelente aderência às previsões teóricas e, para terminar esta seção de conclusões fornecemos a seguir as principais medidas realizadas:

a) para a placa UDC # 7, obtivemos, em função das medidas diretas da taxa de erro de entrada versus taxa de erro de saída, várias curvas indicativas do desempenho, inclusive, por método indireto, a curva E_{bi}/N_o versus BER de saída;

b) na UDC # 7, para uma taxa de erro de bit à saída de $1,0 \times 10^{-4}$, os ganhos estimados da decisão suave sobre a decisão abrupta e sobre a transmissão não-codificada foram, respectivamente, de 1,55 dB e 3,50 dB;

c) idem, para taxa de erro de saída de $1,0 \times 10^{-5}$, os ganhos foram de 2,15 dB e 3,65 dB (estimados indiretamente);

d) os ganhos, na taxa de saída de $1,0 \times 10^{-10}$, foram de 2,85 dB e 4,15 dB (estimados indiretamente), para a UDC # 7, muito próximos aos valores teóricos;

e) pudemos, através das medidas, estabelecer relações lineares entre os logaritmos das taxas de erro. Estas funções de transferência de taxa de erro da UDC # 7 são dadas pelas equações (6.22) e (6.23), para os casos de decisão abrupta e suave, respectivamente;

f) para o conjunto de 10 placas, na taxa de erro de entrada de $2,0 \times 10^{-2}$, a taxa de erro de bit à saída teve média de conjunto de $1,44 \times 10^{-6}$, todas as placas com resultados parecidos (o desvio padrão foi de $0,99 \times 10^{-6}$);

g) para a UDC # 7, com E_{bi}/N_o de 8,5 dB, a taxa de erro de bit à saída foi de $1,0 \times 10^{-9}$ e é de $1,0 \times 10^{-7}$ para E_{bi}/N_o de 7,35 dB; a margem de implementação restante é, portanto, de 1,15 dB.

Logo após este capítulo 7, fornecemos ainda dois apêndices.

O apêndice A fornece métodos de teste convenientes para a caracterização da arquitetura implementada a nível de circuito para o CGE (24, 12).

Por último, fechando este trabalho, apresentamos, no apêndice B, exemplos esclarecedores que nos mostram claramente a forma de atuação do algoritmo de Chase no sentido de aumentar a capacidade corretora do código para padrões de erro com peso de Hamming podendo valer até $(d-1)$.

7.2. EXTENSÕES

Procuramos, neste trabalho, mostrar a concepção de uma arquitetura de um decodificador por decisão suave utilizando o algoritmo 2 de Chase para o CGE (24, 12), que pudesse ser usada no sistema AMDT de transmissão de dados via satélite SAMSAT, e que fosse implementável a partir de CIs digitais "standard" comercialmente disponíveis.

Esta arquitetura pode ser usada para outros códigos corretores de erro de bloco, binários e lineares, e as condições de contorno para esta extensão também foram apresentadas neste trabalho.

Existem, todavia, outras extensões e aplicações desta arquitetura que exigem trabalho não desenvolvido aqui.

Assim, é certo que é necessária uma maior investigação sobre os limites da arquitetura aqui proposta quando usada com outros códigos. Ou seja, apesar de ser

implementável, a arquitetura pode começar a originar circuitos maiores e tornar-se, por conseguinte, anti-econômica. Basta, para isso, investigar para que faixa de valores de n, k e d ela é atrativa e, então, determinar quais códigos de bloco, binários e lineares fornecem desempenho a custos suportáveis usando o decodificador proposto.

Outro campo a explorar é a extensão desta arquitetura a sistemas de transmissão contínua que não possuam uma estrutura de quadro temporal nem usem codificação diferencial. Neste caso, há que se acoplar à arquitetura proposta circuitos que resolvam as ambigüidades de fase dos sistemas de demodulação e que forneçam o sincronismo de bloco (palavra), a partir de limiares de taxa de erro de bit, por exemplo. Técnicas de "interleaving" (rearranjo de dígitos em vários blocos) podem ser úteis neste caso, além de ajudar a combater erros em surtos do canal.

Pode-se ainda investigar se a arquitetura proposta é boa para integração, visando a produção de CIs, ou se continua a ser uma boa opção para uso em velocidades mais altas, implementando-a com CIs da família ECL[38].

Por último, sem se importar com a arquitetura aqui proposta, poderia ser feita uma investigação mais profunda sobre o conceito de função de transferência de taxa de erro como ferramenta de caracterização e comparação de códigos corretores. Introduzimos este conceito no capítulo 6 para exprimir as características de taxa de erro de bit à saída em função da taxa de erro à entrada, através de uma relação linear simples.

APÊNDICE A

MÉTODOS DE TESTE

A.1. INTRODUÇÃO

Procuramos, neste apêndice A, descrever os métodos de teste para a validação da arquitetura e alguns procedimentos de depuração do hardware.

Dividimos os testes em três categorias, com o intuito de realçar suas finalidades. São elas:

a) testes do hardware digital, com as finalidades de validar a arquitetura, mostrar sua viabilidade de implementação e funcionar como procedimentos de depuração do hardware em laboratório e/ou produção;

b) teste do decodificador inserido no canal, com as finalidades de validar a arquitetura funcionando como decodificador em um canal AWGN contínuo e de obter as curvas de desempenho em condições de laboratório, bem controladas.

c) testes no sistema, com a finalidade de validar a arquitetura dentro do contexto de um sistema AMDT, com surtos variáveis e com um modem real, o que introduz degradações no desempenho esperado.

Para facilitar a execução dos testes, introduzimos, no hardware digital que implementa a arquitetura proposta, alguns "jumpers" (estrapes) que permitem a programação manual de determinadas alterações no hardware, necessárias à execução dos testes. São três os "jumpers", com as seguintes funções:

a) **JP1**: seleção de decodificação por decisão suave (pinos 1 e 2 em curto) ou decodificação por decisão abrupta (pinos 2 e 3 em curto). Este "jumper" atua modificando o sinal de "reset" dos registros de armazenamento dos 4 endereços das posições de dígitos menos confiáveis, do bloco caçador de posições menos confiáveis; se selecionamos decisão abrupta, os "latches" ficam permanentemente em nível lógico 0 (zero), produzindo, por bloco, 16 vetores de teste iguais e nulos;

b) **JP2**: seleção de correção de bits ativada (pinos 1 e 2 em curto) ou desativada (pinos 2 e 3 em curto). O "jumper" atua diretamente sobre a entrada da porta ou-exclusivo que recebe os pulsos de inversão de bit; com a correção desativada, esta

entrada da porta fica permanentemente em nível lógico 0, sem receber então os pulsos de correção dos bits;

c) **JP3**: seleção de limiar de alarme de taxa de erro detectada à entrada do decodificador em $2^{-6} = 1{,}56 \times 10^{-2}$ (pinos 1 e 2 em curto) ou $2^{-7} = 7{,}81 \times 10^{-3}$ (pinos 2 e 3 em curto). O jumper atua diretamente sobre o período de observação da taxa de erro, i.e., sobre o número de bits que constituem a população amostral, sem alterar o módulo da contagem de erros, i.e., sem variar o número de eventos "erro em bit" que provocam o estouro ou transbordo do contador de erros.

A.2. TESTE DO DECODIFICADOR INSERIDO NO CANAL

O decodificador está, normalmente, inserido dentro de um modem, que realiza a interface com o canal. Este modem, normalmente, realiza translações em freqüência do sinal digital codificado e devidamente filtrado por um filtro de Nyquist e, na demodulação, ainda há as recuperações de portadora e relógio.

