SINCRONIZADOR PARA MULTIMICROPROCESSADOR

POR

IVAN SEBASTIÃO DE SOUZA E SILVA

TESE DE MESTRADO

Apresentada à Coordenação Setorial de Pós-Graduação e Pesquisa da Pró-Reitoria para Assuntos do Interior da Universidade Federal da Paraíba, em cumprimento às exigências para obtenção do grau de Mestre em Ciências.

CAMPINA GRANDE, SETEMBRO DE 1981

S586s Silva, Ivan Sebastião de Souza e.
Sincronizador para Multimicroprocessador / Ivan Sebastião de Souza e Silva. - Campina Grande, 1981.
50 f.

Dissertação (Mestrado em Ciências) - Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, 1981.
"Orientação : Prof. Dr. Gurdip Singh Deep, Prof. M.Sc. José Homero Feitosa Cavalcanti".
Referências.

1. Sincronizador. 2. Multimicroprocesador. 3. Controle de Processos - Computadores. 4. Dissertação - Ciências. I. Deep, Gurdip Singh. II. Cavalcanti, José Homero Feitosa. III. Universidade Federal da Paraíba - Campina Grande (PB). IV. Título

CDU 621.316.729(043)

CPgEE/CCT-UFP
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
CENTRO DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE FEDERAL DA PARAÍBA

PARECER FINAL DO JULGAMENTO DA DISSERTAÇÃO DO MESTRANDO

IVAN SEBASTIÃO DE SOUSA E SILVA

TITULO: "SINCRONIZADOR PARA MULTIMICROPROCESSADOR"

CONCEITO: Aprovado com distinção

COMISSÃO EXAMINADORA:

PROF. GURDIP SINGH DEEP - Ph.D

PROF. JOSE HOMERO F. CAVALCANTI - M.Sc

PROF. JOSE IVAN CARNAUBA ACCIOLY - M.Sc

PROF. JOBERTO SERGIO B. MARTINS - M.Sc

Campina Grande, 4 de setembro de 1981

I N D I C E

R E S U M O

Um sistema multimicroprocessador possui dois ou mais microcomputadores que compartilham recursos comuns de HARDWARE e SOFTWARE(memória, dispositivos de entrada e saída de dados, etc.). Este trabalho apresenta um estudo de algumas técnicas de controle de acesso dos microcomputadores aos recursos compartilhados. É desenvolvido um microcomputador sincronizador para efetuar o controle do acesso aos recursos compartilhados em um sistema multimicroprocessador que possua até 8(oito) microcomputadores baseados no M6800 da MOTOROLA, e permitir interação com o operador. Um pequeno sistema operacional é implementado para permitir operação em tempo real.

A B S T R A C T

A multimicroprocessor system consist of two or more microcomputers sharing certain common HARDWARE and SOFTWARE resources(e.g. memory, I/O devices, etc.). In this thesis a study of some of the control techniques used for the microcomputers to share the common resources, is attempted. A microprocessor - based synchroniser for controlling of resource sharing by different microcomputers of the multimicroprocessor system is developed. The multimicroprocessor system still under development may contain up to eight microprocessors(M6800). The prototype synchroniser also permits interation with the operator via Video terminal- A small operating system also has been developed to permit real time control applications.

D E D I C A T Ō R I A

À minha esposa **MIRTES**

Aos meus filhos **IVANIZE** e **CLÁUDIO RUI**

À minha mãe **AURELINA**

E em memória do meu pai **RUI**.

A G R A D E C I M E N T O S

Aos Professores G. S. Deep e J. Homero Feitosa Cavalcanti pela valiosa orientação, e a todos que contribuiram na elaboração deste trabalho.

CAPÍTULO I

INTRODUÇÃO

A revolução da Integração em Larga Escala(LSI), e em particular do microprocessador, provocaram um grande impacto nos projetos de sistemas eletrônicos para aplicação na área de controle e outros ramos da engenharia. A proliferação e o baixo custo de produtos como calculadoras, relógios digitais, etc., foi apenas o início e é claro que nunca teria ocorrido sem o rápido aperfeiçoamento da tecnologia de fabricação de dispositivos a semicondutor(01). Novos produtos estão sendo fabricados com uma maior complexidade e com um menor preço(02), permitindo a utilização desses dispositivos em diversas aplicações.

Devido ao baixo custo dos microprocessadores hoje, eles têm sido amplamente utilizados em processamento de dados como unidades dedicadas a uma determinada aplicação, por exemplo, jogos, controlador de temperatura, controle de processos

de um modo geral, processamento de dados convencional, etc.

1.1. Controle de Processos Usando Computadores

Computadores têm sido largamente utilizados em controle de processos de várias maneiras em diferentes aplicações(03).Uma das aplicações é o "DATA LOGGING", no qual os dados do processo sob controle são coletados, analisados, tabulados e colocados à vista do operador por uma requisição deste ou a intervalos regulares de tempo. Os dados resultantes do processamento são utilizados pelo operador para controlar o processo diretamente. Neste caso, o computador desempenha funções como conversão de medidas para unidades de engenharia, verificação das condições de alarme, armazenamentos de dados para posterior verificação,etc.(03).

A outra aplicação dos computadores é o controle supervisório, no qual os dados depois de obtidos e processados, são utilizados para execução e cálculos que fornecem os dados apropriados para os controladores analógicos desempenharem o controle do processo. Por exemplo, um controlador analógico deve controlar um fluxo de algum fluido operando uma válvula. O valor do fluxo deve ser medido e comparado com um valor de referência fornecido pelo computador. A válvula é então manipulada pelo controlador analógico para manter o fluxo no valor especificado. O operador pode modificar os parâmetros de processamento do computador se um novo valor de referência é desejada(03).

A terceira aplicação do computador é o Controle Digital Direto(Direct Digital Control - DDC). Neste caso, o computa-

dor obtém dados do processo e os usa para executar programas equivalentes às funções dos controladores analógicos. O computador, via interfaces especiais, ajusta o atuador final(válvula, chave, etc.) para efetivar o controle apropriado. Os dados de referência desse controle podem ser fornecidos pelo operador ou pelo próprio computador desempenhando cálculos estatísticos, comparações, etc. O computador permite uma ampla seleção de novos algoritmos de controle mais sofisticados que podem ser utilizados em técnicas de controle mais avançadas. O principal problema do DDC ocorre quando um computador é utilizado para controlar vários processos, uma falha neste computador resulta na perda do controle de todos os processos. Por esta razão DDC não tem sido muito usado sem controle analógico associado, para operar quando ocorrer uma falha no computador(03).

Os computadores usados em controle de processos em tempo real necessitam de grande velocidade de operação, para além de fazer o controle do processo, fazer a comunicação com operadores, outros sistemas, etc.(04). Uma solução para aumentar a perfomance de um computador em aplicações de tempo real é utilizar vários processadores interligados num mesmo sistema(04).

Existem várias maneiras diferentes de se ligar computadores para operarem em tempo real(05). Uma das maneiras é o NETWORK, neste tipo de ligação(fig.01) cada computador possui seu processador central e sua própria memória particular. Neste caso a comunicação entre processadores é feita diretamente através de canais de comunicação em série ou em paralelo(05).

Um exemplo de um sistema com esse tipo de ligação é o

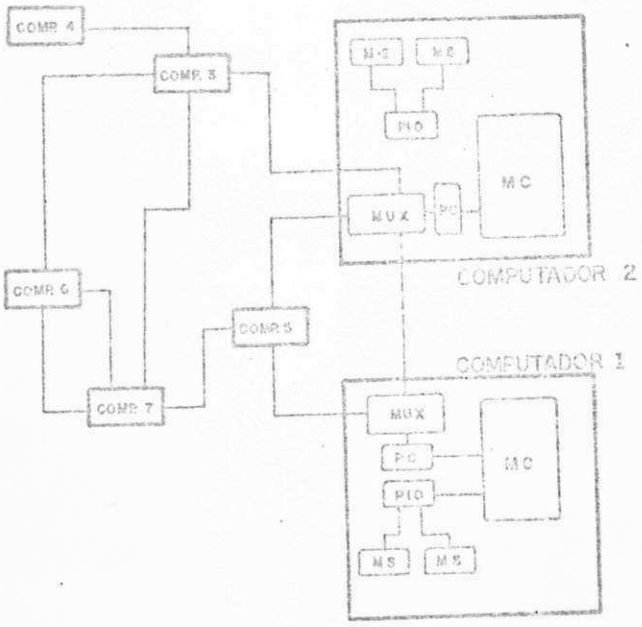
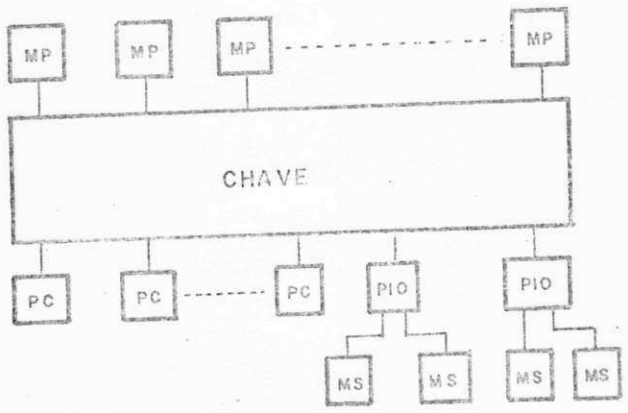


Fig. 1 - Uma estrutura NETWORK



LEGENDA:

PC – Processador Central
PIO – Processador de I/O
MP – Memória Primária
MS – Memória Secundária

Fig. 2 - Uma estrutura multiprocessador

CAMAC(Computer Automated Monitoring and Control) para controle de plantas industriais. O CAMAC é composto de um conjunto de módulos independentes(microcomputadores, interfaces de entrada/saída, etc.) que podem ser interligados em série ou em paralelo sob o controle de um computador central(06).

Uma outra maneira é o MULTIPROCESSADOR, a fig. 02 mostra a estrutura básica de um multiprocessador. Sua principal característica é que a comunicação entre os processadores é feita através de uma memória compartilhada por todos os processadores, isto é, a mensagem a ser transmitida de um computador a outro é sempre armazenada na memória compartilhada(05). Como exemplo de sistemas que utilizam esse tipo de ligação pode ser citado o BURROUGHS D825, o BENDIX G-21 e o IBM 360/65 entre outros (05). Um sistema multiprocessador é melhor definido como um sistema com as seguintes características(07):

a) Um multiprocessador contém dois ou mais processadores de capacidades aproximadamente comparáveis.

b) Todos os processadores compartilham uma memória comum.

c) Todos os processadores compartilham acesso a dispositivos de entrada e saída de dados e unidades de controle.

d) O sistema total é controlado por um sistema operacional que permite interação entre processadores e seus programas.

1.2. Microprocessadores em Controle de Processos

Devido ao relativo baixo custo dos microprocessadores, eles têm sido amplamente utilizados em processamento de dados como unidades dedicadas a uma única aplicação e em particular em controle de processos. A grande vantagem de sistemas que utilizam microprocessadores, além do seu baixo custo, é a modularidade e facilidade de desenvolvimento de sistemas específicos para uma determinada aplicação(05).

Uma desvantagem dos microprocessadores é a sua velocidade relativamente baixa(uma instrução e executada em torno de um microsegundo)(08). Para aplicações em tempo real, um sistema de microprocessador pode ser muito lento(08). Além disso ocorre uma redução no desempenho(aumento no tempo de resposta) de um microcomputador(sistema que utiliza microprocessador) que além de controle de processos externos, faça a gerência de entrada e saída de dados(09).

1.3. Análise de Sistemas Multimicroprocessador

Em um sistema multimicroprocessador deve existir uma VIA(BUS) comum a todos os microprocessadores, de modo a permitir o acesso de cada um desses microprocessadores aos recursos compartilhados(memória, dispositivos de entrada e saída de dados, etc.). O principal problema nesses sistemas é o controle dessa VIA, de modo a não permitir que mais de um microprocessador tenha acesso a ela simultaneamente, assim como evitar que um micropro-

cessador que deseje acesso a essa VIA fique muito tempo desocupado aguardando para que possa utilizá-la(contenção no acesso a VIA). A seguir apresenta-se uma análise do controle dessa VIA utilizada em sistemas multimicroprocessador.

1.3.1. Operação Síncrona e Assíncrona da Via Compartilhada

Uma VIA síncrona, onde cada microprocessador tem acesso a intervalos regulares de tempo, necessita de um circuito de clock centralizado que fornece pulsos de sincronização a cada um dos microprocessadores do sistema. Isso permite o uso de uma única VIA, pelos microprocessadores do sistema, funcionando como VIAS virtuais; desde que o período de cada clock seja convenientemente dividido de modo a não permitir que mais de um microprocessador use a VIA ao mesmo tempo(08). Por exemplo, dois microprocessadores 6800 da MOTOROLA podem facilmente ser conectados a uma mesma VIA, porque eles necessitam da VIA somente durante metade de um período. Assim, basta introduzir um defasamento de 180 graus entre os dois clocks(08).

Para um sistema de multimicroprocessadores com microprocessadores de diferentes famílias, uma solução síncrona não é a melhor, pois cada microprocessador pode ter uma freqüência de operação máxima diferente, e uma sincronização é somente possível se todos os microprocessadores operarem em uma mesma freqüência, isto é, a freqüência de operação dos microprocessadores mais velozes teria que ser reduzida à freqüência de operação do microprocessador mais lento(08). Isso reduz o desempenho

global do sistema, e é melhor escolher uma solução assíncrona, onde cada microprocessador possui seu clock independente(08).

Na operação assíncrona, cada microprocessador requisita a VIA compartilhada sempre que necessitar o uso dessa VIA. O principal problema de uma operação assíncrona é o de requisições simultâneas da VIA, levando a contenção(08). O controle dos microprocessadores no acesso a essa VIA pode ser centralizado ou descentralizado.

Muitos dos sistemas de multimicroprocessador existentes usam um sistema de controle da VIA centralizado(08). Neste controle, um dos microprocessadores é que controla essa VIA, permitindo que qualquer outro microprocessador utilize a VIA através de requisições de acesso. O microprocessador que controla a VIA(microprocessador mestre) pode resolver problemas de contenção no acesso a VIA, considerando um esquema de prioridades conveniente durante requisições simultâneas. Porém esse tipo de controle da VIA limita a flexibilidade do sistema, pois a inclusão de um novo microprocessador a um sistema já existente implica em pelo menos uma mudança no software do mestre(08).

O controle descentralizado da VIA é feito por todos os microprocessadores que constituem o sistema multimicroprocessador. Existem três soluções possíveis para este tipo de controle: Requisições Independentes, Codificação de Prioridades e Encadeamento(08).

A fig. 03 mostra um diagrama em blocos de um sistema multimicroprocessador que efetua controle de requisições inde -

pendentes da VIA. Por exemplo, se o microprocessador 1 deseja acesso a VIA, deverá verificar se as linhas REQUEST 2 e REQUEST 3 não estão ativas, quando isso ocorrer, ativará a linha REQUEST 1 para impedir que outro microprocessador use a VIA até que seja desocupada.

Codificação de prioridades é executada por cada microprocessador colocando seu código de prioridade em uma via de prioridades e verificando, nesta VIA, se pode utilizar a VIA compartilhada. A via de prioridades deve desempenhar uma lógica conveniente de modo a fornecer o controle da VIA compartilhada, em caso de requisições simultâneas, ao microprocessador de maior prioridade. Uma linha de controle "VIA OCUPADA", é suficiente para proibir outras requisições da VIA, enquanto um microprocessador estiver usando a VIA compartilhada. A fig. 04 mostra um esquema de um sistema utilizando esse tipo de controle.

Encadeamento oferece uma solução muito simples para resolver o problema de contenção no acesso a VIA(fig.05). O sinal PRIORIN atravessa cada microprocessador e é transmitido somente se ele não necessita da VIA. Uma linha PROCREQ indica que um microprocessador está requerendo a VIA. Um esquema de prioridades em círculo pode ser obtido se novas requisições são inibidas enquanto o sinal PRIORIN permanecer ativo(08). Se mais que um microprocessador tem sua linha de requisição ativa, eles serão atendidos um após o outro, de modo que o microprocessador que tiver esperado por maior tempo será atendido primeiro.

