



DEPARTAMENTO DE
ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

**SISTEMA DISTRIBUIDO
DE RECOLHA DE DADOS
PARA CONTROLO DE PRODUÇÃO
EM AMBIENTES INDUSTRIAIS**

UNIVERSIDADE DO PORTO
Faculdade de Engenharia
BIBLIOTECA

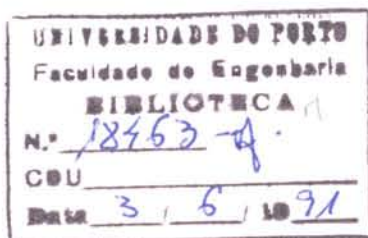
FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Rua dos Bragas, 4099 Porto Codex - PORTUGAL.
Rua dos Bragas, 4099 Porto Codex - PORTUGAL.
Rua dos Bragas, 4099 Porto Codex - PORTUGAL.
Rua dos Bragas, 4099 Porto Codex - PORTUGAL.
Rua dos Bragas, 4099 Porto Codex - PORTUGAL.

Faculdade de Engenharia da Universidade do Porto

Departamento de Engenharia Electrotécnica e dos Computadores

**SISTEMA DISTRIBUIDO
DE RECOLHA DE DADOS
PARA CONTROLO DE PRODUÇÃO
EM AMBIENTES INDUSTRIAIS**



N.º. 6420

TESE DE MESTRADO

681.31043/
SOU/STO

José Carlos Caldeira Pinto de Sousa

1991

043 M
S 715

AGRADECIMENTOS

Gostaria de expressar o meu profundo agradecimento a um conjunto de pessoas que, de uma forma ou de outra, tornaram possível este trabalho.

- Ao Prof. Joaquim Borges Gouveia pelo "empurrão" inicial e pela confiança.
- Ao Prof. José Manuel Mendonça pela sua orientação e encorajamento, e pelo seu empenhamento na elaboração desta Tese.
- Ao Eng. Nelson Castro Neves pela sua grande colaboração e apoio.
- A todos os meus colegas do projecto Automatização Industrial e Gestão de Energia do Inesc - Porto pelas ajudas e sugestões e pelo excelente ambiente de trabalho que me proporcionaram.
- Ao Prof. Eurico Moraes Carrapatoso e ao Prof. Jorge Pinho de Sousa pela colaboração na elaboração do texto.
- À minha mãe e irmã pela sua presença e pela força com que sempre me rodearam.
- À minha mulher, Isabel, pelo imenso carinho, paciência e compreensão.
- Finalmente, e de um modo muito especial, ao meu pai por me ter transmitido a sua força de vontade e por me ter ensinado a não complicar as coisas simples.

INDICE

Agradecimentos.....	I
Indice.....	III
Lista das figuras.....	VI
Resumo.....	XI
 Capítulo 1 - Introdução	
1.1 - Recolha automática de informação para controlo da produção: caracterização da situação actual.....	1.1
1.2 - Funções a implementar.....	1.3
1.3 - Exigências colocadas ao sistema.....	1.5
 Capítulo 2 - Arquitectura proposta	
2.1 - Descrição genérica	2.1
2.2 - Viabilidade técnico-económica	2.1
2.3 - Diagrama geral do sistema	2.2
 Capítulo 3 - Rede de comunicação	
3.1 - Especificação da rede.....	3.1
3.2 - Redes industriais disponíveis no mercado.....	3.1
3.3 - Selecção da rede	3.4
 Capítulo 4 - Metodologia e ferramentas de desenvolvimento	
4.1 - Metodologia de desenvolvimento.....	4.1
4.2 - Ferramentas de desenvolvimento.....	4.2
 Capítulo 5 - Unidades de recolha de dados	
5.1 - Funções	5.1
5.2 - Objectivos	5.1
5.3 - Análise preliminar.....	5.2
5.4 - Redefinição da arquitectura	
5.4.1 - Introdução de um nível hierárquico intermédio.....	5.4

5.4.2 - Reespecificação das unidades de recolha de dados.....	5.6
5.4.3 - Arquitecturas possíveis	5.8

Capítulo 6 - Unidades de aquisição

6.1 - Funções	6.1
6.2 - Objectivos	6.2
6.3 - Análise estruturada	
6.3.1 - Análise de tarefas (Diagrama de Contexto).....	6.2
6.3.2 - Diagramas de Fluxo de Informação	6.4
6.3.3 - Mini-especificações e Dicionário de Informação....	6.11
6.4 - Projecto Estruturado.....	6.13
6.4.1 - Análise de Transacções e Análise de Transformadas.....	6.13
6.4.2 - Mapas de Estrutura	6.14