Assim, se fôssemos verificar o desempenho do decodificador, estaríamos sujeitos a sérias restrições de funcionamento e também a erros, introduzidos por outros circuitos, na distribuição de ruído. Estes erros são associados às não-linearidades da modulação/demodulação e conversões de freqüência, que introduzem intermodulações e espúrios de osciladores, aos erros de fase do modulador, demodulador, aos erros de fase dos recuperadores de portadora e de relógio, ao tremor do relógio recuperado e, em sistemas AMDT, às perdas e falsas detecções de palavra única (que leva à perda total de um surto).

Portanto, para verificar o desempenho do decodificador em um canal AWGN, temos que montar um "set-up" de medição com condições bem controladas, de forma que o ruído gerado seja realmente gaussiano e branco (com o menor erro de distribuição possível) e em que a única degradação admitida seja a associada à quantização em 3 bits do sinal analógico.

Para simplificar este "set-up" valemo-nos novamente da linearidade do código e usamos, como palavra-código transmitida pelo codificador, uma palavra nula (**0**). Com isto, não precisamos do codificador em hardware. Basta deslocarmos o ruído com um nível contínuo, através de uma polarização adequada no conversor A/D, de forma que este apresente em sua saída, na ausência de ruído, o valor 0 para os dígitos de polaridade e 11 para os dígitos de confiabilidade.

Os instrumentos necessários para este teste são o medidor de taxa de erro (detector de erros) e um gerador de ruído branco para multiplex telefônico, além de fonte, sistema de geração de relógios do decodificador, sistema de conversão A/D para o decodificador e amplificador de ruído. A fig. A.1. ilustra a configuração para os testes.

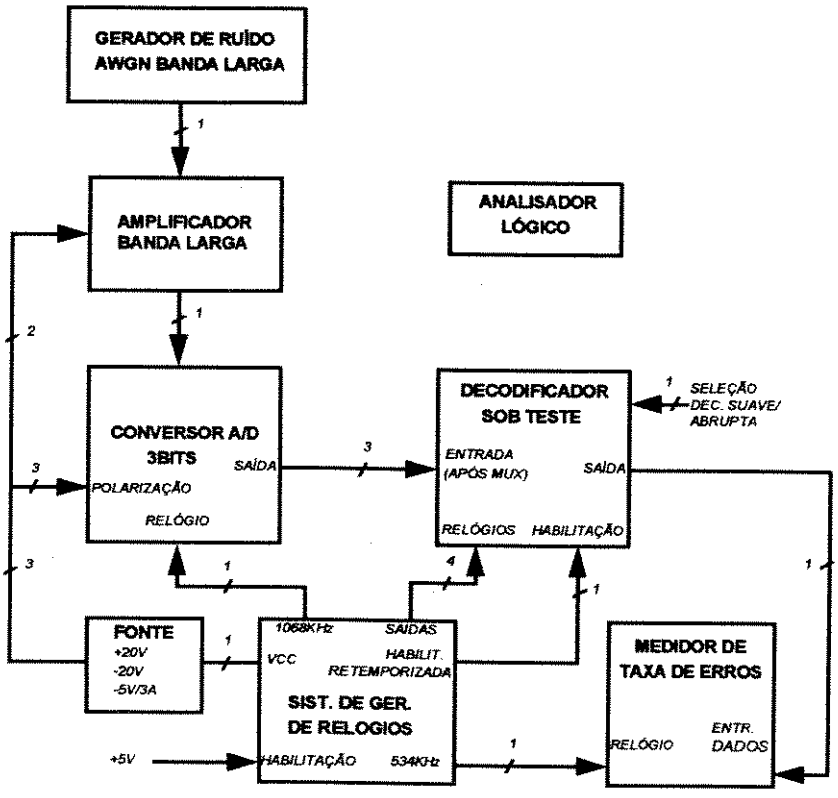


FIGURA A.1. CONFIGURAÇÃO DE TESTE DE DESEMPENHO DO DECODIFICADOR

Observando a fig. A.1., pode-se notar que os dígitos de polaridade e confiabilidades da seqüência de recepção entram no decodificador através de um único canal. Isto é conseguido por meio de um cabo ligado a um soquete de CI de forma que, ao substituirmos o CI encarregado de realizar a multiplexação dos canais no decodificador pelo soquete com cabos, os dígitos do A/D entrem no decodificador nos mesmos pinos de saída do multiplexador. Evita-se, com isso, a duplicação de geradores de ruído, amplificadores e conversores A/D. Outro fato a se notar é que não há necessidade de se fazer uma habilitação de decodificação de forma síncrona, já que a palavra-código transmitida é o vetor nulo e o sincronismo de blocos é desnecessário neste caso.

Cuidados especiais devem ser tomados para garantir a fidelidade, do ruído gerado, à distribuição gaussiana de amplitudes e à distribuição uniforme em freqüências. Na prática, usamos um gerador de ruído para multiplexadores FDM de canais telefônicos, que gera ruído branco e gaussiano em banda larga (8 kHz a 12 MHz) e que possui um conjunto de filtros passa-baixas e passa-altas para as mais diversas canalizações telefônicas. Estes filtros têm como características a altíssima atenuação na faixa de rejeição, faixa de passagem muitíssimo plana e transição brusca da faixa de passagem para faixa de rejeição (na freqüência de corte). Para impedir a entrada de ruídos gerados não-brancos (abaixo de 8 kHz, a densidade

espectral de potência de ruído aumenta muito), e para barrar ruídos que comprometam o desempenho do conversor A/D (acima de 2 MHz, o A/D tem tempos de propagação diferentes na lógica, o slew rate dos comparadores começa a influir), usamos os filtros passa-altas de 12 kHz e passa-baixas de 1052 kHz de forma a termos ruído branco e gaussiano nesta faixa. Devemos observar que esta faixa de ruído é suficiente para todos os efeitos práticos sobre o teste, porque 1052 kHz é bem maior que a mínima faixa necessária para o canal (a freqüência de Nyquist neste caso é de 534 kHz) e porque 12 kHz é bem menor que a componente espectral de ruído que afetaria um bloco de dígitos inteiro (1068/24 = 44,5 kHz). Além disso, para garantir que a potência média de ruído não varie durante o intervalo de medição das taxas de erro, devemos operar em ambiente de temperatura controlada, com oscilações em torno de +/– 2° C, já que o ruído é gerado por semicondutores específicos mas sensíveis à temperatura. O gerador usado nos testes é do tipo TF2091C da Marconi Instruments. Como as amplitudes de ruído geradas não têm nível suficiente para excitar o conversor A/D de forma a gerar altas taxas de erro, é necessário o uso de um amplificador entre o gerador de ruído e o conversor A/D. Este amplificador precisa ser muito plano na faixa de ruído utilizada e ser altamente linear na sua faixa dinâmica, de forma a amplificar linearmente amplitudes 4 vezes maiores que o valor RMS do ruído, ou seja, o ponto de compressão deve ser 12 dB maior que a maior potência média de ruído desejada. Com isto, garantimos que o ruído segue a curva gaussiana sem distorções observáveis na prática, já que o amplificador começa a saturar apenas em amplitudes com probabilidade de ocorrência menor que 4×10^{-5}. Portanto, 99,996% das amplitudes de ruído seguem a distribuição gaussiana. Fizemos um amplificador para ser usado nos testes com a as seguintes características: ganho de 20 dB em 75 Ohms, perdas de retorno de 35 dB na entrada e 26 dB na saída, faixa de passagem (-3 dB) de 3,2 kHz a 3,2 MHz, sem ondulação na faixa de passagem, excursão de 8,8V em aberto e sem compressão, usando os CIs LH0032 (amplificador de vídeo) e LH0033 (amplificador de corrente faixa larga) na configuração de amplificador cascode realimentado.

