A UTILIZAÇÃO DE PROGRAMAS

PARA DETEÇÃO DE FALHAS EM MICROCOMPUTADORES

Humberto dos Santos Melim

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.).

APROVADA POR:

PAULO MARIO BIANCHI FRANÇA
presidente

EBER ASSIS SCHMITZ

LUIZ ANTONIO CARNEIRO
DA CUNHA COUCEIRO

RIO DE JANEIRO , RJ-BRASIL

SETEMBRO DE 1980

MELIM, HUMBERTO DOS SANTOS

A Utilização de Programas para Deteção de Falhas em Microcomputadores [Rio de Janeiro], 1980.

VII, 76 p. 29,7 cm (COPPE-UFRJ,M. Sc., Engenharia de Sistemas e Computação, 1980.

Tese - Universidade Federal do Rio de Janeiro. Núcleo de Computação Eletrônica.

1. Programas de Teste I. COPPE/UFRJ II. Título (série).

AGRADECIMENTOS

Ao orientador, pelas valiosas sugestões apresentadas durante o desenvolvimento deste trabalho, aos demais membros da banca, que muito me honraram com sua participação, e aos colegas do NCE, pelo apoio e colaboração prestados.

RESUMO

O presente trabalho tece considerações sobre a utilização de programas de teste em microcomputadores e descreve um conjunto de testes, residentes em uma ROM de 1Kx8, destinados a detetar falhas nos diversos módulos que compõem um sistema baseado no microprocessador 8080.

ABSTRACT

This work surveys some aspects of self-test programs for microcomputers and presents a 1 Kbyte ROM resident program developed for fault detection in a 8080 based system.

ÍNDICE

I - INTRODUÇÃO

A crescente utilização de microprocessadores nos últimos anos,devido ao seu baixo custo, alto desempenho e grande flexibilidade, ao mesmo tempo que ocasionou mudanças substanciais no projeto de sistemas,dificultou também os procedimentos de teste dos mesmos[22,26,31].

Os métodos tradicionais para encontrar bits presos em blocos combinacionais[4] não são adequados em sistemas com microprocessadores, devido à grande complexidade e inacessibilidade dos circuitos e à existência de outros tipos de erro.

Entre as opções atualmente disponíveis está o emprego da análise de assinaturas[20,21,30,33], uma técnica recentemente desenvolvida e que procura caracterizar os diversos pontos de um circuito através da compressão dos dados ali presentes, amostrados a partir de sinais de referência, ou a utilização de equipamentos automáticos de teste (ATE)[5,6,28,32,34],especialmente desenvolvidos para este fim.

A análise de assinaturas,no entanto,necessita ser definida na fase de projeto,com o intuito de prover os sinais de referência indispensáveis à sua utilização.Requer ainda aparelhagem específica para gerar as assinaturas e compará-las com padrões já estabelecidos.

Por sua vez,o emprego de ATE vai se tornando desaconselhável até mesmo em produção de larga escala, devido ao seu alto preço,que tem se mantido aproximadamente constante ao longo do tempo, absorvendo uma parcela cada vez

maior em relação ao custo de um componente ou sistema, já que os demais gastos associados à fabricação decrescem à medida que a tecnologia evolui[3].

Uma outra alternativa consiste em utilizar a potencialidade inerente dos microprocessadores para executar auto-testes[1,3,14,18,24,29,35]. Os programas de teste, escritos na própria linguagem do processador, permitem avaliar o funcionamento do sistema. Podem ser empregados durante o desenvolvimento, teste de produção, verificação e manutenção.

Seu uso em aplicações de alta confiabilidade, tais como equipamento espacial ou militar, usinas nucleares e sistemas médicos de controle de funções vitais, permite acionar sinais de alarme ou advertência, ou comutar para sistema "back-up".

O grande número de estímulos que pode ser rapidamente aplicado com o uso desta técnica se revela uma valiosa ferramenta para manutenção, reduzindo substancialmente o MTTR do equipamento e possibilitando uma precisa verificação de seu correto funcionamento.

A constante redução do custo por bit de memórias ROM permite até mesmo a inclusão dos testes mais básicos no próprio microprocessador, como já ocorre no MC6805, da Motorola[3].

Uma vez que as únicas despesas envolvidas na utilização de auto-testes consistem na produção de programas apropriados, resultam economias substanciais, por não haver necessidade de adquirir equipamentos específicos.

Para sistemas com um único microprocessador, onde

o custo, tamanho e potência consumida devem ser minimizados, o emprego destes programas surge consequentemente como uma excelente alternativa de teste.

É a este tópico que se prende o presente trabalho,abordando diversos métodos e detalhando a implementação de um programa de auto-teste para um microcomputador típico.

II - TESTES

2.1 - Classificação

Os testes normalmente aplicados a componentes ou sistemas lógicos a fim de verificar seu correto funcionamento se enquadram em três categorias:

I) Teste estático, no qual as correntes e tensões apropriadas são aplicadas a determinados pontos e as correntes e tensões resultantes em outros pontos são verificadas.

II) Teste de funcionamento,procedimento pelo qual se testa a lógica interna de um dispositivo, através da aplicação de padrões binários nas entradas e comparação das saídas com resultados conhecidos.

III) Teste dinâmico, onde se verificam os tempos e formas de onda correspondentes aos parâmetros esperados do circuito em teste.

Em virtude da atual complexidade dos componentes básicos de sistemas lógicos, a utilização de um teste estático permite detetar apenas os defeitos mais grosseiros, antes de se dar início a testes mais sofisticados.

Por outro lado, um dispositivo aprovado no teste de funcionamento tem grande probabilidade de estar perfeito. Deve-se notar que o teste de funcionamento é de todos o mais importante, podendo, dentro de certos limites, englobar implicitamente parte dos demais testes.

Defeitos apontados pelos testes estático e dinâmico só são importantes na medida em que prejudiquem o fun

cionamento do circuito, ao ser este incorporado em um sistema.Seu emprego procura detetar de maneira indireta qualquer discrepância que possa ocasionar problemas de funcionamento. Naturalmente, por desconhecimento do sistema ao qual se destina um componente,bem como por questões de padronização, circuitos são rejeitados após sua fabricação se deixarem de atender a qualquer de suas especificações. Entretanto,ao se tentar localizar falhas em um sistema, é no teste de funcionamento que se deve concentrar especial atenção.

Os testes de funcionamento podem ser realizados através de equipamentos especialmente desenvolvidos para este fim ou através da capacidade de processamento do próprio sistema.Nesta categoria se enquadram todos os testes por programa,independente da forma em que se encontrem implementados.

2.2 - Testes de Processador

O crescente custo do trabalho associado à manutenção e o amplo uso de processadores em aplicações onde se necessita confiabilidade e disponibilidade tem ocasionado o aparecimento de projetos que incluem facilidades embutidas para manutenção, a fim de rapidamente localizar, a nível de subsistema,as partes com defeito e isolar a(s) placa(s) correspondente(s),para devida substituição.

A particular implementação das facilidades de teste difere bastante, até para um mesmo fabricante, como demostram os exemplos a seguir.

No processador KL10 da DEC[2,10] para sistemas de

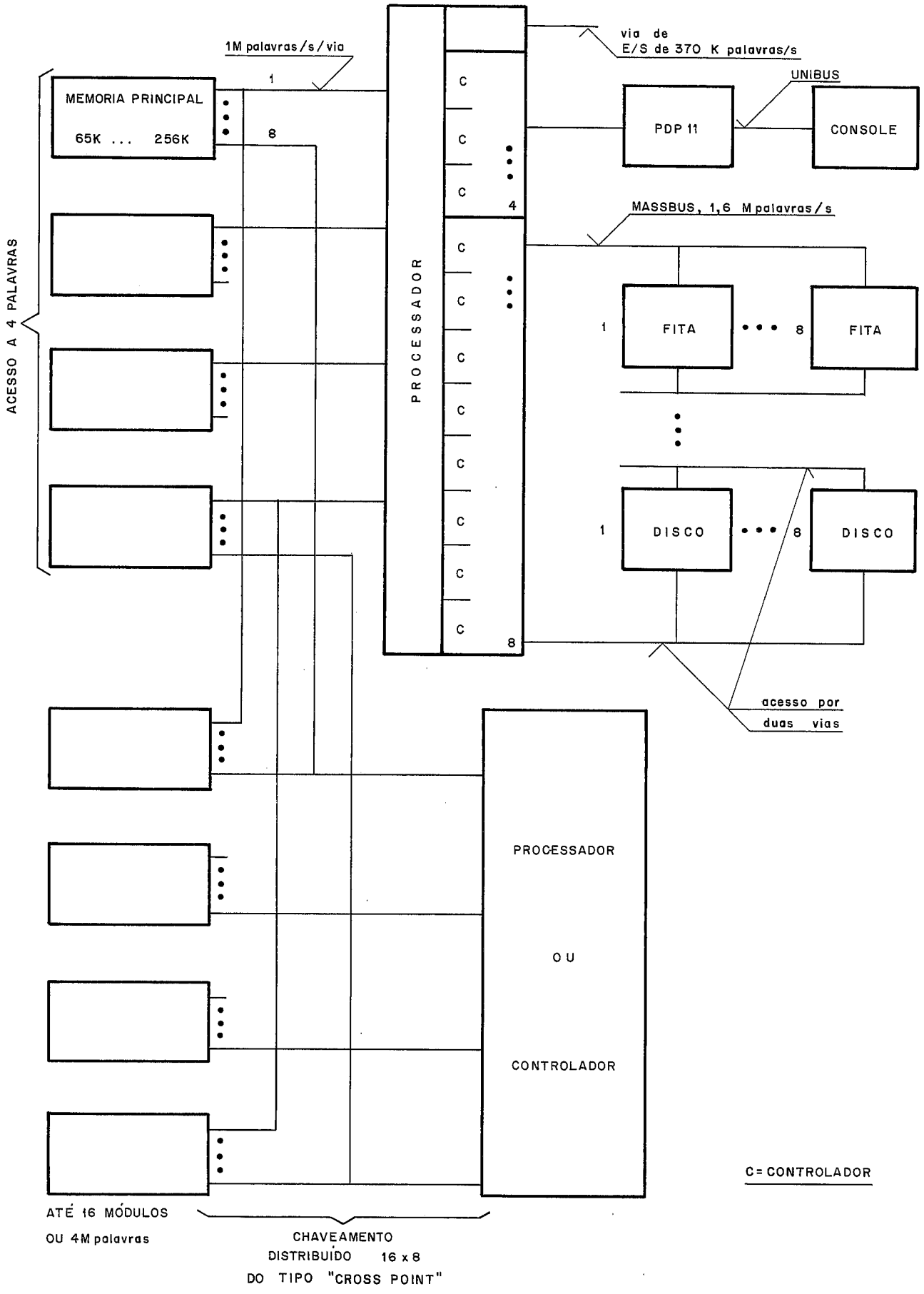

FIG 2-1 – DIAGRAMA DE BLOCOS DO KL 10.

DES. Cleber

grande porte,mostrado na figura 2-1, é possível ligar até quatro computadores PDP 11, que operam como processadores de entrada e saída. Um deles funciona também como console e processador para diagnósticos, com possibilidade de carregar microprogramas na memória de controle, examinar as trajetórias de dados e a lógica de controle do KL10 através de uma via especial,mesmo quando este se encontra completamente inoperante, podendo ainda verificar todas as vias de dados e realizar diagnóstico remoto através de linha telefônica.

Já o PDP 11/60[2], mostrado na figura 2-2,dispõe de uma unidade independente de microdiagnóstico,que testa as placas no processador,indicando,através de um conjunto de LEDs, o código associado a quaisquer anormalidades detetadas,e que,juntamente com o uso de um diretório,permite determinar a(s) placa(s) a substituir.

Para correta indicação,esta unidade supõe que uma pequena parte do processador, mais especificamente, o sequenciador de micropalavras, esteja operacional.

O PDP 11/70[11,12,13],cujo diagrama de blocos se vê na figura 2-3, possui um módulo acoplado à via padrão UNIBUS, onde se encontra uma ROM de 512 palavras de 16 bits, contendo um programa de diagnósticos e também a parte referente ao carregamento do sistema operacional. Este programa testa as instruções do processador,o cache, o gerenciamento de memória,o mapeador do UNIBUS,a memória principal e,parcialmente,o controlador e o periférico de UNIBUS ou MASSBUS a partir de onde o sistema operacional é carregado. A deteção de quaisquer erros provoca uma parada e o

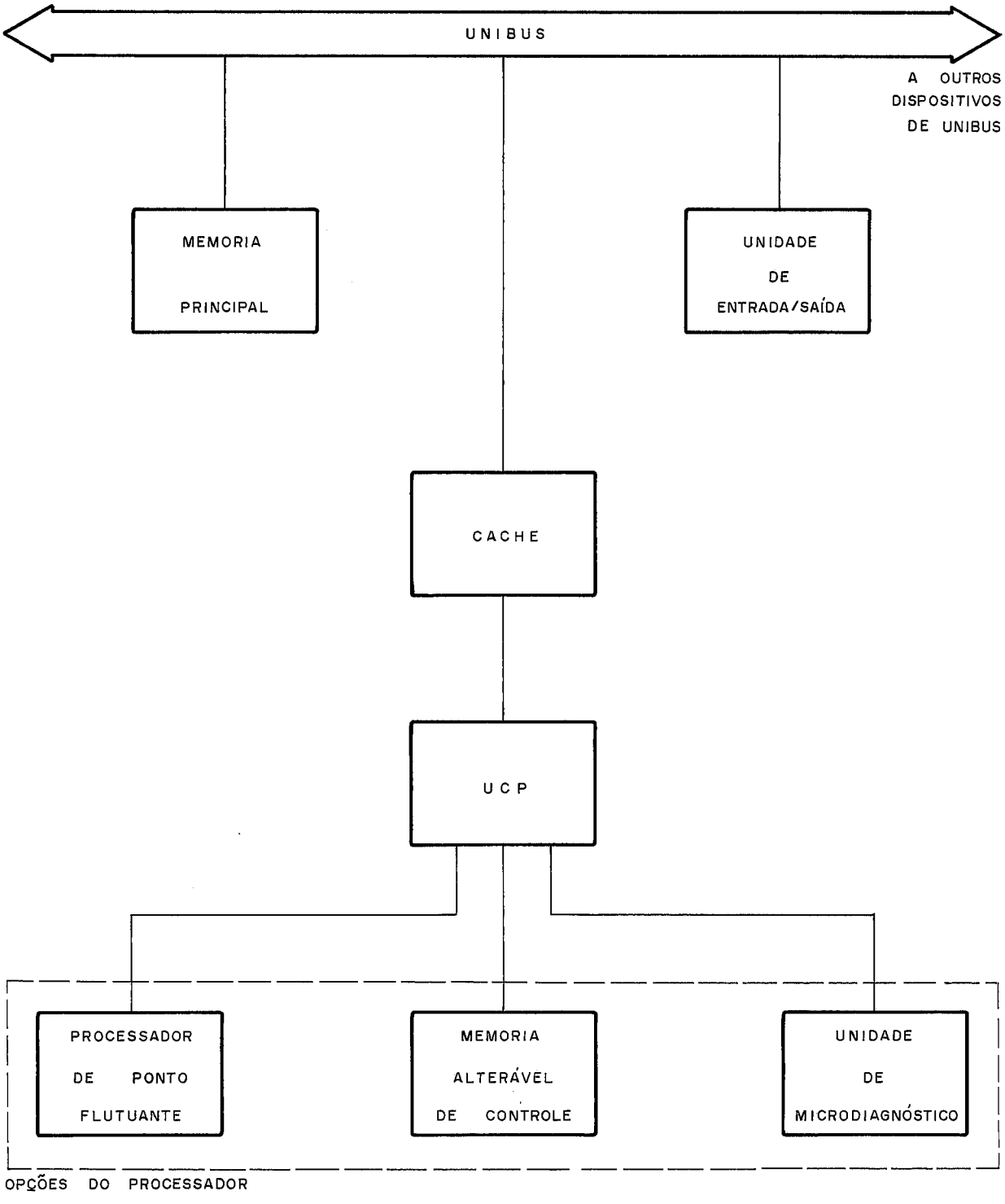

FIG. 2-2 — DIAGRAMA DE BLOCOS DO PDP 11/60.