Capítulo 7 - Unidades concentradoras

7.1 - Funções	7.1
7.2 - Objectivos	7.1
7.3 - Análise Estruturada	
7.3.1 - Análise de Tarefas.....	7.2
7.3.2 - Diagramas de Fluxo de Informação	7.5
7.3.3 - Mini-especificações.....	7.11
7.4 - Projecto Estruturado	
7.4.1 - Análise de Transacções e Análise de Transformadas....	7.11
7.4.2 - Mapas de Estrutura	7.13

Capítulo 8 - Rede própria

8.1 - Introdução.....	8.1
8.2 - Características.....	8.1
8.3 - Serviços a implementar na rede (nível aplicação).....	8.2
8.4 - Protocolo de ligação (nível de ligação lógica)	8.6
8.5 - Nível físico	8.9

Capítulo 9 - Unidade central

9.1 - Funções	9.1
9.2 - Objectivos	9.2
9.3 - Análise Estruturada	
9.3.1 - Análise de Tarefas.....	9.2
9.3.2 - Estrutura básica da informação na base de dados...	9.4
9.3.3 - Diagramas de Fluxo de Informação	9.5

9.3.4 - Mini-especificações.....	9.10
9.4 - Projecto Estruturado	
9.4.1 - Mapas de Estrutura	9.12

Capítulo 10 - Implementação e testes

10.1 - Implementação	10.1
10.1.1 - Unidades de aquisição	
10.1.1.1 - Hardware	10.2
10.1.1.2 - Software.....	10.6
10.1.2 - Unidades concentradoras	
10.1.2.1 - Hardware.....	10.7
10.1.2.2 - Software	
10.1.2.2.1 - Implementação.....	10.12
10.1.2.2.2 - Integridade da informação.....	10.16
10.1.3 - Unidade central	
10.1.3.1 - Hardware.....	10.18
10.1.3.2 - Software	
10.1.3.2.1 - O sistema operativo	10.18
10.1.3.2.2 - O sistema gestor de base de dados	10.19
10.1.3.2.3 - O programa de controlo da produção.....	10.20
10.2 - Testes	10.27
10.2.1 - Unidades de aquisição.....	10.28
10.2.2 - Rede própria.....	10.28
10.2.3 - Unidades concentradoras	10.29
10.2.4 - Unidade central.....	10.29

Capítulo 11 - Conclusões e evolução futura

11.1 - Conclusões	11.1
11.2 - Evolução futura	11.4

Referências e bibliografia.....	R&B.1
---------------------------------	-------

Apêndices

Apêndice A - Descrição da arquitectura BITBUS	A1
Apêndice B - Descrição do IDCX51	B1
Apêndice C - Descrição do microcontrolador 8044	C1
Apêndice D - Pormenores físicos de implementação	D1

LISTA DE FIGURAS

Figura 2.1 - Diagrama geral do sistema	2.3
Figura 5.1 - Unidade de recolha de dados: Diagrama de Contexto.....	5.3
Figura 5.2 - Architecturas possíveis I.....	5.8
Figura 5.3 - Architecturas possíveis II.....	5.9
Figura 5.4 - Architecturas possíveis III.....	5.10
Figura 5.5 - Diagrama geral da nova arquitectura do sistema.....	5.11
Figura 6.1 - Unidade de aquisição: Diagrama de Transição de Estados.....	6.3
Figura 6.2 - Unidade de aquisição: DFI de ordem zero	6.5
Figura 6.3 - Inicialização: DFI de ordem um	6.7
Figura 6.4 - Inicialização: DFI de ordem dois.....	6.8
Figura 6.5 - Monitorização: DFI de ordem um	6.9
Figura 6.6 - Comunicação: DFI de ordem um.....	6.10
Figura 6.7 - Reset do contador parcial: DFI de ordem um.....	6.10
Figura 6.8 - Mini-especificação da tarefa "Monitorização".....	6.12
Figura 6.9 - Extrato do Dicionário de Informação.....	6.13
Figura 6.10 - Unidade de aquisição: ME de ordem zero.....	6.15
Figura 6.11 - Inicialização: ME de ordem um.....	6.16
Figura 6.12 - Monitorização: ME de ordem um.....	6.17
Figura 6.13 - Comunicação: ME de ordem um.....	6.18