As maiores desvantagens do encadeamento são a prioridade dependente da posição dos microprocessadores e a transmissão

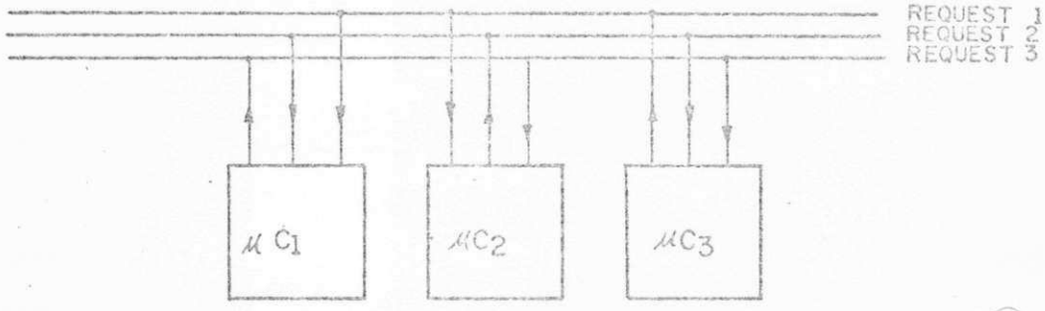


Fig. 3 - Requisições independentes

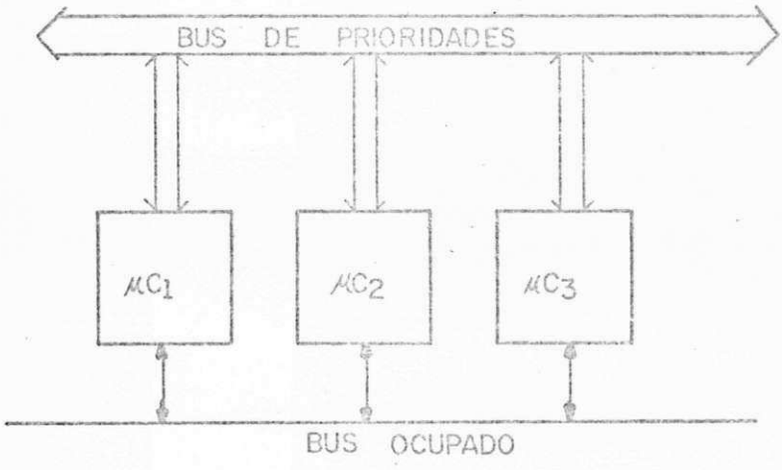


Fig. 4 - Codificação de prioridades

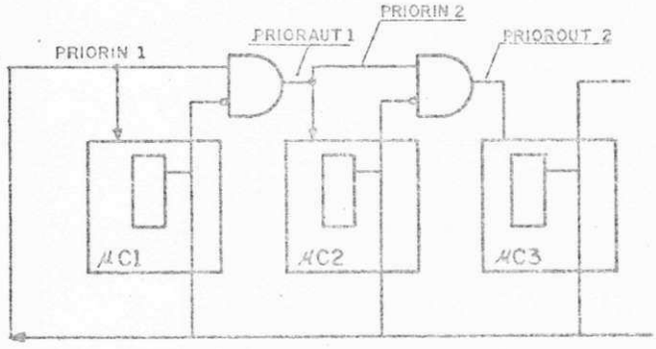


Fig. 5 - Encadeamento

de atrasos dentro da cadeia. Para um grande número de microprocessadores, este atraso pode tornar-se excessivo(08).

1.4. O MATER

O MATER(Multimicroprocessadores para Aplicações em Tempo Real) é um sistema multimicroprocessador para operar com oito microprocessadores, da família M6800 da MOTOROLA, desenvolvido para controle de processos em tempo real. O controle dos microprocessadores do MATER, no acesso aos recursos compartilhados, é assíncrono e centralizado em um microcomputador sincronizador.

A estrutura geral do MATER, como mostrado no diagrama em blocos da fig.06, permite interação entre os microcomputadores principais(µcp) pela memória compartilhada(MC). Esta comunicação deve ser sincronizada pelo microcomputador SINCRONIZADOR (µcs), de modo a resolver problemas de requisições simultâneas e supervisão do acesso à memória compartilhada.

O µcs é um sistema de microprocessador que permite, dependendo do algoritmo de controle, selecionar prioridades independentes da posição de cada µcp, além de efetuar levantamento de dados estatísticos, que permitirão ao operador verificar a performance do sistema. O µcs também pode verificar as condições de alarme emitindo mensagens ao operador quando tais condições são atingidas.

Este trabalho trata do projeto de um sincronizador

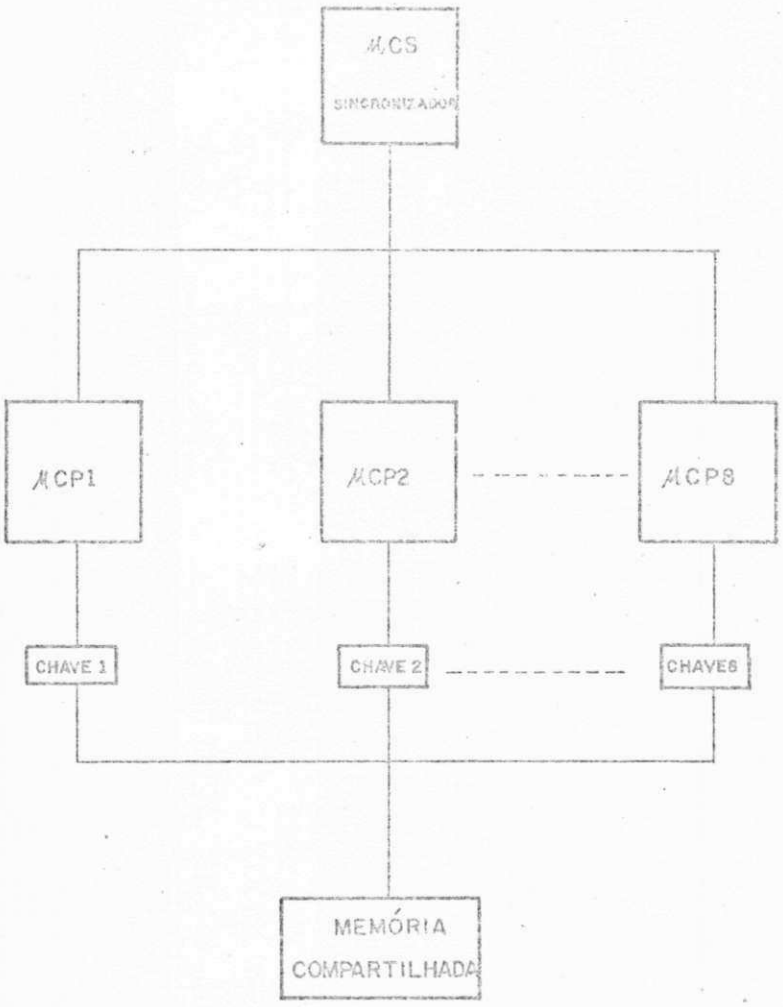


Fig. 6 - Arquitetura geral do MATER

para o sistema supra citado. O capítulo 2 apresenta uma descrição mais detalhada do MATER, o capítulo 3 descreve detalhadamente o hardware do sincronizador, o capítulo 4 mostra um pequeno sistema operacional para aplicação em tempo real e os programas executados pelo µcs, e finalmente no capítulo 5 são feitos comentários dos resultados e algumas importantes conclusões.

CAPÍTULO II

DESCRIÇÃO GERAL DO MATER

O MATER é um sistema multimicroprocessador modular, baseado na família M6800 da Motorola, desenvolvido para controle de processos em tempo real. A fig.07 mostra a arquitetura geral do MATER, compõe-se de oito microcomputadores principais(µcp's), cada um com memórias e interfaces de entrada/saída particular, podendo se comunicarem via um banco de memória, de acordo com um programa gerenciador de memória compartilhada. O acesso dos µcp's à memória compartilhada(MC) é controlado por um microcomputador sincronizador(µcs). O µcs compõe-se de memória e interfaces de entrada/saída para comunicação com usuário e controle do acesso dos µcp's à MC.

Durante o funcionamento normal do MATER, cada µcp pode se comunicar com outro µcp através da MC. Por exemplo, se o μcp_1 deseja enviar uma mensagem ao μcp_2, deve primeiramente se comunicar com o µcs para verificar se a MC está dispo-

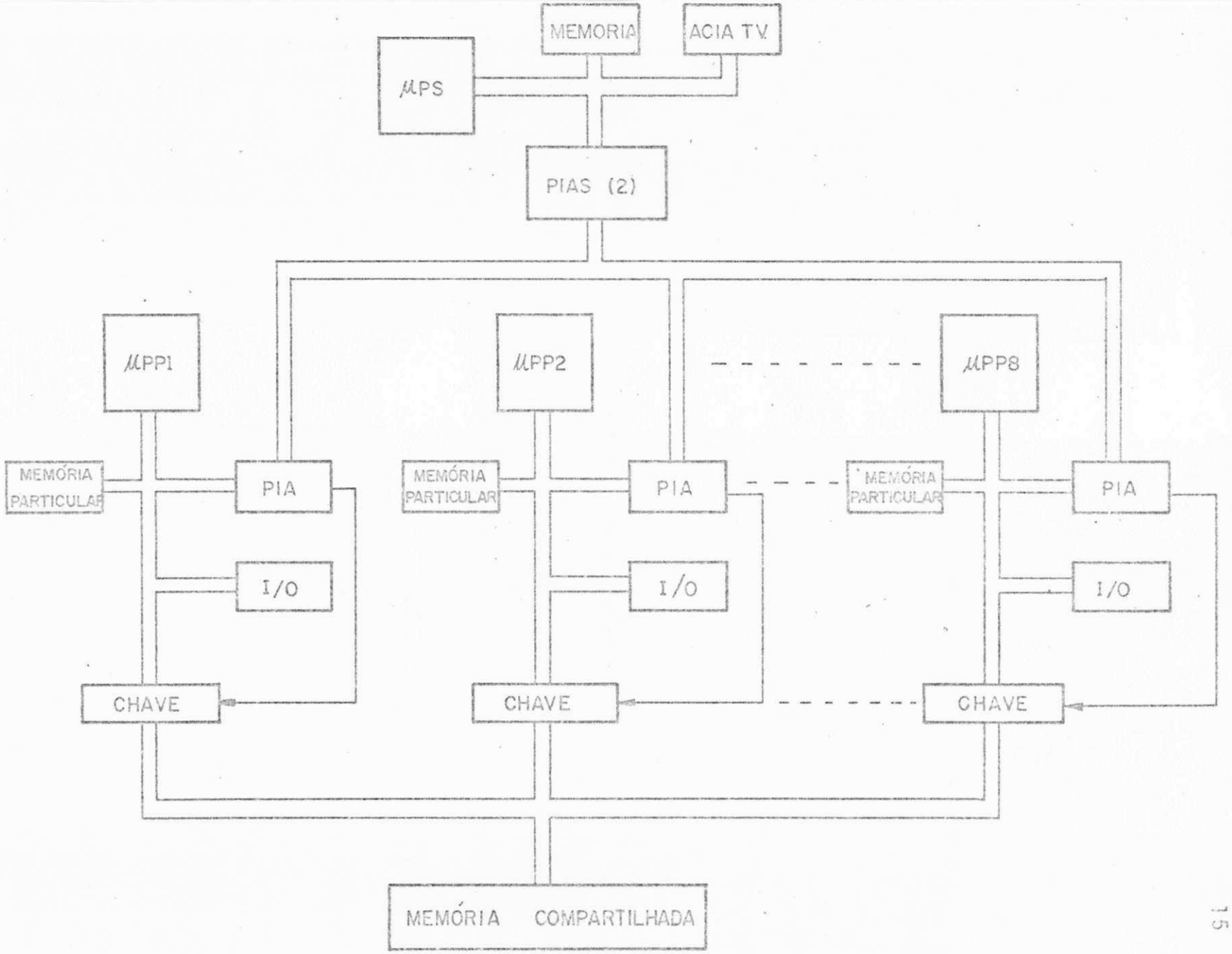


Fig. 7 - Esquema detalhado do MATER

nível, quando isso ocorrer o μcp_1 ativa a chave que dá acesso à MC(Fig. 07) e então escreve a mensagem de acordo com o gerenciador da MC(GMC). Após isso, o μcp_1 deve desativar a chave e informar ao μcs à que μcp se destina aquela mensagem. O μcs fica verificando(Polling) os μcp's um a um ciclicamente, até detectar um que deseje acesso à MC ou deseje sinalizar(interromper um μcp para ler na MC) outro μcp. Após a deteção de um μcp requisitante, o μcs informa-o sobre o estado da MC(disponível ou não). Caso a MC esteja disponível, o μcs, após informar ao μcp requisitante, fica aguardando nova informação deste μcp, para que um outro μcp seja sinalizado ou para liberar o μcs para eventuais atendimentos de outros μcp's. O μcs faz aquisição de dados estatísticos quanto à utilização da MC, pelos μcp's(número de acessos à MC, tempo de uso da MC, etc.), e verifica periodicamente se algum μcp foi ativado ou desativado durante o funcionamento do MATER. Esses dados estatísticos poderão ser utilizados pelo operador para verificar o desempenho do sistema.

O GMC é um programa, armazenado em memória EPROM na MC, desenvolvido para controlar o fluxo de mensagens entre os μcp's. As mensagens são protocoladas como dados, tarefas ou mensagens de comunicação. Devido às características dinâmicas das mensagens trocadas pelos μcp's, a memória RAM da MC deve ser dividida em páginas. Cada mensagem será dividida em páginas referenciadas por uma tabela de mensagens(TMSG). Na TMSG existirá um apontador para a primeira página da mensagem, número de páginas da mensagem, e os números dos μcp's transmisso-

res a receptores. O µcp executará o GMC até que a transmissão da mensagem seja encerrada.

O MATER se comunica com o usuário através de dispositivos de entrada e saída de caracteres cuja velocidade é lenta comparada com a velocidade de operação do microprocessador(18, 19), de modo que o tempo perdido pelo microprocessador no processo de entrada e saída de caracteres pelos periféricos é prejudicial ao desempenho do sistema. Além disso, existem tarefas que devem ser acionadas por meio de interrupções do processo externo sob controle. O sistema operacional do MATER, baseado em um programa chamado KERNEL(20), minimiza o tempo gasto pelo processador, no processo de entrada e saída de caracteres via terminal de vídeo/teclado, e também, permite o acionamento de tarefas por requisições do processo sob controle e do operador.

CAPÍTULO III

HARDWARE DO SINCRONIZADOR PARA O MATER

Como visto no capítulo 1, existem várias maneiras de se controlar o acesso dos microprocessadores, de um sistema multimicroprocessador, a uma via comum, que permita a utilização dos recursos compartilhados. Além desse controle, é desejado para o MATER, que seja feita aquisição de dados estatísticos sobre os µcp's de modo a permitir ao usuário, uma avaliação do desempenho do sistema. Por este motivo optou-se por um pequeno sistema microprocessador, que além de efetuar o controle dos µcp's no acesso aos recursos compartilhados, permita uma comunicação com o usuário via terminal de vídeo/teclado, possibilitando assim, o envio de relatórios sobre o comportamento dos µcp's, de acordo com os algoritmos implementados.

A grande vantagem de se utilizar um microcomputador para executar as funções de sincronizador, é que o microcomputador pode implementar vários algoritmos de controle com esquemas

de prioridades diferentes sem mudança no hardware, além de permitir a interação com o operador via terminal de vídeo/teclado.

Este capítulo mostra o esquema detalhado do circuito do µcs, as ligações com os oito µcp's do MATER e descreve como se processa a comunicação dos µcp's.

3.1. Arquitetura do Sincronizador

A fig.08 mostra um esquema em diagrama de blocos da arquitetura utilizada para o SINCRONIZADOR(µcs). Compõe-se de um microprocessador M6800 da MOTOROLA(10), uma ACIA M6850(11) para comunicação com o usuário via vídeo/teclado, 2 Kbytes de memória EPROM 2708(12) para armazenamento do sistema de desenvolvimento e do sistema operacional, 1 Kbyte de RAM 2114(13) disponível para desenvolvimento de novos algoritmos de controle, 256 bytes de RAM 2112(14) para armazenamento temporário de dados usados pelo sistema de desenvolvimento e para armazenamento dos dados estatísticos. Duas PIAS M6820(11) para comunicação com os µcp's e um relógio de tempo real implementado com o LM555(15) que gera interrupções NMI a cada 200 milisegundos. A fig.9 mostra o mapa de memória do µcs.

O circuito de clock mostrado na fig.10 é implementado com dois monoestáveis(74123)(16) de modo a gerar dois sinais de forma de onda quadrada com freqüências de 1MHz defasados de 180 graus, que constituem os clocks Ø1 e Ø2 necessários ao funcionamento do microprocessador. O circuito de reset é um latch implementado com duas portas NAND's de modo a eliminar o ruído da cha