O conversor A/D de 3 bits usado nestes testes é o mesmo que se usa no demodulador QPSK, e é do tipo de comparadores em paralelo, com referências dadas por uma rede resistiva, para atender aos requisitos de velocidade de operação. As saídas dos comparadores alimentam uma lógica de interface que gera os níveis lógicos segundo a convenção dada na fig. 1.9.

Para somar a palavra-código nula ao ruído que entra no A/D, alteramos as tensões de alimentação da rede resistiva de referência de V(1) = +V1 e V(-1) = -V1 para V(1) = +2V1 e V(-1) = 0V, onde V1 representa o nível máximo do "olho demodulado", no instante ótimo de amostragem. Devemos reparar que, neste caso, não há olho demodulado, apenas mudamos as tensões da rede de referência para fornecer nível lógico 0 para o dígito de polaridade e nível 1 para os dígitos de confiabilidade na ausência de ruído, i.e., com a entrada do A/D aterrada. Para atender os requisitos de velocidade, os comparadores usados no conversor A/D precisam ser muito rápidos e ter alto "slew rate"; usamos então comparadores do

tipo LM760. Os resistores da rede resistiva de referência devem ser de precisão (tolerâncias máximas de 1% em relação ao valor teórico calculado).

O procedimento do teste deve ser o seguinte:

a) desativar a correção de erros através do JP2;

b) selecionar, através do atenuador de precisão do gerador de ruído, uma potência média de ruído de forma que a média das taxas de erro médias sem correção seja igual a uma taxa de erro de canal desejada; estas taxas de erro médias são medidas com o medidor de taxa de erro operando em modo BER (bit error rate);

c) selecionar operação em decisão abrupta através do JP1;

d) ativar a correção de erros através do JP2;

e) medir as taxas de erro médias para correção ativada, e calcular a média das taxas médias;

f) selecionar operação em decisão suave através do JP1;

g) repetir os passos d) e e) deste procedimento, obtendo a média das taxas de erro médias no caso de decisão suave;

h) repetir o procedimento todo para outras taxas de erro de canal desejadas.

Devemos aqui alertar para o cuidado que se deve ter na medida de taxas de erro médias. O medidor de taxa de erros que utilizamos completa uma medida de taxa de erro toda vez que o número de erros detectados for igual a 100, quando está em modo BER. Isto fornece uma estimação da taxa de erro média bastante boa, com tolerância e nível de confiabilidade adequadas, em torno de +/–20% e 75%, respectivamente [22]. Se não dispusermos de um detector de erros que opere desta forma mas apenas em contagem de erros, devemos sempre medir sobre uma temporização adequada de forma que o número de erros medidos seja maior ou igual a 100, sob pena de degradar a qualidade da medida.

Para comprovar a afirmação acima, lembramos que em um canal sem memória, o processo de contagem de erros de bit em uma seqüência de bits é de natureza binomial onde cada um dos N bits recebidos tem uma probabilidade p de estar em erro. Como, na prática, normalmente temos $p << 0{,}01$ e $N >>>> 1.000$, podemos aproximar a distribuição binomial por uma de Poisson. Usando então a expressão de desvio padrão da distribuição de Poisson na desigualdade de Tchebycheff, obtemos a seguinte expressão [22]:

$$\text{PROB}\left[N < \frac{1}{(1-P)\cdot p\cdot E^2}\right] \geq P \qquad \text{(A.1)}$$

onde

N = número de bits observados, i.e., população amostral
p = valor verdadeiro da taxa de erro

P = probabilidade da taxa de erro medida estar dentro do intervalo $\pm E$ de tolerância do valor verdadeiro da taxa de erro para N bits observados

E = limite de erro de medida, definido por:

$$E = \frac{|p - p'|}{p} \qquad \text{(A.2)}$$

com p' sendo a taxa de erro medida.

Manipulando algebricamente (A.1), obtemos, dados P e E desejados, o valor mínimo necessário do produto $(N.p)$. Entretanto, $(N.p)$ é o número de erros observados na população amostral. Assim se estabelecermos um erro máximo de 20% na medida com um nível de confiança de 75%, veremos que o número mínimo de erros a observar numa seqüência de tamanho qualquer é $(N.p) = 100$. Portanto, medidas feitas com menos de 100 erros são perigosamente enganosas porque além de terem mais tolerância, ultrapassam mais freqüentemente esta tolerância.

Como resultado do teste, obtemos uma série de pontos para traçar as curvas de desempenho do decodificador operando tanto em decisão suave como em decisão abrupta. Podemos então traçar as curvas de taxa de erro de saída versus taxa de erro de entrada, para os dois casos, diretamente, ou podemos traçar, por meio indireto, as curvas de taxa de erro de saída versus E_{bi}/N_O, nos dois casos. Para a curva E_{bi}/N_O ser traçada, devemos adotar o seguinte procedimento:

a) tendo a tabela dos pontos E_b/N_O versus BER para QPSK sem codificação, entramos com o BER à entrada do decodificador correspondente a um BER de saída desejado, e obtemos o E_b/N_O sem codificação;

b) para achar o ponto de E_{bi}/N_O adicionamos 3 dB ao valor de E_b/N_O encontrado acima, porque o CGE (24, 12) tem taxa 1/2;

c) plotamos o ponto de coordenadas (E_b/N_O + 3 dB, BER de saída).

Estes resultados podem, então, ser comparados aos resultados teóricos obtidos por simulações. Podemos além disso, obter o ganho em taxa de erro ponto a ponto.

O teste descrito aqui serve também para a verificação dos limiares do alarme de taxa de erro excessiva à entrada do decodificador.

A.3. TESTES NO SISTEMA

Os testes propostos nos itens anteriores são mais que suficientes para a caracterização e validação da arquitetura proposta. Ficariam faltando, no entanto, os testes do decodificador integrado ao sistema AMDT. Todavia, descrevê-los aqui foge em muito ao escopo deste trabalho, já que eles envolvem um universo muito extenso de outras unidades constituintes do sistema e também muitos instrumentos de teste, tornando inviável as suas descrições dentro deste trabalho. Além disso, a única informação adicional que obteríamos para a caracterização do decodificador é a execução de testes com transmissão em surtos e, mesmo esta informação é mais

dependente da unidade que envia o comando de habilitação de decodificação do que do decodificador, já que este foi testado, com verificação de aceitação do comando nos testes anteriores e também foi verificado o atraso de decodificação. Descrevemos, então, os testes com o decodificador funcionando no sistema de forma bastante resumida.

Uma estação do sistema possui, de forma resumida, as seguintes unidades de interface com o decodificador: modem, controlador de modem e geração de freqüências. A seqüência decodificada é, na verdade, uma linha de dados multiplexada de diversos usuários que se conectam ao sistema através de portas de acesso de diferentes velocidades e interfaces programáveis (RS-232, V.36, etc).

Assim, a medida de taxa de erro passa a ser feita escolhendo-se uma das portas e nela conectando o medidor de taxa de erro. A seqüência a ser programada na recepção deve coincidir com a seqüência de dados enviados por um gerador de padrões conectado à porta parceira do enlace estabelecido por duas estações do sistema.