Figura 6.14 - Reset do contador parcial: ME de ordem um.....	6.19
Figura 7.1 - Unidade concentradora: Diagrama de Transição de Estados.....	7.3
Figura 7.2 - Unidade concentradora: DFI de ordem zero.....	7.5
Figura 7.3 - Inicialização: DFI de ordem um.....	7.7
Figura 7.4 - Varrimento: DFI de ordem um.....	7.8
Figura 7.5 - Varrimento: DFI de ordem dois.....	7.9
Figura 7.6 - Comunicação BITBUS: DFI de ordem um.....	7.10
Figura 7.7 - Interface com o operador: DFI de ordem um.....	7.11
Figura 7.8 - Mini-especificação da tarefa "Recolhe da U.A. corrente o estado e a medida de produção".....	7.12
Figura 7.9 - Unidade concentradora: ME de ordem zero.....	7.13
Figura 7.10 - Inicialização: ME de ordem um.....	7.14
Figura 7.11 - Varrimento: ME de ordem um.....	7.15
Figura 7.12 - Comunicação Bitbus: ME de ordem um.....	7.16
Figura 7.13 - Interface com o operador: ME de ordem um.....	7.17
Figura 8.1 - Serviço "Recebe status".....	8.3
Figura 8.2 - Serviço "Reset do contador".....	8.4
Figura 8.3 - Serviço "Envia bloco".....	8.4
Figura 8.4 - Serviço "Recebe bloco".....	8.5
Figura 8.5 - Códigos identificativos utilizados nos serviços da rede própria.....	8.6

Figura 8.6 - Estrutura de uma trama.....	8.9
Figura 8.7 - Serviço "recebe status": Tramas enviada e recebida	8.9
Figura 9.1 - Unidade central: Diagrama de Transição de Estados.....	9.3
Figura 9.2 - Unidade central: DFI de ordem zero.....	9.6
Figura 9.3 - Inicialização: DFI de ordem um	9.7
Figura 9.4 - Opções: DFI de ordem um.....	9.8
Figura 9.5 - Configuração/Erros: DFI de ordem dois	9.8
Figura 9.6 - Monitorização/Históricos: DFI de ordem um.....	9.9
Figura 9.7 - Actualização: DFI de ordem um	9.10
Figura 9.8 - Mini-especificação da tarefa "Actualização"	9.11
Figura 9.9 - Unidade central: ME de ordem zero	9.13
Figura 9.10 - Monitorização: ME de ordem um.....	9.14
Figura 9.11 - Históricos: ME de ordem um.....	9.15
Figura 9.12 - Selecção e visualização da informação: ME de ordem um.....	9.16
Figura 10.1 - Núcleo central da unidade de aquisição: diagrama de blocos.....	10.3
Figura 10.2 - Interface com o operador da unidade de aquisição: diagrama de blocos.....	10.4
Figura 10.3 - Interface com sinais de entrada e saída da unidade de aquisição: diagrama de blocos	10.4
Figura 10.4 - Interface com a rede própria da unidade de aquisição: diagrama de blocos.....	10.5

Figura 10.5 - Alimentação da unidade de aquisição: diagrama de blocos.....	10.6
Figura 10.6 - Mensagens enviadas para o afixador alfanumérico da unidade de aquisição	10.7
Figura 10.7 - Núcleo central da unidade concentradora: diagrama de blocos.....	10.8
Figura 10.8 - Interface com o operador da unidade concentradora: diagrama de blocos	10.9
Figura 10.9 - Interface com a rede BITBUS da unidade concentradora: diagrama de blocos	10.10
Figura 10.10 - Interface com a rede própria da unidade concentradora: diagrama de blocos.....	10.11
Figura 10.11 - Interface com o relógio de tempo real da unidade concentradora: diagrama de blocos.....	10.12
Figura 10.12 - Mensagens enviadas para o afixador alfanumérico da unidade concentradora durante uma sinalização de início de operação.....	10.14
Figura 10.13 - "Buffer" de dados de uma unidade de aquisição	10.15
Figura 10.14 - "Buffer" de transmissão de blocos.....	10.16
Figura 10.15 - Selecção da opção "Monitorização".....	10.23
Figura 10.16 - Introdução da máscara de selecção.....	10.24
Figura 10.17 - Máscara de selecção definida.....	10.25
Figura 10.18 - Visualização da informação referente às linhas seleccionadas.....	10.26
Figura 10.19 - Opção para nova selecção ou regresso ao menu anterior.....	10.27

RESUMO

A recolha automática, da planta fabril, de informações necessárias ao planeamento e controlo da produção é uma actividade essencial para maximizar os benefícios que é possível obter com este tipo de aplicações. Este trabalho tem como objectivo o desenvolvimento de um sistema distribuído de recolha de informações em ambientes industriais, vocacionado para pequenas e médias empresas, que permita recolher dos postos de trabalho as informações atrás referidas.