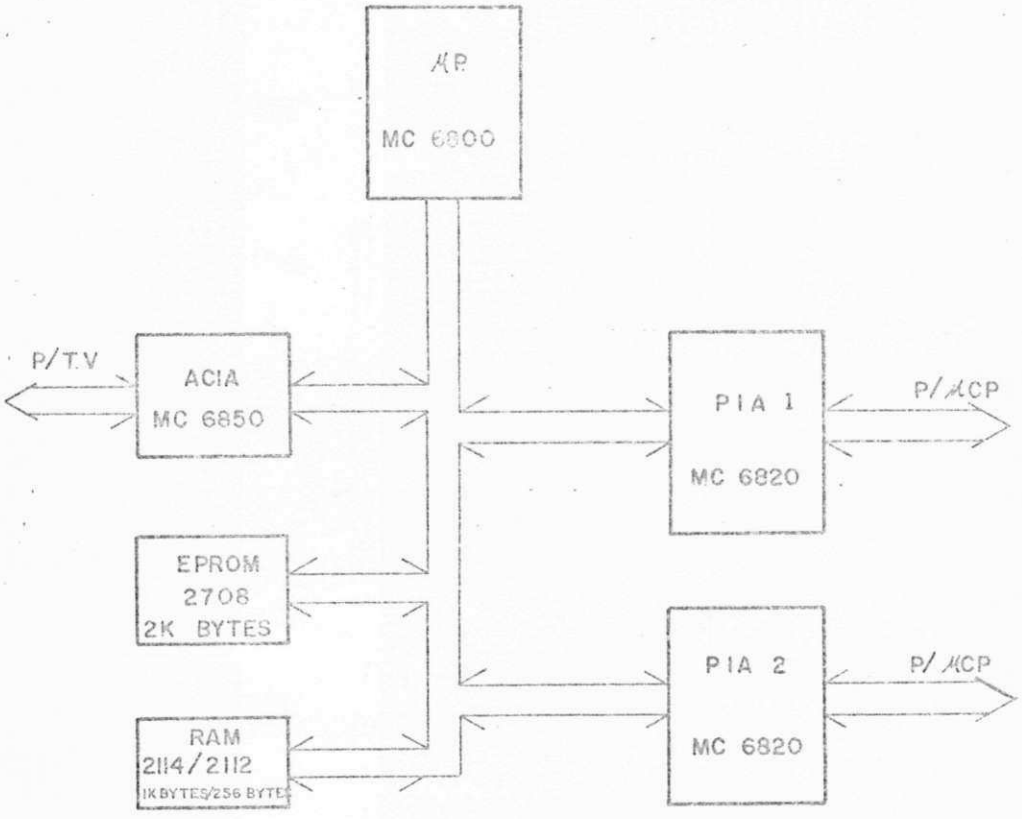


Fig. 8 - Arquitetura do μcs

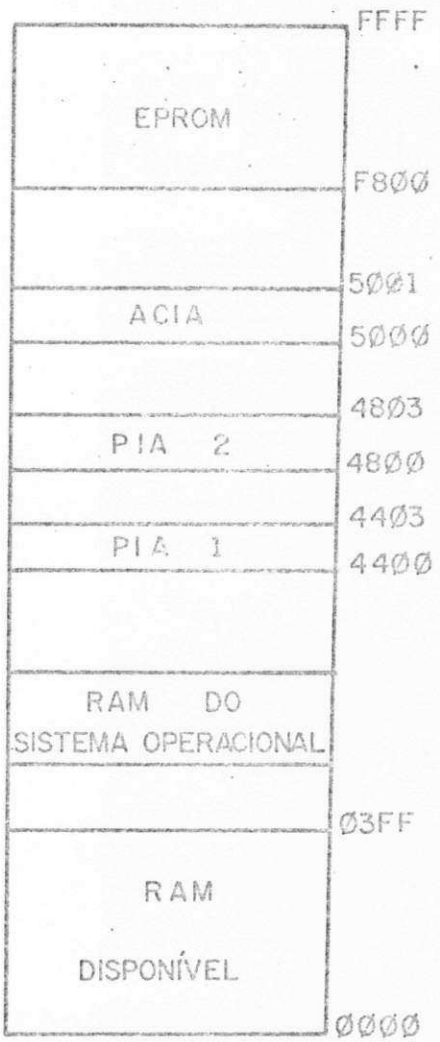
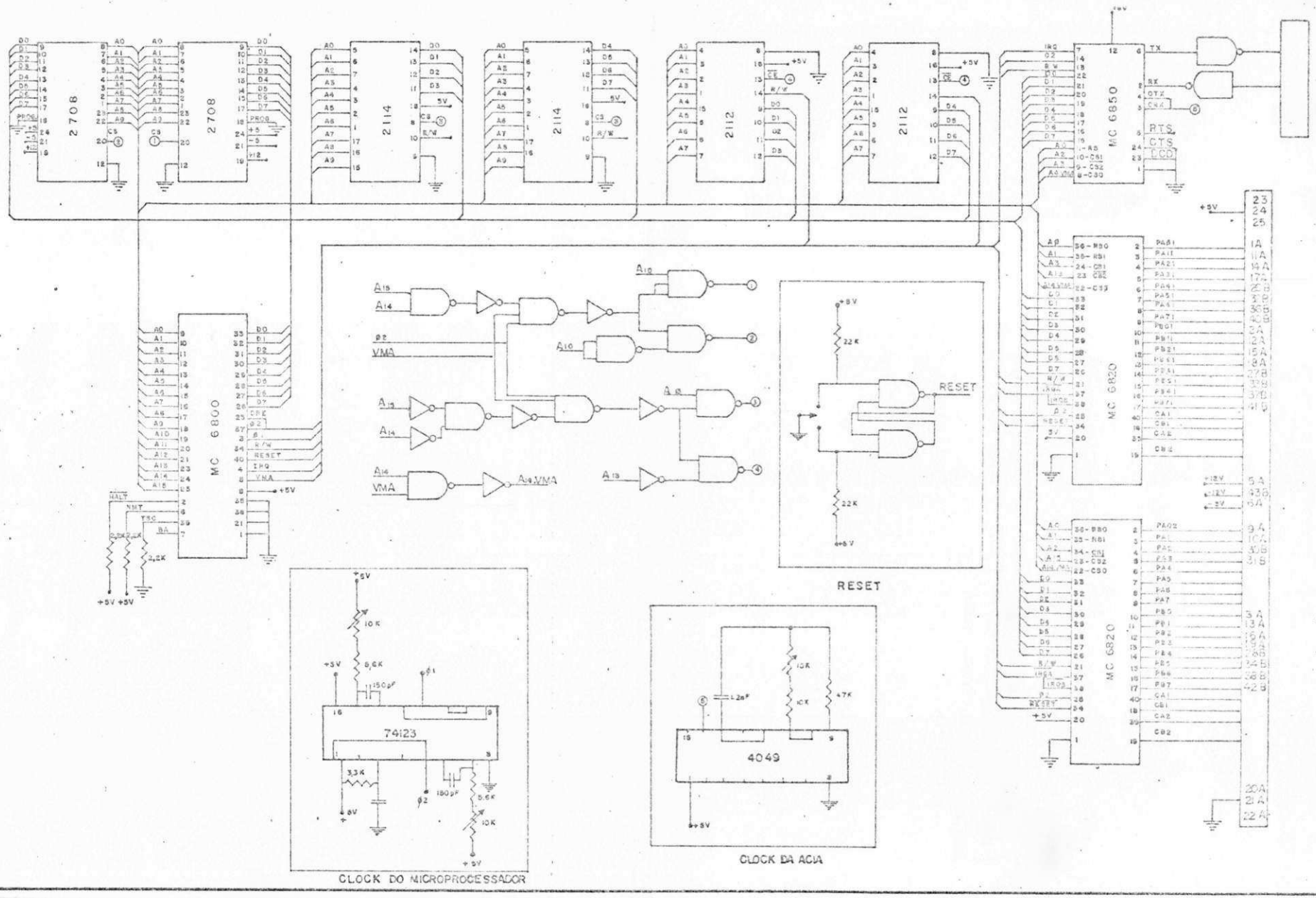


Fig. 9 - Mapa de memória do µcs

Fig. 10 - Esquema de ligações do ACS



ve RESET.

Os clocks de transmissão e recepção de dados da ACIA são gerados por um circuito oscilador que usa três portas inversoras CMOS(4049)(17).

3.2. Ligação do µcs com os µcp's

Como visto no ítem 3.1. deste capítulo, o µcs se comunica com os µcp's através de PIAS. A fig.11 mostra o esquema de ligação entre as PIAS do µcs e as PIAS dos oito µcp do MATER. A Tabela 1 apresenta cada linha desta ligação, com as respectivas ligações ao µcs e ao µcp, e a programação dessas linhas pelo µcs e µcp. A nomenclatura utilizada na Tabela 1 é a seguinte:

REQ_i - Linha utilizada pelo μcp_i para requisitar acesso a MC.

$RESP_i$ - Linha utilizada pelo µcs para informar ao μcp_i que o acesso foi concedido ou não.

SIN_i - Linha Utilizada pelo µcs para interromper(sinalizar) o μcp_i, indicando que deve usar a MC para ler mensagem.

VIA DE CÓDIGO - Via utilizada pelos µcp's para informar ao µcs que outro µcp deve ser sinalizado ou para liberar a MC, de acordo com os códigos da Tabela 2.

INT - Linha utilizada por todos os µcp's para enviar um sinal de IRQ ao µcs informando que estão ativos.

PAi1 - Linha de dados i da parte A da PIA1 (i=Ø,1,...,7)

PAi2 - Linha de dados i da parte A da PIA2 (i=Ø,1,...,7)

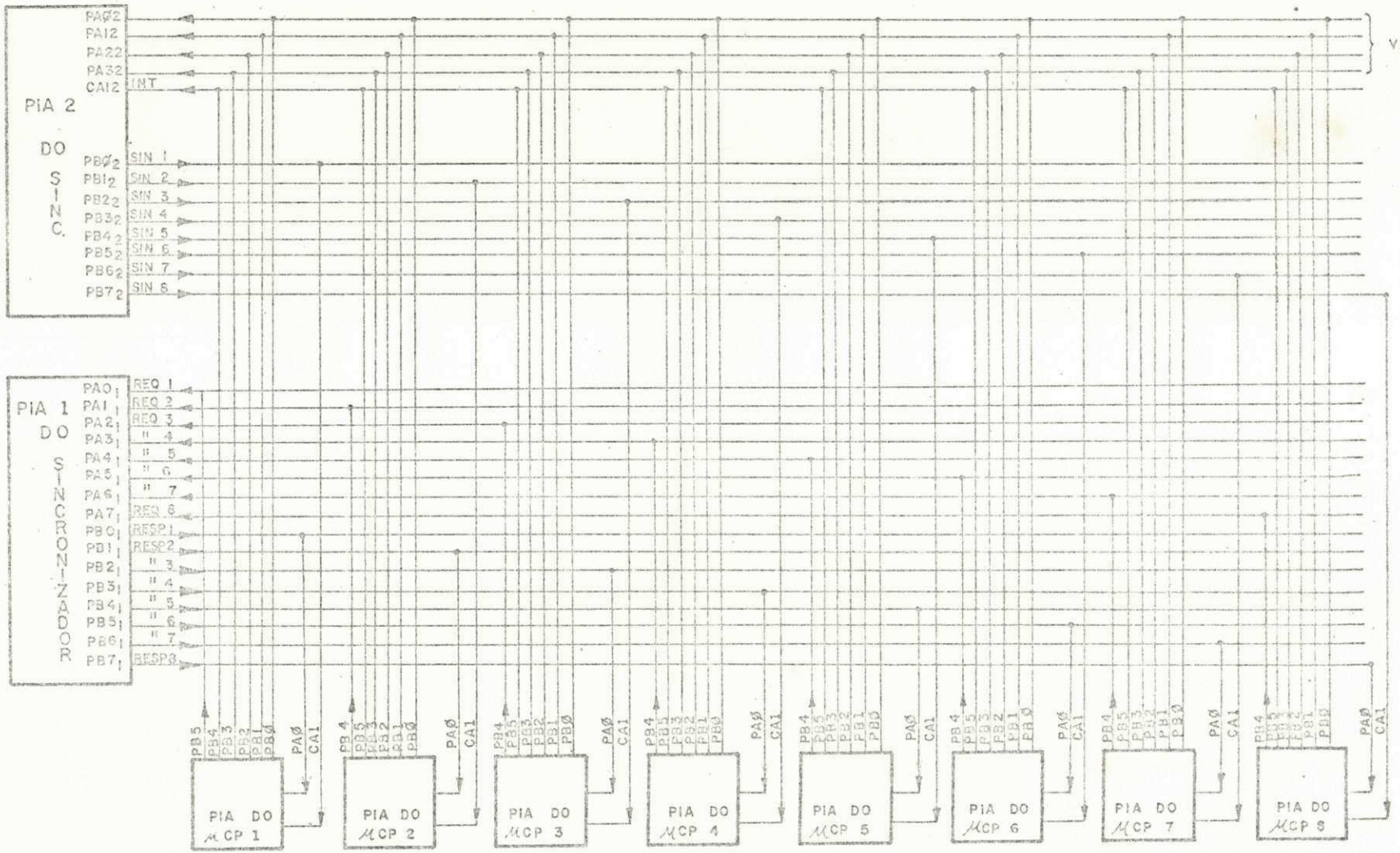


Fig. 11 - Ligação do μcs com os oito μcp's.

PBi1 - Linha de dados i da parte B da PIA1

PBi2 - Linha de dados i da parte B da PIA2

CA12 - Linha de controle de interrupção da parte A da PIA2

CA1i - Linha de controle de interrupção da parte A da PIA do µcp

LINHA	Lig.no µcs	Lig. no µcp	Prog.do µcs	Prog.do µcp
REQ_i	PAi1	PB4 do $µcp_i$	ENTRADA	SAÍDA
$RESP_i$	PBi1	PAØ do cp_i	SAÍDA	ENTRADA
SIN_i	PBi2	CA1i	SAÍDA	ENTRADA
VIAS DE CÓDIGO	PAØ2	PBØ de todos	ENTRADA	ENTRADA/SAÍDA
	$PA1_2$	PB1 de todos	ENTRADA	ENTRADA/SAÍDA
	$PA2_2$	PB1 de todos	ENTRADA	ENTRADA/SAÍDA
	$PA3_2$	PB3 de todos	ENTRADA	ENTRADA/SAÍDA
INT	$CA1_2$	PB5 de todos	ENTRADA	SAÍDA

Tabela 1 - Ligação do µcs aos µcp's

3.3. Acesso à MC

No acesso à MC, o $µcp_i$ deve executar os seguintes passos:

a) ativa a linha REQ_i

b) verifica o estado da linha $RESP_i$, até encontrá-la ativa

c) ativa chave de acesso à MC(Fig.07)

d) se comunica com a MC de acordo com o gerenciador de

acesso à MC(GMC)

e) desativa as chaves de acesso à MC

f) coloca código na via de código, mostrado na Tabela 2 indicando ao µcs outro µcp a ser sinalizado ou liberando o µcs

g) desativa linha REQ

O µcs desempenha os seguintes passos:

a) verifica o estado de todas as linhas REQ, até encontrar pelo menos uma ativada

b) verifica ciclicamente qual o µcp correspondente à linha REQ ativada

c) ativa a linha RESP correspondente ao µcp requisitante

d) verifica a via de códigos até encontrar código diferente de inválido

e) se código encontrado representa liberação do µcs (Tabela 2) volta para o passo(a), caso contrário sinaliza o µcp de acordo com o código encontrado, mostrado na Tabela 2, e volta ao passo(a).

PB_3	PB_2	PB_1	$PB_\emptyset$	µcp a ser sinalizado
Ø	Ø	Ø	Ø	$µcp_1$
Ø	Ø	Ø	1	$µcp_2$
Ø	Ø	1	Ø	$µcp_3$
Ø	Ø	1	1	$µcp_4$
Ø	1	Ø	Ø	$µcp_5$
Ø	1	Ø	1	$µcp_6$
Ø	1	1	Ø	$µcp_7$
Ø	1	1	1	$µcp_8$
1	X	X	Ø	
1	X	Ø	X	Libera o µcs
1	Ø	X	X	
1	1	1	1	Código Inválido

Tabela 2 - Codificação da Via de Códigos

CAPÍTULO IV

SOFTWARE DO µCS

Neste capítulo são apresentados os programas executados pelo µcs e o sistema operacional que gerencia esses programas. A apresentação dos algoritmos será feita através de fluxogramas, que são encontrados no fim deste capítulo. As listagens em linguagem Assembly destes programas podem ser vistas no apêndice 2.

4.1. Programas do µcs

Durante a operação do µcs, os seguintes programas(tarefas) podem ser executados.

1 - INICIALIZAÇÃO - Após a operação RESET, o operador deve executar o comando G FC4B do SATER.2(apêndice 1), para que o µcs atualize todos os parâmetros, definindo o algoritmo de controle que deve ser executado e os µcp's ativos, de acordo com comandos do operador. Durante a execução deste programa,

o µcs pergunta ao operador, através da mensagem "QUAIS?", quais os µcp's que estão ativos, o que deverá ser informado, teclando-se um número de dois caracteres hexadecimais, cujos bits são formados de acordo com a primeira coluna da Tabela 3. Após isso, o µcs pergunta, através da mensagem "ALG?", que algoritmo de controle deve ser executado. Se o algoritmo a ser executado é o CONTROLE SEM MESTRE um "S" deve ser teclado pelo operador e se o algoritmo a ser executado é o CONTROLE COM MESTRE um "M" deve ser teclado pelo operador.

2. CONTROLE SEM MESTRE - Neste programa o µcs fica verificando, através de leitura na PIA utilizada para comunicação com os µcp's, se existe algum µcp requisitando acesso a MC. Quando um µcp requisitante é detectado, o µcs responde a esta requisição através da subrotina SINAL e retorna ao início do programa de controle, após a liberação da MC.

3. CONTROLE COM MESTRE - Neste programa o µcpl é definido como mestre, e portanto possui maior prioridade no atendimento de requisições simultâneas da VIA, isto é, sempre que o µcs detecta algum µcp requisitando a VIA, primeiramente é verificado se foi o µcpl que requisitou.

4. SUBROTINA SINAL - Este programa identifica qual o µcp requisitante e sinaliza-o, indicando que a MC está disponível. Atualiza o número de vezes em que tal µcp obteve acesso ao µcs. Conta o tempo de uso da MC e aguarda resposta do µcp liberando o µcs ou indicando outro µcp a ser sinalizado. Caso o µcp não queira sinalizar outro µcp, o número de vezes em

que somente a MC foi usada é atualizado. Se o tempo de uso da MC atinge FFFF(hexadecimal), ou o número de vezes que tal µcp obteve acesso ao µcs atinge FF(hexadecimal), a tarefa VIDEO é acionada para envio da mensagem ao operador, indicando o fato.

5. VIDEO - É uma tarefa acionada por IRQ, na saída de caracteres pela ACIA, que identifica que mensagem deve ser enviada para o video, e desvia para o programa que transmite um caracter daquela mensagem.

6. TECLADO - acionada por IRQ na entrada de caracteres pela ACIA, identifica a mensagem requerida pelo operador,e caso não exista uma outra mensagem sendo enviada, atualiza o desvio que o microprocessador deve executar quando ocorrer uma interrupção IRQ na saída de caracteres pela ACIA, e envia um caracter inválido para que a tarefa VIDEO seja acionada.

7. RELÓGIO - Acionado por NMI, que ocorre de 200 a 200 milisegundos, conta o tempo de operação do µcs. Esta tarefa também verifica se existe proteção contra NMI, caso afirmativo retorna ao programa interrompido, caso negativo aciona a tarefa VRFMIC.

8. VRFMIC - envia um sinal de IRQ a cada um dos oito µcp's para verificar seu funcionamento(se está ativo ou não). A cada vez que esta tarefa é acionada um µcp é interrompido.Após os oitos µcp's serem interrompidos por esta tarefa, é feito um teste para verificar se todos os µcp's que estavam inicialmente ativos permanecem ativos, e se algum dos µcp's que

não estavam ativos foi ativado. Caso seja verificado que algum μcp foi ativado ou desativado, a tarefa VIDEO é acionada para envio de mensagem ao operador.