DES. Cleber

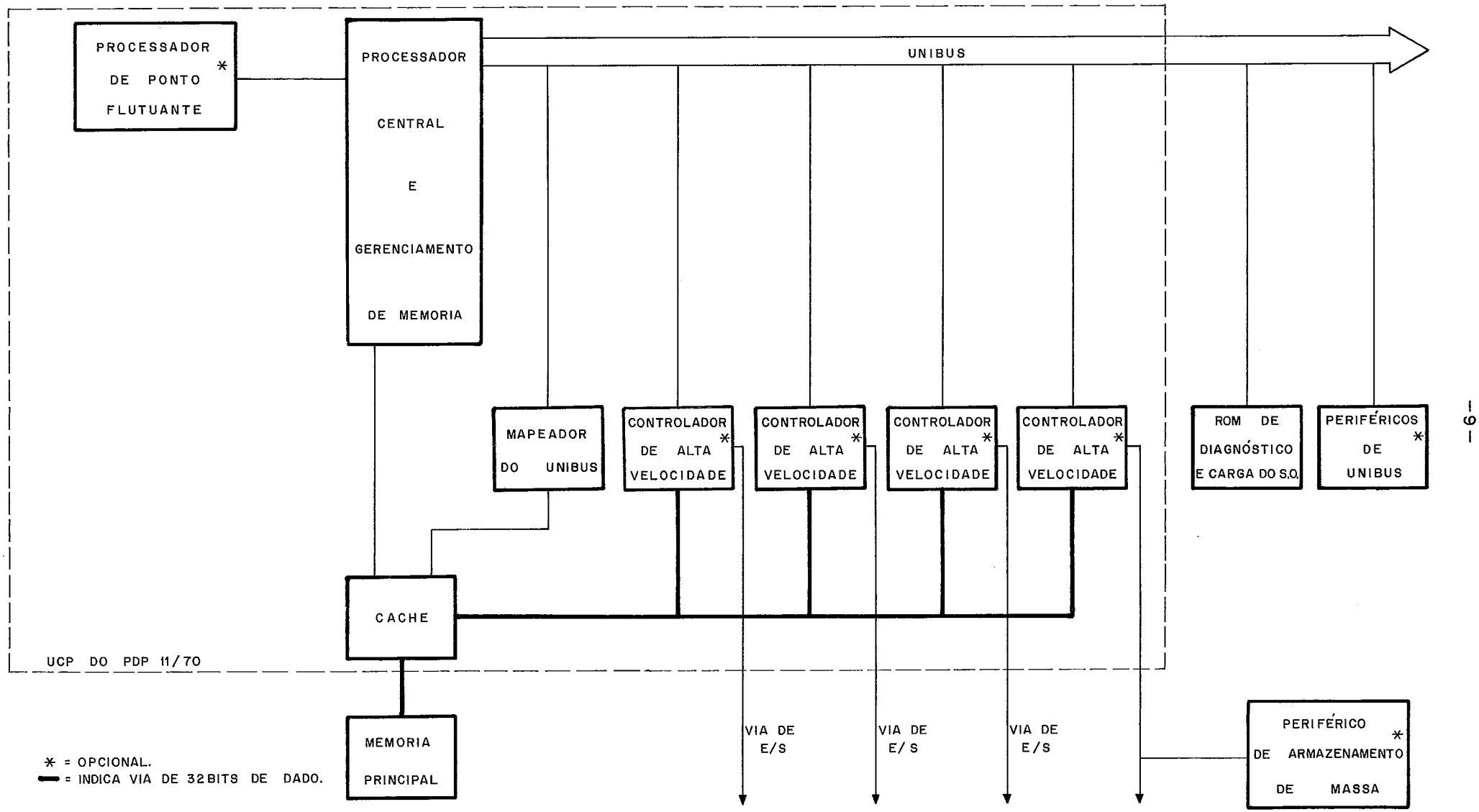

FIG. 2-3 - DIAGRAMA DE BLOCOS DO PDP 11/70.

DES. Cleber

endereço correspondente, disponível no painel,indica o tipo do erro encontrado.

O VAX 11/780[2,8], apresentado na figura 2-4, volta a empregar esquema semelhante ao do KL10,com uma memória de controle que pode ser carregada com o microprograma normal ou de diagnóstico, através do subsistema console, constituido por um microcomputador LSI-11 com disco flexível e capacidade para diagnóstico remoto e auto-teste, refletindo a evolução tecnológica da idéia original.

A IBM, em seu recente modelo 4341[9,17], leva ainda mais adiante a mesma idéia, como ilustra a figura 2-5, valendo-se de um microprocessador em separado para suporte e manutenção, que não apenas é capaz de carregar microprogramas de operação e diagnóstico, realizar diagnóstico remoto e auto-teste, como ainda registra em diskette o estado do processador principal, em caso de erro, para subsequente análise,uma vez que este dispõe de lógica especificamente incluída em sua arquitetura para permitir o acesso a seus estados internos. Este método permite até mesmo a deteção da causa de erros intermitentes,pois as informações são gravadas já na primeira ocorrência da anormalidade, acompanhada de dados adicionais relativos a condições ambientais de temperatura e de voltagem de alimentação,de forma que não se faz necessário reconstruir o erro para localizar o problema.

Embora cada um dos sistemas descritos, visto como um todo,apresente capacidade de auto-teste,quem melhor caracteriza este aspecto é o PDP 11/70, que utiliza as próprias instruções de seu único processador para realizar

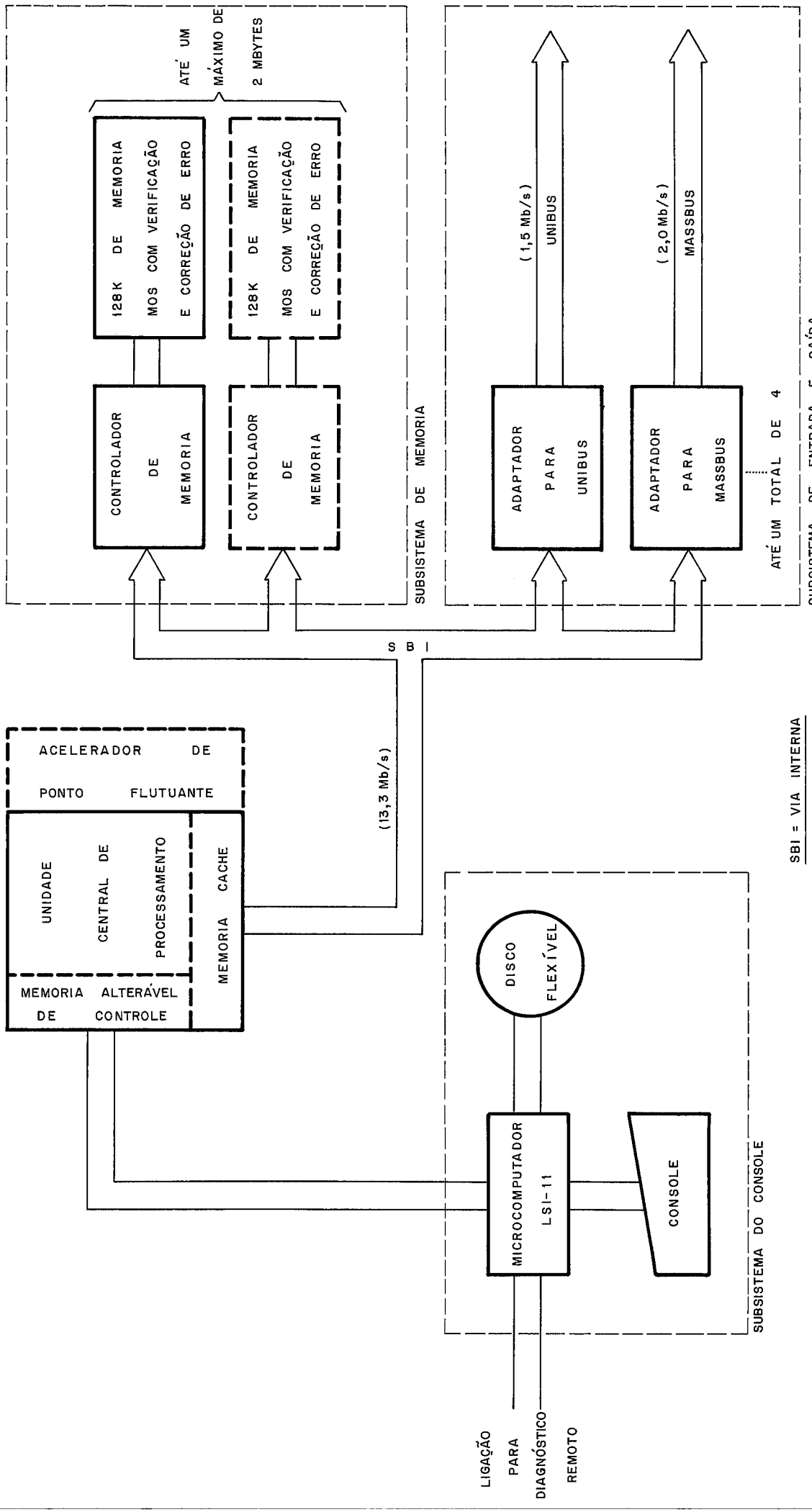

FIG. 2-4 — DIAGRAMA DE BLOCOS DO VAX 11/780.

DES. Cleber

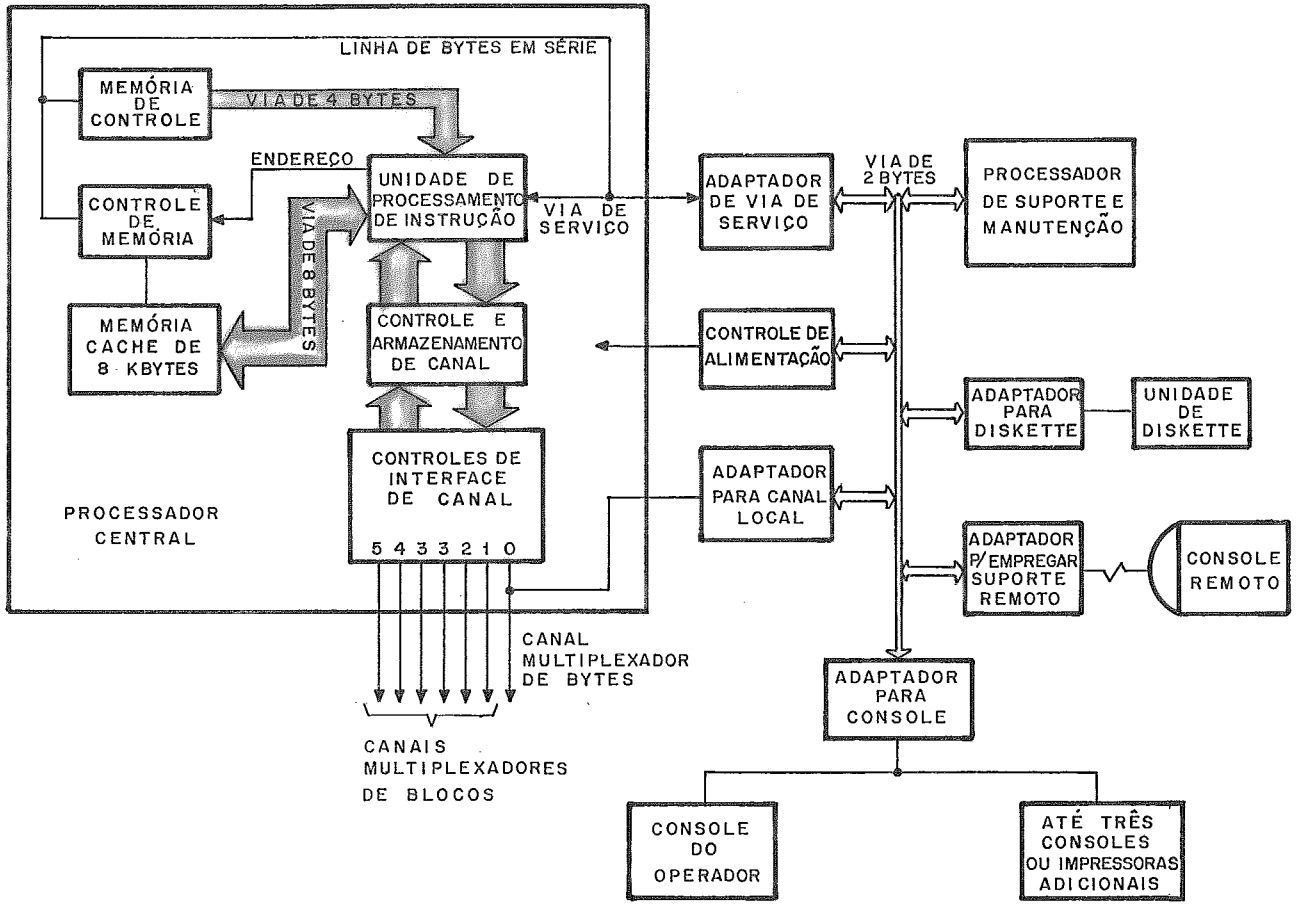

Fig. 2-5- Diagrama de Blocos do 4341

auto-diagnóstico.

Deve-se notar que há um problema fundamental com programas de auto-teste:uma falha pode anular o efeito de outra,de modo que ambas passem despercebidas. Para contornar este problema, pode-se testar inicialmente as instruções mais confiáveis e a seguir usa-lás para verificar as mais complexas.

O PDP 11/70, por exemplo,supõe o correto funcionamento do contador de programas(PC) e da instrução HALT, e aplicando sequências de instruções, organizadas em grupos funcionais, compara os resultados obtidos com os esperados,de modo a,na ocorrência de qualquer discrepância, executar a instrução HALT e determinar,pelo endereço onde se encontra, o grupo de instruções que deixou de operar,e,indiretamente, a parte da lógica que corresponde às mesmas.

A suposição inicial pode ser facilmente verificada executando-se a instrução HALT e observando-se a evolução do conteúdo do PC, através do acompanhamento passo a passo das instruções, nos indicadores do painel frontal.

2.3 - Testes de Memória RAM

Uma vez que se torna totalmente impraticável aplicar e verificar todos os padrões em uma pastilha de memória semicondutora,em virtude do extenso número de configurações possíveis, há que se fazer uma seleção dos padrões a utilizar, com base na estrutura interna da mesma e dos tipos mais comuns de defeito, entre os quais pode-se mencionar[23] :

D1 - Fuga em uma das entradas, que necessita mais

do que a corrente especificada para funcionamento.

D2 - Decodificador aberto, impedindo a utilização de parte da matriz de armazenamento.

D3 - Múltiplo endereçamento, onde os dados são lidos de ou escritos em outras células além da que estiver especificada pelo endereço.

D4 - Inabilidade em manter os dados entre os ciclos de restauração ("REFRESH"), em memórias dinâmicas.

D5 - Circuitos abertos ou em curto, devido à metalização insuficiente da excessiva em uma das fases de fabricação.

D6 - Sensibilidades a padrões, o que implica em menor confiabilidade para determinadas configurações de dados armazenados.

D7 - Defeito em algum elemento interno da pastilha em relação à sua especificação.

D8 - Tempo de acesso lento, causado por cargas armazenadas nos circuitos de saída ou por longas linhas.

D9 - Recuperação lenta no amplificador de leitura, notada pela tendência do amplificador de permanecer no mesmo estado após uma longa série de bits similares, devido à carga impropriamente acumulada.

D10 - Recuperação lenta de escrita, provocando um acréscimo no tempo de acesso para a realização de uma leitura imediatamente após uma escrita.

A organização intena de memórias semicondutoras consiste de u'a matriz com dois decodificadores, cada um recebendo parte das linhas de endereço e ativando a linha e a coluna correspondente ao mesmo. A célula que estiver

na interseção será selecionada para escrita ou leitura quando os sinais apropriados forem ativados.Normalmente a matriz de células é quadrada,havendo,entretanto,variações na geometria devido ao processo de fabricação e ao projeto dos semicondutores empregados.