No capítulo 1, faz-se uma caracterização do "estado da arte" nesta área, identificando os problemas que se colocam às indústrias e a inadequação das soluções disponíveis no mercado. São ainda identificadas as funções que se pretendem implementar e as exigências a colocar ao sistema.

No capítulo 2, é apresentada uma arquitectura genérica para o sistema, e é analisada a sua viabilidade técnico-económica.

O hardware e software a utilizar no desenvolvimento depende, em grande parte, da rede de comunicação utilizada. No capítulo 3 é feito um estudo comparativo entre as várias redes de comunicação industriais disponíveis comercialmente, do qual resultou a opção por uma delas.

A anteceder o desenvolvimento do sistema são definidas, no capítulo 4, as metodologias e as ferramentas a utilizar.

No capítulo 5, são definidas as funções e objectivos das unidades de recolha de dados, a partir dos quais é feita uma análise preliminar. Dessa análise resultou uma redefinição da arquitectura do sistema surgindo agora, em substituição da unidade de recolha de dados, unidades concentradoras, unidades de aquisição e uma rede própria a desenvolver para suportar a comunicação entre elas.

Nos capítulos 6 e 7, são identificadas as funções e objectivos para as unidades de aquisição e para as unidades concentradoras. Seguidamente foram utilizadas Técnicas de Análise e Projecto Estruturado para produzir a especificação destas unidades.

No capítulo 8, é definida a rede própria, nomeadamente as suas características e o conjunto de serviços a implementar. São ainda definidos os níveis físico e de ligação lógica.

No capítulo 9, é apresentada a Especificação Estruturada da unidade central utilizando a mesma metodologia que foi usada nos capítulos 6 e 7 para as unidades de aquisição e unidades concentradoras.

No capítulo 10, é apresentada uma descrição da implementação, hardware e software, das várias unidades referidas anteriormente. São ainda apresentados os resultados de um conjunto de testes efectuados para avaliação do desempenho dos diversos componentes do sistema.

Finalmente, são apresentadas no capítulo 11 as conclusões retiradas do desenvolvimento deste sistema e é perspectivada a sua evolução futura.

CAPITULO 1

Introdução

1. INTRODUÇÃO

1.1 - Recolha automática de informação para controlo da produção - caracterização da situação actual

A crescente concorrência dos países com mão de obra barata, associada ao inevitável aumento dos níveis salariais, obriga a indústria de um país como o nosso a um esforço cada vez maior no sentido do aumento da produtividade.

Esse esforço tem como componentes fundamentais por um lado os aspectos tecnológicos, e por outro as actividades de planeamento e o controlo da produção. Enquanto que o planeamento tem como objectivo afectar as necessidades de fabrico aos recursos disponíveis, procurando maximizar o seu aproveitamento e minimizar os custos de produção, o controlo tem como função verificar o cumprimento do plano de fabrico e a validade dos parâmetros utilizados no planeamento (capacidade dos postos de trabalho, tempos médios das operações, etc.).

É fácil concluir que um fluxo eficaz de informação entre estas duas actividades, e entre elas e os postos de trabalho, é fundamental para a qualidade do funcionamento do sistema produtivo. É necessário que as directivas do planeamento, por exemplo ordens de fabrico, cheguem à produção em tempo útil, e é igualmente importante que as informações referentes a eventuais anomalias que provoquem ou possam provocar, em sentido inverso, alterações ao plano de fabrico (avarias em máquinas, falhas de matéria prima, etc.) sejam comunicadas o mais rapidamente possível ao planeamento, para que assim se possam minimizar ou mesmo anular os efeitos dessas alterações [1], [2].

Faz-se pois sentir a necessidade em ambientes industriais, de sistemas de recolha, processamento e transmissão de informação, que respondam em tempo útil e com elevada fiabilidade.