9. TSTMIC - Acionada por IRQ a pedido de um dos μcp's, detecta o μcp que enviou tal interrupção, e armazena esta informação em uma locação de memória indicando que tal μcp está ativo, para posterior utilização da tarefa VRFMIC.

BITS DO NÚMERO		μCP CORRESPONDENTE	VALOR QUE REPRESENTA ESTADO ATIVO
MENOS SIG.	b_0	μcp_1	1
	b_1	μcp_2	1
	b_2	μcp_3	1
	b_3	μcp_4	1
	b_4	μcp_5	1
	b_5	μcp_6	1
	b_6	μcp_7	1
MAIS SIG.	b_7	μcp_8	1

Tabela 3 - Formação do número que representa os μcp's ativos

4.2. Sistema Operacional do μcs

A comunicação do operador com um microcomputador é geralmente feita através de dispositivos, cuja velocidade de operação é baixa comparada com a velocidade de operação do micro-

processador central(18,19),pode ocorrer que o microprocessador fique inativo à espera de dados de entrada ou saída do periférico. No caso do µcs, que usa como dispositivo de interface, com o operador, uma ACIA M6850 da MOTOROLA(11), cuja velocidade de operação é bem menor que a velocidade de operação do microprocessador central, o tempo perdido pela CPU no processo de entrada e saída de caracteres pelo terminal de video/teclado, é prejudicial ao desempenho do sistema, pois deseja-se que o µcs dedique a maior parte do seu tempo de operação ao controle do processo externo(controle dos µcp's). Além disso, como visto no ítem anterior, existem tarefas que devem ser acionadas através de requisições do processo externo sob controle e do operador, e também existem tarefas que são acionadas periodicamente. Por esses motivos, foi projetado um sistema operacional de modo a minimizar o tempo gasto pelo processador nas tarefas de entrada e saída de dados via terminal de video/teclado, e permitir o controle do processo em tempo real.

4.2.1. Arquitetura do Sistema Operacional

O sistema operacional do µcs baseia-se em um programa chamado KERNEL(20), que sincroniza e aciona as diferentes tarefas do µcs, conforme o tipo de interrupção e da sua procedência. A fig.12 mostra a arquitetura do sistema operacional do µcs. O KERNEL aceita os seguintes tipos de interrupção:

a) IRQ que pode ser acionada por entrada de caracteres via teclado, saída de caracteres via video e inter-

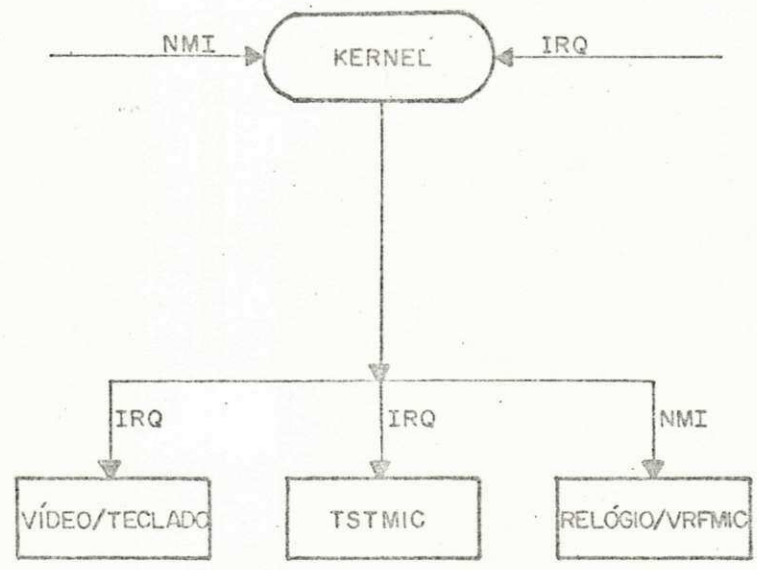


Fig. 12 - Arquitetura do sistema operacional do cs.

rupção de hardware externa devido ao processo sob controle.

b) NMI que é acionada por um relógio de tempo real de 200 a 200 milisegundos.

O sistema operacional do µcs aceita tarefas nos seguintes estados:

a) Executando Interrompível - Associado a uma tarefa quando a mesma está utilizando o processador, podendo ser interrompida.

b) Executando não Interrompível - Associado a uma tarefa que está utilizando o processador, não podendo ser interrompida.

c) Bloqueada - Tarefa que teve sua execução interrompida para que outra tarefa seja executada.

d) Pronta para Executar - Tarefa cuja excução foi solicitada e está à espera do processador.

O KERNEL é um programa desenvolvido para controlar a ativação e desativação das tarefas. A entrada no KERNEL é feita através de interrupções NMI e IRQ. Se uma tarefa está sendo executada no estado Executando Interrompível e uma interrupção ocorrer, seu estado muda para bloqueada, e o controle do processador é passado ao KERNEL, que no caso de IRQ, verifica através de um polling qual a tarefa que deve ser acionada. Se IRQ foi gera ra pela PIA, a tarefa TSTMIC é acionada; se IRQ foi gerada pela

ACIA, o KERNEL verifica se foi entrada ou saída de caracteres,e aciona a tarefa VIDEO ou TECLADO, conforme seja saída ou entrada de caracteres, respectivamente.

Se uma interrupção NMI ocorrer durante a execução de uma tarefa no estado Executando não Interrompível, apenas a tarefa RELÓGIO será acionada, e após sua execução, o controle do processador voltará à tarefa que foi interrompida.

4.3. Operação do µcs

Durante a operação do µcs, o sistema operacional permite o envio de mensagens ao terminal de vídeo, através de requisições do operador ou de requisição dos algoritmos de controle, conforme foi visto na descrição da tarefa teclado e subrotina sinal no ítem 4.1.

4.3.1. Mensagens Solicitadas Pelo Operador

O operador pode solicitar ao µcs, através de comandos, o envio de mensagens que permitem uma avaliação do desempenho do MATER. Os comandos usados no µcs são os seguintes:

a) N - Após o operador teclar N, o µcs envia ao terminal de vídeo uma mensagem contendo todos os dados estatísticos obtidos durante a operação do sistema. A mensagem é escrita no terminal de vídeo na seguinte forma:

XXXX

KK_1 MM_1 $NNNN_1$ $PPPP_1$

KK_2 MM_2 $NNNN_2$ $PPPP_2$

KK_3 MM_3 $NNNN_3$ $PPPP_3$

KK_4 MM_4 $NNNN_4$ $PPPP_4$

KK_5 MM_5 $NNNN_5$ $PPPP_5$

KK_6 MM_6 $NNNN_6$ $PPPP_6$

KK_7 MM_7 $NNNN_7$ $PPPP_7$

KK_8 MM_8 $NNNN_8$ $PPPP_8$

onde XXXX representa o tempo de operação do µcs. Uma unidade deste número corresponde a 200 milisegundos.

KK_i (i=1,2...,8) representa o número de vezes que o μcp_i foi atendido pelo µcs.

MM_i (i=1,2,...,8) representa o número de vezes que o μcp_i utilizou a MC e não requereu sinalização de outro µcp.

$NNNN_i$ (i=1,2,...,8) representa o maior tempo que o μcp_i ocupou a MC. Cada unidade deste número corresponde a 20 microsegundos, conforme algoritmo implementado(Apêndice 2)

$PPPP_i$ (i=1,2,...,8) representa o tempo total que o μcp_i ocupou a MC. Cada unidade deste número corresponde a 20 microsegundos, conforme algoritmo implemen

tado(Apêndice 2).

b) D - Após o operador teclar D, o μcs envia ao terminal de vídeo uma mensagem indicando os μcp's ativos. O exemplo abaixo indica que apenas os μcp_1, μcp_3, μcp_4 e μcp_7 estão ativos.

MICRO 1 OK
MICRO 3 OK
MICRO 4 OK
MICRO 7 OK

c) i(i=1,2,...,8) - Após o operador teclar i, o μcs envia ao terminal de vídeo uma mensagem contendo o tempo de operação do sistema e todos os dados estatísticos referentes ao μcp_i. A mensagem é escrita no terminal de vídeo na seguinte forma:

XXXX
KK_i MM_i $NNNN_i$ $PPPP_i$

onde XXXX, KK_i MM_i, $NNNN_i$ e $PPPP_i$ segue a mesma definição do ítem a.

4.3.2. Mensagens Solicitadas Pelo Algoritmo de Controle

De acordo com a subrotina sinal vista no ítem 4.1, a tarefa vídeo pode ser acionada para envio de mensagens ao operador,

para indicar que alguma condição limite foi atingida. Neste caso, as seguintes mensagens podem ser enviadas pelo µcs:

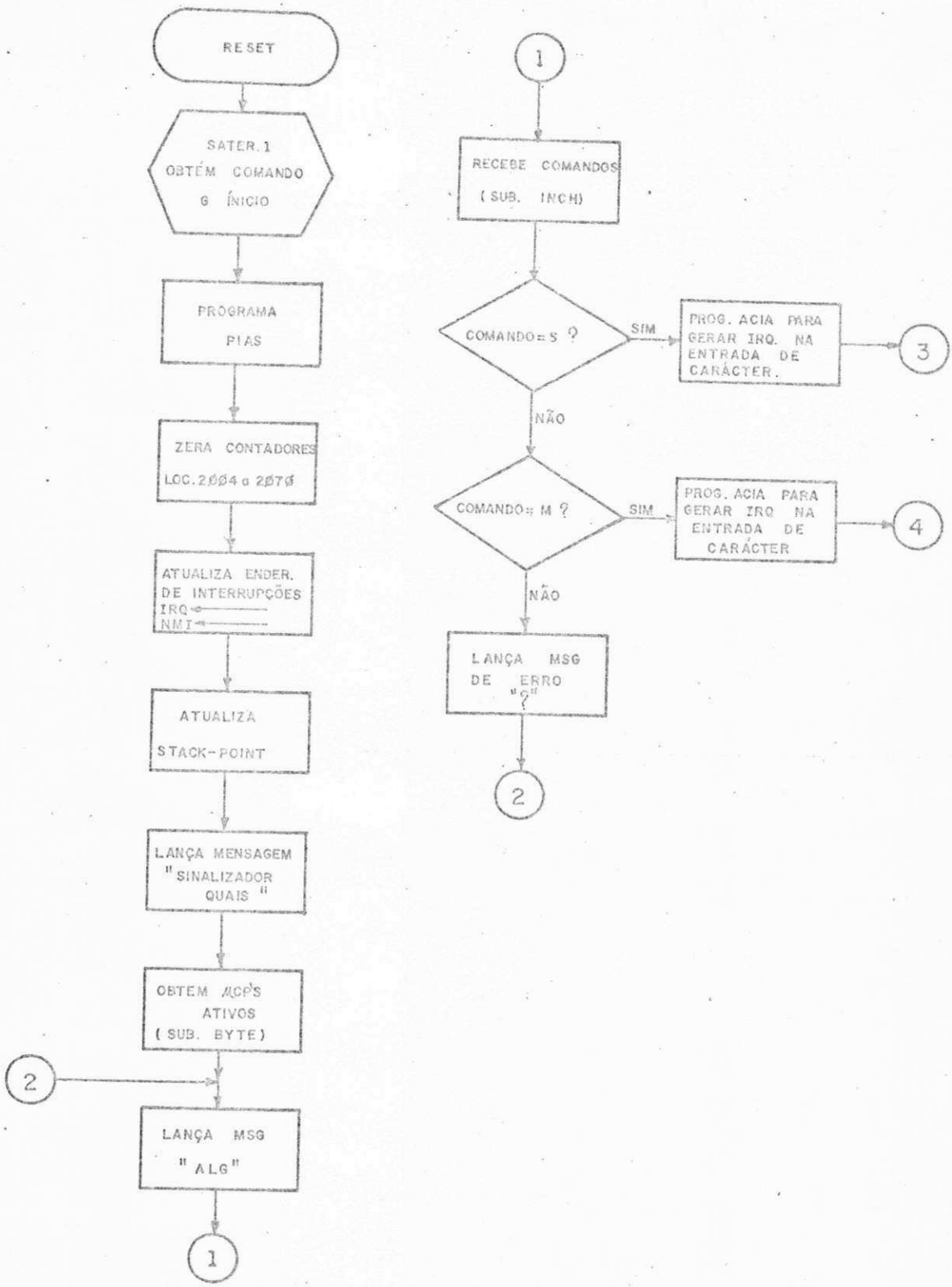
a) "MICRO i PIROU no MT" - Indica que o μcp_i está ocupando a MC a um tempo maior que FFFFx20 microsegundos.

b) "MICRO i PIROU NO TT" - Indica que o tempo total de uso da MC pelo µcpi ultrapassou FFFFx20 microsegundos.

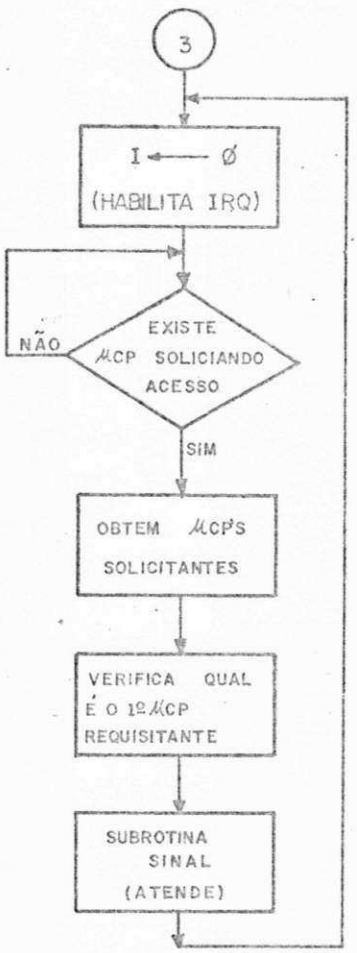
c) "XXXX"
"FF_i MM_i $NNNN_i$ $PPPP_i$",
indica que o número de acessos do μcp_i ao µcs atingiu o valor FF(hexadecimal).

Além dessas, a mensagem descrita em (b) no ítem 4.3.1, é enviada pelo µcs sempre que algum µcp for ativado ou desativado durante a operação do sistema.

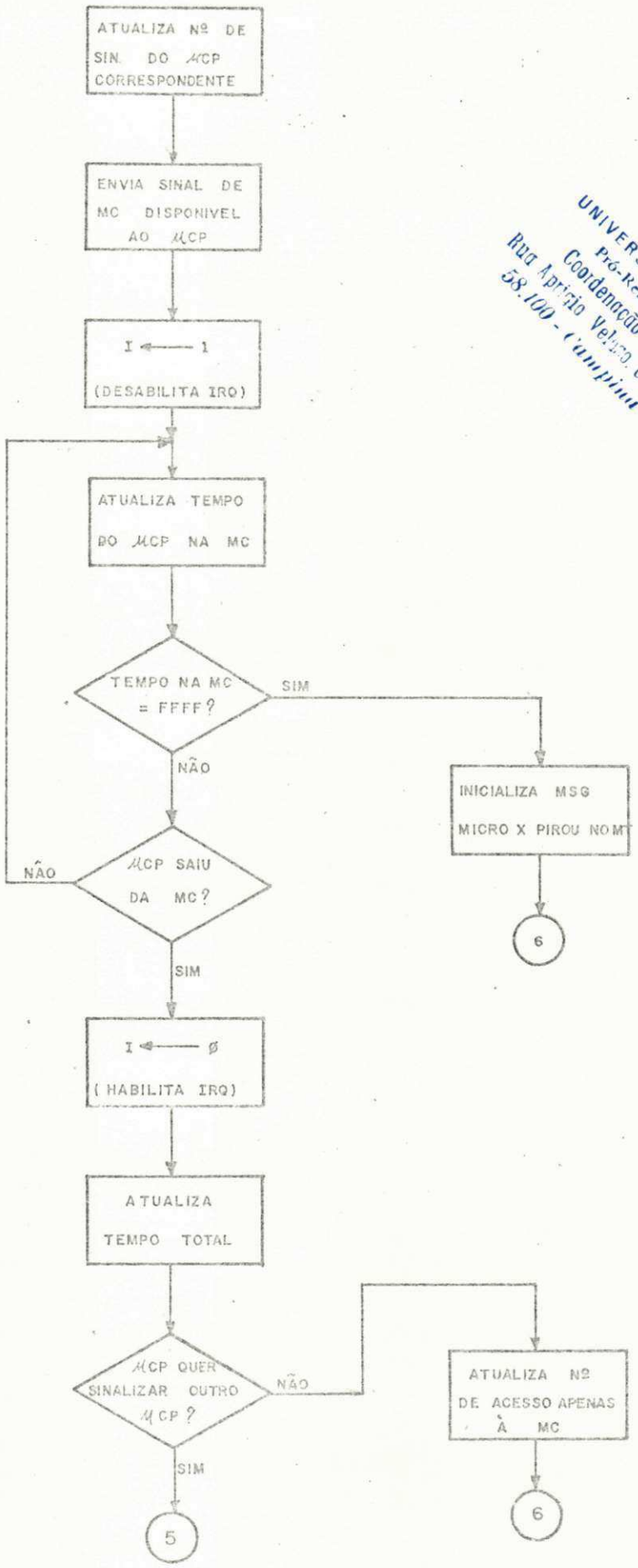
Nota: Os índices i, utilizados no ítem 4.3, não aparecem nas mensagens escritas no terminal de vídeo, foram usados apenas para facilitar a explicação.