Com o propósito de descobrir um ou vários dos possíveis defeitos citados anteriormente, vários métodos de teste tem sido propostos[7,19,23,27], entre os quais:

T1 - Teste de Seleção Múltipla de Endereço:

Os e 1s alternados são escritos em posições sucessivas, começando com o endereço zero. Cada célula é então lida e verificada,enquanto se utiliza uma sequência de endereço,complemento do endereço,endereço,endereço mais um, e assim por diante. Esta sequência do endereçamento testa todas as possíveis trocas de N bits de endereço, onde N é o número de linhas de endereço da memória. Ao final deste procedimento, a memória é lida sequencialmente e o padrão de dados inicialmente gravado é conferido.

T2 - Teste de Deslocamento da Diagonal:

Este teste inicia com a memória contendo zeros em todas as células,exceto os da diagonal da matriz, que contem 1s. As células são todas lidas e verificadas, pela ordem,resultando longas sequências de zeros seguidas por um bit 1.A matriz é então carregada com a diagonal de 1s deslocada uma célula para a direita, e novamente verificada. Este processo se repete até que a diagonal retorne à posição inicial,quando então o dado 1 já foi armazenado em cada célula de memória. O teste é a seguir repetido com uma diagonal de zeros e com as demais células em 1.

T3 - Teste de 1s e 0s Marchando:

A memória é inicialmente preenchida com um padrão e,a seguir,lida sequencialmente,a partir do endereço zero e até o final da mesma. A cada endereço, o dado correspondente é conferido,complementado e reescrito na mesma posição. Ao se atingir o final,a memória é novamente percorrida, em ordem reversa, seguindo o mesmo procedimento até o endereço inicial, quando então o padrão inicial é invertido e todos os passos novamente executados.

T4 - Teste de Paridade Armazenada:

Similar ao anterior, mas consome menos tempo. Consiste em preencher a memória de forma que cada célula contenha a paridade do endereço a que corresponde.As células são da mesma forma lidas em ordem crescente e verificadas, sendo, ao final, todos os passos repetidos para a paridade inversa.

T5 - Teste de 1s e 0s Caminhando:

A memória é inicializada com um mesmo padrão.A seguir,cada célula é testada,escrevendo-se na mesma o inverso do padrão inicial, verificando todas as demais células para averiguar se permaneceram inalteradas e voltando à célula sob teste para confirmar se contém o inverso das demais.A célula é então reescrita com o padrão original e a sequência de operações,repetida para a próxima, até que todas tenham sido testadas. O procedimento é a seguir executado para a memória inicializada com o padrão inverso.

T6 - Teste de 1s e 0s Galopando:

Análogo ao anterior,com a diferença que,durante a verificação das demais células, a posição em teste é lida

após a leitura de cada uma delas. Da mesma forma que no teste anterior, a sequência é repetida para cada célula e todo o procedimento é executado também com o padrão inverso.

T7 - Teste Ping-Pong:

Consiste de uma variação simplificada de T6, com um único passo,que, ao invés de verificar o estado de todas as demais células durante o teste de cada uma delas, apenas inverte os bits de endereço,um por vez, e verifica a célula correspondente, voltando, após cada leitura,para a célula em teste.

T8 - Teste de Recuperação de Escrita:

Este teste verifica o efeito de se escrever dados, cada escrita seguida por uma leitura, para todos os possíveis pares de células. A memória inicia com qualquer valor; a célula em teste recebe um padrão e,para cada uma das demais células,segue-se a sequência: escrever o inverso do padrão na célula selecionada,ler a célula em teste, inverter o padrão da célula selecionada e ler novamente a célula em teste. Quando toda a sequência terminar em relação à primeira célula,testa-se a segunda,e assim por diante. Ao final, todo o procedimento é repetido com o padrão inverso do que foi inicialmente adotado nas células em teste.

O número de acessos exigido pelos testes,para u'a memória de tamanho N,é de,respectivamente, 5N, 2N $(\sqrt{N}+3)$, 10N, 4N, 2N (N+3), 2N (2N+1), N $(2\log_2 N+3)$, 2N (4N-3), incluindo o carregamento inicial da memória com os padrões que se fizerem necessários e supondo a matriz quadrada,no

caso de T2.

Uma vez que o ciclo de instrução de microprocessadores é bastante superior ao ciclo de leitura ou escrita de memória, torna-se impossível testar por programa acessos com a máxima taxa permitida. Desta forma,as falhas D9 e D10 não serão normalmente detetadas por esta modalidade de teste. Ocorre ainda que a falha D1 só poderá ser detetada se aparecer em extensão suficiente para prejudicar o funcionamento da memória.

Outro ponto a considerar no emprego de testes por programa é que a sequência de endereços aplicada tem pouco significado. Uma vez que as linhas correspondentes voltam sempre ao seu estado quiescente,entre dois ciclos consecutivos de memória.Desta forma,testes semelhantes a T1 perdem sua utilidade nestes casos.

As falhas D2,D3,D5,D6,D7 e D8 são todas detetadas pelos testes T2,T3,T4,T5,T6,T7 e T8.Os procedimentos mais complexos verificam melhor alguns aspectos,como, por exemplo, D6,com a consequente desvantagem de exigirem um número de acessos mais elevado.

A falha D4 necessita de um procedimento específico para ser detetada,consistindo no preenchimento da memória com determinado conteúdo e sua verificação,após decorrido um intervalo superior ao tempo de varredura das colunas ou linhas da matriz de memória das pastilhas constituintes. Para este caso,pode-se fazer uso de,por exemplo, sequências de endereços crescentes ou decrescentes, dados pseudo-aleatórios ou padrões básicos.

Deve-se testar ainda a independência entre bits

de uma mesma palavra de memória, a fim de assegurar a inexistência de acoplamentos unidirecionais ou bidirecionais entre os mesmos.

Quando as pastilhas constituintes apresentam apenas um bit de largura,esta verificação pode ser realizada em uma única palavra de cada conjunto de pastilhas e é útil para localizar curto-circuitos que estejam presentes na placa. Nos demais casos,a verificação deve se estender ao longo de toda a memória.

O emprego de um padrão circulante, correspondente à largura da pastilha, com um bit 1 e os demais em 0, bem como o inverso desta configuração, se mostra bastante adequado e simples de gerar para este tipo de teste.

Um outro método mais eficiente consiste em escrever e conferir K máscaras com metade dos bits em 1 e a outra metade em 0, sendo a máscara I constituída por I repetições de $N/(2^I)$ bits 1 e $N/(2^I)$ bits 0,com I variando de 1 a $\log_2(N)$,onde N é a menor potência de 2 maior ou igual à largura da palavra. Esta sequência de máscaras deteta qualquer acoplamento bidirecional entre os bits de uma palavra e,se complementada com o inverso das máscaras,cobre igualmente os acoplamentos unidirecionais.

2.4 - Testes de Memória ROM

Para verificar o conteúdo de memórias ROM,é usual o cálculo de algum tipo de informação redundante com os dados gravados e posterior comparação com o valor esperado.

Tem-se,por exemplo, o LRCC (Longitudinal Redundan

cy Check Code),que consiste na paridade longitudinal para cada um dos bits da palavra,o CRCC (Cyclic Redundancy Check Code), obtido através de um polinômio gerador aplicado aos bits de cada palavra,e a soma acumulada em um determinado número de bits,entre as formas mais empregadas de informação redundante para um bloco de dados.

Todos estes métodos permitem detetar trocas simples de bits e uma percentagem de trocas múltiplas,servindo como uma primeira verificação do correto conteúdo de uma ROM.

Quando se dispõe de um periférico de armazenamento,pode-se guardar em um arquivo uma cópia do conteúdo de cada uma das ROMs e,através da comparação palavra a palavra, detetar quaisquer trocas de bits que ocorram.

2.5 - Testes de Periféricos

A idéia básica em testes de periféricos consiste em procurar utilizá-los de maneira a garantir seu funcionamento em condições normais de operação.

2.5.1 - Periféricos de Saída

Em periféricos de saída, é usual o procedimento de gerar um conjunto de linhas de modo a ter em cada posição todos os caracteres utilizados pelo menos uma vez.

Testes mais específicos dependem do tipo de periférico de saída;em dispositivos com partes mecânicas,tais como teletipos e impressoras,costuma-se ainda verificar o alinhamento das partes móveis através da impressão de linhas de tamanho crescente ou decrescente, com vértice de

convergência em uma das bordas ou no centro; a regularidade do espaçamento entre caracteres por meio da repetição do mesmo símbolo em toda uma linha; espaçamento entre linhas, troca de folha, etc. Em vídeos, testa-se também a memória RAM associada aos mesmos.

2.5.2 - Periféricos de Entrada

Para periféricos de entrada,deve-se gerar u'a massa de dados para ser lida e conferida, visual ou automáticamente. A primeira alternativa pode ser vantajosamente empregada no caso de teclados, onde o eco visual dos caracteres enviados ou o valor do código gerado facilita sobremaneira a deteção de quaisquer anormalidades. Já a segunda se mostra mais adequada para leitoras de cartões ou de fitas de papel, onde é possível verificar a correspondência entre as perfurações e sua deteção ótica ou mecânica, usando um conjunto de caracteres fixo e preestabelecido.

Em qualquer dos casos, informações disponíveis,no decorrer dos testes, provenientes dos registros de estado do periférico em teste, constituem valiosos elementos para a verificação do correto funcionamento dos mesmos.

2.5.3 - Periféricos de Armazenamento

Em periféricos de armazenamento,tais como discos, diskettes,fitas,memórias de bolhas,K7,etc., a execução de um maior número de comandos é normalmente necessária para verificar seu perfeito funcionamento. Entretanto, uma vez que estes periféricos são usualmente enpregados para man

ter informações, este tipo de teste costuma residir em um dos periféricos e ser selecionado e executado apenas quando o núcleo do sistema estiver em ordem.

III - DESCRIÇÃO DO MICROCOMPUTADOR POTI

3.1 - Apresentação

O POTI é um microcomputador integralmente desenvolvido pelo Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro. Consiste,em sua configuração mínima, de uma unidade de processamento com microprocessador 8080, memórias (ROM e RAM) interfaces e periféricos: teclado, vídeo, impressora e disco.

Dispõe de um sistema operacional em disco(SOCO) e de um compilador para uma linguagem de alto nível (PLTI), semelhante ao PL/1, o que lhe permite desenvolver e executar programas localmente.Este compilador foi implementado através de um interpretador, a fim de propiciar maior portabilidade do software.

3.2 - Organização

Os diversos componentes do sistema se interligam através de uma via padrão de 44 linhas,descritas na tabela 3-1.

As linhas,com exceção de INTA IN, INTA OUT, HOLDA IN e HOLDA OUT, são conectadas em paralelo a todas as interfaces. As quatro linhas mencionadas são ligadas em cadeia, a partir da UCP. As saídas INTA OUT e HOLDA OUT de uma interface vão as linhas INTA IN e HOLDA IN da próxima. As interfaces que não se utilizam de interrupção e DMA tem estes sinais simplesmente conectados da forma descrita, diretamente no painel de interconexões.

A figura 3-1 apresenta a configuração do sistema

16	Linhas de Endereço	A00 - A15
8	Linhas de Dado	D0 - D7
6	Linhas de Alimentação	+12 V +5 V (2 LINHAS) 0 V (2 LINHAS) -12 V
5	Linhas de Sincronismo	I/OR I/OW MEMR MEMW NOREADY
2	Linhas de Controle	REFRESH RESET
3	Linhas para Interrupção	INT INTA IN INTA OUT
3	Linhas para DMA	HOLD HOLDA IN HOLDA OUT
1	Linha de Referência	Ø2 TTL

TABELA 3-1 - Linhas da via Padrão

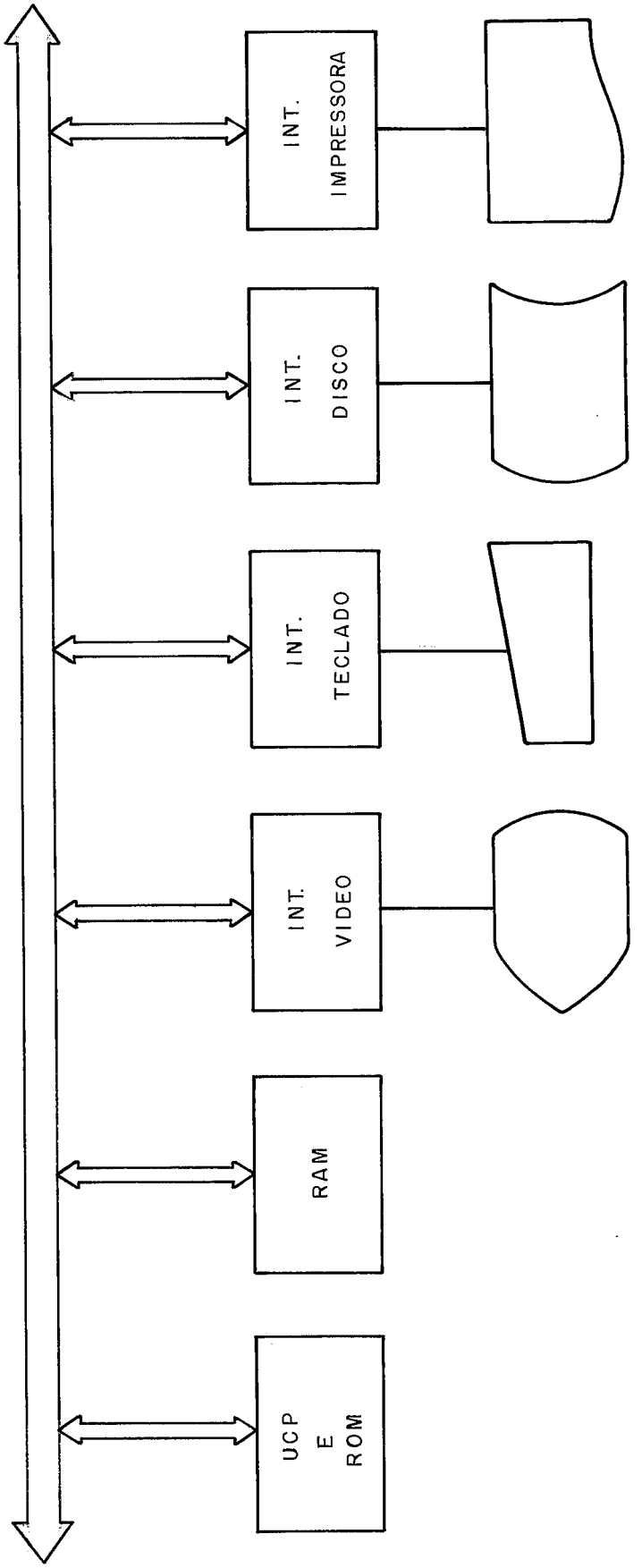

Fig. 3-1

Arquitetura do Poti (Configuração mínima)

básico, com 16 Kbytes de memória RAM.

3.3 - Unidade Central de Processamento

A UCP do sistema é constituída por um microprocessador 8080, com capacidade de atender pedidos de interrupção e de acesso direto à memória. A largura das palavras que o processador opera em paralelo é de 8 bits(1 byte) e sua capacidade de endereçamento é de 65536 palavras. O ciclo de máquina é de 500 ns, suas 72 instruções levam de 4 a 17 ciclos para execução,havendo necessidade de um ciclo adicional para cada escrita em memória, devido ao protocolo envolvido.

A figura 3-2 ilustra a alocação de endereços para os registros de entrada e saída, por onde se faz a comunicação com os periféricos, bem como a distribuição do espaço de endereçamento do processador.

3.4 - Memória RAM

A memória principal é organizada em bytes, denominando-se bit 7 ao mais significativo e bit 0 ao menos significativo bit de cada byte.

O sistema pode conter um máximo de 48 Kbytes de memória RAM, ocupando os endereços físicos entre 4000 e FFFF(HEX). Utilizam-se em sua implementação pastilhas de memória dinâmica, sendo o conteudo das mesmas restaurado periodicamente pela UCP, à razão de um acesso a cada instrução executada, durante os intervalos em que a via não estiver sendo usada pela instrução corrente, não havendo portanto degradação no desempenho da UCP.