As vantagens da utilização de computadores no processamento de quantidades apreciáveis de informação levaram ao aparecimento, há já alguns anos, de pacotes de software para planeamento e controlo da produção, de uma ou outra forma integráveis com pacotes para gestão de stocks, manutenção, facturação, salários, contabilidade, etc. A partilha de informações entre os vários módulos aplicativos traz benefícios evidentes ao nível da velocidade de processamento e da fiabilidade da informação permitindo, a todos os níveis, uma melhor gestão. A melhor prova do sucesso da informática no processamento de informação industrial é o número cada vez maior de empresas que

a utiliza nas áreas contabilística, financeira, da gestão de stocks, da manutenção e do controlo estatístico de qualidade. Ao nível do planeamento e controlo da produção a evolução e aceitação de soluções informáticas não foram porém tão rápidas, pelas razões que a seguir se apresentam.

Em primeiro lugar, é difícil justificar, perante os potenciais utilizadores, fazer planeamento sem fazer controlo, i.e. sem dispôr de meios de avaliação do grau de exactidão dos cálculos e previsões efectuadas.

Por outro lado, fazer controlo da produção implica proceder à recolha de informação dos postos de trabalho, em tempo útil. Os meios tradicionais de recolha (fichas ou relatórios de produção preenchidos quer por pessoal encarregado exclusivamente dessa tarefa, quer pelos próprios operadores das máquinas) são todavia muito sujeitos a erros, quer involuntários (devido a cansaço ou desatenção), quer voluntários (por desleixo ou quando estão em jogo prémios de produção). Esta situação, não pouco vulgar, além de dificilmente controlável por parte dos responsáveis, pode acarretar graves consequências ao nível do processo de controlo. Além disso, os processos de preenchimento, recolha e introdução no sistema informático, geralmente através de teclado, dos dados das fichas de produção são caros e demorados, levando a que uma dada ocorrência seja geralmente processada com 24 a 48 horas de atraso.

Os sistemas de controlo de produção existentes no mercado, dotados de recolha automática de informação nos postos de trabalho (Barco, Picotron, Captor, System Electronics, Siemens, ABB, etc.), baseiam-se em redes de autómatos ou de unidades, dedicadas ou não, de recolha de informação. As primeiras implementações de soluções deste tipo foram realizadas na indústria automóvel e nas suas subsidiárias de componentes. São indústrias com um elevado grau de automatização e que tiram partido de tecnologias disponíveis ao nível da produção (autómatos, redes industriais, computadores, etc.) para implementar a recolha e o processamento de informação. São no entanto indústrias com características muito particulares, quer ao nível dos meios e técnicas de produção quer ao nível dos investimentos em jogo.

O tecido industrial português, e especialmente as PME's, é caracterizado por reduzidas disponibilidades financeiras o que, conjugado com a escassez de meios técnicos, limita a

implementação directa das soluções acima referidas, devido ao seu elevado custo e à complexidade da sua utilização e manutenção.

Existe pois a nível nacional, um mercado significativo para sistemas dimensionados para pequenas e médias empresas. Tais sistemas deverão:

- ser modulares, de forma a poderem ser configurados à medida das necessidades;
- fornecer a informação necessária, de um modo o mais transparente possível para os operadores dos postos de trabalhos;
- ser de fácil utilização por pessoas sem conhecimentos de informática;
- ter baixo custo, para que a sua aplicação em PME's possa ser técnica e economicamente viável.

Este mercado potencial inclui sobretudo empresas de mão de obra intensiva e com níveis de automatização relativamente baixos, casos das indústrias têxteis e de vestuário, calçado, metalomecânicas, etc.

A análise que acaba de apresentar-se explica a motivação para um trabalho de I&D direccionado para a obtenção de soluções de recolha automática de informação fabril adequadas a uma percentagem significativa das PME's no nosso país.

1.2 - Funções a implementar

Pretende-se pois desenvolver um sistema que permita fazer a recolha, a partir da planta fabril, da informação relevante para o processo de planeamento e controlo da produção (este processo será designado a partir deste momento, unicamente por gestão da produção). O primeiro passo deste esforço será definir qual é essa informação relevante.