Fluxograma da tarefa INICIALIZAÇÃO



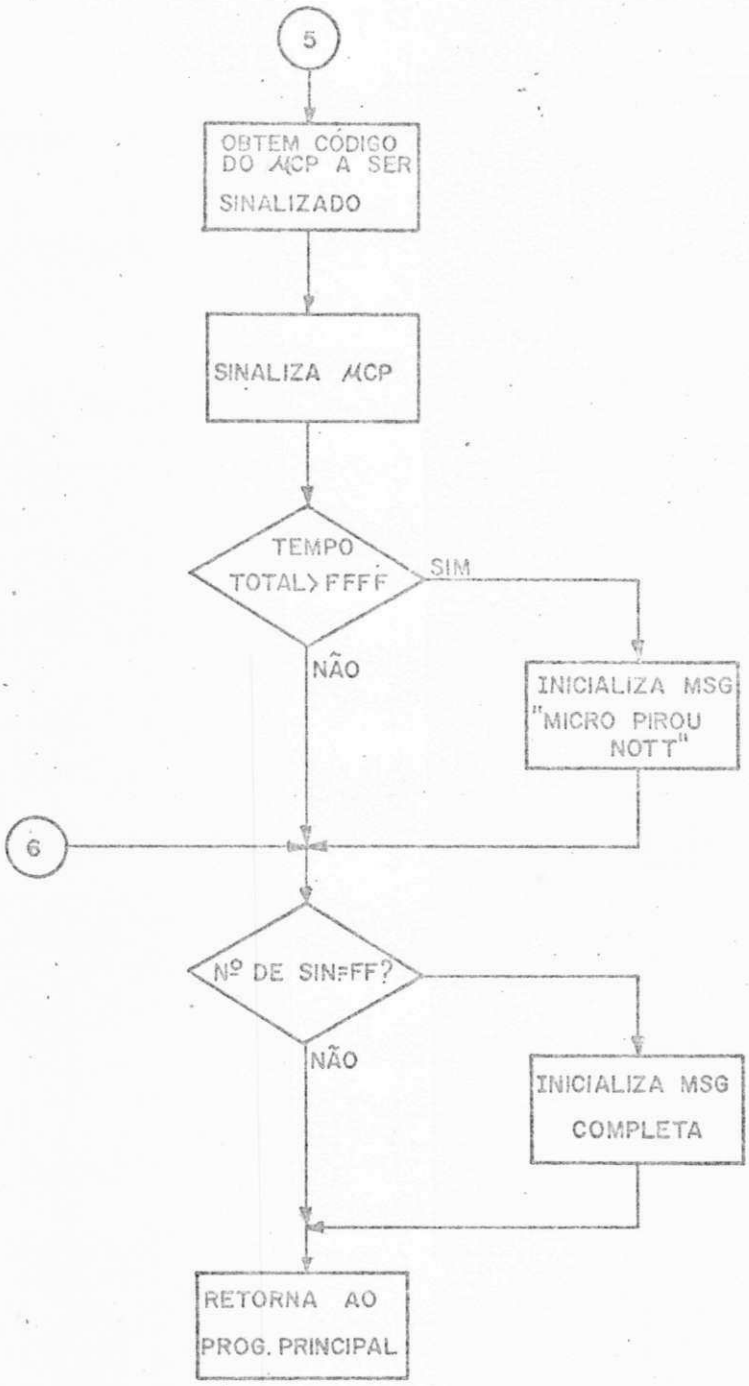
Fluxograma da tarefa CONTROLE

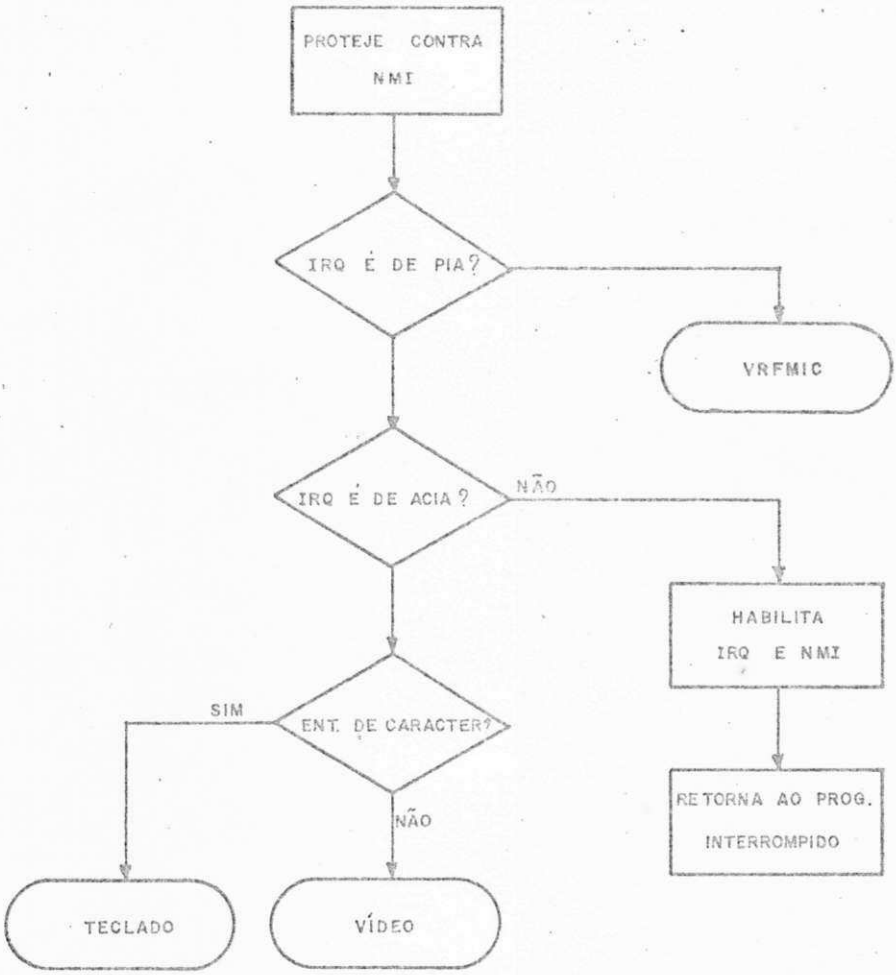


Fluxograma da Subrotina SINAL

segue na folha seguinte...

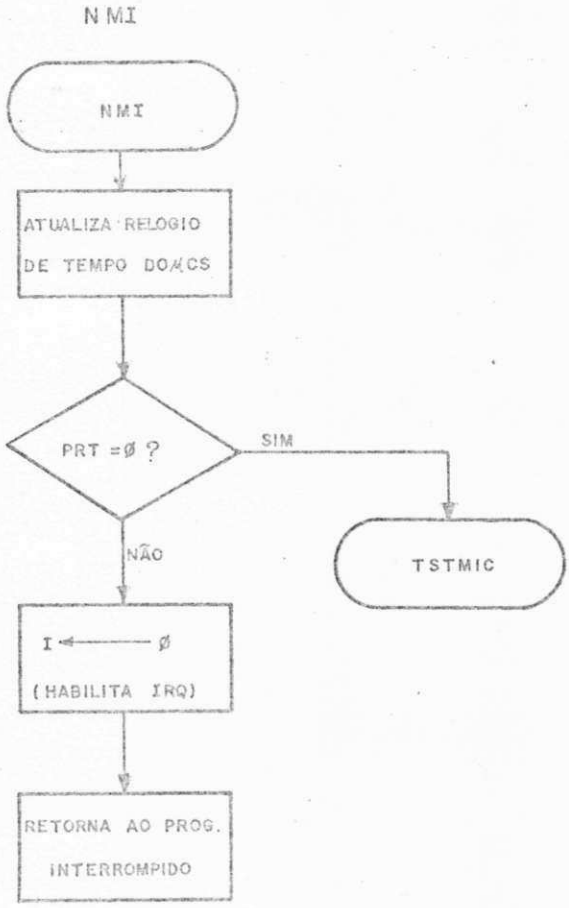
Continuação da Subrotina SINAL



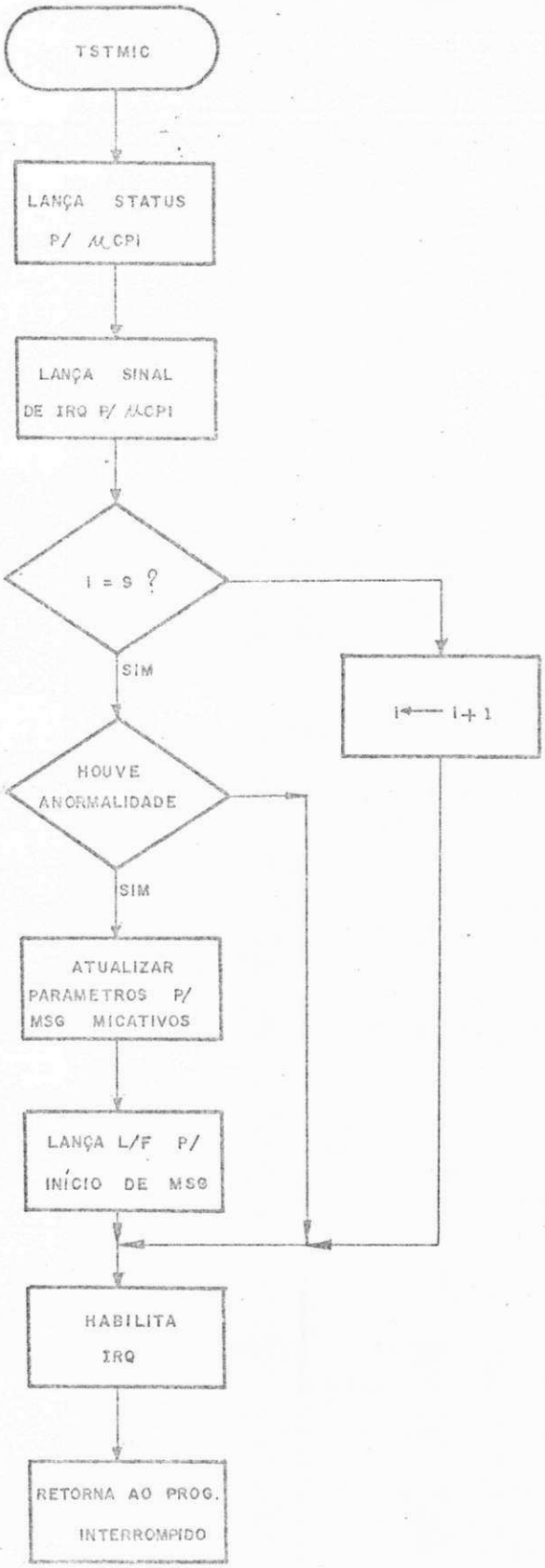


Fluxograma do KERNEL

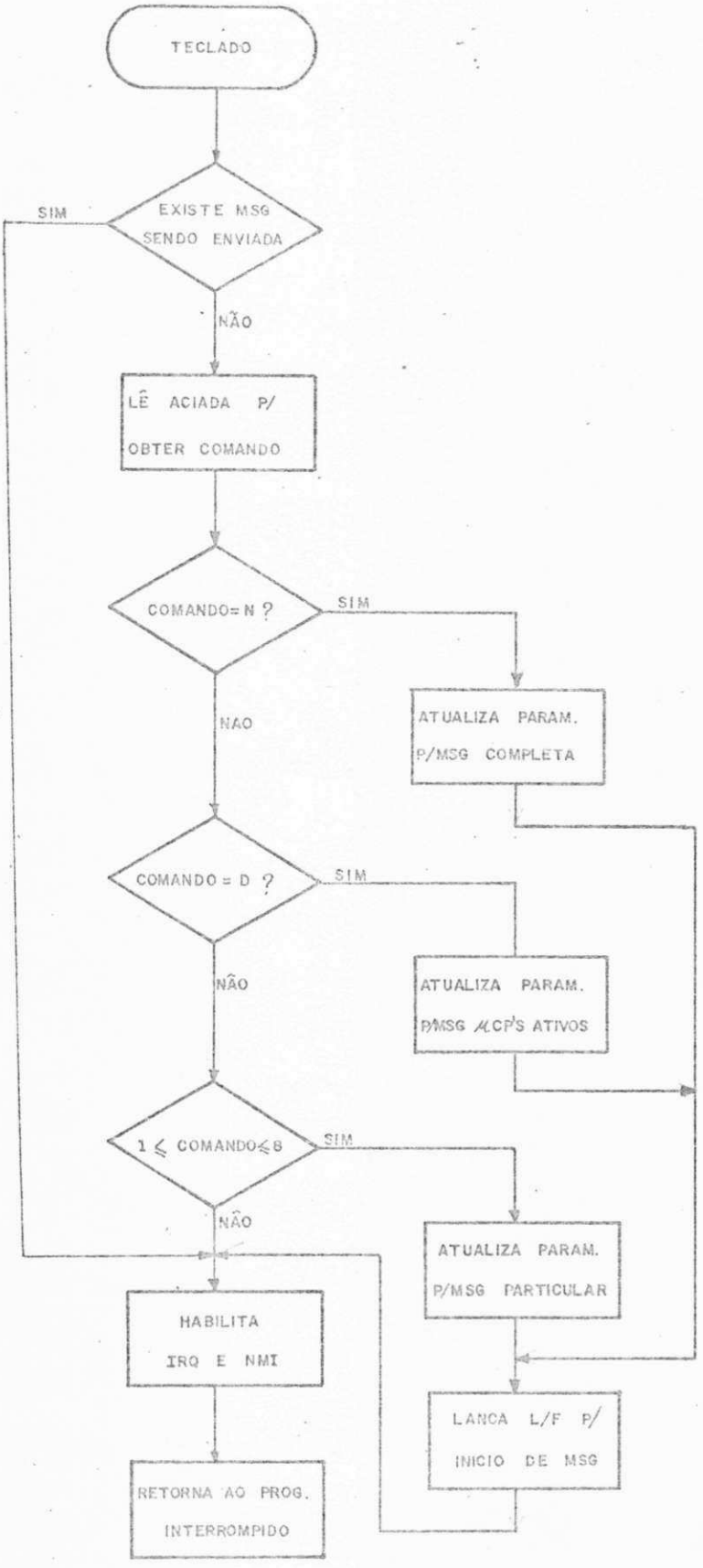
Fluxograma da tarefa VRFMIC



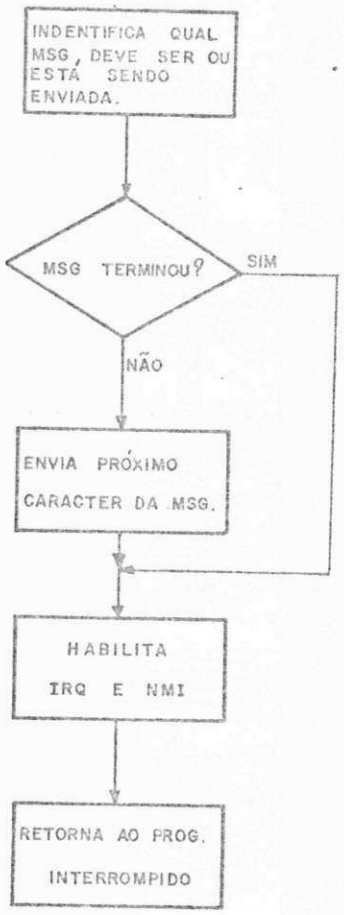
Fluxograma da tarefa RELÓGIO



Fluxograma da tarefa TSTMIC



Fluxograma da tarefa TECLADO



VÍDEO

Fluxograma da tarefa VÍDEO

CAPÍTULO V

CONCLUSÃO

Neste capítulo, apresenta-se uma descrição do protótipo do MATER que está sendo desenvolvido, uma análise do tempo de execução das principais operações e algumas sugestões para melhoria do projeto do MATER.

O MATER está sendo montado com três µcp's, uma MC e um µcs. Cada µcp possui 4 Kbytes de EPROM(2708) e 4 Kbytes e RAM (2114) na memória particular; duas ou três PIAS(M6820) para interface na comunicação com o µcs e com o processo sob controle; uma ou duas ACIAS(M6850) para comunicação com operador via terminal de vídeo/teclado, TTY e/ou ligação com um minicomputador; chaves(três estados)(8216) para acesso a MC e relógio de tempo real para gerar interrupções NMI. O µcs possui 256 Bytes de RAM (2112) utilizada para armazenamento de dados temporários e dados estatísticos de operação dos µcp's; 1 Kbyte de RAM(2114) disponível para depuração de novos algoritmos; 1 ACIA(6850)para comunicação com operador via terminal de vídeo/teclado; 2 PIAS (M6820) para comunicação com os µcp's; um relógio de tempo real para gerar interrupções NMI e 2 Kbytes de EPROM(2708) para armazenamen-

to dos algoritmos de controle, do sistema de desenvolvimento (SATER.2), e do sistema operacional. A memória compartilhada é composta de 14 Kbytes de RAM(2114) para armazenamento de mensagens de comunicação entre os µcp's e para armazenamento das tarefas a serem executadas pelos µcp's; e 2 Kbytes de EPROM(2708) para armazenamento do programa gerenciador da MC(GMC). O sistema de desenvolvimento SATER.2 foi implementado com o auxílio do microcomputador EXORCISER da MOTOROLA, todos os outros programas foram implementados e testados no próprio µcs.

O µcs, através da execução do seu algoritmo de controle, verifica a intervalos de 8 microsegundos se algum µcp deseja acesso à MC. Após uma requisição ser encontrada, o µcs dispende em média, 100 microsegundos para identificar e responder ao µcp requisitante. Esta verificação é feita ciclicamente como foi mostrado no capítulo 4, porém novos algoritmos de controle podem ser implementados. Por exemplo, se o µcp mestre acessa maior número de vezes à MC, deve ser atendido com maior prioridade. Uma solução possível para este caso, é implementar um algoritmo que verifique as requisições de todos os µcp's alternadamente com a verificação de requisição do µcp mestre. O sistema operacional do µcs, implementado para operação em tempo real de modo a aumentar o tempo dedicado ao controle dos µcp's, dispende 47 microsegundos após uma interrupção IRQ pela PIA, 120 microsegundos após uma interrupção IRQ na entrada de carácter pela ACIA, 75 microsegundos após uma interrupção IRQ na saída de carácter pela ACIA, e 200 microsegundos após uma interrupção NMI, para atualização dos parâmetros e acionamento de tarefas.