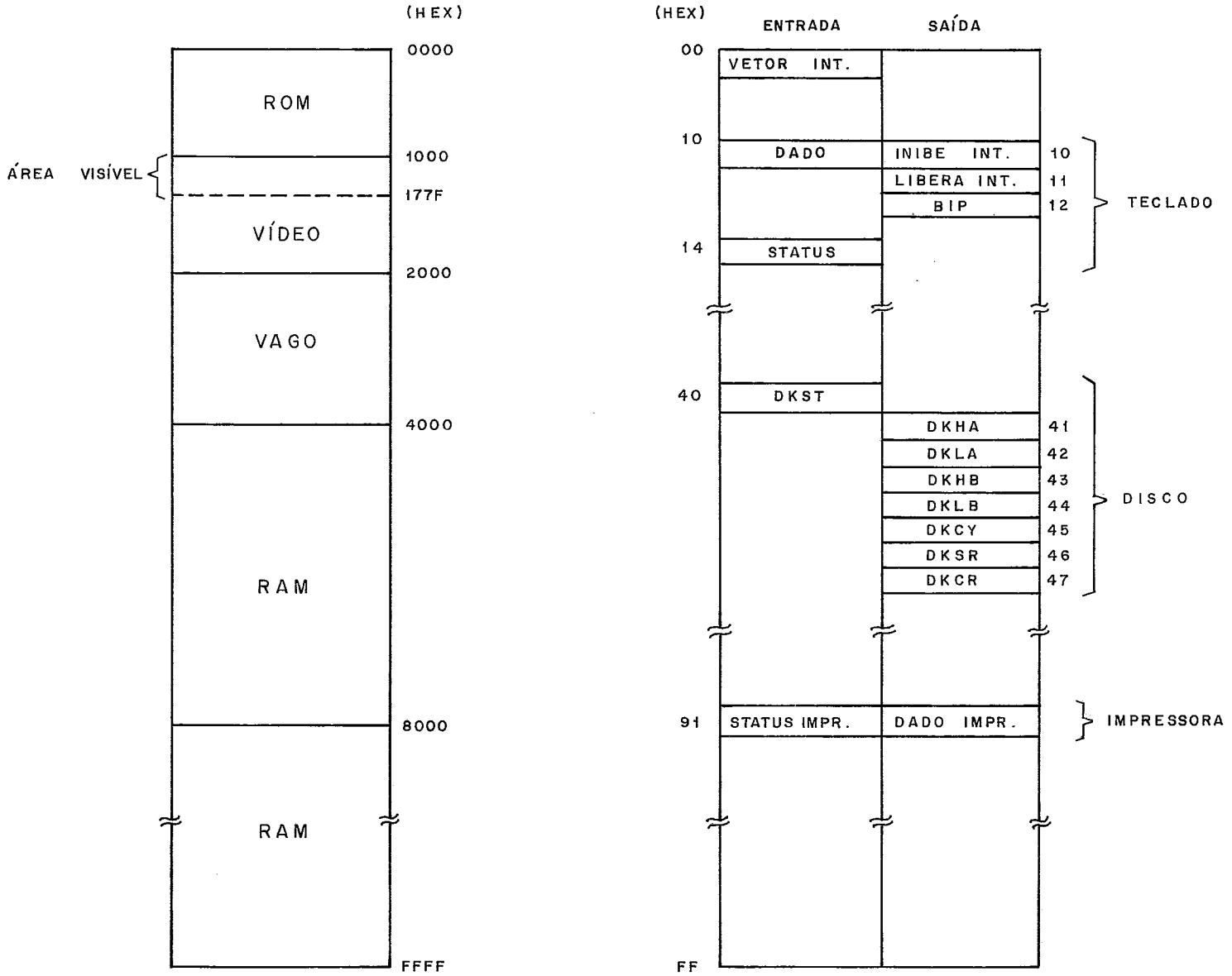

Fig. 3-2

Alocação dos Endereços dos Registros e de Espaço de Endereçamento

3.5 - Memória ROM

Os endereços entre 0000 e 1000(HEX) estão reservados para 4 Kbytes de memória ROM,onde estão gravados o interpretador da linguagem PLTI, algumas rotinas auxiliares e os procedimentos de inicialização do sistema e de tratamento de interrupção.

Ao ser ligado ou inicializado o sistema,a UCP passa a executar instruções a partir do endereço zero de seu espaço de endereçamento, onde se encontra o procedimento de carga do sistema operacional(partida fria).

3.6 - Pefifférico Teclado

O teclado compreende 52 teclas, sendo 47 codificadas,2 de função,2 teclas de deslocamento e 1 tecla de controle; sua saída apresenta código ASCII de 7 bits,em paralelo.

O código de saída, juntamente com informações de estado,sofre um processo de serialização para ser transmitido à interface de teclado, que por sua vez os restaura, recuperando a forma original. Este procedimento diminui o número de fios necessários à interligação do teclado com sua interface.

A interface de teclado é a única que pode gerar interrupções, desde que habilitada através de um de seus registros.

3.7 - Periférico Vídeo

A memória de vídeo ocupa os endereços entre 1000 e 2000(HEX), podendo ser acessada por qualquer instrução,

da mesma forma que uma palavra da memória principal.É também constituida por pastilhas de memória dinâmica,que, entretanto, não necessitam de restauração externa, uma vez que são constantemente lidas para manter visível,na tela, os caracteres correspondentes aos códigos ASCII presentes nos primeiros 1920 endereços (24 linhas por 80 caracteres). O restante dos 4 Kbytes é utilizado de modo normal, como memória do sistema, devendo-se notar que, durante a ocorrência de acessos pelo controlador de vídeo, a UCP pode ter que esperar até 88 us para completar um acesso à memória do vídeo, o que representa, no pior caso, uma degradação média de 15% na velocidade de execução das instruções.

3.8 - Periférico Disco

A interface de disco inclui circuitos de DMA e um controlador/formatador para até oito unidades, cada uma com capacidade formatada de 2,4 Mbytes.

As unidades aceitam discos removíveis, tipo cartucho,com duas superfícies magnéticas, 203 trilhas por superfície e 12 setores por trilha. Os setores são delimitados por fendas detetáveis oticamente,sendo cada setor formatado com um preâmbulo de 26 bytes zero,1 bit de sincronismo,2 bytes de cabeçalho, 512 bytes de dado, 2 bytes de soma acumulada dos dados e um postâmbulo com 2 bytes zero,pela ordem.O cabeçalho contém o número do cilindro correspondente,que é automaticamente conferido durante leituras ou escritas. Da mesma forma,a soma acumulada consiste de informação redundante com a área de dados,e é verifica

da durante leituras e gerada durante escritas, a fim de propiciar maior confiabilidade nas transferências de dados.

Há comandos de leitura, verificação e escrita de setor, além de posicionamento,inicialização e proteção da unidade contra escrita. Durante a área de dados do setor, ocorrem entre 1 e 512 transferências de dado por DMA entre a memória principal e a interface de disco, sem intervenção da UCP, que apenas inicia o comando. O sentido da transferência depende do comando em execução.

3.9 - Periférico Impressora

O sistema comporta uma impressora do tipo série, com matriz de pontos,velocidade até 165 caracteres por segundo, 132 caracteres por linha, código ASCII de 7 bits, impressão de caracteres normais ou elongados e armazenamento interno para uma linha. A impressora se conecta ao sistema por meio de uma interface paralela.

3.10 - Periféricos Adicionais

A configuração descrita se refere ao sistema básico. Outros periféricos podem ser conectados através da via padrão, desde que se utilize uma interface apropriada para cada um deles.

IV - TESTES APLICADOS AO POTI

4.1 - Sequenciamento dos Testes

O programa de teste desenvolvido para o POTI reside em uma ROM de 1 Kbytes,que substitui a primeira ROM do sistema quando se deseja testá-lo ou verificá-lo.Este procedimento permite que o programa tenha início ao se ligar o sistema, sem necessidade de chaves para comutar do modo de operação normal para manutenção;evita ainda o dispendio de uma ROM para cada unidade e possibilita sua implantação mesmo em sistemas já definidos, sem necessidade de quaisquer alterações em circuitos e interconexões.

O programa foi implementado no sentido de inicialmente testar a parte mais central do sistema,e, a seguir, fazendo uso das partes já verificadas, ir testando os demais módulos, como mostra a figura 4-1.

De acordo com esta filosofia, o programa tem início com o teste da UCP,onde,em caso de erro,o microprocessador para. Em funcionamento normal,um sinal audível(BIP) é emitido ao final do teste, para indicar que a parte central se encontra em perfeita ordem.

O próximo módulo a ser examinado é o vídeo, que é submetido a dois tipos de teste: um, visual, para a parte de varredura e geração de caracteres,e outro, automático, para a memória de armazenamento.

Deste ponto em diante, o vídeo,já verificado,está disponível para emitir quaisquer mensagens referentes aos demais módulos. A primeira mensagem,quando não ocorrem falhas, indica que a UCP e o vídeo estão funcionando.

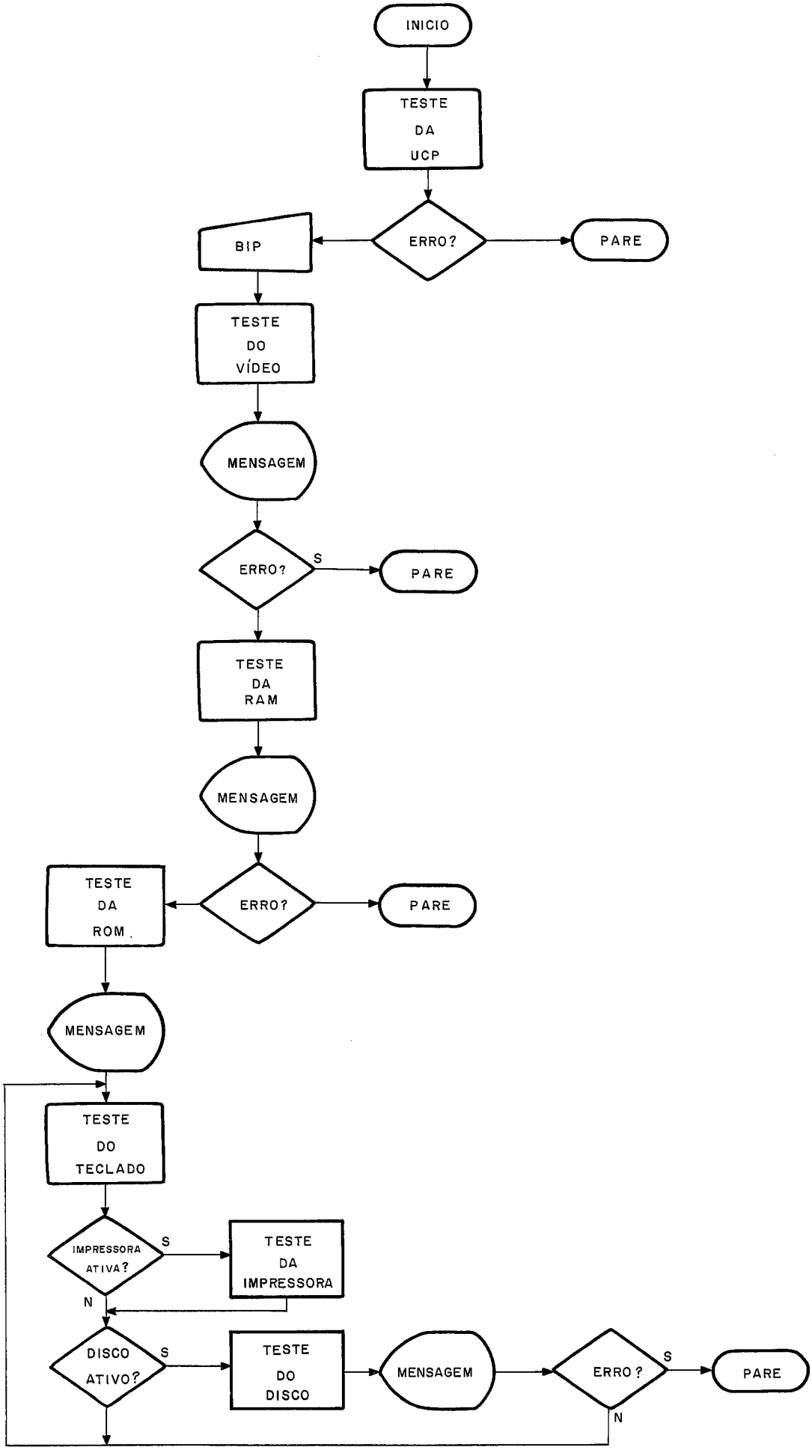

FIG. 4-1 · FLUXOGRAMA DO PROGRAMA DE TESTE.

DES. Cleber

A seguir é testada a memória principal do sistema, de modo a assegurar que cada uma de suas células pode armazenar 1 ou 0,que os ciclos de restauração ("REFRESH") estão ocorrendo com a cadência adequada para que os dados não se volatilizem, que não há curtos entre as linhas de dado ou endereço e que os decodificadores internos de cada pastilha de memória operam corretamente.

Logo após,são apresentadas, no vídeo, informações geradas a partir do conteúdo de cada uma das pastilhas de memória ROM do sistema,possibilitando detetar a maior parte das irregularidades que possam ocorrer nas mesmas.

A parte final do programa verifica os periféricos,um por vez.Os testes de disco e impressora tem início assim que a unidade envolvida for ligada e ativada.Enquanto ambos permanecerem inativos,o teclado estará implicitamente selecionado para teste.

4.2 - Teste da UCP

A complexibilidade dos CIs de integração de pequena e média escala fez com que os mesmos passassem a ser considerados agrupamentos lógicos ao invés de transistores, e cada pastilha fosse testada como uma unidade, sem dar importância aos parâmetros dos dispositivos constituintes. Da mesma forma, a maior complexidade dos CIs de integração de larga escala leva ao teste por módulos, onde os agrupamentos lógicos são ignorados em favor de blocos maiores: os subsistemas funcionais ou módulos.

Assim,os microprocessadores podem ser testados em termos de decodificação de registros e instruções, armaze

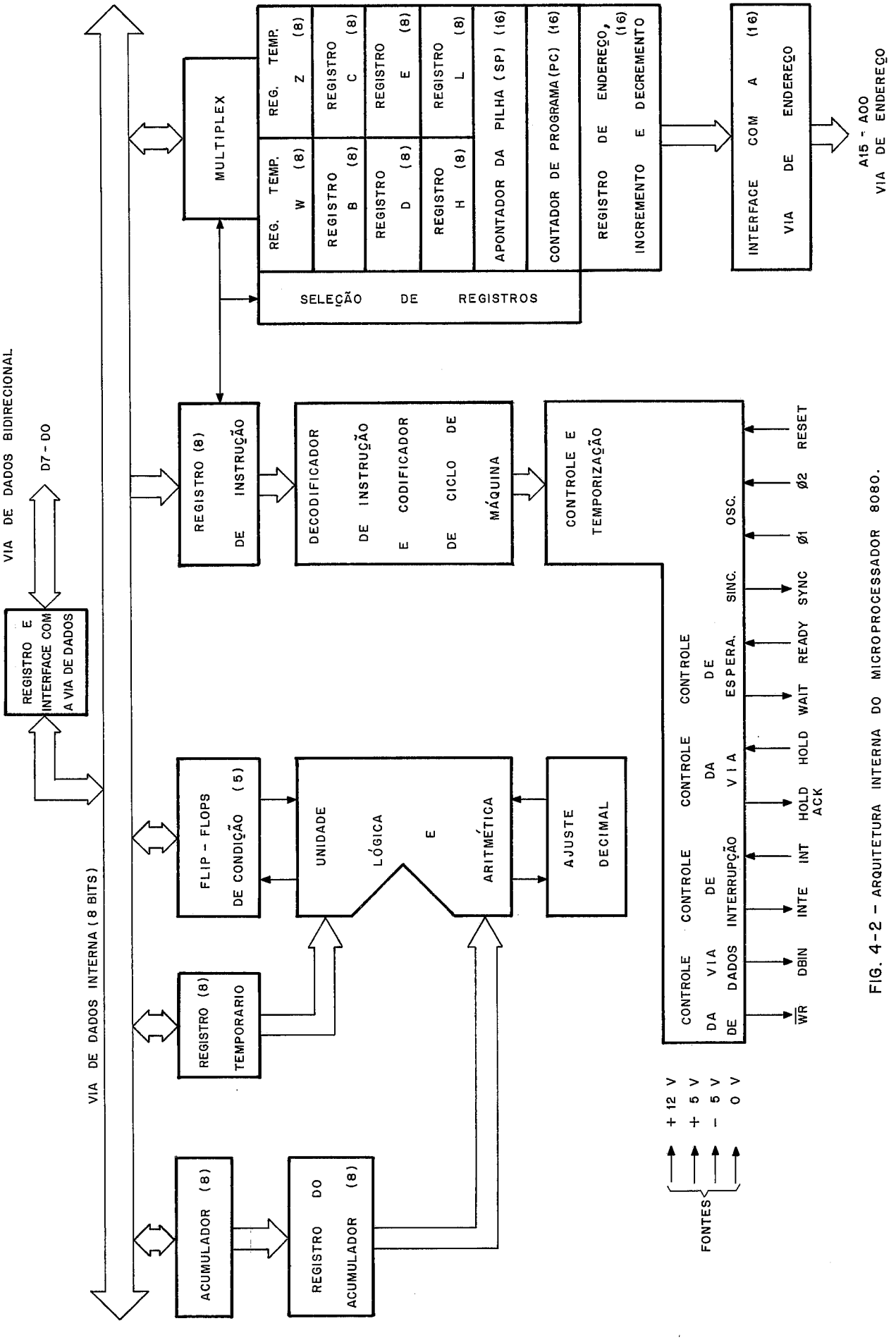

FIG. 4-2 - ARQUITETURA INTERNA DO MICROPROCESSADOR 8080.

namento, transferência e manipulação de dados.