A especificação funcional que nos propusemos fazer neste trabalho visa sobretudo o caso das PME's nacionais, resultando essencialmente de contactos com responsáveis dos sectores de gestão da produção de diversas empresas. Foi possível definir um conjunto de informações caracterizador dos postos de trabalho, do

ponto de vista da gestão da produção, nomeadamente:

- código identificador do posto de trabalho;
- código identificador do(s) operador(es);
- código identificador da operação que está a ser efectuada (código de operação ou número de ordem de fabrico);
- uma medida de produção (por exemplo, número de operações efectuadas, de unidades de produto fabricadas, de horas de trabalho das máquinas, etc.);
- código identificador do estado do posto de trabalho. Os equipamentos de produção podem ser caracterizados pelo conjunto de estados em que se podem encontrar (desligado, ligado, avariado, em manutenção, em mudança de ferramenta ou de tipo de produto, etc.).

Através do registo dos valores destas variáveis de informação, assim como da data e hora a que se verificarem eventuais alterações, é possível conhecer o estado das ordens de fabrico e verificar quais as alterações, se as houver, relativamente ao plano de fabrico. No caso de se registarem atrasos é ainda possível identificar o motivo desses atrasos (avarias, produções médias inferiores às previstas pelo plano, problemas com stocks, absentismo, etc.). Toda esta informação é vital para a tomada, em tempo útil, de decisões de re-sequenciamento ou de alteração dos planos de fabrico. Podem ainda, desta forma, obter-se informações úteis para o planeamento, como valores reais da capacidade máxima e média dos postos de trabalho, tempos médios de cada operação, etc.

É fácil observar que este conjunto de informações permite gerar indicadores a ser utilizados em várias outras áreas da gestão industrial, como por exemplo no cálculo de custos, na manutenção preventiva, no controlo horário dos operadores, na atribuição de prémios de produção, etc.

É ainda importante salientar que este conjunto de informações é razoavelmente independente do tipo de indústria, podendo ser aplicado à maior parte das empresas industriais de pequena e média dimensão, quer tenham produções contínuas quer descontínuas, como têxteis, calçado, metalomecânicas, fundição, etc.

Podemos, portanto, concluir que as principais funções do sistema a desenvolver são:

- a recolha, através de unidades de recolha de dados e com intervenção mínima dos operadores, da planta fabril da informação acima referida;
- o envio dessa informação, através de uma rede de comunicação, para uma unidade central;
- o seu processamento de forma a permitir a consulta directa ou a utilização por outros programas, nomeadamente pacotes de planeamento e controlo da produção.

1.3 - Exigências colocadas ao sistema

Tendo identificado o conjunto de funções a implementar, são agora definidas as exigências a colocar ao sistema a desenvolver, do ponto de vista funcional, técnico e económico:

- o sistema deve dispôr de unidades de recolha de dados capazes de monitorizar um conjunto de variáveis úteis à gestão da produção, como por exemplo as acima referidas, com o mínimo de intervenção por parte dos operadores. Sempre que se verificar alguma alteração no funcionamento do sistema produtivo, deve ser registada a hora a que se verificou essa alteração e o novo valor da variável afectada. Essas unidades deverão estar localizadas o mais próximo possível dos postos de trabalho e, sempre que tal seja economicamente viável, deverá existir uma por posto de trabalho por forma a evitar que o operador tenha de se deslocar quando necessitar de dialogar com o sistema;
- as unidades de recolha de dados devem estar ligadas, através de uma rede de comunicação industrial, a uma unidade central capaz de recolher de cada uma delas as informações relativas a cada posto de trabalho e de as armazenar através de um sistema gestor de base de dados;
- a unidade central deve dispôr de software que permita a consulta da informação contida na base de dados e a sua utilização por outros programas, nomeadamente de planeamento da produção, gestão da manutenção, orçamentação, gestão de stocks, etc. Esta ligação poderá

ser feita por acesso directo à base de dados, ou através de um processo intermédio de escrita da informação em ficheiros com um formato a definir;

- deve ser garantida a modularidade e expansibilidade físicas e funcionais do sistema, i.e. deve ser possível, de uma forma simples e económica, expandir o sistema quer em número de unidades de recolha de dados quer em número de funções disponíveis. É importante garantir que o aumento da quantidade de informação gerada pela expansão do sistema não acarrete sobrecargas da rede de comunicação ou tempos de processamento elevados, de modo a poderem ser garantidas as características de desempenho em tempo útil;
- o sistema deve ser fiável e de baixo custo. É de salientar que, na maior parte das situações práticas, as unidades de recolha de dados são os componentes que maior peso vão ter no custo global do sistema, pelo facto do seu número ser naturalmente elevado. É portanto no seu desenvolvimento que se deverão concentrar os principais esforços de redução de custos.