O tempo perdido pelo µcs e µcp mestre no processo de entrada e saída de caracteres via terminal de vídeo/teclado é prejudicial ao desempenho do sistema. O microprocessador M6800 da MOTOROLA necessita das vias de endereço e dados apenas na metade do período do clock, assim sugere-se o uso de dois microprocessadores interligados sincronicamente em uma mesma via(cada microprocessador utiliza a via em uma metade do período do clock), um para controle de entrada e saída de dados via terminal de vídeo/teclado e o outro para execução dos programas de controle dos processos externos, possibilitando um maior desempenho do sistema.

Os recursos oferecidos pelo microprocessador M6800, em alguns casos podem ser insuficientes para implementação de algoritmos de controle mais complexos. Por este motivo, sugere-se que o MATER seja ligado, através do seu µcp mestre, a um minicomputador que além de oferecer maiores recursos com linguagens de alto nível, permita o uso de uma memória auxiliar para o MATER.

APÊNDICE 1

DESCRIÇÃO DO SATER.2.

APÊNDICE 1

DESCRIÇÃO DO SATER.2

O SATER.2(Sistema de Desenvolvimento para Microcomputadores) é um programa de controle que pode ser usado para desenvolvimento de sistemas microprocessadores utilizando a família M6800 da MOTOROLA(10), permitindo uma interação entre usuário e microprocessador via terminal de vídeo/teclado. Além disto,as subrotinas do SATER.2 poderão ser usadas como suporte por sistemas a serem desenvolvidos. O SATER.2 foi desenvolvido baseado em sistemas semelhantes já existentes como MKIBUG/MINIBUG(21) e SATER(22).

1. Hardware do SATER.2

A fig. 13 mostra uma estrutura básica para um microcomputador que use o SATER.2 Compõe-se de uma unidade central de processamento M6800, uma memória EPROM de 1 Kbyte para armazenamento do SATER.2 uma memória RAM de 256 Bytes para armazenamento temporário de dados, uma ACIA para interface com um terminal de vídeo/teclado, e memórias RAM para armazenamento de programas do usuário.

A fig.14 mostra o mapa de memória do microcomputador da fig. 13.Neste caso, o SATER.2. fica armazenado em uma EPROM que é colocada a partir do endereço FC00 até FFFF(hexadecimal). A memória RAM para armazenamento temporário de dados é colocada a partir do endereço 2000 até 20FF(hexadecimal). A memória do usuário pode ser colocada a partir do endereço 0000 até 1FFF (hexadecimal), permitindo assim, a utilização de 8 Kbytes de memória. A ACIA ocupa os endereços 500 e 5001.

2. Operação do SATER.2.

O SATER.2. se comunica com o usuário através das seguintes funções:

1) Examina/troca o conteúdo de uma locação de memória

2) Examina conteúdo dos registradores da MPU

3) Desvia para programas do usuário

Após o RESET, o controle do microprocessador é transferido ao SATER.2, que imprime na próxima linha, do terminal de vídeo, a palavra MATER seguida de um L/F(linha completa),um C/R (retorno do carro) e o símbolo @ , e fica aguardando que o usuário tecle um dos comandos permitidos para desempenhar uma das funções acima citadas.

Durante o funcionamento normal do sistema, o SATER.2 poderá responder aos quatro tipos de interrupções do MOTOROLA, e o usuário possui controle sobre as interrupções IRQ e NMI, isto é, o usuário pode escolher o endereço para o qual o

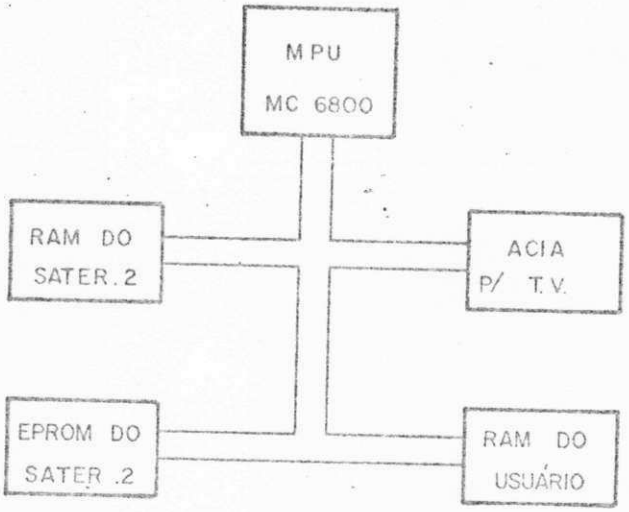


Fig. 13 - Estrutura básica de um microcomputador que usa o SATER.2

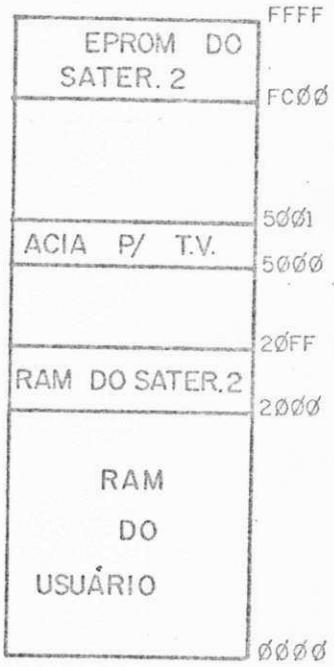


Fig. 14 - Mapa de memória do microcomputador da Fig. 13.

microprocessador deve desviar quando uma dessas interrupções ocorrer.

2.1. Examina/Troca Conteúdo da Memória

Após o SATER.2 imprimir o símbolo @ , o usuário tecla M, o SATER.2 responde com um espaço e fica aguardando que o usuário tecle os quatro números hexadecimais correspondentes ao endereço que se quer examinar/trocar o conteúdo. Quando isso ocorre, o SATER.2 responde com um novo espaço seguido de dois números hexadecimais correspondentes ao conteúdo daquela posição de memória e de um outro espaço. Se o usuário deseja modificar o conteúdo desta locação, deverá teclar dois números hexadecimais correspondentes ao novo conteúdo. Após isso o SATER.2 responde mostrando o endereço e conteúdo da locação seguinte. Se o usuário não deseja mudar o conteúdo e sim examinar a locação seguinte, deverá teclar um L/F. Caso o usuário não necessite mais desta função um C/R deverá ser teclado. O exemplo abaixo mostra a utilização desta função.

```
@M 0400 FA AD
   0401 5A L/F
   0402 67 C/R
@
```

2.2. Examina Conteúdo dos Registradores

Após o SATER.2 imprimir um @ no início da próxima linha, o usuário pode verificar o conteúdo dos registradores do mi

croprocessador teclando R. O SATER.2 mostra o conteúdo dos registradores na próxima linha, na seguinte ordem: Registro de Código de Condições(C), Acumulador B(B), Acumulador A(A), Registrador de Indexação(X), Contador de Programa(P) e Apontador de Pilha(S). Após mostrar os registradores, o controle voltará ao SATER.2 imprimindo um @ no início da próxima linha, aguardando um novo comando do operador. O exemplo abaixo mostra a utilização desta função.

```
@R
  C=E5 B=40 A=FD X=FDC4 P=0100 S=20F3
```

2.3. Desvio para Programa do Usuário

Após o SATER.2 imprimir um @ , o usuário tecla G seguido de quatro números hexadecimais correspondentes ao endereço do programa a ser executado. O controle do microprocessador é então transferido para o programa do usuário. O exemplo abaixo mostra a utilização desta função.

```
@G 0300
```

2.4. Interrupções

As locações de memória 2000 e 2003 são reservadas para indicar o endereço para o qual o microprocessador deve desviar após uma interrupção IRQ ou NMI. Sempre que uma interrupção IRQ ocorrer, o contador de programa é carregado com o conteúdo das

locações 2000(PC HI) e 2001(PC LOW) e o controle é transferido ao programa existente naquela posição. Da mesma maneira, se uma interrupção NMI ocorrer,o controle do microprocessador é transferido ao programa cujo endereço é o conteúdo das locações 2002 e 2003. É possível, através do comando M do SATER.2 modificar o conteúdo dessas locações, permitindo uma modificação dos endereços de desvio de IRQ e NMI.

CODIFICAÇÃO ASSEMBY DO SATER . 2

```
                ACIACS  EQU     $5000
                ACIADA  EQU     $5001
                        ORG     $F800

F800    B6 5000 INCH    LDAA    ACIACS      Subrotina INCH (recebe
F803    47              ASRA                um carater do TV e arma
F804    2A FA           BCC     INCH        zena-o no ACCA.
F806    B6 5001         LDAA    ACIADA
F809    84 7F           ANDA    # $7F
F80B    81 7F           CMPA    # $7F
F80D    27 F1           BEQ     INCH
F80F    7E F877         JMP     OUTCH       Desvio para eco

F812    8D EC   INHEX   BSR     INCH        Subrotina INHEX (recebe
F814    81 0A           CMPA    # 'L/F      um carater do TV, trans
F816    27 16           BEQ     C2          forma-o em hexadecimal'
F818    81 30           CMPA    # $30       e armazena-o no ACCA
F81A    2B 0F           BMI     C1
```

```
F81C  81 39          CMPA  #$39
F81E  2F 0A          BLE   INIGH
F820  81 41          CMPA  #'A
F822  2B 07          BMI   C1
F824  81 46          CMPA  #'F
F826  2E 03          BGT   C1
F828  80 07          SUBA  #$07
F82A  39       INIGH RTS
F82B  7E       C1    JMP   INICIO
F82E  09       C2    DEX
F82F  7E F937        JMP   OUTRO

F832  8D 0C    BADDR BSR   BYTE    Subrotina BADDR (recebe
F834  B7 20FE        STAA  XHI     quatro caracteres hexa-
F837  8D 07          BSR   BYTE    decimais do TV e armaze
F839  B7 20FE        STAA  XLOW    na-o no registrador X.
F83C  FE 20FE        LDX   XHI
F83F  39             RTS

F840  8D D0    BYTE  BSR   INHEX   Subrotina BYTE (recebe'
F842  48             ASLA          dois caracteres hexade-
F843  48             ASLA          cimais do TV e armaze -
F844  48             ASLA          na-o no ACCA.
F845  48             ASLA
F846  16             TAB           Salva acumulador A.
F847  8D C9          BSR   INHEX
F849  84 0F          ANDA  #$0F
F84B  1B             ABA
F84C  16             TAB
F84D  FB 20FC        ADDB  CKSUM
F850  F7 20FC        STAB  CKSUM
F853  39             RTS
```

```
F854   8D DC    CHANGE  BSR   BADDR   Subrotina CHANGE: modifica
F856   8D 38    BYTEN   BSR   OUTS    conteúdo de uma locação de
F858   8D 34            BSR   OUT2HS  memória de acordo com coman
F85A   8D EA            BSR   BYTE    do do operador.
F85C   09               DEX
F85D   A7 00            STAA  0,X
F85F   A1 00            CMPA  0,X
F861   27 03            BEQ   #$+3
F863   7E F92A          JMP   SAI?
F866   7E F932          JMP   NOVO

F869   44      OUTL    LSRA            Subrotina OUTL: envia ao TV
F86A   44               LSRA            um caracter hexadecimal.
F86B   44               LSRA
F86C   44               LSRA

F86D   84 0F    OUTHR   ANDA  #$0F      Subrotina OUTHR: envia ca -
F86F   8B 30            ADDA  #$30      racter menos significativo
F871   81 39            CMPA  #$39      de uma byte ao TV.
F873   23 02            BLS   OUTCH
F875   8B 07            ADDA  #$07

F877   37      OUTCH   PSHB            Subrotina OUTCH: envia  ao
F878   F6 5000 OUTC1   LDAB  ACIACS    TV um caracter ASCII do
F87B   57               ASRB            ACCA
F87C   57               ASRB
F87D   24 F9            BCC   OUTC1
F87F   B7 5001          STAA  ACIADA
F882   33               PULB            Recupera res. B.
F883   39               RTS

F884   A6 00    OUT2H   LDAA  0,X       Subrotina OUT2H: envia ao
F886   8D E1            BSR   OUTHL     TV dois caracters hexac-
F888   A6 00            LDAA  0,X       teres hexadecimais.
F88A   8D E1            BSR   OUTHR
F88C   08               INX
F88D   39               RTS
```

```
F88E  8D F4    OUT2HS  BSR   OUT2H
F890  86 20    OUTS    LDAA  # $20
F892  20 E3            BRA   OUTCH
```

Subrotina OUT2HS: envia ao TV dois caracteres hexadecimais, seguido de um espaço

```
F894  30       PRINT   TSX
F895  FF 20FA          STX   SP
F898  86 0D            LDAA  # 'C/R
F89A  BD F877          JSR   OUTCH
F89D  86 0A            LDAA  # 'L/F
F89F  BD F877          JSR   OUTCH
F8A2  86 43            LDAA  # 'C
F8A4  BD F877          JSR   OUTCH
F8A7  BD F99C          JSR   SAITR
F8AA  BD F88E          JSR   OUT2HS
F8AD  86 42            LDAA  # 'B
F8AF  BD F877          JSR   OUTCH
F8B2  BD F99C          JSR   SAITR
F8B5  BD F88E          JSR   OUT2HS
F8B8  86 41            LDAA  # 'A
F8BA  BD F877          JSR   OUTCH
F8BD  BD F99C          JSR   SAITR
F8C0  BD F88E          JSR   OUT2HS
F8C3  86 58            LDAA  # 'X
F8C5  BD F877          JSR   OUTCH
F8C8  BD F99C          JSR   SAITR
F8CB  BD F884          JSR   OUT2H
F8CE  BD F88E          JSR   OUT2HS
F8D1  86 50            LDAA  # 'P
F8D3  BD F877          JSR   OUTCH
F8D6  BD F99C          JSR   SAITR
F8D9  BD F884          JSR   OUT2H
F8DC  BD F88E          JSR   OUT2HS
F8DF  86 53            LDAA  # 'S
F8E1  BD F877          JSR   OUTCH
F8E4  BD F99C          JSR   SAITR
```

Subrotina PRINT: imprime no TV o conteudo dos regs.

```
F8E7    BD F884    JSR     OUT2H
F8EA    BD F884    JSR     OUT2H
F8ED    7E F984    JMP     INICIO

F8F0    86 03      RESET   LDAA    # $03     Reseta ACIA
F8F2    B7 5000            STAA    ACIACS
F8F5    86 11              LDAA    # $11
F8F7    B7 5000            STAA    ACIACS    Programa ACIA
F8FA    8E 20F2    CNTRL   LDS     # $TACK    Define stack-pointer
F8FD    86 0D              LDAA    # 'C/R
F8FF    BD F877            JSR     OUTCH
F902    86 0A              LDAA    # 'L/F
F904    BD F877            JSR     OUTCH
F907    7E F969            JMP     MENS
F90A    16         SEGUE   TAB
F90B    BD F890            JSR     OUTS
F90E    C1 4D              CMPB    # 'M        Obtem comandos do TV.
F910    27 51              BEQ     CHANG1
F912    C1 52              CMPB    # 'R
F914    27 50              BEQ     PRINT1
F916    C1 47              CMPB    # 'G
F918    26 10              BNE     SAI?
F91A    BD F832            JSR     BADDR
F91D    B6 20FE            LDAA    XHIGH
F920    B7 20F8            STAA    PHIGH
F923    B6 20FF            LDAA    XLOW
F926    B7 20F9            STAA    PLOW
F929    3B                 RTI               Desvia para programa do
                                             usuario.
F92A    86 3F      SAI?    LDAA    # '?
F92C    BD F877            JSR     OUTCH
F92F    7E F984            JMP     INICIO
F932    8E 20F2    NOVO    LDS     STACK
F935    20 05              BRA     NOVO1
F937    8E 20F2    OUTRO   LDS     STACK
F93A    20 05              BRA     OUTRO1
```

```
F93C  86 0A     NOVO1   LDAA    # 'L/F
F93E  BD F877           JSR     OUTCH
F941  86 0D     OUTRO1  LDAA    # 'C/R
F943  BD F877           JSR     OUTCH
F946  08                INX
F947  FF 20FE           STX     XHIGH
F94A  B6 20FE           LDAA    XHIGH
F94D  16                TAB
F94E  BD F869           JSR     OUTHL
F951  17                TBA
F952  BD F86D           JSR     OUTHR
F955  B6 20FF           LDAA    XLOW
F958  16                TAB
F959  BD F869           JSR     OUTHL
F95C  17                TBA
F95D  BD F86D           JSR     OUTHR
F960  7E F856           JMP     BYTEN
F963  7E F854   CHANG1  JMP     CHANGE
F966  7E F894   PRINT1  JMP     PRINT
F969  86 4D     MENS    LDAA    # 'M    Envia msg. "MATER" ao TV.
F96B  BD F877           JSR     OUTCH
F96E  86 41             LDAA    # 'A
F970  BD F877           JSR     OUTCH
F973  86 54             LDAA    # 'T
F975  BD F877           JSR     OUTCH
F978  86 45             LDAA    # 'E
F97A  BD F877           JSR     OUTCH
F97D  86 52             LDAA    # 'R
F97F  BD F877           JSR     OUTCH
F982  20 03             BRA     REPETE
F984  8E 20F2   INICIO  LDS     STACK  Inicializa Stack Pointer.
F987  86 0D     REPETE  LDAA    # 'C/R
F989  BD F877           JSR     OUTCH
F98C  86 0A             LDAA    # 'L/F
F98E  BD F877           JSR     OUTCH
F991  86 40             LDAA    # '@    Envia
```

```
F993    BD F877          JSR     OUTCH
F996    BD F800          JSR     INCH
F999    7E F90A          JMP     SEGUE
F99C    86 3D    SAITR   LDAA    # ' =    Subordina SAITR: envia ao
F99E    BD F877          JSR     OUTCH    TV o sinal =.
F9A1    39               RTS

F9A2    FE 2000          LDX     ENDIRQ   Prepara desvio de IRQ.
F9A5    6E 00            JMP     0,X

F9A7    FE 2002          LDX     ENDNMI   Prepara desvio de NMI.
F9AA    6E 00            JMP     0,X
```