O teste de UCP desenvolvido para o POTI consiste de uma sequência de instruções que procura verificar o funcionamento da arquitetura interna do microprocessador 8080[25], apresentada na figura 4-2. Supõe o correto funcionamento da instrução HALT e da lógica de atualização do contador de programa (PC),empregando estes fatos para verificar os demais módulos.

O método empregado é semelhante ao do PDP 11/70, citado no capítulo II. Vale notar, entretanto, que,quando se trata de um microprocessador, de pouca valia se torna localizar em que parte do mesmo se encontra o defeito,uma vez que toda a lógica reside normalmente em uma única pastilha. Desta forma, os testes são orientados no sentido de indicar se o microprocessador se encontra em ordem ou não, sem haver necessidade de dispor as instruções por grupo funcional.

Um teste funcional completo torna-se imprático,devido ao grande número de instruções envolvidas.As figuras 4-3A, 4-3B e 4-3C apresentam a sequência de operações aplicada, com correspondente evolução do conteúdo dos registros internos do microprocessador.

Cada um dos registros internos é empregado como fonte e destino, todas as instruções aritméticas e lógicas são utilizadas ao menos uma vez, cada um dos códigos de condição é testado em seus dois estados, e as instruções de desvio condicional são verificadas para ambas as condições. O resultado acumulado de uma sequência preestabelecida de operações aritméticas e lógicas é comparado,no fi

INST	LXI H	LXI D	XRAA	SUI 2	RLC	DAA	RAL	MOVCA	ORAD	MOVBA	ANAE	MOVHA	ADDC
A	-		[0]	[FE]	[FD]	[63]	[C7]	[C7]	[F7]	[F7]	[56]	[56]	1D
B	-									F7			
C	-							C7				[C7]	
D	-	B6						[B6]					
E	-	5E								[5E]			
H	00											56	
L	06												
SP	-												
Z	-		(1)	0		0			0		0	(0)	0
S	-		0	1		0			1	(1)	0		0
P	-		1	0		1			0	(0)	1	(1)	1
AC	-		0	1	[1]	0			0		0		1
CY	-		0	1		[1]	0		0		0	[0]	1

(·) — INDICA QUE A CONDIÇÃO É TESTADA.

FIG. 4-3A · EVOLUÇÃO DO CONTEÚDO DOS REGISTROS NO DECORRER DO TESTE DA UCP.

INST		MOVLA	SPHL	DCXSP	ADDH	RAR	SBBL	CMC	RAR	CMC	SBBB	CMA	CMC	RAL	STC	STC
A	1D			1D	73	39	1B	1B	8D	8D	96	69	69	D2		
B	F7															
C	C7															
D	B6															
E	5E															
H	56	56		56												
L	06	1D				1D										
SP	-		561D	561C												
Z	0				0		0				0					
S	0				0	(0)	0				1					
P	1			(1)	0		1				1					
AC	1				1		1				0					
CY	1				0	1	(0)	1	1	(0)	1	(1)	0	0	1	1

□ — INDICA QUE O VALOR É UTILIZADO NA PRÓXIMA INSTRUÇÃO.

FIG. 4 – 3B · EVOLUÇÃO DO CONTEÚDO DOS REGISTROS NO DECORRER DO TESTE DA UCP.

INST		ADCA	RRC	INRE	DCRD	INXB	DADSP	DADB	DADD	CPI/D2	MOVAH	CPI/59	MOVAL	CPI/60
A	D2	A5	D2						D2		59		60	
B	F7				F7	F7	F7							
C	C7				C7	C8	C8							
D	B6			B6	B5			B5						
E	5E		5E	5F				5F						
H	56					56	AC	A4	59	59				
L	1D					1D	39	01	60			60		
SP	561C					561C								
Z	0	0		0	0			0		1		1		1
S	1	1		0	1	1				0		0		0
P	1	1		1	0					1		1		1
AC	0	0		0	0					0		0		0
CY	1	1	1						1	0		0		0

TODOS OS VALORES ESTÃO EM HEXADECIMAL.

FIG.4-3C · EVOLUÇÃO DO CONTEÚDO DOS REGISTROS NO DECORRER DO TESTE DA UCP.

DES. Cleber

nal, com o valor esperado, e, em caso de discrepância, o microprocessador executa a instrução HALT e permanece parado. Para funcionamento normal, passa-se ao próximo teste, emitindo antes um sinal audível (BIP).

4.3 - Teste do Vídeo

O teste aplicado ao vídeo visa duas finalidades distintas; inicialmente sua área visível é preenchida com caracteres ASCII, um por vez, possibilitando a observação visual do correto funcionamento dos circuitos de comutação de endereço, escrita na memória do vídeo, varredura e geração dos caracteres por matriz de pontos, o que permite, a partir deste momento, usar o vídeo para dar mensagens sobre o bom ou mau funcionamento de módulos do sistema. Os códigos são gerados em ordem crescente, começando em zero, estando os caracteres correspondentes apresentados na tabela 4-1. O valor zero ou um para o bit mais significativo de cada posição de memória da área visível do vídeo indica que o símbolo correspondente ao código ASCII presente nos demais bits aparecerá na tela, respectivamente, em fundo preto ou em fundo branco.

Logo a seguir,a memória de armazenamento do vídeo é testada, empregando-se o já conhecido método de teste de 1s e 0s marchando (T3), apresentado no capítulo II. Quaisquer erros detetados no decorrer deste teste forçam o processador a parar, enquanto que, em caso normal, comuta-se para o próximo teste. Em qualquer dos casos, mensagens indicativas sobre os módulos já verificados são apresentadas na tela.

BITS 765 ▶ 4321 ▼	000	001	010	011	100	101	110	111
0000	■			0	@	P	`	p
0001			!	1	A	Q	a	q
0010			"	2	B	R	b	r
0011			#	3	C	S	c	s
0100			$	4	D	T	d	f
0101			%	5	E	U	e	u
0110			&	6	F	V	f	v
0111			'	7	G	W	g	w
1000		▶	(	8	H	X	h	x
1001			)	9	I	Y	i	y
1010			*	:	J	Z	j	z
1011			+	;	K	[	k	{
1100		∴	,	<	L	\	l	¦
1101			-	=	M	]	m	}
1110			.	>	N	Λ	n	~
1111			/	?	O	_	o	iv

TABELA 4-1

CARACTERES MOSTRADOS NO VÍDEO DE ACORDO COM O CÓDIGO ASCII.

4.4 - Teste da Memória RAM

O teste de memória RAM compreende três fases distintas: durante a primeira, os endereços inicial e final da memória são estabelecidos e o teste de K máscaras,para testar a independência entre os bits de uma mesma palavra de memória,é aplicado em sua versão simplificada, uma vez que as pastilhas empregadas tem apenas um bit de largura. A segunda fase corresponde ao teste de 1s e 0s marchando, já descrito anteriormente.Este teste foi selecionado devido às suas características de deteção de falhas, simplicidade e linearidade do número de acessos em função do tamanho das pastilhas de memória, o que assegura uma rápida execução e permite verificar a memória como um todo, sem a correspondente penalidade relativa ao crescimento excessivo do tempo necessário para completá-lo. Finalmente, a terceira fase procura simular o funcionamento real da memória, através do emprego de sequências de dados pseudo-aleatórios e retardamento do período entre ciclos sucessivos de restauração ("REFRESH") valendo-se da execução repetitiva da mais lenta instrução do microprocessador (XTHL), durante um intervalo superior a 100 vezes o período estabelecido pelo fabricante para varrer todas as linhas da matriz de memória (2 ms), findo o qual toda a memória é lida e conferida.

Indicações sobre o bom ou mau funcionamento deste módulo são também apresentadas no vídeo. Em caso normal, tem seguimento o próximo teste, ao passo que, na ocorrencia de anomalias, a UCP para.

4.5 - Teste da Memória ROM

A caracterização do conteúdo de cada uma das pastilhas ROM se efetua através da apresentação da assinatura em 16 bits, um particular tipo de CRCC (Cyclic Redundancy Check Code), capaz de detetar qualquer mudança em 1 bit e 99,998% de todas as possíveis trocas de bit[21].

O resultado não é automaticamente comparado com o valor esperado, a fim de permitir alterações nas ROMs do sistema sem necessidade de haver modificações no conteúdo da ROM de teste e também para possibilitar a obtenção da assinatura associada à primeira ROM do sistema, colocando a no lugar de uma das demais, que se situam na faixa de endereço entre 0800 e 0FFF(HEX), uma vez que o espaço correspondente à mesma se encontra tomado pela ROM de teste.

A maior complexidade no cálculo da assinatura é plenamente compensada pela excelente margem de deteção de erro, muito superior em relação ao que se pode obter com outros tipos de informação redundante com um bloco de dados.

4.6 - Teste do Teclado

O teclado é testado através da apresentação visual, no vídeo, dos caracteres batidos, do registro de estado e do código correspondente, sendo também submetido a uma verificação automática da ocorrência da interrupções, a cada vez que uma tecla é comprimida.

4.7 - Teste da Impressora

Todos os símbolos ASCII comportados pela impressora são transmitidos à sua interface, a fim de verificar o funcionamento da matriz de agulhas. Adicionalmente, o formato de impressão foi escolhido de forma a salientar quaisquer problemas relativos ao espaçamento entre linhas e caracteres, ao posicionamento da cabeça móvel e à perda de caracteres transmitidos.

A comparação visual dos resultados impressos com padrões já conhecidos permite detetar a existência de falhas nos mecanismos da impressora ou na lógica correspondente.

4.8 - Teste do Disco

O teste do disco compreende inicialmente uma sequência de operações para testar os indicadores de painel das unidades e os estados associados à mesmas.

A seguir, são realizadas leituras a cada trilha para verificar o mecanismo de posicionamento e a formatação do disco.

A última parte do teste realiza sucessivos comandos de verificação para uma mesma trilha, com o propósito de garantir que os dados na memória não se alterem com o uso repetitivo de ciclos de DMA.

V - CONCLUSÃO

Os testes implementados se mostraram extremamente úteis durante o desenvolvimento do protótipo do POTI,para verificação geral de unidades prontas e para encontrar falhas em caso de manutenção.

Outros testes mais detalhados para os periféricos já mencionados, ou relativos a periféricos adicionais, podem residir em disco e ser carregados para execução, após a conclusão dos testes preliminares.

Neste sentido, já existe um conjunto de testes,escritos em PLTI, para o periférico disco, com a finalidade de auxiliar o alinhamento das cabeças e testar de modo extensivo os comandos, principalmente os de escrita,que,por questões de segurança, não foram incluídos na ROM de teste.

Com vistas a futuras expansões, há também espaço alocado em disco,fora da área utilizada pelo SOCO, previsto para testar escrita e formatação de setores e para comportar um pequeno sistema operacional de manutenção.

Os programas de teste, apesar de não apontarem diretamente a causa dos defeitos, se revelaram uma preciosa ferramenta para indicar a ocorrência de falhas e fornecer informações relativas às mesmas que muito facilitaram sua resolução. Através de seu uso foram detetados vários problemas relativos à temporização dos ciclos de memória RAM e do canal DMA do disco,endereçamento do vídeo,insuficiência da taxa de restauração ("REFRESH") da memória, pastilhas com defeito e até mesmo acoplamento mútuo entre as linhas da via padrão ("CROSS-TALK").

Apesar de aplicado a um microcomputador específico, o programa desenvolvido pode ser facilmente adaptado, no todo ou em parte, a sistemas semelhantes que utilizem microprocessadores 8080, 8085 ou Z80,uma vez que todos englobam o conjunto de instruções empregado, havendo compatibilidade a nível de código de máquina.

Uma versão modificada deste programa tem sido utilizada, com bons resultados, no sistema de entrada de dados SDE-40, industrialmente produzido pela EMBRACOMP S.A., para agilizar a manutenção e prover um mecanismo de aprovação dos equipamentos prontos.

BIBLIOGRAFIA

1. BALLARD,D,R. - Designing fail-safe microprocessor systems,ELECTRONICS, Vol. 52, Nº 1, Jan 4,1979.
2. BELL,C.G.;J.C. MUDGE;J.E. MACNAMARA - COMPUTER ENGINEERING:A DEC VIEW OF HARDWARE SYSTEMS DESIGN,Digital Press, 1978.
3. BONEY,J. - Let your next microcomputer check itself and cut down your testing overhead, ELECTRONIC DESIGN, Vol. 27, Nº 18, Sep 1,1979.
4. CHANG,H.Y.;E. MANNING;G. METZE - FAULT DIAGNOSIS OF DIGITAL SYSTEMS,Wiley, 1970.
5. CHIANG,A.C.L.;R. MCCASKILL - Two new approaches simplify testing of microprocessors,ELECTRONICS, Vol. 49, Nº 2, Jan 22,1976.
6. CHIANG,A.C.L.;D. HACKMEISTER - Microprocessor test technique reveals instruction pattern sensitivity, COMPUTER DESIGN, Vol. 14, Nº 12, Dec 1975.
7. CHIANG,A.C.L.;R. STANDRIDGE - Pattern sensitivity on 4K RAM devices,COMPUTER DESIGN, Vol. 14, Nº 2, Feb 1975.
8. CHRISTY,P. - Minicomputer architecture links past and future generations, ELECTRONICS, Vol. 51,Nº 14, Jul 6,1978.
9. CORDERO,H.JR. - 4341's infrastructure is new from the substrate up,ELECTRONICS,Vol. 52, Nº 23, Nov 8, 1979.
10. DIGITAL EQUIPMENT CORPORATION - Decsystem-10 technical summary, 1976.

11. DIGITAL EQUIPMENT CORPORATION - PDP 11/70 maintenance and instalation manual, Jan 1977.

12. DIGITAL EQUIPMENT CORPORATION - PDP 11/70 processor handbook, 1976.

13. DIGITAL EQUIPMENT CORPORATION - M9301 bootstrap/terminator module maintenance and operator's manual, Jun 1977.

14. DONN,E.S.;M.D.LIPPMAN - Design forethought promotes easier testing of microcomputer boards, ELECTRONICS, Vol. 52, Nº 2, Jan 18,1979.

15. FIRMAN,A.H,;FOLEY,E.B. - Testing microcomputer boards automatically, COMPUTER DESIGN, Vol. 15, Nº 12, Dec 1976.

16. FOOSE,R. - Module minimizes repair time of process--control systems, ELECTRONICS, Vol. 51,Nº 5,Mar 2,1978.

17. FRECHETTE,T.J.;F. TANNER - Support processor analyzes errors caught by latches, ELECTRONICS, Vol. 52, Nº 23, Nov 8,1979.