Podemos então realizar os seguintes testes: todos os testes do hardware digital (os procedimentos mudam) quando as estações estão na mesma localização geográfica e conectadas em FI (de 70 MHz), de forma a podermos ter ruído AWGN controlado, ou o teste de desempenho através do canal satélite, para qualquer localização das estações. Lembramos que neste teste, além do ruído térmico (AWGN), influem as interferências geradas por intermodulação no transponder do satélite, por polarização cruzada, espúrios de outros transponders, enfim todos os espúrios e interferências geradas pelas não-linearidades do sistema e do canal satélite.

Influem também as degradações do sistema associadas às outras unidades (filtros de Nyquist, erros de fase da demodulação, recuperação de portadora, recuperação de relógio, efeito Doppler do movimento do satélite, perdas de palavras única, etc).

APÊNDICE B

EXEMPLOS ESCLARECEDORES
- PADRÕES ALTERNATIVOS PARA O TESTE DO DECODIFICADOR -

Os padrões que serão descritos neste apêndice servem para ilustrar como é que a decisão suave pode aumentar a capacidade corretora do sistema de codificação-decodificação. Devemos ressaltar que estes mesmos padrões auxiliaram a depuração (em laboratório) do hardware projetado.

B.1. PADRÃO DE TESTE A1

Seja o padrão de teste dado pelos seguintes vetores de polaridade e de confiabilidade:

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A1)	0	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1	1
α (A1)	3	1	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3

(B.1)

Estes vetores representam a forma de onda quantizada de uma seqüência de dígitos recebidos para ser decodificada.

Vamos mostrar o processo de decodificação passo por passo para os casos de decisão abrupta e de decisão suave, e os resultados das seqüências decodificadas comparadas com a palavra recebida estão ilustrados nas figuras B.1 a), b) e c).

Inicialmente, suponhamos o processo de decodificação por decisão abrupta como sendo o escolhido.

Neste caso, o único vetor a ser observado é o vetor dos dígitos de polaridade Y (A1) e todos os cálculos são feitos com base nas distância e peso de Hamming. No hardware, programado para decisão abrupta através de um estrape JP1 – ver apêndice A – o vetor das confiabilidades α(A1) é desprezado, forçando-se os quatro registradores de armazenamento dos endereços de posições dos dígitos menos

confiáveis a terem valor 0 permanentemente, através de suas linhas de reset. Isto acarreta que as 16 decodificações realizadas pelo hardware, no tempo de duração de um bloco, serão todas iguais e correspondentes àquela que seria realizada por um decodificador binário, já que as sucessivas métricas calculadas corresponderão não aos pesos analógicos dos padrões de erro associados, mas apenas ao peso de Hamming do padrão de erro dado pela tabela decodificadora multiplicado pelo valor correspondente de confiabilidade que os dígitos podem assumir (em nosso caso, vale 3 no máximo).

Então, lembrando o processo de construção da tabela decodificadora analisado na seção 5.8, que, para padrões de 4 erros, só contempla aqueles que englobem um erro no primeiro dígito do bloco, vemos que a palavra-código que possui a menor distância de Hamming em relação à seqüência **Y** (A1) é a palavra que tem os 24 dígitos **"1"**, isto é:

$$\mathbf{C}_H\,(A1) = \mathbf{1} = (1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1) \qquad (B.2)$$

Neste caso, o padrão de erro correspondente seria o de posições em erro dadas por PE (A1)= (1, 7, 13 e 19):

$$\mathbf{E}\,(A1) = (1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0) \qquad (B.3)$$

cujo peso de Hamming vale 4. A síndrome correspondente vale o ou-exclusivo bit-a-bit das colunas 1, 7, 13 e 19 da matriz de verificação de paridade do CGE (24, 12), ou seja:

$$\mathbf{S}(A1) = \mathbf{Y}.[H]^T = (0001\ 1111\ 0100) = 2F8_H \qquad (B.4)$$

e **[H]** é a matriz de verificação de paridade do CGE (24, 12) dada pela eq. (1.21c).

Os resultados acima são exatamente aqueles que são obtidos se acompanharmos o processo executado pelo hardware:

a) deslocamento sucessivo da seqüência **Y** (A1) através do registrador de síndromes da figura 5.5, 23 vezes e através do acumulador de paridade total (fig. 5.5), 24 vezes. Obtemos a mesma síndrome primária;

b) endereçamento da tabela decodificadora através da síndrome primária, já que as síndromes-testes serão sempre **0**. Usando a notação da fig. 5.10., obteremos, por construção da tabela:

$$PE1\,(A1) = 1;\ PE2\,(A1) = 7;\ PE3\,(A1) = 13;\ PE4\,(A1) = 19 \qquad (B.5)$$

c) uso destes PEi, $1 \leq i \leq 4$, e dos dígitos de teste DTi:

$$DT1 = DT2 = DT3 = DT4 = 0 \qquad (B.6)$$

como endereços das confiabilidades a serem somadas pelo somador de métricas,

d) obtenção de 16 métricas iguais que valerão:

$$W\alpha\ (\mathbf{A1}) = \alpha(1) + \alpha(7) + \alpha(13) + \alpha(19) + 0 + 0 + 0 + 0 = 12 \qquad (B.7)$$

devemos reparar que este valor é o peso de Hamming do padrão de erro obtido multiplicado pelo valor das confiabilidades associadas;

e) inversão dos dígitos acusados como estando em erro pelo padrão de erros PEi (A1), $1 \leq i \leq 4$, obtendo a palavra-código com mínima distância de Hamming em relação ao vetor **Y** (A1) recebido, ou seja o vetor 1.

f) transferência dos 12 primeiros dígitos à saída, pois, sendo o CGE (24, 12) utilizado um código sistemático, estes são os próprios bits de informação.

Suponhamos agora que o processo de decodificação seja o por decisão suave. No hardware, devemos usar a programação normal do estrape JP1, e agora tanto **Y** (A1) como α(A1) serão usados nos cálculos, feitos agora com base nos conceitos de distância e peso analógicos introduzidos por Chase e definidos no capítulo 4.

Para a decisão suave usando o algoritmo 2 de Chase devemos inicialmente buscar, por inspeção no vetor α, quais as 4 posições dos dígitos de menor confiabilidade. Assim, teremos, inspecionando (B.1):

$$P1 = 2;\ P2 = 8;\ P3 = 14;\ P4 = 20 \qquad (B.8)$$

posições cujos α valem 1. No hardware, o caçador de posições menos confiáveis indicará exatamente estas posições, nesta mesma ordenação.

Em seguida, pelo algoritmo 2 de Chase, perturbamos **Y** (A1) com todas as 16 combinações possíveis dos Pi, $1 \leq i \leq 4$, calculamos 16 síndromes e decodificamos binariamente, calculando em seguida 16 pesos analógicos e selecionando o padrão de menor peso.

Equivalentemente, o macro-processo implementado no hardware calcula 16 síndromes a partir da soma da síndrome primária com as síndromes teste e depois continua de forma igual ao algoritmo de Chase.