CAPITULO 2

Arquitectura propuesta

2. ARQUITECTURA PROPOSTA

2.1 - Descrição genérica

Com base na especificação funcional apresentada no capítulo anterior é possível fazer uma descrição genérica do sistema a desenvolver:

- sistema distribuído, baseado em unidades remotas simples de recolha de dados;
- rede de comunicação adequada a ambiente industrial;
- unidade central PC-compatível com sistema operativo MS-DOS e/ou XENIX/UNIX 386;
- interfaces com os operadores de simples compreensão e utilização, quer na unidade central quer nas unidades de recolha de dados;
- expansão das unidades de recolha de dados simples e modular.

2.2 - Viabilidade técnico-económica

Sendo o baixo custo um dos principais objectivos a atingir, é importante que sejam identificados os elementos do sistema que mais vão influenciar esse custo e que o seu desenvolvimento tenha em conta esse importante factor.

Na maioria dos casos a principal componente do custo do sistema será o custo das unidades de recolha de dados, necessariamente em grande número. Com efeito, pretendendo colocar uma unidade destas por posto de trabalho, o seu número poderá atingir facilmente as muitas dezenas ou mesmo centenas.

Prosseguindo por ordem decrescente de influência no preço final do sistema, segue-se a rede de comunicação. No custo da rede deve incluir-se o custo da instalação (cabos, acessórios, etc.), do hardware e software de ligação da rede à unidade central e às unidades de recolha de dados, e os custos de manutenção e expansão da rede.

No que respeita à unidade central o problema é fácil de equacionar. Existindo apenas uma unidade central no sistema e considerando os preços actuais para o tipo de equipamento

proposto e a sua provável e contínua descida, poderemos dizer que este elemento não irá, na maior parte dos casos, influenciar significativamente o custo final.

Finalmente, entra o custo do software a adquirir a terceiros, nomeadamente o do sistema gestor de base de dados. Deve ser analisada a razão custo do software/custo de desenvolvimento, pois pode ser vantajosa a utilização de software mais dispendioso mas que possua linguagens de programação e ferramentas poderosas que acelerem o desenvolvimento de aplicações e facilitem a sua manutenção.

Tornou-se pois necessário acompanhar o projecto e o desenvolvimento, ou especificação, de todos os elementos do sistema com uma análise rigorosa da razão custo/desempenho.

2.3 - Diagrama geral do sistema

Na figura 2.1 apresenta-se um diagrama do blocos ilustrativo da estrutura preliminar do sistema.

Existe uma unidade de recolha de dados em cada posto de trabalho encarregada de recolher as informações necessárias à gestão da produção. Essa unidade dispõe de duas interfaces: uma para o operador, constituída por um teclado e um afixador alfanumérico, e outra com o posto de trabalho que implementa a recolha automática de informações. As informações são posteriormente enviadas, através de uma rede de comunicação industrial, para a unidade central onde são armazenadas por um sistema gestor de base de dados para posterior consulta local ou remota.