18. GILLOW,G. - Simplifying processor maintenence with a carefully designed maintenance panel, COMPUTER DESIGN, Vol. 14, Nº 7, Jul 1975.

19. GOLDBLATT,R.C. - How computers can test their own memories,COMPUTER DESIGN, Vol. 15, Nº 7,Jul 1976.

20. GOODNER,R.;M. NEIL - Designing a serviceman's needs into microprocessor-based systems, ELECTRONICS, Vol. 52, Nº 2, Mar 1,1979.

21. GORDON,G.;H. NADIG - Hexadecimal signatures identify troublespots in microprocessor systems,ELECTRO

NICS, Vol. 50, Nº 5, Mar 3,1977.

22. HNATEK,E.R. - Checking microprocessors?, ELECTRONIC DESIGN, Vol. 23, Nº 22, Oct 25,1975.

23. HNATEK,E.R. - 4-kilobit memories present a challenge to testing, COMPUTER DESIGN, Vol. 14, Nº 5, May 1975.

24. HOLDERBY,W.S. - Diagnostic structures and formats for complex computer systems, COMPUTER DESIGN, Vol. 14, Nº 5, May 1975.

25. INTEL CORPORATION - 8080 microcomputer systems user's manual, Sep 1975.

26. JONES,T.;P. THOMAS - Challenges in microprocessor system design,COMPUTER DESIGN, Vol. 15, Nº 11, Nov 1976.

27. JONGE,J.H. DE;A.J. SMULDERS - Moving invertions test pattern is thorough,yet speedy,COMPUTER DESIGN, Vol. 15, Nº 5, May 1976.

28. LEMKE,D.;D. SMITH;T. TUNDER - Hardware emulation conquers the testing mountain created by 16-bit uPs, ELECTRONIC DESIGN, Vol. 27, Nº 14, Jul 5, 1979.

29. NEESE,J.W. - Microprocessor system validation and failure isolation with portable tester,COMPUTER DESIGN, Vol 16, Nº 9, Sep 1977.

30. PYNN,C. - In-circuit tester using signature analysis adds digital MSI to its range, ELECTRONICS, Vol. 52, Nº 11, May 4,1979.

31. RAMAMOORTHY,C.V. - A structural theory of machine diagnosis, spring joint computer conference, AFIPS

CONFERENCE PROCEEDINGS, Vol. 30,1967.

32. SCHUSHEIM,B. - A flexible approach to microprocessor testing,COMPUTER DESIGN, Vol.15, Nº 3,Mar 1976.

33. SHARRIT,D. - Team up a uP with signature analysis and ease troubleshooting in the field,ELECTRONIC DESIGN, Vol. 27, Nº 1, Jan 4,1979.