Analisamos abaixo os resultados de cada padrão de teste, DTi, $1 \leq i \leq 4$

01) $DT = (0,0,0,0) \Rightarrow \mathbf{S'}_1 = (0001\ 1111\ 0100) = 2F8_{\ H}$
$\therefore PE = (1,7,13,19),\ W\alpha = 12,\ d_H = 4 \Rightarrow \mathbf{C}_1 = \mathbf{1}$ (decisão abrupta)

02) $DT = (2,0,0,0) \Rightarrow \mathbf{S'}_2 = (1110\ 0110\ 0111) = E67_{\ H}$
$\therefore PE = (4,15,20,0),\ W\alpha = 8,\ d_H = 4$
$\Rightarrow \mathbf{C}_2 = (001011011111010111001111)$

03) DT = (0,8,0,0) $\Rightarrow$ S'_3 = (101010001101) = B15 $_H$
$\therefore$ PE = (11,16,24,0), Wα = 10, d_H = 4
$\Rightarrow$ C_3 = (0111 1100 1101 0110 1101 1110)

04) DT = (2,8,0,0) $\Rightarrow$ S'_4 = (1101 0001 1110) = 78A $_H$
$\therefore$ PE = (1,3,10,21), Wα = 14, d_H = 6
$\Rightarrow$ C_4 = (1001 1100 1011 0111 1101 0111)

05) DT = (0,0,14,0) $\Rightarrow$ S'_5 = (0101 1111 0101) = AFA $_H$
$\therefore$ PE = (9,18,21,0), Wα = 10, d_H = 4
$\Rightarrow$ C_5 = (0111 1101 0111 0011 1001 0111)

06) DT = (2,0,14,0) $\Rightarrow$ S'_6 = (1010 0110 0110) = 665 $_H$
$\therefore$ PE = (1,17,22,24), Wα = 14, d_H = 6
$\Rightarrow$ C_6 = (1011 1101 1111 0011 0101 1010)

07) DT = (0,8,14,0) $\Rightarrow$ S'_7 = (1110 1000 1100) = 317 $_H$
$\therefore$ PE = (1,6,15,23), Wα = 14, d_H = 6
$\Rightarrow$ C_7 = (1111 1000 1111 0001 1101 1101)

08) DT = (2,8,14,0) $\Rightarrow$ S'_8 = (0001 0001 1111) = F88 $_H$
$\therefore$ PE = (5,12,13,0), Wα = 12, d_H = 6
$\Rightarrow$ C_8 = (0011 0100 1110 1011 1101 1111)

09) DT = (0,0,0,20) $\Rightarrow$ S'_9 = (0001 1110 0101) = A78 $_H$
$\therefore$ PE = (2,4,15,0), Wα = 8, d_H = 4
$\Rightarrow$ C_9 = (0010 1001 1111 0101 1100 1111)

10) DT = (2,0,0,20) $\Rightarrow$ S'_{10} = (1110 0111 0110) = 6E7 $_H$
$\therefore$ PE = (4,15,0,0), Wα = 8, d_H = 4
$\Rightarrow$ C_{10} = (0010 1101 1111 0101 1100 1111)

11) DT = (0,8,0,20) $\Rightarrow$ S'_{11} = (1010 1001 1100) = 395 $_H$
$\therefore$ PE = (1,9,12,22), Wα = 14, d_H = 6
$\Rightarrow$ C_{11} = (1111 1100 0110 0111 1100 1011)

12) DT = (2,8,0,20) $\Rightarrow$ S'_{12} = (0101 0000 1111) = FOA $_H$
$\therefore$ PE = (4,8,15,20), Wα = 8, d_H = 4
$\Rightarrow$ C_{12} = (0010 1101 1111 0101 1100 1111)

13) DT = (0,0,14,20) $\Rightarrow$ S'_{13} = (0101 1110 0100) = 27A $_H$
$\therefore$ PE = (1,5,10,16), Wα = 14, d_H = 6

$\Rightarrow C_{13} = (1111\ 0101\ 1011\ 0010\ 1100\ 1111)$

14) $DT = (2,0,14,20) \Rightarrow S'_{14} = (1010\ 0111\ 0111) = EE5_H$
$\therefore PE = (4,14,15,0),\ W\alpha = 8,\ d_H = 4$
$\Rightarrow C_{14} = (0010\ 1101\ 1111\ 0101\ 1100\ 1111)$

15) $DT = (0,8,14,20) \Rightarrow S'_{15} = (1110\ 1001\ 1101) = B97_H$
$\therefore PE = (3,7,17,0),\ W\alpha = 12,\ d_H = 6$
$\Rightarrow C_{15} = (0101\ 1110\ 1111\ 0011\ 0100\ 1111)$

16) $DT = (2,8,14,20) \Rightarrow S'_{16} = (0001\ 0000\ 1110) = 708_H$
$\therefore PE = (1,11,18,19),\ W\alpha = 16,\ d_H = 8$
$\Rightarrow C_{15} = (1011\ 1100\ 1111\ 0011\ 1010\ 1111)$

Ressaltamos que os resultados acima podem ser checados da seguinte forma:

a) fazemos o ou-exclusivo bit-a-bit da síndrome primária **S** com as colunas da matriz **[H]** (eq. (1.21.c)) correspondentes às posições dadas no conjunto de dígitos **"1"** do padrão de teste (DT), obtendo assim a síndrome **S'**;

b) o padrão de erros dado pelo conjunto PE deve checar **S'** através do ou-exclusivo bit-a-bit das colunas da matriz **[H]** endereçadas pelas posições em erro dadas por PE;

c) o peso analógico $W\alpha$ é a mera aplicação da fórmula da eq. (4.3), facilitada porque somamos apenas as confiabilidades associadas aos dígitos endereçados pela justaposição dos conjuntos DT e PE, expurgados aqueles que pertençam simultaneamente aos dois conjuntos;

d) a distância de Hamming d_H é o número de endereços diferentes de zero no conjunto obtido pela justaposição dos conjuntos DT e PE, expurgados os que coincidem entre si.

Enfim, como o menor peso analógico vale 8 e corresponde ao segundo padrão de teste, a palavra-código decodificada valerá:

$$C_S\ (A1) = C_2 = (00101101\ 1111\ 0101\ 1100\ 1111) \qquad (B.9)$$

O vetor-erro correspondente a $PE \cup DT = (2, 4, 15, 20)$ será:

$$E(A1)=(010100000000001000010000) \qquad (B.10)$$

E a seqüência de bits à saída será dada pelos 12 primeiros dígitos de C2.

É interessante, ainda aproveitando este exemplo, realçarmos várias características importantes do macro-processo de decodificação que aqui aparecem nitidamente.

A primeira dela diz respeito à expansão da capacidade corretora do sistema de codificação-decodificação. Enquanto na decodificação por decisão abrupta todos os

padrões de erro resultantes têm 0, 1, 2 ou 3 erros em qualquer posição dentro do bloco ou ainda 4 erros desde que um deles seja no primeiro bit, a decodificação por decisão suave pode levar a padrões de erro com até 7 erros. No exemplo dado, o padrão de erros foi de 4 erros sem que tenha o primeiro bit em erro, o que seria impossível de se decodificar com o processo em decisão abrupta.

A segunda característica é a de que, qualquer que seja o vetor-teste dado no conjunto DT, a palavra decodificada sempre deve ser uma palavra-código do CGE (24, 12), porque o decodificador binário utilizado é completo.

A terceira característica está relacionada à necessidade do analisador de coincidência de endereços. Neste exemplo, nos padrões de teste 12 e 14 há coincidência entre o DT2 e PE2 e entre o DT3 e PE2, respectivamente.

A quarta característica é que muitas vezes padrões de teste originados de combinações diferentes levam a um conjunto igual de vetor-erro final. No exemplo, os padrões de teste 2, 9, 10, 12, e 14 levam ao mesmo padrão de erro, dado pela justaposição dos DT e PE, excluindo os coincidentes, que é, por sinal, o padrão de erro de menor peso analógico.

A quinta e última característica é exemplificada no padrão de teste 16: o padrão compõe-se de um conjunto de 8 posições em erro, indicando que a palavra-código que está sendo testada dista, do vetor **Y** (A1) recebido, a distância mínima do CGE (24, 12), o que indica que o vetor-erro correspondente jamais poderia ser o de mínimo peso analógico, conforme já havíamos frisado na seção 5.9.