DADOS DA RAM:

20F2	STACK
20F3	RCC
20F4	ACCB
20F5	ACCA
20F6	SXIGH
20F7	XLOW
20F8	PHIGH
20F9	PLOW
20FA	SHIGH
20FB	SLOW
20FC	CRSM
20FD	BYTEC
20FE	REGXH
20FF	REGXL

APÊNDICE 2

CODIFICAÇÃO ASSEMBY DOS PROGRAMAS DO μCS

APENDICE 2

CODIFICAÇÃO ASSEMBLY DOS PROGRAMAS DO μCS

```
INTIRQ   EQU    $2000
INTNMI   EQU    $2002
MICTST   EQU    $200C
MICATI   EQU    $200D
MEMJ     EQU    $201A
MEMY     EQU    $201C
MEMA     EQU    $204B
LINHA    EQU    $2050
INDICE   EQU    $2052
NMIC     EQU    $2054
MEMZ     EQU    $2055
PRTMSG   EQU    $2056
TEMP     EQU    $2057
MODO     EQU    $205A
PRTNMI   EQU    $205B
TEMP2    EQU    $205C
MEMX     EQU    $205D
RELGO    EQU    $205E
MEMG     EQU    $2061
PCA1     EQU    $4401
PDDB1    EQU    $4402
PCB1     EQU    $4403
PCA2     EQU    $4801
PDDB2    EQU    $4802
PCB2     EQU    $4803
ACIACS   EQU    $5000
ACIADA   EQU    $5001
```

PROGRAMA QUE ENVIA AO TERMINAL DE VIDEO A MENSAGEM
SOBRE OS ESTADOS DOS UCP'S (ATIVO OU NÃO)

```
                        ORG     $F9EF
F9EF    B6 200D DESV1   LDAA    MEMOD       Obtem ucp's ativos
F9F2    B5 2055         BITA    MEMZ
F9F5    26 16           BNE     SAI
F9F7    7C 2054         INC     NMIC
F9FA    78 2055 SEGUE   ASL     MEMZ
F9FD    B6 2055         LDAA    MEMZ
FA00    81 00           CMPA    #$00
FA02    26 EB           BNE     DESV1
FA04    86 91           LDAA    # $91
FA06    B7 5000         STAA    ACIACS      Pros.ACIA p/ gerar IRQ na
FA09    F7 2056         CLR     PRTMSG      entrada de carct.
FA0C    3B              RTI
FA0D    FE 2050 SAI     LDX     LINHA
FA10    7C 2051         INC     LINHA
FA13    8C F9B4         CPX     # $F9B4
FA16    27 06           BEQ     A
FA18    A6 00           LDAA    0,X         Obtem carct. da msg.
FA1A    B7 5001         STAA    ACIADA      Envia carct. da msg.
FA1D    3B              RTI

FA1E    CE F9Ac A       LDX     # $F9AC
FA21    FF 2050         STX     LINHA
FA24    CE FA34         LDX     # $FA34
FA27    FF 2052         STX     INDICE
FA2A    7C 2054         INC     NMIC
FA2D    B6 2054         LDAA    NMIC        Obtem número do micro.
FA30    B7 5001         STAA    ACIADA      Envia número do micro.
FA34    CE FA40 DESV2   LDX     # $FA40
FA37    FF 2052         STX     INDICE
FA3A    86 20           LDAA    # 'SP
FA3C    B7 5001         STAA    ACIDA       Envia espaço.
FA3F    3B              RTI
```

```
FA40    CE FA4C             LDX     # $FA4C
FA43    FF 2052             STX     INDICE
FA46    86 4F               LDAA    # 'O
FA48    B7 5001             STAA    ACIDA       Envia O.
FA4B    3B                  RTI

FA4C    CE FA58             LDX     # $FA58
FA4F    FF 2052             STX     INDICE
FA52    86 4B               LDAA    # 'K
FA54    B7 5001             STAA    ACIADA      Envia K.
FA57    3B          RTI

FA58    CE F9EF             LDX     # $F9EF
FA5B    FF 2052             STX     INDICE
FA5E    7E F9FA             JMP     SEGUE
```

PROGRAMA QUE ENVIA AO TERMINAL DE VIDEO
A MENSAGEM MICRI i PIROU NO MT.

```
                              ORG     $FA61
FA61    FE 2050     INIC      LDX     LINHA
FA64    7C 2051               INC     LINHA
FA67    8C F9B4               CPX     # $F9B4     Verf. se msg terminou.
FA6A    27 06                 BEQ     A
FA6C    A6 00                 LDAA    0,X
FA6E    B7 5001               STAA    ACIADA      Envia um carct. da msg.
FA71    3B                    RTI

FA72    CE FA85     A         LDX     # $FA85
FA75    FF 2052               STX     INDICE
FA78    CE F9D5               LDX     # $F9D5
FA7B    FF 2050               STX     LINHA
FA7E    B6 2054               LDAA    NMIC
FA81    B7 5001               STAA    ACIADA      Envia número do micro.
FA84    3B                    RTI

FA85    FE 2050               LDX     LINHA
FA88    7C 2051               INC     LINHA
FA8B    8C F9E1               CPX     # $F9E1     Verf. se msg terminou.
FA8E    27 06                 BEQ     B
FA90    A6 00                 LDAA    0,X
FA92    B7 5001               STAA    ACIDA       Lança um carct. da msg.
FA95    3B                    RTI

FA96    86 91       B         LDAA    # $91
FA98    B7 5000               STAA    ACIACS      Pros. ACIA.
FA9B    7F 2056               CLR     PRTMSG      Habilita p/ nova msg.
FA9E    3B                    RTI
```

PROGRAMA QUE ENVIA AO TERMINAL DE VIDEO

A MENSAGEM MICRI i PIROU NO TT.

```
                                ORG     $FA9F
FA9F    FF 2050                 LDX     LINHA
FAA2    7C 2051                 INC     LINHA
FAA5    8C F9B4                 CPX     # $F9B4     Verf. se msg terminou.
FAA8    27 06                   BEQ     A
FAAA    A6 00                   LDAA    0,X
FAAC    B7 5001                 STAA    ACIADA      Lança um carct. da msg.
FAAF    3B                      RTI

FAB0    CE FAC3         A       LDX     #'$FAC3
FAB3    FF 2052                 STX     INDICE
FAB6    CE F9E2                 LDX     #'$F9E2
FAB9    FF 2050                 STX     LINHA
FABC    B6 2054                 LDAA    NMIC
FABF    B7 5001                 STAA    ACIADA      Envia número do micro.
FAC2    3B                      RTI

FAC3    FE 2050                 LDX     LINHA
FAC6    7C 2051                 INC     LINHA
FAC9    8C F9EE                 CPX     #'F9EE      Verf. se msg terminou.
FACC    27 06                   BEQ     B
FACE    A6 00                   LDAA    0,X
FAD0    B7 5001                 STAA    ACIADA      Lança um carct. da msg.
FAD3    3B                      RTI

FAD4    86 91           B       LDAA    #'$91
FAD6    B7 5000                 STAA    ACIACS      Prog. ACIA.
FAD9    79 2056                 CLR     PRTMSG      Habilia p/ nova msg.
FADC    3B                      RTI
```

PROGRAMA QUE ENVIA AO TERMINAL DE VIDEO A MENSAGEM QUE
CONTEM TODOS OS DADOS ESTATÍSTICOS OBTIDOS PELO UCS.

```
                        ORG     $FADD
FADD    CE FAE9         LDX     #'$FAE9
FAE0    FF 2052         STX     INDICE
FAE3    86 0A           LDAA    #'L/F
FAE5    B7 5001         STAA    ACIADA      Envia L/F.
FAEB    3B              RTI

FAE9    CE FAF6         LDX     #$FAF6
FAEC    FF 2052         STX     INDICE
FAEF    B6 205E         LDAA    RELHI       Obtem relogio hish.
FAF2    BD F869         JSR     OUTL        Envia 1 carct. do relg.
FAF5    3B              RTI

FAF6    CE FB03         LDX     #'$B03
FAF9    FF 2052         SIT     INDICE
FAFC    B6 205E         LDAA    RELHI
FAFF    BD F86D         JSR     OUTR        Envia seg. carct. do rel.
FB02    3B              RTI

FB03    CE FB10         LDX     #'$FB10
FB06    FF 2052         STX     INDICE
FB09    B6 205F         LDAA    RELLOW      Obtem relogio low.
FB0C    BD F869         JSR     OUTL        Lança terc. carct. do rel.
FB0F    3B              RTI

FB10    CE FB1D         LDX     #$FB1D
FB13    FF 2052         STX     INDICE
FB16    B6 205F         LDAA    RELLOW
FB19    BD F86D         JSR     OUTR        Lança quarto caracter
FB1C    3B              RTI                 do relogio.

FB1D    CE FB29         LDX     #$FB29
FB20    FF 2052         STX     INDICE
FB23    86 0A           LDAA    #'L/F
```

```
FB25    B7 5001         STAA    ACIADA      Envia L/F.
FB28    3B              RTI

FB29    CE FB35         LDX     #$FB35
FB2C    FF 2052         STX     INDICE
FB2F    86 0D           LDAA    #'C/R
FB31    B7 5001         STAA    ACIADA      Envia C/R.
FB34    3B              RTI

FB35    CE FB44         LDX     #$FB44
FB38    FF 2052         STX     INDICE
FB3B    FE 2057         LDX     TEMP        Recupera deg. X.
FB3E    A6 00           LDAA    0,X
FB40    BD F869         JSR     OUTL        Envia primeiro carct.
FB43    3B              RTI                 do número de vezes que o
                                            ucp usou o ucs.

FB44    CE FB53         LDX     #$FB53
FB47    FF 2052         STX     INDICE
FB4A    FE 2057         LDX     TEMP        Recupera reg. X.
FB4D    A6 00           LDAA    0,X
FB4F    BD F86D         JSR     OUTR        Envia segundo carct. do
FB52    3B              RTI                 número de vezes que o
                                            ucp usou o ucs.
FB53    CE FB5F         LDX     #$FB5F
FB56    FF 2052         STX     INDICE
FB59    86 20           LDAA    #'SP
FB5B    B7 5001         STAA    ACIADA      Envia espaço.
FB5E    3B              RTI

FB5F    CE FB6E         LDX     #$FB6E
FB62    FF 2052         STX     INDICE
FB65    FE 2057         LDX     TEMP        Recupera reg.X.
FB68    A6 0A           LDAA    0,X
FB6A    BD F869         JSR     OUTL        Envia primeoro carct. do
FB6D    3B              RTI                 número de acessos a MC.
```

```
FB6E    CE FB7D         LDX     # $FB7D
FB71    FF 2052         STX     INDICE
FB74    FE 2057         LDX     TEMP        Recupera indexador.
FB77    A6 0A           LDAA    0A,X
FB79    BD F86D         JSR     OUTR        Envia segundo carct.
FB7C    3B              RTI                 do número de acessos a MC.

FB7D    CE FB89         LDX     # $FB89
FB80    FF 2052         STX     INDICE
FB83    86 20           LDAA    # 'SP
FB85    B7 5001         STAA    ACIADA      Envia espaço.
FB88    3B              RTI

FB89    CE FB98         LDX     # $FB98
FB8C    FF 2052         STX     INDICE
FB8F    FE 2057         LDX     TEMP        Recupera indexador.
FB92    A6 20           LDAA    20,X
FB94    BD F869         JSR     OUTL        Envia primeiro carct. do
FB97    3B              RTI                 maior tempo que o ucp
                                            ocupou a MC.

FB98    CE FBA7         LDX     # $FBA7
FB9B    FF 2052         STX     INDICE
FB9E    FE 2057         LDX     TEMP        Recupera indexador.
FBA1    A6 20           LDAA    20,X
FBA3    F86D            JSR     OUTR        Envia segundo carct. do
FBA6    3B              RTI                 maior tempo que o ucp
                                            ocupou a MC.

FBA7    CE FBB6         LDX     # $FBB6
FBAA    FF 2052         STX     INDICE
FBAD    FE 2057         LDX     TEMP        Recupera indecador.
FBB0    A6 28           LDAA    28,X
FBB2    BD F869         JSR     OUTL        Envia terceiro carct. do
FBB5    3B              RTI                 maior tempo que o ucp
                                            ocupou a MC.

FBB6    CE FB65         LDX     # $FBC5
FBB9    FF 2052         STX     INDICE
```

```
FBBC    FE 2057     LDX   TEMP      Recupera indexador.
FBBF    A6 28       LDAA  28,X
FBC1    BD F86D     JSR   OUTR      Envia quarto carct. do
                                    maior tempo que o ucp
                                    ocupou a MC.
FBC5    CE FBD1     LDX   # $FBD1
FBC8    FF 2052     STX   INDICE
FBCB    86 20       LDAA  # 'SP
FBCD    B7 5001     STAA  ACIADA    Envia espaço.
FBD0    3B          RTI

FBC5    CE FBD1     LDX   # $FBD1
FBC8    FF 2052     STX   INDICE
FBCB    86 20       LDAA  # 'SP
FBCD    B7 5001     STAA  ACIADA    Envia espaço.
FBD0    3B          RTI

FBD1    CE FBE0     LDX   # $FBE0
FBD4    FF 2052     STX   INDICE
FBD7    FE 2057     LDX   TEMP      Recupera indexador.
FBDA    A6 30       LDAA  30,X
FBDC    BD F869     JSR   OUTL      Envia primeiro carct. do
FBDF    3B          RTI             tempo  total que o ucp
                                    ocupou a MC.
FBE0    CE FBEF     LDX   # $FBEF
FBE3    FF 2052     STX   INDICE
FBE6    FE 2057     LDX   TEMP      Recupera indexador.
FBE9    A6 30       LDAA  30,X
FBEB    BD F86D     JSR   OUTR      Envia segundo carct. do
FBEE    3B          RTI             tempo total que o ucp
                                    ocupou a MC.
FBEF    CE FBFE     LDX   # $FBFE
FBF2    FF 2052     STX   INDICE
FBF5    FE 2057     LDX   TEMP      Recupera indexador .
FBF8    A6 38       LDAA  38,X
FBFA    BD F869     JSR   OUTL      Envia terceiro carct. do
FBFD    3B          RTI             maior tempo que o ucp ocu
                                    pou a MC.
```

```
FBFE    CE FC0D          LDX     # $FC0D
FC01    FF 2052          STX     INDICE
FC04    FE 2057          LDX     TEMP        Recupera indexador.
FC07    A6 38            LDAA    38,X
FC09    BD F86D          JSR     OUTR        Envia quarto carct. do
FC0C    3B               RTI                 maior tempo que o ucp
                                             ocupou a MC.

FC0D    B6 205A          LDAA    MODO
FC10    27 12            BEQ     FIM         Verf. modo de msd.
FC12    B6 2058          LDAA    TEMPHI
FF15    81 0B            CMPA    # $0B       Verf. se msd. terminou.
FC17    27 0B            BEQ     FIM
FC19    7C 2058          INC     TEMPHI
FC1C    CE FB1D          LDX     # $FB1D
FC1F    FF 2052          STX     INDICE
FC22    6E 00            JMP     0,X
FC24    86 91    FIM     LDAA    # $91
FC26    B7 5000          STAA    ACIACS      Prog. ACIA.
FC29    7F 2056          CLR     PRTMSG      Habilita para nova msd.
FC2C    3B               RTI
```

INICIALIZAÇÃO DO µCS

```
                                ORG     $FC4B
FC4B    86 FF       INICIO      LDAA    # $FF
FC$D    B7 4402                 STAA    PDDB1       Programa PIAS.
FC50    B7 4802                 STAA    PDDB2
FC53    86 04                   LDAA    # $04
FC55    B7 4401                 STAA    PCA1
FC58    B7 4403                 STAA    PCB1
FC5B    B7 4803                 STAA    PCB2
FC5E    86 07                   LDAA    # $07
EC60    B7 4801                 STAA    PCA2
FC63    CE 2004                 LDX     INIC
EC66    6F 00       RET         CLR     0,X         Prepara memoria para arma
FC68    08                      INX                 zenamento dos dados esta-
FC6C    26 F8                   BNE     RET         tisticos.
FC6E    86 01                   LDAA    # $01
FC70    B7 200C                 STAA    MICTST
FC73    CE FCC6                 LDX     # $FCC6
FC76    FF 2000                 STX     INTIRQ      Prepara desvio de IRQ.
FC79    CE FD8C                 LDX     # $FD8C
FC7C    FF 2002                 STX     INTNMI      Prepara desvio de NMI.

FC7F    8E 20F2                 LDS     STACK       Atualiza stack pointer.
FC82    CE FC2D                 LDX     # $FC2D
FC85    A6 00                   LDAA    0,X
FC87    BD F877                 JSR     OUTCH       Envia msd. QUAIS ?
FC8A    08                      INX
FC8B    8C FC44                 CPX     # $FC44      Verf. se msd terminou.
FC8E    26 F5                   BNE     VOLTA
FC90    BD F840                 JSR     BYTE        Obtem ucp's ativos.
FC93    B7 200D                 STAA    MICATI
FC96    CE FC44     MSG2        LDX     # $FC44
FC99    A6 00       NOVO        LDAA    0,X
FC9B    BD F877                 JSR     OUTCH       Envia msg. ALG ?
FC9E    08                      INX
```

```
FC9F    8C FC4B          CPX    # $FC4B
FCA2    26 F5            BNE    NOVO
FCA4    BD F800          JSR    INCH       Obtem algoritmo.
FCA7    81 53            CMPA   # 'S
FCA9    27 0B            BEQ    PONTE1
FCAB    81 4D            CMPA   # 'M
FCAD    27 0F            BEQ    PONTE2
FCAF    86 3F            LDAA   # '?
FCB1    BD F877          JSR    OUTCH      Envia codigo de erro.
FCB4    20 E0            BRA    MSG2

FCB6    86 91    PONTE1  LDAA   # $91
FCBS    B7 5000          STAA   ACIACS     Prog. ACIA.
FCBB    7E FDE8          JMP    SINS
FCBE    86 91    PONTE2  LDAA   # $91
FCC0    B7 5000          STAA   ACIACS     Prog. ACIA.
FCC3    7E F8F0          JMP    SINC
```