34. THOMAS,J.J. - Common misconceptions in digital test generation, COMPUTER DESIGN, Vol. 16, nº 1, Jan 1977.

35. WASER, S. - PROM card simplifies computer diagnosis, COMPUTER DESIGN, Vol. 16, Nº 1, Jan 1977.

APÊNDICE

LISTAGEM DO PROGRAMA DE TESTE

P1 P2 P3 P4 P5

```
*****************************************************************************
*                                                                           *
*                        FAULT DETECTION PROGRAM                            *
*                                                                           *
*                          ***** V2.0/1980 *****                            *
*                                                                           *
*                   POTI - NCE/UFRJ - RIO DE JANEIRO                        *
*                                                                           *
*                        HUMBERTO DOS SANTOS MELIM                          *
*                                                                           *
*****************************************************************************
```

```
                        ********************************************************************************
                        *                                                                              *
                        *                                CPU/ROM                                       *
                        *                                                                              *
                        ********************************************************************************
                              ABS     0
                        *                                                                              *
F3                            DI              *    DISABLE INTERRUPT                                  *
21 06 00                      LXTH    L1      *    (HL)=L1                                            *
E9                            PCHL            *    (PC)=L1                                            *
76                            HLT             *    STOP ON ERROR                                      *
11 5E B6              L1      LXTD    =/49A2  *    (DE)=B65E                                          *
AF                            XRAA            *    (A)=0, Z BIT=1                                     *
C2 2A 00                      JNZ     ERROR   *    FALSE CONDITION,NO JUMP                            *
CA 11 00                      JZ      L2      *    TRUE CONDITION,JUMP                                *
76                            HLT             *    STOP ON ERROR                                      *
D6 02                 L2      SUI     2       *    (A)=FE                                             *
07                            RLC             *    (A)=FD                                             *
27                            DAA             *    (A)=63, C BIT=1                                    *
17                            RAL             *    (A)=C7                                             *
4F                            MOVCA           *    (C)=C7                                             *
B2                            ORAD            *    (A)=F7, P BIT=0, S BIT=1                           *
47                            MOVBA           *    (B)=F7                                             *
EA 2A 00                      JPE     ERROR   *    FALSE CONDITION,NO JUMP                            *
FA 20 00                      JM      L3      *    TRUE CONDITION,JUMP                                *
76                            HLT             *    STOP ON ERROR                                      *
A4                    L3      ANAE            *    (A)=56, P BIT=1, Z BIT=0, C BIT=0                  *
67                            MOVHA           *    (H)=56                                             *
EA 26 00                      JPE     L4      *    TRUE CONDITION,JUMP                                *
76                            HLT             *    STOP ON ERROR                                      *
```

```
P1 P2 P3 P4 P5

00                        L4    NOP          *                                  *
C2 2B 00                        JNZ    L5    *   TRUE CONDITION,JUMP            *
76                        ERROR HLT          *   STOP ON ERROR                  *
89                        L5    ADCC         *   (A)=1D, P BIT=1                *
6F                              MOV L,A      *   (L)=1D                         *
F9                              SPHL         *   (SP)=561D                      *
3B                              DCX SP       *   (SP)=561C                      *
E2 2A 00                        JPO    ERROR *   FALSE CONDITION,NO JUMP       *
84                              ADDH         *   (A)=73, S BIT=0, C BIT=0       *
1F                              RAR          *   (A)=39, C BIT=1                *
C3 3A 00                        JMP    INT+2 *   #38 IS RESERVED FOR INTERRUPT  *
76                              HLT          *   STOP ON ERROR                  *
77                        INT   MOV M,A      * DISPLAY KEYBOARD CHARACTER       *
C9                              RET          * RETURN                           *
FA 2A 00                        JM     ERROR *   FALSE CONDITION,NO JUMP       *
9D                              SBB L        *   (A)=1B, C BIT=0                *
DA 2A 00                        JC     ERROR *   FALSE CONDITION,NO JUMP       *
3F                              CMC          *   C BIT=1                        *
1F                              RAR          *   (A)=8D, C BIT=1                *
3F                              CMC          *   C BIT=0                        *
D2 48 00                        JNC    L6    *   TRUE CONDITION,JUMP            *
76                              HLT          *   STOP ON ERROR                  *
98                        L6    SBB B        *   (A)=96, C BIT=1                *
2F                              CMA          *   (A)=69                         *
DA 4E 00                        JC     L7    *   TRUE CONDITION,JUMP            *
76                              HLT          *   STOP ON ERROR                  *
```

```
P1 P2 P3 P4 P5

3F                      L7      CMC             *   C BIT=0                                                 *
17                              RAL             *   (A)=D2, C BIT=0                                         *
37                              STC             *   C BIT=1                                                 *
37                              STC             *   C BIT=1                                                 *
8F                              ADCA            *   (A)=A5                                                  *
0F                              RRC             *   (A)=D2                                                  *
1C                              TNRF            *   (E)=5F, S BIT=0                                         *
F2 59 00                        JP      L8      *   TRUE CONDITION,JUMP                                     *
76                              HLT             *   STOP ON ERROR                                           *
15                      L8      DCRD            *   (D)=A5, Z BIT=0, S BIT=1, P BIT=0                       *
E2 5E 00                        JPO     L9      *   TRUE CONDITION,JUMP                                     *
76                              HLT             *   STOP ON ERROR                                           *
03                      L9      TNXB            *   (BC)=F7C8                                               *
F2 2A 00                        JP      ERROR   *   FALSE CONDITION,NO JUMP                                 *
39                              DADSP           *   (HL)=AC39                                               *
09                              DADB            *   (HL)=A401                                               *
DA 2A 00                        JZ      ERROR   *   FALSE CONDITION,NO JUMP                                 *
19                              DADD            *   (HL)=5960, C BIT=1                                      *
D2 2A 00                        JNC     ERROR   *   FALSE CONDITION,NO JUMP                                 *
FE D2                           CPI     /D2     *   Z BIT=1                                                 *
C2 2A 00                        JNZ     ERROR   *   FALSE CONDITION,NO JUMP                                 *
7C                              MOVAH           *   (A)=59                                                  *
FE 59                           CPI     /59     *   Z BIT=1                                                 *
C2 2A 00                        JNZ     ERROR   *   FALSE CONDITION,NO JUMP                                 *
7D                              MOVAL           *   (A)=60                                                  *
FE 60                           CPI     /60     *   Z BIT=1                                                 *
C2 2A 00                        JNZ     ERROR   *   FALSE CONDITION,NO JUMP                                 *
D3 12                           OUT     /12     *   SIGNAL END OF CPU TEST(BIP)                             *
C3 BF 01                        JMP     VTEST   *   UNCONDITIONAL JUMP                                      *
                        *                                                                                   *
                        ****************************************************************************************
```

```
P1 P2 P3 P4 P5

                        ********************************************************************
                        *                                                                  *
                        *                        SYSTEM SUBROUTINES                        *
                        *                                                                  *
                        ********************************************************************
                        *                                                                  *
E5                      CLEAR PUSHH             *   SAVE DISPLAY POINTER                   *
36 20                   SPC   MVIM    /20       *   ASCII SPACE                            *
23                            INXH              *   MOVE DISPLAY POINTER TO NEXT POSITION  *
7C                            MOVAH             *   REPEAT                                 *
FE 17                         CPI     /17       *    WHILE                                 *
C2 82 00                      JNZ     SPC       *     (H)(L)</1780                        *
7D                            MOVAL             *      (END OF                            *
FE 80                         CPI     /80       *       DISPLAY                          *
C2 82 00                      JNZ     SPC       *        AREA)                           *
E1                            POPH              *   RESTORE DISPLAY POINTER                *
C9                            RET               *   RETURN                                 *
                        *                       ********************************************
21 00 10                ROLL  LXIH    /1000     *   FIRST VIDEO POSITION                   *
F5                            PUSHPS            *   SAVE (A)                               *
C5                            PUSHB             *   SAVE (B)(C)                            *
01 50 10                      LXIB    /1050     *   BOTH VIDEO POSITION                    *
0A                      UP    LDAXB             *   GET VIDEO CHARACTER                    *
77                            MOVMA             *   MOVE ONE LINE UP                       *
03                            INXB              *   MOVE POINTERS TO                       *
23                            INXH              *    NEXT POSITION                         *
78                            MOVAB             *   REPEAT                                 *
FF 17                         CPI     /17       *     WHILE                                *
C2 9B 00                      JNZ     UP        *     (B)(C)</1780                         *
79                            MOVAC             *      (END OF                            *
FE 80                         CPI     /80       *       DISPLAY                          *
C2 9B 00                      JNZ     UP        *        AREA)                           *
CD 81 00                      CALL    CLEAR     *   CLEAR LAST LINE                        *
C1                            POPB              *   RESTORE (B)(C)                         *
F1                            POPPSW            *   RESTORE (A)                            *
C9                            RET               *   RETURN                                 *
                        *                       ********************************************
```

```
P1 P2 P3 P4 P5

                                *                       ***********************************************************
1A                              MSG     LDAXD           * READ CHARACTER                                          *
13                                      INXD            * UPDATE POINTER                                          *
FE 0A                                   CPI     /0A     * ASCII LF                                                *
C2 BE 00                                JNZ     NX1     * ON LINE FEED,                                           *
CD 93 00                                CALL    ROLL    *  ROLL DISPLAY                                           *
C3 B1 00                                JMP     MSG     *   AND CONTINUE                                          *
FE 03                           NX1     CPI     /03     * ASCII ETX                                               *
C8                                      RZ              * END OF TEXT                                             *
77                                      MOVMA           * DISPLAY CHARACTER                                       *
23                                      INXH            * UPDATE DISPLAY POINTER                                  *
C3 B1 00                                JMP     MSG     * REPEAT                                                  *
                                *                       ***********************************************************
30 31 32 33 34                  CONVT   DC      C'0123456789ABCDEF'                                               *
35 36 37 38 39
41 42 43 44 45
46

0F                              ASCXH   RRC             *   SWAP                                                  *
0F                                      RRC             *    HEXADECIMAL                                          *
0F                                      RRC             *     DIGITS                                              *
0F                                      RRC             *       IN (A)                                            *
E6 0F                           ASCXL   ANI     /0F     *   MASK LOWER ORDER DIGIT                                *
D5                                      PUSHD           * SAVE (DE)                                               *
11 C6 00                                LXID    CONVT   * POINTER TO CONVERSION TABLE                             *
83                                      ADDE            * ADD RELATIVE POSITION                                   *
5F                                      MOVEA           *  TO TABLE POINTER                                       *
1A                                      LDAXD           * GET CONVERTED CHARACTER                                 *
77                                      MOVMA           * AND MOVE IT TO VISIBLE AREA                             *
23                                      INXH            * UPDATE VIDEO POINTER                                    *
D1                                      POPD            * RESTORE (DE)                                            *
C9                                      RET             *                                                         *
                                *                       ***********************************************************
```

```
P1 P2 P3 P4 P5

                        *                       ********************************************************
F5                      ASCX    PUSHPS          * SAVE (A)                                             *
CD D6 00                        CALL    ASCXH   *   DISPLAY MOST SIGNIFICANT DIGIT                      *
F1                              POPPSW          * RESTORE (A)                                          *
F5                              PUSHPS          * SAVE (A)                                             *
CD DA 00                        CALL    ASCXL   *   DISPLAY LESS SIGNIFICANT DIGIT                     *
F1                              POPPSW          * RESTORE (A)                                          *
C9                              RET             *   RETURN                                             *
                        *                       ********************************************************
21 F0 1F                INIT    LXIH    ERTAB   * POINTER TO MEMORY ERROR TABLE                        *
01 00 0C                        LXIB    /0C00   * TABLE SIZE AND INITIAL VALUE                         *
71                      CLR     MOVMC           * CLEAR MEMORY                                         *
23                              INXH            * UPDATE POINTER                                       *
05                              DCRB            * REPEAT                                               *
C2 F8 00                        JNZ     CLR     *  12 TIMES                                            *
C9                              RET             * RETURN                                               *
                        *                       ********************************************************
E5                      RAMER   PUSHH           * SAVE CURRENT ADDRESS                                 *
32 FD 1F                        STA     REFLG   * SET RAM ERROR FLAG                                   *
4F                              MOVCA           * SAVE BAD BITS MASK                                   *
7C                              MOVAH           * FIND                                                 *
0F                              RRC             *  4K BLOCK                                            *
0F                              RRC             *   NUMBER,                                            *
0F                              RRC             *    STARTING                                          *
0F                              RRC             *     AT                                               *
E6 0F                           ANI     /0F     *      NUMBER 0                                        *
D6 04                           SUI     4       *       FOR FIRST BLOCK                                *
21 F0 1F                        LXIH    ERTAB   * FIND RELATIVE                                       *
85                              ADDL            *  POSITION INSIDE                                     *
6F                              MOVLA           *   ERROR TABLE                                        *
7E                              MOVAM           * READ CONTENT                                         *
B1                              ORAC            *  AND SET                                             *
77                              MOVMA           *   BAD BITS                                           *
E1                              POPH            * RESTORE CURRENT ADDRESS                              *
C9                              RET             *                                                      *
                        *                       ********************************************************
```

```
P1 P2 P3 P4 P5

                        *
                        ************************************************************
11 F0 1F                SWERR   LXTD    ERTAB * POINTER TO MEMORY ERROR TABLE                    *
1A                      VRFY    LDAXD         * GET MEMORY BLOCK ERROR WORD                      *
B7                              ORAA          * WORD IS ZERO                                     *
CA 29 01                        JZ      BKOK  *  FOR GOOD BLOCKS                                 *
7B                              MOVAE         * GET WRONG BLOCK NUMBER                           *
CD DA 00                        CALL    ASCXI * CONVERT AND DISPLAY IT                           *
23                              INXH          * MOVE DISPLAY POINTER TO NEXT POSITION            *
1A                              LDAXD         * GET BAD BITS MASK                                *
CD E7 00                        CALL    ASCX  * CONVERT AND DISPLAY IT                           *
23                              INXH          * MOVE DISPLAY POINTER TO NEXT POSITION            *
13                      BKOK    INXD          * CHANGE POINTER TO NEXT BLOCK ERROR WORD           *
7B                              MOVAE         * REPEAT FOR                                       *
FE FC                           CPI     /FC   *  ALL MEMORY                                      *
C2 1A 01                        JNZ     VRFY  *   BLOCKS                                         *
76                              HLT           * STOP                                             *
                        *       ************************************************************
3A FE 1F                SETMA   LDA     LMADR * INITIAL MEMORY ADDRESS                           *
67                              MOVHA         * MEMORY POINTER, HIGH BYTE                        *
2E 00                           MVIL    0     * MEMORY POINTER, LOW BYTE                         *
16 52                           MVID    82    * INITIAL TERM FOR PSEUDO-RANDOM DATA              *
C9                              RET           * RETURN                                           *
                        *       ************************************************************
7A                      RAND    MOVAD         * LAST PSEUDO RANDOM NUMBER                        *
C6 BD                           ADI     189   *  PLUS 189                                        *
57                              MOVDA         *   MOD 256                                        *
C9                              RET           * RETURN                                           *
                        *       ************************************************************
37                      ONES    STC           * C BIT=1                                          *
3F                              CMC           * C BIT=0                                          *
17                              RAL           * CHECK IF                                         *
D2 46 01                        JNC     ZEROR *  MSB=1                                          *
0C                              INRC          * INCREMENT ONES' COUNTER                          *
B7                      ZEROR   ORAA          * REPEAT IF                                        *
C2 3F 01                        JNZ     ONES  *  ANY ONE BIT REMAINS                             *
C9                              RET           * RETURN                                           *
                        *       ************************************************************
```

```
P1 P2 P3 P4 P5

                        *
                        WTDSK   IN      DKST    ********************************************************
DB 40                           ANI     /C1     * READ DISK STATUS                                     *
E6 C1                           CPI     /C1     * WAIT CONTROL READY,                                  *
FF C1                           JNZ     WTDSK   *  R/W/S READY AND                                     *
C2 4B 01                        RET             *   FILE READY                                         *
C9                      *                       * RETURN                                              *
                        SKEX    MOVAD           ********************************************************
7A                              PUSHH           * CYLINDER                                             *
E5                              INXH            * SAVE DISPLAY POINTER                                 *
23                              CALL    ASCX    * MOVE DISPLAY POINTER TO NEXT POSITION                *
CD E7 00                        POPH            * DISPLAY CYLINDER ADDRESS                             *
E1                              OUT     DKCY    * RESTORE DISPLAY POINTER                              *
D3 45                           MVIA    3       *  ADDRESS                                             *
3E 03                           OUT     DKCR    * SEEK                                                 *
D3 47                           LDA     FMADR   *  COMMAND                                             *
3A FF 1F                        OUT     DKHA    * MEMORY ADDRESS,                                      *
D3 41                           XRAA            *  HIGH BYTE                                           *
AF                              OUT     DKLA    * MEMORY ADDRESS,                                      *
D3 42                           INRA            *  LOW BYTE                                            *
3C                              OUT     DKHB    * BYTE COUNT,                                          *
D3 43                           MVIA    /FF     *  HIGH BYTE                                           *
3E FF                           OUT     DKLB    * BYTE COUNT,                                          *
D3 44                           CALL    WTDSK   *  LOW BYTE                                            *
CD 4B 01                        MOVAE           * WAIT END OF COMMAND                                  *
7B                              OUT     DKCR    * DISK                                                 *
D3 47                           CALL    WTDSK   *  COMMAND                                             *
CD 4B 01                        IN      DKST    * WAIT END OF COMMAND                                  *
DB 40                           ANI     /1E     * READ STATUS REGISTER                                 *
E6 1E                           RZ              * LOOK FOR ERRORS                                      *
C8                              INXH            * RETURN FOR NORMAL OPERATION                          *
23                              IN      DKST    * MOVE DISPLAY POINTER TO NEXT POSITION                *
DB 40                           CALL    ASCX    * READ DISK STATUS                                     *
CD E7 00                        HLT             * DISPLAY ERROR                                        *
76                      *                       * STOP                                                *
                        *                       ********************************************************
```

```
P1 P2 P3 P4 P5

                        *  ************************************************************
21 44 17                DPRS    LXIH    /1744 * 30TH POSITION AT LAST VIDEO LINE                 *
36 50                           MVIH    /50   * ASCII: 'P'                                      *
23                              INXH          * MOVE DISPLAY POINTER TO NEXT POSITION           *
36 52                           MVIM    /52   * ASCII: 'R'                                      *
23                              INXH          * MOVE DISPLAY POINTER TO NEXT POSITION           *
23                              INXH          * MOVE DISPLAY POINTER TO NEXT POSITION           *
DB 91                           IN      /91   * PRINTER STATUS                                  *
CD E7 00                        CALL    ASCX  * DISPLAY STATUS                                  *
FE 81                           CPI     /81   * SELECT,READY AND NOT BUSY                       *
C9                              RET           *                                                 *
                        *  ************************************************************
CD 86 01                WPR     CALL    DPRS  * DISPLAY PRINTER STATUS                          *
C8                              RZ            * RETURN IF PRINTER IS IDLE                       *
E6 80                           ANI     /80   * SELECT BIT                                      *
C8                              RZ            * RETURN IF PRINTER IS OFF                        *
C3 98 01                        JMP     WPR   * REPEAT                                          *
                        *  ************************************************************
                        *                                                                      *
0A                      MSG1    DC      /0A   * ASCII: LF                                       *
43 50 55 20 4F                  DC      C'CPU OK'                                               *
4B                                                                                              *
0A                              DC      /0A   * ASCII: LF                                       *
56 49 44 20 4F                  DC      C'VID OK'                                               *
4B                                                                                              *
03                              DC      /03   * ASCII: ETX                                      *
                        *                                                                       *
0A                      MSG2    DC      /0A   * ASCII: LF                                       *
52 41 4D 20                     DC      C'RAM '                                                 *
03                              DC      /03   * ASCII: ETX                                      *
                        *                                                                       *
4F 4B                   MSG3    DC      C'OK'                                                   *
0A                              DC      /0A   * ASCII: LF                                       *
52 4F 4D 20                     DC      C'ROM '                                                 *
03                              DC      /03   * ASCII: ETX                                      *
                        *                                                                       *
                        ********************************************************** **********
```

```
                ********************************************************************************
                *                                                                              *
                *                              VIDEO                                           *
                *                                                                              *
                ********************************************************************************
                *                                                                              *
                ********************************************************************************
                *                                                                              *
                *                              PART1: VISUAL TEST                              *
                *                                                                              *
                ********************************************************************************
                *                                                                              *
06 00           VTEST MVIB     0       * SET FIRST VISUAL PATTERN                            *
21 00 10        NPAT  LXIH     /1000   * POINTER TO FIRST VIDEO POSITION                     *
70              AGAIN MOVMB            * FILL VIDEO                                          *
23                    INXH             *  MEMORY                                             *
7C                    MOVAH            *   WITH                                              *
FE 80                 CPI      /80     *    CURRENT                                          *
FA C4 01              JM       AGAIN   *     PATTERN                                         *
04                    INRB             * CHANGE PATTERN                                      *
C2 C1 01              JNZ      NPAT    *  AND REPEAT FOR ALL OF THEM                         *
                *                                                                              *
                ********************************************************************************
```

```
P1 P2 P3 P4 P5

                        ********************************************************************************
                        *                                                                              *
                        *                          PART2: STORAGE TEST                                 *
                        *                                                                              *
                        ********************************************************************************
                        *                                                                              *
01 00 55                        LXIB    /5500 * INITIAL TEST PATTERN AND ERROR MASK                  *
21 FF 1F                MARCH   LXIH    /1FFF * POINTER TO LAST VIDEO POSITION
70                      FILL    MOVMB         * MOVE PATTERN TO VIDEO MEMORY                         *
2B                              DCXH          * UPDATE POINTER                                       *
7C                              MOVAH         * REPEAT                                               *
FE 0F                           CPI     /0F   *  UNTIL FIRST                                         *
C2 D6 01                        JNZ     FILL  *   VIDEO POSITION                                     *
                        *                                                                            *
23                              INXH          * SET POINTER TO FIRST VIDEO POSITION                  *
78                      CHKFW   MOVAB         * VERIFY IF                                            *
AE                              XRAM          *  MEMORY CONTENT AND PATTERN                          *
CA E6 01                        JZ      FWOK  *   ARE THE SAME                                       *
B1                              ORAC          * ON MISMATCH,                                         *
4F                              MOVCA         *  SET ERROR MASK                                      *
78                      FWOK    MOVAB         * INVERT PATTERN                                       *
2F                              CMA           *  AND STORE AT                                        *
77                              MOVMA         *   THE SAME ADDRESS                                   *
23                              INXH          * UPDATE POINTER                                       *
7C                              MOVAH         * REPEAT FORWARD                                       *
FE 20                           CPI     /20   *  CHECK FOR THE                                       *
C2 DF 01                        JNZ     CHKFW *   WHOLE MEMORY                                      *
                        *                                                                            *
```

```
P1 P2 P3 P4 P5

2B                      DCXH            * SET POINTER TO LAST VIDEO POSITION                *
78                CHKBW MOVAB           * VERIFY IF                                         *
2F                      CMA             *  MEMORY CONTENT                                   *
AE                      XRAH            *   AND PATTERN                                     *
CA F9 01                JZ      BWOK    *    ARE INVERTED                                   *
B1                      ORAC            * ON MISMATCH,                                      *
4F                      MOVCA           *  SET ERROR MASK                                   *
70                BWOK  MOVMB           * RESTORE TEST PATTERN                              *
2B                      DCXH            * UPDATE POINTER                                    *
7C                      MOVAH           * REPEAT BACKWARD                                   *
FE 0F                   CPI     /0F     *  CHECK FOR THE                                    *
C2 F1 01                JNZ     CHKBW   *   WHOLE MEMORY                                    *
78                      MOVAB           * INVERT                                            *
2F                      CMA             *  TEST                                             *
47                      MOVBA           *   PATTERN                                         *
B7                      ORAA            *    AND REPEAT                                     *
FA D3 01                JM      MARCH   *     EVERYTHING                                    *
                *                                                                           *
79                      MOVAC           * READ ERROR MASK                                   *
B7                      ORAA            *  AND STOP                                         *
C2 2A 00                JNZ     ERROR   *   ON ERROR                                        *
                *                                                                           *
31 F0 1F                LXISP   /1FF0 * INITIALIZE STACK POINTER                            *
21 00 10                LXIH    /1000 * START OF VISIBLE AREA                               *
CD B1 00                CALL    CLEAR * CLEAR DISPLAY                                       *
11 A2 01                LXID    MSG1  * 'CPU OK       VID OK'                               *
CD B1 00                CALL    MSG   * DISPLAY MSG1                                        *
                *                                                                           *
*****************************************************************************************
```

```
                ********************************************************************************
                *                                                                              *
                *                              MAIN MEMORY                                     *
                *                                                                              *
                ********************************************************************************
                *                                                                              *
                ********************************************************************************
                *                                                                              *
                *                        PART1: CONFIGURATION AND K-MASK TEST                  *
                *                                                                              *
                ********************************************************************************
                *                                                                              *
AF                      XRAA            * INITIALIZE                                         *
32 FD 1F                STA     REFLG   *  RAM ERROR FLAG AND                                 *
CD F2 00                CALL    INIT    *    RAM ERROR TABLE                                 *
21 00 40                LXIH    /4000   * START OF RAM AREA                                  *
45                      MOVBL           * FLAG FOR FIRST ADDRESS                             *
7D                      MOVAL           * INITIAL VALUE FOR                                  *
32 FF 1F                STA     FMADR   *  FIRST MEMORY ADDRESS                              *
36 FF           CONF    MVIM    /FF     * CHECK FOR                                          *
7E                      MOVAM           *   EMPTY                                            *
B7                      ORAA            *    MEMORY                                          *
CA 52 02                JZ      EMPTY   *     ADDRESS                                        *
3E F0                   MVIA    /F0     * FIRST MASK                                         *
77                      MOVMA           * MOVE TO MEMORY                                     *
AE                      XRAM            * COMPARE MEMORY                                     *
C4 FF 00                CNZ     RAMER   *  AND MASK                                          *
3E CC                   MVIA    /CC     * SECOND MASK                                        *
77                      MOVMA           * MOVE TO MEMORY                                     *
AE                      XRAM            * COMPARE MEMORY                                     *
C4 FF 00                CNZ     RAMER   *  AND MASK                                          *
78                      MOVAB           * CHECK                                              *
B7                      ORAA            *  FLAG                                              *
C2 4B 02                JNZ     LBK     *   AND STORE                                        *
06 01                   MVIB    1       *    FIRST                                           *
7C                      MOVAH           *     RAM                                           *
32 FF 1F                STA     FMADR   *      ADDRESS                                       *
```

```
P1 P2 P3 P4 P5

7C                      LBK     MOVAH           * STORE LAST BLOCK                                              *
32 FE 1F                        STA     LMADR   *  INITIAL ADDRESS                                               *
C3 97 02                        JMP     NXBL    *                                                               *
78                      EMPTY   MOVAR           * CHECK                                                         *
B7                              ORAA            *  FOR                                                          *
C2 9E 02                        JNZ     ENDM    *   END OF MEMORY                                               *
7C                      NXBL    MOVAH           * UPDATE POINTER                                                *
C6 10                           ADI     /10     *  TO NEXT 4K-BLOCK                                             *
67                              MOVHA           *   AND CHECK                                                   *
C2 2B 02                        JNZ     CONF    *    FOR LAST ONE                                               *
11 B1 01                ENDM    LXID    MSG2    * 'RAM '                                                        *
CD B1 00                        CALL    MSG     * DISPLAY MSG2                                                  *
3A FF 1F                        LDA     FMADR   * GET FIRST MEMORY ADDRESS                                      *
CD E7 00                        CALL    ASCX    * CONVERT AND DISPLAY IT                                        *
B7                              ORAA            * FMADR=0 IF
CA FF 02                        JZ      RAMOK   *  MEMORY IS NOT PRESENT                                       *
AF                              XRAA            * (A)=00                                                        *
CD E7 00                        CALL    ASCX    * CONVERT AND DISPLAY IT                                        *
36 2D                           MVIM    /2D     * ASCII '-'                                                     *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION                         *
3A FE 1F                        LDA     LMADR   * GET LAST BLOCK INITIAL ADDRESS                                *
C6 10                           ADI     /10     * EVALUATE AND STORE                                            *
32 FE 1F                        STA     LMADR   *  LAST MEMORY ADDRESS+1                                        *
3D                              DCRA            * FIND LAST ADDRESS                                             *
CD E7 00                        CALL    ASCX    * CONVERT AND DISPLAY IT                                        *
3E FF                           MVIA    /FF     * (A)=/FF                                                       *
CD E7 00                        CALL    ASCX    * CONVERT AND DISPLAY IT                                        *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION                         *
                        *                                                                                       *
                        ********************************************************************************
```

```
P1 P2 P3 P4 P5

                        ********************************************************************************
                        *                                                                              *
                        *                    PART2: MARCHING 1'S AND 0'S TEST                          *
                        *                                                                              *
                        ********************************************************************************
                        *                                                                              *
E5                              PUSHH           * SAVE DISPLAY POINTER                                 *
06 55                           MVIB    /55     * INITIAL TEST PATTERN
CD 31 01                STEST   CALL    SETMA   * SET INITIAL ADDRESS FOR MEMORY TEST                  *
70                      MFILL   MOVMB           * MOVE PATTERN TO MEMORY                               *
2B                              DCXH            * UPDATE POINTER                                       *
3A FF 1F                        LDA     FMADR   * REPEAT                                               *
3D                              DCRA            *  UNTIL                                               *
BC                              CMPH            *   FIRST                                              *
C2 8D 02                        JNZ     MFILL   *    MEMORY POSITION                                   *
                        *                                                                              *
23                              INXH            * SET POINTER TO FIRST MEMORY POSITION                 *
78                      MCHKF   MOVAB           * VERIFY IF                                            *
AE                              XRAM            *  MEMORY CONTENT AND PATTERN                          *
C4 FF 00                        CNZ     RAMER   *   ARE THE SAME                                       *
78                              MOVAB           * INVERT PATTERN                                       *
2F                              CMA             *  AND STORE AT                                        *
77                              MOVMA           *   THE SAME ADDRESS                                   *
23                              INXH            * UPDATE POINTER                                       *
3A FE 1F                        LDA     LMADR   * REPEAT FORWARD                                       *
BC                              CMPH            *  CHECK FOR                                           *
C2 98 02                        JNZ     MCHKF   *   THE WHOLE MEMORY                                   *
                        *                                                                              *

                                                                                                -99-
```

```
p1 p2 p3 p4 p5

2B                          DCXH          * SET POINTER TO LAST MEMORY POSITION                *
78                    MCHKB MOVAB         * VERIFY IF                                          *
2F                          CMA           *  MEMORY CONTENT                                    *
AE                          XRAM          *   AND PATTERN                                      *
C4 FF 00                    CNZ     RAMER *    ARE INVERTED                                     *
70                          MOVMA         * RESTORE TEST PATTERN                               *
2B                          DCXH          * UPDATE POINTER                                     *
3A FF 1F                    LDA     FMADR * REPEAT BACKWARD                                    *
3D                          DCRA          *  CHECK FOR                                         *
BC                          CMPH          *   THE WHOLE                                        *
C2 A9 02                    JNZ     MCHKB *    MEMORY                                          *
78                          MOVAB         * INVERT                                             *
2F                          CMA           *  TEST                                              *
47                          MOVBA         *   PATTERN                                          *
B7                          ORAA          *    AND REPEAT                                      *
FA 8A 02                    JM      STEST *     EVERYTHING                                     *
                      *                   ********************************************************
```

```
P1 P2 P3 P4 P5

                                      *************************************************************************
                                      *                                                                       *
                                      *                      PART5: PSEUDO-RANDON DATA AND REFRESH TEST       *
                                      *                                                                       *
                                      *************************************************************************
                                      *                                                                       *
CD 31 01                                    CALL   SETMA * SET INITIAL ADDRESS FOR MEMORY TEST                *
CD 3A 01                              RANDG CALL   RAND  * RANDON DATA GENERATION                             *
72                                          MOVMD        * MOVE DATA TO MEMORY                                *
23                                          INXH         * UPDATE POINTER                                     *
3A FE 1F                                    LDA    LMADR * REPEAT                                             *
BC                                          CMPH         *  UNTIL LAST                                        *
C2 C3 02                                    JNZ    RANDG *   MEMORY POSITION                                  *
                                      *                  **************************************************
01 20 00                                    LXIB   32    *    (B)(C)=32                                       *
60                                          MOVHB        *    (H)=0                                           *
68                                          MOVLB        *    (L)=0                                           *
E3                                    SRFF  XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
E3                                          XTHL         *    SLOW DOWN REFRESH RATE                          *
09                                          DADB         *    (H)(L)=(H)(L)+32                                *
D2 D4 02                                    JNC    SRFF  *    LOOP BACK WHILE C BIT=0                         *
                                      *                  **************************************************
```

```
P1 P2 P3 P4 P5

                *                       ************************************************************
                        CALL    SETMA   * SET INITIAL ADDRESS FOR MEMORY TEST                        *
CD 31 01        CRAND   CALL    RAND    * RANDOM DATA GENERATION (SAME SEQUENCE)                    *
CD 3A 01                MOVAD           * CHECK IF                                                  *
7A                      XRAM            *  SEQUENCE AND MEMORY DATA                                 *
AE                      CZ      RAMER   *   ARE THE SAME                                           *
CC FF 00                INXH            * UPDATE POINTER                                          *
23                      LDA     LHADR   * REPEAT                                                  *
3A FE 1F                CMPH            *  UNTIL LAST                                             *
BC                      JNZ     CRAND   *   MEMORY POSITION                                       *
C2 E7 02                POPH            * RESTORE DISPLAY POINTER                                 *
E1                      LDA     REFLG   * GET RAM ERROR FLAG                                      *
3A FD 1F                ORAA            * IF ANY ERRORS WERE DETECTED,                            *
B7                      CNZ     SWERR   *  SHOW ERROR REPORT AND STOP                            *
C4 17 01        RAMOK   LXID    MSG3    * 'OK           ROM'                                      *
11 B7 01                CALL    MSG     * DISPLAY MSG3                                            *
CD B1 00        *                       ************************************************************
```

```
P1 P2 P3 P4 P5

                        ********************************************************************************
                        *                                                                              *
                        *                              ROM                                             *
                        *                                                                              *
                        ********************************************************************************
                        *                                                                              *
E5                      PUSHH           * SAVE DISPLAY POINTER                                       *
21 00 00                LXIH    0       * ROM INITIAL ADDRESS                                        *
11 00 00        STSIG   LXID    0       * INITIAL SIGNATURE                                          *
7E              SIG     MOVAM           * GET BYTE                                                   *
E5                      PUSHH           * SAVE ROM POINTER                                           *
67                      MOVHA           * SAVE BYTE                                                  *
06 08                   MVIB    8       * BIT COUNTER                                                *
0E 00           XBIT    MVIC    0       * INITIAL VALUE FOR ONES' COUNTER                            *
7C                      MOVAH           * EVALUATE                                                   *
E6 01                   ANI     1       *  MODULO 2                                                  *
CD 3F 01                CALL    ONES    *   SUM                                                      *
7A                      MOVAD           *    OF INCOMING                                             *
E6 80                   ANI     /80     *     BIT AND                                                *
CD 3F 01                CALL    ONES    *      SIGNATURE                                             *
7B                      MOVAE           *       BITS                                                 *
E6 40                   ANI     /40     *        15,11,                                              *
CD 3F 01                CALL    ONES    *         8 AND 6,                                           *
                        ********************************************************************************
79                      MOVAC           * SHIFT                                                      *
1F                      RAR             *  SIGNATURE                                                 *
7B                      MOVAE           *   LEFT,                                                    *
17                      RAL             *    TAKING                                                  *
5F                      MOVEA           *     PREVIOUS                                               *
7A                      MOVAD           *      SUM                                                   *
17                      RAL             *       AS FIRST                                             *
57                      MOVDA           *        BIT                                                 *
                        ********************************************************************************
```

```
P1 P2 P3 P4 P5

                        *                       ***********************************************************
7C                              MOVAH           * PREPARE                                                 *
1F                              RAR             *  NEXT                                                   *
67                              MOVHA           *   INCOMING BIT                                          *
05                              DCRB            * REPEAT                                                  *
C2 11 03                        JNZ     XBIT    *  8 TIMES                                                *
E1                              POPH            * RESTORE ROM POINTER                                     *
7C                              MOVAH           * CHECK                                                   *
E6 02                           ANI     /02     *  FOR                                                    *
23                              INXH            *   LAST                                                  *
CA 0C 03                        JZ      SIG     *    ADDRESS                                              *
7C                              MOVAH           *     ON                                                 *
E6 02                           ANI     /02     *      CURRENT                                            *
C2 0C 03                        JNZ     SIG     *       ROM                                               *
                        *                       ***********************************************************
E3                              XTHL            * GET DISPLAY POINTER                                     *
7A                              MOVAD           * DISPLAY                                                 *
CD E7 00                        CALL    ASCX    *  SIGNATURE                                              *
7B                              MOVAE           *   AS FOUR                                               *
CD E7 00                        CALL    ASCX    *    HEXADECIMAL DIGITS                                   *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION                   *
E3                              XTHL            * RESTORE ROM POINTER                                     *
                        *                       ***********************************************************
7C                              MOVAH           * CHECK                                                   *
E6 10                           ANI     /10     *  FOR                                                    *
CA 09 03                        JZ      STSIG   *   LAST ROM                                              *
CD 93 00                        CALL    ROLL    * ROLL DISPLAY                                            *
                        *                       *                                                         *
                        ***********************************************************************************
```

```
                ********************************************************************************
                *                                                                              *
                *                                KEYBOARD                                      *
                *                                                                              *
                ********************************************************************************
                *                                                                              *
D3 11           KB      OUT     /11     * KEYBOARD INTERRUPT ENABLE                          *
21 30 17                LXIH    /1730   * BEGIN OF LAST VISIBLE VIDEO LINE                   *
36 4B                   MVIM    /4B     * ASCII: 'K'                                         *
23                      INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
36 42                   MVIM    /42     * ASCII: 'B'                                         *
23                      INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
23                      INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
DB 14                   IN      /14     * KEYBOARD STATUS REGISTER                           *
CD E7 00                CALL    ASCX    * DISPLAY IT                                         *@
23                      INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
E6 40                   ANI     /40     * BRANCH TO PRINTER TEST                             *
CA 78 03                JZ      PRINT   *  IF NO DATA IS AVAILABLE                           *
FB                      EI              * SYSTEM INTERRUPT ENABLE                            *
DB 10                   IN      /10     * GET KEYBOARD CHARACTER                             *
F3                      DI              * SYSTEM INTERRUPT DISABLE                           *
23                      INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
36 3D                   MVIM    /3D     * ASCII: '='                                         *
23                      INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
CD E7 00                CALL    ASCX    * DISPLAY IT                                         *@
                *                                                                              *
                ********************************************************************************
```

```
                        ********************************************************************************
                        *                                                                              *
                        *                              PRINTER                                         *
                        *                                                                              *
                        ********************************************************************************
                        *                                                                              *
CD 86 01                PRINT   CALL    DPRS    * SHOW PRINTER STATUS                                  *
C2 A7 03                        JNZ     DISK    * GO TO NEXT TEST IF PRINTER IS INATIVE                *
01 21 21                        LXIB    /2121   * INICIAL PATTERNS FOR PRINTER                         *
16 84                   NLINE   MVID    132     * D REGISTER IS CHARACTER COUNTER                      *
78                      SLINE   MOVAB           * GET CURRENT CHARACTER                                *
3D                              DCRA            *  DECREMENT ONE TO CORRECT RANGE 20-7E                *
D3 91                           OUT     /91     *   SEND IT TO PRINTER BUFFER                          *
CD 98 01                        CALL    WPR     * WAIT                                                 *
04                              INRB            * UPDATE CURRENT CHARACTER                             *
F2 90 03                        JP      NXCH    * CORRECT RANGE                                        *
06 21                           MVIB    /21     *  TO 21-7F                                            *
15                      NXCH    DCRD            * UPDATE CHARACTER COUNTER                             *
C2 83 03                        JNZ     SLINE   * SEND ANOTHER IF LINE IS NOT FULL                     *
3E 0D                           MVIA    /0D     * ASCII: CR                                            *
D3 91                           OUT     /91     * PRINT BUFFER                                         *
CD 98 01                        CALL    WPR     * WAIT                                                 *
3E 0A                           MVIA    /0A     * ASCII: LF                                            *
D3 91                           OUT     /91     * FEED A NEW LINE                                      *
CD 98 01                        CALL    WPR     * WAIT                                                 *
0C                              INRC            * CHANGE INITIAL CHARACTER FOR NEXT LINE               *
41                              MOVBC           * INITIALIZE CURRENT CHARACTER                         *
F2 81 03                        JP      NLINE   * REPEAT FOR ALL CHARACTERS                            *
                        *                                                                              *
                        ********************************************************************************
```

```
                        ********************************************************************************
                        *                                                                              *
                        *                              DISK                                            *
                        *                                                                              *
                        ********************************************************************************
                        *                                                                              *
06 00                   DISK    MVTB    0       * DRIVE 0, SECTOR 0, UPPER SIDE                      *
21 62 17                NXDR    LXIH    /1762   * 50TH POSITION AT LAST VIDEO LINE                   *
36 44                           MVIM    /44     * ASCII: 'D'                                         *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
36 4B                           MVIM    /4B     * ASCII: 'K'                                         *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
78                              MOVAB           * SELECT                                             *
D3 46                           OUT     DKSR    *  DRIVE                                             *
0F                              RRC             * DISPLAY                                            *
CD D6 00                        CALL    ASCXH   *  DRIVE NUMBER                                      *
36 3A                           MVIM    /3A     * ASCII: ':'                                         *
23                              INXH            * MOVE DISPLAY POINTER TO NEXT POSITION              *
DB 40                           IN      DKST    * READ STATUS                                        *
CD E7 00                        CALL    ASCX    *  AND DISPLAY IT                                    *
E6 01                           ANI     1       * ON FILE READY,                                     *
C2 D1 03                        JNZ     DKTST   *  START DISK TEST                                   *
78                              MOVAB           * OTHERWISE                                          *
C6 20                           ADI     /20     *  TRY                                               *
47                              MOVBA           *   NEXT                                             *
C2 A9 03                        JNZ     NXDR    *    DRIVE                                           *
C3 58 03                        JMP     KB      * ON LAST DRIVE, RETURN TO KEYBOARD TEST             *
                        *                                                                              *
```

```
P1 P2 P3 P4 P5

3E 02          DKTST  MVIA    2       * RESET                                *
D3 47                 OUT     DKCR    *  COMMAND                             *
CD 40 01              CALL    WTDSK   * WAIT END OF OPERATION                *
3E 01                 MVIA    1       * WRITE PROTECT                        *
D3 47                 OUT     DKCR    *  COMMAND                             *
DB 40                 IN      DKST    * READ STATUS                          *
E6 20                 ANI     /20     * FILE MUST                            *
CA 2A 00              JZ      ERROR   *  BE PROTECTED NOW                    *
                *                                                            *
1E 04                 MVIC    4       * READ COMMAND CODE                    *
01 CA 00              LXIB    /00CA   * FIRST AND LAST CYLINDER              *
50             POSIT  MOVDB           * DESTINATION 1                        *
CD 55 01              CALL    SKEX    * SEEK AND READ                        *
04                    INRB           * FIND NEXT DESTINATION 1              *
51                    MOVDC          * DESTINATION 2                        *
CD 55 01              CALL    SKEX    * SEEK AND READ                        *
0D                    DCRC           * FIND NEXT DESTINATION 2              *
C2 F8 03              JNZ     POSIT   * TEST FOR END OF DISK                 *
1C                    INRE           * WRITE CHECK CODE                     *
CD 55 01       WRCK   CALL    SKEX    * SEEK AND WRITE CHECK                 *
0D                    DCRC           * REPEAT                               *
C2 F6 03              JNZ     WRCK    *  256 TIMES                           *
C3 58 03              JMP     KB      * RESTART PERIPHERALS TESTS            *
                *                                                            *
*************************************************************************
```