As figuras B.1 a), b) e c) mostram, respectivamente: a forma de onda recebida, reconstituída por um conversor D/A hipotético à saída do quantizador; a forma de onda decodificada por decisão abrupta; a forma de onda decodificada por decisão suave. Supusemos também um mapeamento dos dígitos de forma que 0 é representado por -4 e 1 por 4.

A partir da comparação das figuras podemos compreender graficamente como a decisão suave aumenta a capacidade corretora do sistema.

Uma primeira interpretação das figuras diz respeito à diferença de energias entre as formas de onda. Esta diferença é proporcional à diferença das áreas sob as curvas e é proporcional à potência de ruído necessária para transformar uma forma de onda em outra. A melhor decodificação possível é aquela que minimiza a diferença de áreas sob as formas de onda recebida e forma de onda representativa da palavra-código originalmente transmitida, já que ruídos de baixas energias são mais prováveis que ruídos de altas energias em um sistema de comunicação bem dimensionado. Na figura B.1.a), hachuriada com retas inclinadas à direita, temos a diferença de área entre y (t) e uma suposta C (t) = 1 originalmente transmitida. Esta suposta C (t) seria o resultado da decodificação por decisão abrupta, conforme já vimos. A diferença entre as áreas vale 44. Já na figura B.1 c), hachuriada com retas inclinadas à esquerda, temos a diferença de área entre y (t) e uma suposta C (t) = [representação temporal de C2] originalmente transmitida. Esta suposta C (t) seria o resultado da decodificação por decisão suave, conforme já vimos. A diferença entre

as áreas vale 32. Então, a forma de onda que minimiza a potência de ruído e portanto a probabilidade de erro é dada pela figura B.1.c).

Outra interpretação, que não altera o resultado final, é aquela onde observamos qual palavra-código tem a maior correlação com y(t), palavra recebida.

B.2. PADRÃO DE TESTE A2

O padrão A2 é formado por um vetor de polaridade que é a inversão booleana de Y (A1) e o mesmo α(A1). As saídas da decodificação tanto em decisão abrupta como em decisão suave são inversão booleana das saídas anteriores, pois o CGE (24, 12) é simétrico (ver cap. 1), e os padrões de teste são os mesmos da seção B.1. Assim teremos:

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A2)	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
α (A2)	3	1	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3
C_H (A2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_S (A2)	1	1	0	1	0	0	1	0	0	0	0	0	1	0	1	0	0	0	1	1	0	0	0	0

B.3. PADRÃO DE TESTE A3

Está dado em (B.11), junto com as saídas para os casos de decisão abrupta e suave.

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A3)	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
α (A3)	3	3	0	2	0	3	3	0	3	3	1	3	2	3	3	3	3	0	3	3	3	3	3	3
C_H (A3)	1	0	0	0	1	0	1	1	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0
C_S (A3)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(B.11)

O conjunto dos 16 padrões de teste e os respectivos pesos-analógicos estão dados na tabela B.1. a seguir, já com as coincidências eliminadas.

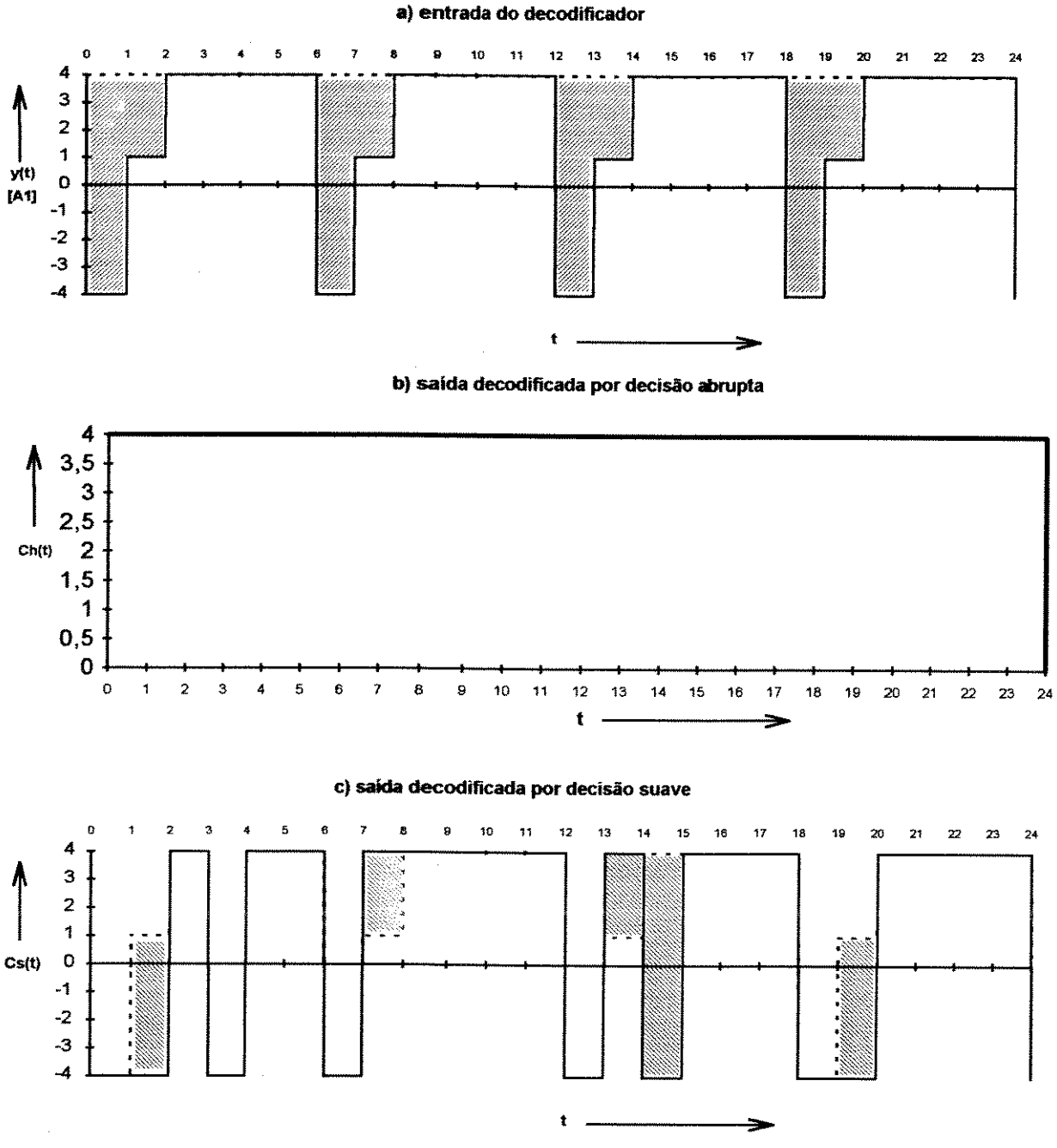


FIGURA B.1. FORMAS DE ONDA DO PADRÃO DE TESTE A1

#	PE	DT	Wa	#	PE	DT	Wa
1	1,7,9,22	0,0,0,0	12	9	5,11,18,0	0,0,0,8	1
2	12,14,17,0	3,0,0,0	9	10	1,10,15,23	3,0,0,8	12
3	5,8,11,0	0,18,0,0	1	11	5,11,0,0	0,18,0,8	1
4	1,2,4,13	3,18,0,0	10	12	0,5,11,0	0,18,0,8	1
5	8,11,18,0	0,0,5,0	1	13	11,18,0,0	0,0,5,8	1
6	1,6,16,20	3,0,5,0	12	14	0,11,18,0	0,0,5,8	1
7	8,11,0,0	0,18,5,0	1	15	11,0,0,0	0,18,5,8	1
8	0,8,11,0	0,18,5,0	1	16	0,11,0,0	0,18,5,8	1

TABELA B.1. RESULTADOS DO PADRÃO A3

B.4. PADRÃO DE TESTE A4

Está dado em (B.12),junto com as saídas para decisão abrupta e suave.