KERNEL DO UCS

```
                                ORG     $FCC6
FCC6    7C 205B     IRQ         INC     PRTNMI      Proteje contra NMI.
FCC9    0F                      SEI                 Proteje contra IRQ.
FCCA    B6                      LDAA    PCA2
FCCD    85 80                   BITA    #$80        IRQ foi da PIA ?
FCCF    26 5B                   BNE     TSTMIC
FCD1    B6 5000                 LDAA    ACIACS
FCD4    85 01                   BITA    #$01        IRQ foi ent. de carc ?
FCD6    26 08                   BNE     TECLA
FCD8    FE 2052                 LDX     INDICE      Obtem desvio.
FCDB    7F 205B                 CLR     PRTMSG      Habilita NMI.

FCDE    6E 00                   JMP     0,X         Desvio p/ tarefa.
FCE0    B6 2056     TECLA       LDAA    PRTMSG
FCE3    27 05                   BEQ     VERF        Verf. prot. de msd.
FCE5    B6 5001                 LDAA    ACIADA      Reseta flag de IRQ.
FCE8    0E                      CLI
FCE9    3B                      RTI

FCEA    B6 5001     VERF        LDAA    ACIADA      Obtem carct. que inter-
FCED    84 7F                   ANDA    #$7F        rompeu o ucs.
FCEF    81 4E                   CMPA    #'N
FCF1    27 47                   BEQ     MSG3
FCF3    81 44                   CMPA    #'D
FCF5    27 0D                   BEQ     MSG4
FCF7    81 30                   CMPA    $30
FCF9    2F 08                   BLE     RTNO
FCFB    81 3B                   CMPA    #$38
FCFD    2F 65                   BLE     MSG3
FCFF    0E                      CLI
FD00    7F 205B                 CLR     PRTNMI      Habilita NMI.
FD03    3B          RTNO        RTI
```

```
FD04    7C 2056   MSG4    INC   PRTMSG
FD07    CE F9EF           LDX   #$F9EF
FD0A    FF 2052           STX   INDICE
FD0D    CE F9AC           LDX   #$F9AC
FD10    FF 2050           STX   LINHA
FD13    86 30             LDAA  #$30
FD15    B7 2054           STAA  NMIC
FD18    86 01             LDAA  #$01
FD1A    B7 2055           STAA  MENZ
FD1D    86 B1             LDAA  #$B1
FD1F    B7 5000           STAA  ACIACS    Prog. ACIA.
FD22    86 0A             LDAA  #'L/F
FD24    7F 205B           CLR   PRTMSG
FD27    B7 5001           STAA  ACIADA    Envia L/F para ativação
FD2A    0E                CLI             da tarefa video.
FD2B    3B                RTI
FD2C    B6 205C   TSTMIC  LDAA  TEMP2     Obtem ucp ativos.
FD2F    BB 205D           ADDA  MEMX
FD32    B7 205D           STAA  MEMX
FD35    0E                CLI
FD36    7E 205B           CLR   PRTNMI
FD39    3B                RTI

FD3A    7C 2056   MSG3    INC   PRTMSG    Atualiza parametros para
FD3D    CE FADD           LDX   #$FADD    envio de msg.
FD40    FF 2052           SIT   INDICE
FD43    CE 2004           LDX   #$2004
FD46    FF 2057           STX   TEMP
FD49    7C 205A           INC   MODO
FD4C    CE F9AC           LDX   #$F9AC
FDAF    FF 2050           STX   LINHA
FD52    86 B1             LDAA  #$B1
FD54    B7 5000           STAA  ACIACS
FD57    86 0A             LDAA  #'L/F
FD59    B7 50001          STAA  ACIADA
FD5C    0E                CLI
```

```
FD5C    7F 205B          CLR     PRTNMI
FD60    3B               RTI

FD61    7C 2056  MIC     INC     PRTMSG      Atualiza parametros
FD64    80 2D            SUBA    # $2D       para envio de msg.
FD66    B7 2058          STAA    TEMPL
FD69    7D 205A          CLR     MODO
FD6C    CE FADD          LDX     # $FADD
FD6F    FF 2052          STX     INDICE
FD72    86 20            LDAA    # $20
FD74    B7 2057          STAA    TEMP
FD77    CE F9AC          LDX     # $F9AC
FD7A    FF 2050          STX     LINHA
FD7D    86 B1            LDAA    # $B1
FD7F    B7 5000          STAA    ACIACS
FD82    86 0A            LDAA    # 'L/F
FD84    B7 5001          STAA    ACIADA
FD87    0E               CLI
FD88    7F 205B          CLR     PRTMSG
FD8B    3B               RTI
```

TAREFA RELOGIO

```
                        ORG     $FD8C
FD8C    0F              SEI
FD8D    FE 205E         LDX     RELGO       Conta tempo de operação
FD90    08              INX                 do ucs.
FD91    FF 205E         STX     RELGO
```

TAREFA VRFMIC

```
                        ORG     FD94
FD94    B6 205B VRFMIC  LDAA    PRTNMI
FD97    26 1B           BNE     FIM
FD99    B6 200C         LDAA    MICTST
FD9C    27 18           BEQ     A
FD9E    B7 4802         STAA    PDB2        Envia sinal de IRQ a
FDA1    7F 4802         CLR     PDB2        um ucp.
FDA4    B7 4402         STAA    PDB1
FDA7    86 FF           LDAA    #$FF
FDA9    4A      LOOP    DECA
FDAA    81 00           CMPA    #$00
FDAC    26 FB           BNE     LOOP
FDAE    B7 205C         STAA    TEMP2
FDB1    78 200C         ASL     MICTST
FDB4    0E      FIM     CLI
FDB5    3B              RTI

FDB6    86 01   A       LDAA    #$01        Verifica quais os ucp's
FDB8    B7 200C         STAA    MICTST      estão ativos.
FDBB    B6 205D         LDAA    MEMX
FDBE    B1 200D         CMPA    MEMOD
FDC1    27 F1           BEQ     FIM
FDC3    B7 200D         STAA    MEMOD       Atualiza parametros para
FDC6    CE F9EF         LDX     #$F9EF      ativação de tarefa video.
```

```
FDC9    FF 2052     STX   INDICE
FDCC    CE F9AC     LDX   # $F9AC
FDCF    FF 2050     STX   LINHA
FDD2    86 30       LDAA  # $30
FDD4    B7 2054     STAA  NMIC
FDD7    86 01       LDAA  # $01
FDD9    B7 2055     STAA  MEMZ
FDDC    86 B1       LDAA  # $B1
FDDE    B7 5000     STAA  ACIACS
FDE1    86 0A       LDAA  # 'L/F
FDE3    B7 5001     STAA  ACIADA
FDE6    0E          CLI
FDE7    3B          RTI
```

CONTROLE SEM MESTRE

```
                        ORG     $FDE8

FDE8    0E              INICIO  CLI             Habilita IRQ
FDE9    CE 2004                 LDX     #$2004
FDEC    C9 01                   LDAB    #$01
FDEE    F7 204B                 STAB    MEMA
FDE1    F6 4400         F1S     LDAB    PDA1    Obtem ucp's requisitan-
FDE4    F4 200B                 ANDB    MEMD    tes
FDF7    27 F8                   BEQ     F1S
FDF9    F5 204B                 BITB    MEMA    Verf. qual o ucp que
FDFC    26 06                   BNE     SINAL1  requisita a VIA.
FDFE    78 204B         SEGUE   ASL     MEMA
FE01    08                      INX
FE02    20 DD                   BRA     F1S
FE04    BD FE36         SINAL1  JSR     SINAL
FE07    B6 204B                 LDAA    MEMA
FE0A    85 80                   BITA    #$80
FE0C    27 F0                   BEQ     SEGUE
FE0E    20 D8                   BRA     INICIO

FE10    B6 2056         OPERA1  LDAA    PRTMSG  Atualiza parametros
FE13    26 30                   BNE     REFAZ   para ativação da tare-
FE16    B6 201B                 LDAA    MEMJL   fa video.
FE19    8B 2D                   ADDA    #$2D    Obtem num. do ucp.
FE1B    B7 2054                 STAA    NMIC
FE1E    CE FA61                 LDX     #$FA61
FE21    FF 2052                 STX     INDICE
FE24    CE F9AC                 LDX     #$F9AC
FE27    FF 2050                 STX     LINHA
FE2A    86 B1                   LDAA    #$B1
FE2C    B7 5000                 STAA    ACIACS  Prog. ACIA.
FE2F    86 0A                   LDAA    #'L/F
FE31    B7 5001                 STAA    ACIADA
FE34    20 0F                   BRA     REFAZ
```

SUBROTINA SINAL

```
                                ORG     $FE36

FE36        6C 00       SINAL   INC     0,X     Atualiza número de vezes
FE38        B6 204B             LDAB    MEMA    que o ucp se comunicou
FE3B        7C 205B             INC     PRTNMI  o ucs.
FE3E        0F                  SEI
FE3E        F7 4402             STAB    PDB1    Responde ao ucp.
FE42        FF 201A             STX     MEMJ    Salva indexador.
FE45        CE 0000     REFAZ   LDX     # $0000 Aciona contador de tem-
FE48        08          NOVO    INX             po na MC.
FE49        8C FFFF             CPX     # $FFFF
FE4C        27 C2               BEQ     OPERA1
FE4E        F6 4800             LDAB    PDA2    Obtem codigo.
FE51        C4 0F               ANDB    # $0F
FE53        C1 0F               CMPB    # $0F
FE55        27 F1               BEQ     NOVO
FE57        7F 4402             CLR     PDB1
FE5A        FF 201C             STX     MEMY
FE5D        FE 201A             LDX     MEMJ    Recupera indexador.
FE60        7F 205B             CLR     PRTNMI
FE63        0E                  CLI
FE64        A6 20               LDAA    20,X
FE66        B7 2061             STAA    MEMG
FE69        A6 28               LDAA    28,X
FE6B        B7 2062             STAA    MEMG+1
FE6E        FE 201C             LDX     MEMY
FE71        BC 2061             CPX     MEMG
FE74        2B 0F               BMI     SOMA
FE76        27 0D               BEQ     SOMA
FE78        FE 201A     ARMAZ   LDX     MENJ    Atualiza maior tempo do
FE7B        B6 201C             LDAA    MEMY    ucp na MC.
FE7E        A7 20               STAA    20,X
FE80        B6 201D             LDAA    MEMY+1
FE83        A7 28               STAA    28,X
```

```
FE85    FE 201A   SOMA   LDX    MEMJ     Atualiza tempo total
FE88    B6 201D          LDAA   MEMY+1   do cup na MC.
FE8B    AB 38            DDA    38,X
FE8D    A7 38            STAA   38,X
FE8F    B6 201C          LDAA   MEMY
FE92    A9 30            ADCA   30,X

FE94    A7 30            STAA   30,X
FE96    07               TPA
FE97    B7 201E          STAA   MEMW
FE9A    C5 08            BITB   # $08
FE9C    27 02            BEQ    END
FE9E    20 27            BRA    CONTA
FEA0    86 01     END    LDAA   # $01    Decodifica codigo.
FEA2    B7 204D          STAA   MEM2
FEA5    4F               CLRA
FEA6    C4 07            ANDB   # $07
FEA8    11        REP    CBA
FEA9    27 06            BEQ    IRQ
FEAB    4C               INCA
FEAC    78 204D          ASL    MEM2
FEAF    20 F7            BRA    REP
FEB1    F6 204D   IRQ    LDAB   MEM2     Sinaliza ucp corres-
FEB4    F7 4802          STAB   PDB2     pondente ao codigo.
FEB7    7F 4802          CLR    PDB2
FEBA    E6 00     TST0   LDAB   0,X      Verifica se numero de
FEBC    C1 FF            CMPB   # $FF    ucs atingiu FF.
FEC0    B6 201E   TST1   LDAA   MEMW     Verifica se tempo total
FEC3    06               TAP             na MC e maior que FFFF.
FEC4    25 28            BCS    OPERA2
FEC6    39               RTS
FEC7    6C 0A     CONTA  INC    0A,X     Atualiza numero de aces-
FEC9    20 EF            BRA    TST0     sos a MC.

FECB    B6 2056   MSG5   LDAA   PRTMSG
FECE    27 01            BEQ    FORA
```

```
FED0    39                  RTS
FED1    FF 2057    FORA     STX     TEMP        Atualiza parametros
FED4    CE FADD             LDX     #$FADD      para ativação da ta-
FED7    FF 2052             STX     INDICE      refa video.
FEDA    7F 205A             CLR     MODO
FEDD    FE 201A             LDX     MEMJ
FEE0    86 B1               LDAA    #$B1
FEE2    B7 5000             STAA    ACIACS      Prog. ACIA.
FEE5    B6 2058             LDAA    TEMP+1
FEE8    8B 2D               ADDA    #$2D        Obtem num. do ucp.
FEEA    B7 5001             STAA    ACIADA
FEED    39                  RTS

FEEE    B6 2056    OPERA2   LDAA    PRTMSG
FEF1    26 21               BNE     RT
FEF3    B6 201B             LDAA    MEMJ+1      Atualiza parametros
FEF6    8B 2D               ADDA    #$2D        para ativação da ta
FEF8    B7 205A             STAA    NMIC        refa video.
FEFB    CE FA9F             LDX     #$FA9F
FEFE    FF 2052             STX     INDICE
FF01    CE F9AC             LDX     #$F9AC
FF04    FF 2050             STX     LINHA
FF07    FE 201A             LDA     MEMJ
FF0A    86 B1               LDAA    #$B1
FF0C    B7 5000             STAA    ACIACS      Prog. ACIA.
FF0F    86 0A               LDAA    #'L/F
FF11    B7 5001             STAA    ACIADA      Lança L/F.
FF14    39          RT      RTS
```

B I B L I O G R A F I A

1. RUSSO, P. M. etc. "Microprocessors in Consumer Products", Proceeding of IEEE, Vol. 66, Nº 2, Fevereiro de 1978, PP. 131 - 141.

2. SEARLE, B. C. e FREBERG, D. E. "Tutorial: Microprocessor Application in Multiple Processor Systems". Computer, Outubro de 1975, pp. 22 - 30.

3. ROSICA, G. A. "The Digital Computer Interface". Electronics Engineer's Handbook, 1a. edição, McGraw-Hill Book Company, 1975, pp. 30 - 46.

4. SATYANARAYANAN, M. "Commercial Multiprocessing Systems" Computer, maio de 1980, pp. 75 - 96.

5. FULLER, S. H. etc. "Multi-Microprocessors: An Overview and Working Example", Proceeding of IEEE, Vol. 66, Nº 2, fevereiro de 1978, pp. 216 - 228.

6. STUCKENBERG, Hans J. "CAMAC for Newcomers", CAMAC Bulletin Issure Nº 13, setembro de 1975, Supplement A.(A Publication of the esone comittee).

7. ESLON Jr, PHILIP H. "MUltiprocessor Organization - A Survey". ACM - Computing Surveys, Vol. 9, Nº 1, março-1977.

8. ROTHLISBERGER, H. "A Standard bus for Multiprocessor Architecture" - Third EUROMICRO symposium on microproces - sing and microprograming - 3 a 6 de outubro, 1977 - pp. 23 - 34.

9. CAVALCANTI, J. H. F. etc. "Multiprocessador para aplicações em tempo real, Anais do 1º SICOP, Rio de Janeiro, 1981, pp. 207 - 209.

10. M6800 Microprocessor Applications Manual - MOTOROLA,1975.

11. S6800 Microprocessing Family Selection Guide, A.M.I.6800, 1976.

12. SIGNETICS BIPOLAR & MOS MEMORY - Data Manual, 1977.

13. NATIONAL SEMICONDUCTOR- MEMORY Databook - 1977,pp.22 - 25.

14. NATIONAL SEMICONDUCTOR- MEMORY Databook - 1978-pp.17 - 21.

15. NATIONAL SEMICONDUCTOR - LINEAR Databook - 1978.

16. "The TTL Data book for Design Engineers" - Texas Instruments, 2a. edição, 1976.

17. NATIONAL SEMICONDUCTOR - CMOS Databook, 1978.

18. Holt, R. C., etc. "Structure Concurrent Programming with Operating Systems Applications", Addison Wesley Publishing Company, Toronto, Canadá, 1979.

19. GUIMARÃES, C. Cardoso. "Princípios de Sistemas Operacionais", Editora Campus Ltda, Rio de Janeiro, 1980.

20. CAVALCANTI, J. H. F., Deep, G. S. "Um Kernel para Controle de Processos Industriais em Tempo Real usando Micro - computadores". Anais do VII Seminário Integrado de SOFTWARE e HARDWARE, Campinas - SP, 1980.

21. MCM6830L7 MIKBUG/MINIBUG ROM - MOTOROLA Semiconductor Products Inc. 1975.

22. CAVALCANTI, J. H. F., Deep, G. S., ALVES, R. N. C, "Sistema de Desenvolvimento de SOFTWARE de Microprocessadores para Aplicações em Tempo Real", Panel 81 EXPODATA , Buenos Aires - Argentina.