P1 P2 P3 P4 P5

```
*************************************************************************
*                                                                       *
DKST   EQU      /40    * DISK STATUS REGISTER                           *
DKHA   EQU      /41    * MEMORY ADDRESS REGISTER,HIGH BYTE              *
DKLA   EQU      /42    * MEMORY ADDRESS REGISTER,LOW BYTE               *
DKHB   EQU      /43    * BYTE COUNT REGISTER,HIGH BYTE                  *
DKLB   EQU      /44    * BYTE COUNT REGISTER,LOW BYTE                   *
DKCY   EQU      /45    * CYLINDER ADDRESS REGISTER                      *
DKSR   EQU      /46    * SECTOR REGISTER                                *
DKCR   EQU      /47    * CONTROL REGISTER                               *
FMADR  EQU      /1FFF  * FIRST MEMORY ADDRESS                          *
LMADR  EQU      /1FFE  * LAST MEMORY ADDRESS                           *
REFLG  EQU      /1FFD  * RAM ERROR FLAG                                *
ERTAB  EQU      /1FF0  * MEMORY ERROR TABLE                            *
*                                                                       *
*************************************************************************
*                                                                       *
       END
```

UTILIZADA 1024 BYTES   MONTAGEM BEM SUCEDIDA   0 ADVERTENCIA(S)   0 ERROS(S)

OPCOES EM EFEITO NXREF LIST NTSIM

EXECUCAO DO PASS01/PASS02 59,876/ 47,489 SEG