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A4)	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0
α(A4)	1	2	2	2	2	2	2	2	0	2	3	1	3	2	3	3	3	3	0	0	2	2	2	2
C_H (A4)	1	0	0	1	0	0	0	0	1	1	0	1	0	1	0	0	0	0	1	1	0	0	0	0
C_S (A4)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(B.12)

O conjunto dos 16 padrões de teste e os respectivos pesos analógicos podem ser vistos na tabela B.2. abaixo, já com os endereços coincidentes eliminados dos conjuntos PE e DT.

#	PE	DT	W_α	#	PE	DT	W_α
1	1,4,10,14	0,0,0,0	7	9, 9,12,19	0,0,0,0	20	1
2	4,10,14,0	1,0,0,0	7	10, 0,9,12	19,0,0,0	20	1
3	12,19,20,0	0,9,0,0	1	11, 12,19,0	0, 0,9,0	20	1
4	0,12,19,20	0,9,0,0	1	12, 0,12,19	0, 0,9,0	20	1
5	9,12,20,0	0,0,19,0	1	13, 9,12,0	0, 0,0,19	20	1
6	0,9,12,20	0,0,19,0	1	14, 0,9,12	0, 0,0,19	20	1
7	12,20,0,0	0,9,19,0	1	15, 12,0,0	0, 0,9,19	20	1
8	0,12,20,0	0,9,19,0	1	16, 0,12,0	0, 0,9,19	20	1

TABELA B.2. RESULTADOS DO PADRÃO A4

B.5. PADRÃO DE TESTE A5

A equação (B.13) fornece o padrão e as saídas esperadas:

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A5)	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0
α (A5)	3	3	3	3	0	1	3	3	3	0	3	3	3	1	0	0	3	0	3	3	0	3	3	3
C_H (A5)	0	0	0	0	1	0	0	0	0	1	0	1	1	0	1	1	1	1	0	0	0	0	0	0
C_S (A5)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(B.13)

O conjunto dos 16 padrões de teste e os respectivos pesos estão na tabela B.3, já com as coincidências de endereços removidas.

#	PE	DT	W_α	#	PE	DT	W_α
1	12,13,17,0	0,0,0,0	9	9	1,9,20,21	0,0,0,15	9
2	1,9,15,20	21,0,0,0	9	10	1,9,20,0	21,0,0,15	9
3	1,3,11,23	0,16,0,0	12	11	5,10,18,0	0,16,0,15	0
4	7,8,14,0	21,16,0,0	7	12	1,9,0,20	21,0,0,15	9
5	1,6,7,19	0,0,10,0	10	13	5,16,18,0	0,0,10,15	0
6	2,23,24,0	21,0,10,0	9	14	1,9,0,20	21,0,0,15	9
7	5,15,18,0	0,16,10,0	0	15	5,18,0,0	0,16,10,15	0
8	1,4,13,22	21,16,10,0	12	16	5,18,0,0	0,16,10,15	0

TABELA B.3. RESULTADOS DO PADRÃO A5

B.6. PADRÃO DE TESTE A6

Está dado, junto com as saídas de decisão abrupta e suave, na equação (B.14):

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A6)	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0
α (A6)	3	3	3	0	3	0	3	3	3	3	1	3	3	3	0	3	3	1	0	3	3	0	3	3
C_H (A6)	1	0	0	0	1	1	1	0	0	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0
C_S (A6)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(B.14)

O conjunto dos 16 padrões de teste, já com os endereços coincidentes eliminados, está na tabela B.4. a seguir.

#	PE	DT	W_α	#	PE	DT	W_α
1	1,5,7,16	0,0,0,0	12	9	6,18,19,0	0,0,0,15	1
2	14,21,24,0	22,0,0,0	9	10	1,2,8,13	22,0,0,15	12
3	9,12,13,0	0,4,0,0	9	11	1,17,20,21	0,4,0,15	12
4	1,10,18,23	22,4,0,0	10	12	3,5,11,0	22,4,0,15	7
5	15,18,19,0	0,0,6,0	1	13	18,19,0,0	0,0,6,15	1
6	1,3,9,20	22,0,6,0	10	14	18,19,0,0	0,0,6,15	1
7	1,8,11,14	0,4,6,0	12	15	0,18,19,0	0,0,6,15	1
8	2,16,17,0	22,4,6,0	9	16	1,7,12,24	22,4,6,15	12

TABELA B.4. RESULTADOS DO PADRÃO A6

B.7. PADRÃO DE TESTE A7

Está dado em (B.15), junto com as saídas da decodificação em decisão abrupta e em decisão suave.

ÍNDICE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Y (A7)	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0
α(A7)	3	3	3	3	3	3	3	3	1	3	1	2	1	3	3	3	1	3	3	3	3	3	1	3
C_H (A7)	1	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	1	1	0	1	0	0	1	0
C_S (A7)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(B.15)

O conjunto dos 16 padrões de teste, já com os endereços coincidentes eliminados, está na tabela B.5. abaixo:

#	PE	DT	W_α	#	PE	DT	W_α
1	1,14,18,20	0,0,0,0	12	9	6,10,15,0	0,0,0,13	10
2	9,11,17,0	23,0,0,0	4	10	1,4,19,22	23,0,0,13	14
3	11,17,23,0	0,9,0,0	4	11	1,3,12,16	0,9,0,13	13
4	11,17,0,0	23,9,0,0	4	12	11,0,17,0	23,9,0,0	4
5	9,17,23,0	0,0,11,0	4	13	1,2,7,8	0,0,11,13	12
6	9,17,0,0	23,0,11,0	4	14	9,0,17,0	23,0,11,0	4
7	17,23,0,0	0,9,11,0	4	15	0,17,23,0	0,9,11,0	4
8	17,0,0,0	23,9,11,0	4	16	0,17,0,0	23,9,11,0	4

TABELA B.5. RESULTADOS DO PADRÃO A7

É interessante notarmos nestes padrões de teste de A1 a A6, e em especial em A7, como opera o mecanismo que alivia a carga de trabalho do decodificador binário (de decisão abrupta) através das inversões a priori nos dígitos menos confiáveis que foram selecionados para serem combinados.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] ---(diversos)---, "Completion Report: Project 155, Part A, Coded Phase Modem at 64 Kbps"; INTELSAT, Contribution of Director General, Oct. 31,1983.

[2] ---(diversos)---, "Intel Memory Data Book"; (INTEL Corp., 1983).

[3] ---(diversos)---, "Introduction to the 80386"; (INTEL Corp., April 1986).

[4] ---(diversos)---, "SAB 8051 User's Manual"; (SIEMENS AG, July 1981).

[5] ---(diversos)---, "The HCMOS Data Book";(TEXAS Instr., 1985).

[6] ---(diversos)---, "The TTL Data Book, Volume 2";(TEXAS Instr., 1985).

[7] ABBASZADEH, A. D. & RUSHFORTH, C. K. "VLSI Implementation of a Maximum-Likelihood Decoder for the Golay (24,12) Code"; IEEE Journal on Selec. Areas in Communications; vol 6, no. 3, pp 558-565, April 1988.

[8] ALMEIDA, V. A. F.- "Supercomputadores: A Batalha dos Nanosegundos"; Ciência Hoje, SBPC; vol. 5, nº 25, pp62-72, Julho/Agosto 1986.

[9] BARNEY, C.- "Logic Designers Toss out the Clock"; Electronics; pp 42-45, Dec. 9, 1985.

[10] BARTEE, T. C.- "Digital Computer Fundamentals";(4th. ed.; McGraw Hill Kogakusha,Ltd., Tokio, 1977).

[11] BERLEKAMP, E. R.- "Algebraic Coding Theory";(McGraw Hill Book Comp.; New York, 1968).

[12] CARLSON, A. B.- "Sistemas de Comunicação: Uma Introdução aos Sinais e Ruído em Comunicação Elétrica"; (Ed. McGraw Hill do Brasil; São Paulo, 1981).

[13] CHASE, D. "A Class of Algorithms for Decoding Block Codes with Channel Measurement Information"; IEEE Trans. on Information Theory ; vol IT-18, no. 1, pp 170-182, Jan 1972.

[14] CLARK, G. C. & CAIN, J. B. "Error-Correction Coding for Digital Communications"; (Plenum Press, New York,1982).

[15] CONWAY, J. H. & SLOANE, N. J. A. "Soft Decoding Techniques for Codes and Lattices, Including the Golay Code and the Leech Lattice"; IEEE Trans. on Information Theory, vol IT-28, 1986.

[16] HACKETT, C. M. "An Efficient Algorithm for Soft-Decision Decoding of the (24,12) Extended Golay Code"; IEEE Trans. on Communications; vol. COM-29, no. 6, pp 909-911, June 1981.

[17] HELLER, J. A. & JACOBS, J. M.- "Viterbi Decoding for Satellite and Space Communication"; IEEE Trans. on Comm Tech.; vol. COM-19, no. 5, pp 835-848, Oct 1971.

[18] HILL, F. J. & PETERSON, G. R.- "Digital Systems: Hardware Organization and Design"; (2nd. ed., 6th. prt., John Wiley & Sons, New York, 1978).

[19] HILL, F. J. & PETERSON, G. R.- "Introduction to Switching Theory and Logical Design"; (2nd. ed., 11th. prt., John Wiley & Sons, New York, 1974).

[20] HWANG, T. Y.- "Efficient Optimal Decoding of Linear Block Codes"; IEEE Trans. on Information Theory; vol. IT-26, no. 5, pp 603-606, Sept. 1980.

[21] KASAMI, T.- " A Decoding Procedure for Multiple-Error-Correcting Cyclic Codes"; IEEE Trans. on Information Theory; vol. IT-10, pp 134-139, Apr. 1964.

[22] KERCZEWSKI, R. J.; DAUGHERTY, E. S. & KRAMARCHUK, I. "Gauge Bit-Error Rate as a Function of Signal-to-Noise Ratio"; Microwaves & RF; vol 27, no. 2, pp 107-113, February 1988.

[23] KLIMOV, G.- "Probability Theory and Mathematical Statistics"; (1st. ed., Mir Publishers, Moscow, 1986).

[24] KOGA, K. "High-Speed Operable Soft-Decision FEC Decoding of Block Codes"; Proceedings of the 8th. International Conference on Digital Satellite Communications, Guadeloupe; pp 249-256, 1989.

[25] LATHI, B. P.- "An Introduction to Random Signals and Communication Theory"; (International Textbook Company, Scranton, Pa., 1968).

[26] LEE, L. N. & WENG, M. J.- "2-Bit Soft-Decision Weighted Erasure Decoding for Binary Block Codes"; COMSAT Technical Review; vol. 10, no. 2, pp 397-415, Fall 1980.

[27] LIN, S.- "An Introduction to Error-Correcting Codes"; (1st. ed., 2nd. prt., Prentice-Hall,Inc., Englewood Cliffs, NJ, 1970).

[28] LUCKY, R. W.; SALZ, J. & WELDON, E. J.- "Principles of Data Communication"; (1st. ed., McGraw Hill Book Comp., New York, 1968).

[29] MARTIN, J.- "Communications Satellite Systems"; (Prentice-Hall,Inc., Englewood Cliffs, NJ, 1978).

[30] OPPENHEIM, A. V. & SCHAFER, R. W.- "Digital Signal Processsing"; (Prentice-Hall,Inc., Englewood Cliffs, NJ, 1975).

[31] PESSOA, A. C. F. & ARANTES, D. S.- "Simulação do Algoritmo de Chase II para a Decodificação de Códigos de Bloco"; Relatório Técnico RT-143, Contrato TELEBRÁS-UNICAMP 024A/83, Junho 1986.

[32] PESSOA, A. C. F.- "Considerações sobre a Sensibilidade do Decodificador de Chase com Relação a Variações no Controle Automático de Ganho da Recepção"; Relatório Interno TELEBRÁS-UNICAMP, não publicado, Agosto 1986.

[33] PETERSON, W. W. & WELDON, E. J.- "Error-Correcting Codes"; (2nd. ed., The MIT Press, Cambridge, Mass., 1972).

[34] PROAKIS, J. G.- "Digital Communications"; (1st. ed., McGraw Hill International Book Comp., Tokio, 1983).

[35] ROWE, H. E.- "Signals and Noise in Communications Systems"; (Van Nostrand, Princeton, NJ, 1965).

[36] SALLES, H. C. A. S. & BORELLI, W. C.;"Um Novo Método para Implementação em Hardware de Decodificadores de Códigos de Bloco por Decisão Suave: Um Estudo Comparativo"; Anais do Nono Simpósio Brasileiro de Telecomunicações; 1991.

[37] SALLES, H. C. A. S. & BORELLI, W. C.;"Procedimento Alternativo para Caracterização do Desempenho de Decodificadores de Canal"; Anais do Nono Simpósio Brasileiro de Telecomunicações; 1991.

[38] SALLES, H. C. A. S. & BORELLI, W. C.;"Optimisation of Chase Decoding Algorithm for Application in Medium Data Rate Decoder Architecture"; First International Symposium on DSP for Communication Systems University of Warwick; England; 7-9 Sept. 1992.

[39] SALLES, H. C. A. S. & BORELLI, W. C."Design and Evaluation of Chase Decoder Architecture for Medium Capacity TDMA Satellite Systems"; Revista da Sociedade Brasileira de Telecomunicações(SBT) - Edição Especial em Códigos Corretores de Erro, vol 5, no. 1, Junho 1990.

[40] SALLES, H. C. A. S.- "Caderno de Laboratório TELEBRÁS: Depuração da Unidade Decodificadora"; não publicado,1987-88-89.

[41] SALLES, H. C. A. S.- "Decodificação por Decisão Suave: Caderno de Notas"; Relatório de Estudos Especiais I; não publicado, 1986.

[42] SALLES, H. C. A. S.- "Notas de Projeto do Decodificador de Chase - Caderno de Projetos"; não publicado, 1986.

[43] TENDOLKAR, N. N. & HARTMANN, C. R. P.- "Generalization of Chase Algorithms for Soft-Decision Decoding of Binary Linear Codes"; IEEE Trans. on Information Theory; vol. IT-30, no. 5, pp 714-721, Sept. 1984.

[44] VITERBI, A. J. & OMURA, J. K. "Principles of Digital Communication and Coding"; (McGraw Hill Kogakusha,Ltd.; Tokio, 1979).