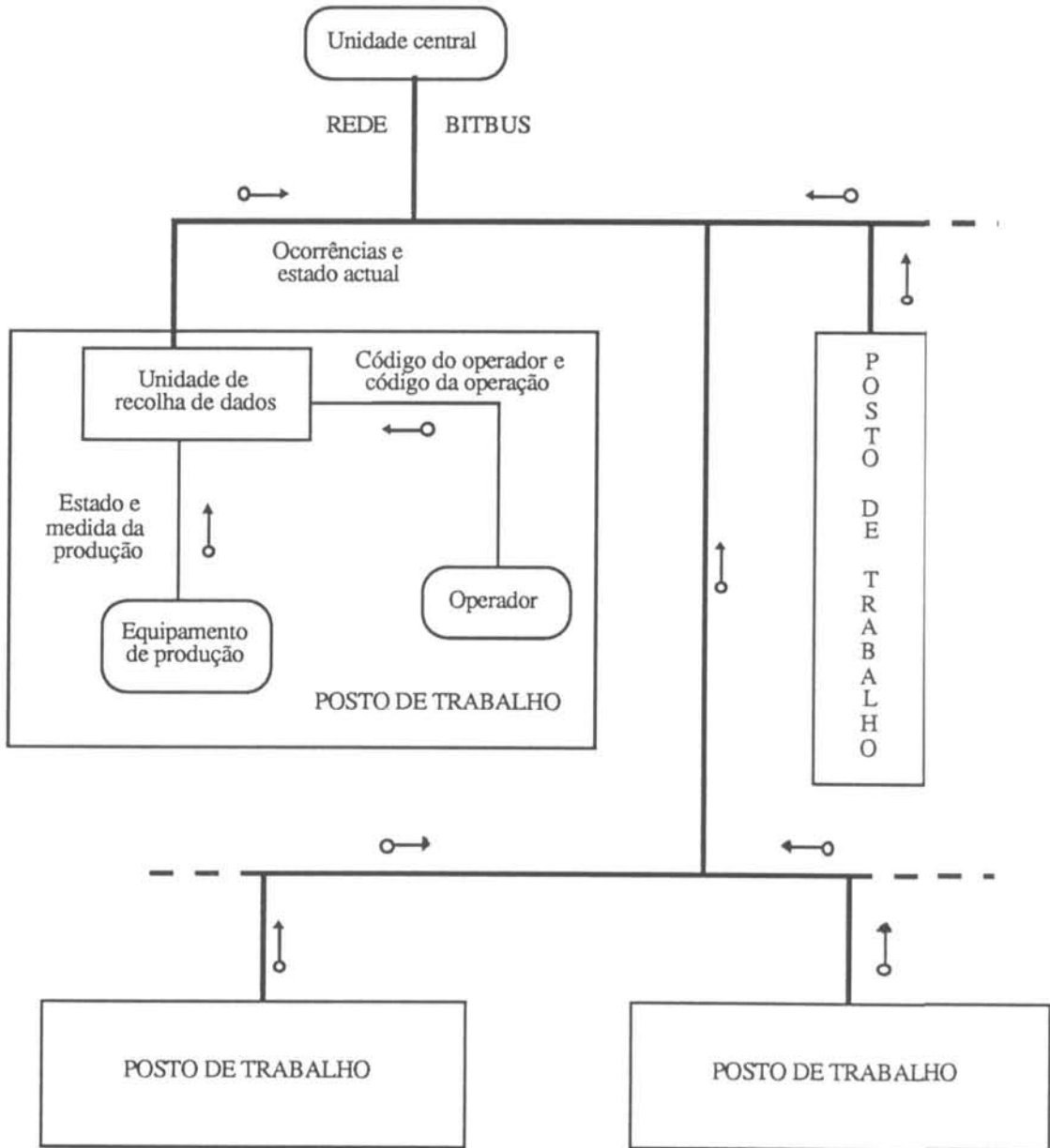


Figura 2.1 - Diagrama geral do sistema

CAPITULO 3

Rede de comunicação

3. REDE DE COMUNICAÇÃO

3.1 - Especificação da rede

Um dos primeiros elementos do sistema a definir foi a rede de comunicação, pois da sua escolha dependia o conjunto de produtos de hardware e software disponíveis no mercado cuja utilização poderia ser considerada. Apresenta-se a seguir o conjunto de características que essa rede deve possuir [3], [4]:

- capacidade de garantir as suas características de funcionamento mesmo em ambientes adversos, como o são grande parte das instalações fabris, com elevados níveis de ruído electromagnético, presença de cabos de alta tensão, temperaturas elevadas, etc.;
- elevada fiabilidade;
- baixo custo de instalação e manutenção;
- disponibilidade no mercado de hardware e software, a preços acessíveis, que facilitem o desenvolvimento e manutenção de soluções;
- disponibilidade de informação detalhada sobre todos os seus aspectos constituintes, salvaguardando assim a hipótese do desenvolvimento de hardware e software específicos compatíveis com a rede;
- adequação ao tráfego de blocos de informação de pequeno e médio comprimento, dado ser relativamente pequena a quantidade de informação a recolher de cada posto de trabalho;
- garantia de atrasos limitados no atendimento de serviços solicitados à rede.

3.2 - Redes industriais disponíveis no mercado

Foi efectuado um levantamento das redes de comunicação industriais existentes no mercado, com o intuito de seleccionar a que melhor se adaptava às exigências e necessidades do sistema.

Esse estudo debruçou-se sobre um conjunto de redes, cujas características mais relevantes são resumidas a seguir.

Redes de campo ("field buses").

O seu suporte físico é geralmente um cabo de um ou dois pares de condutores de cobre entrançados, existindo no entanto algumas redes que utilizam cabo coaxial ou fibra óptica.

São redes em "bus" que suportam até algumas centenas de nós (tipicamente entre 200 e 250 nós), com velocidades de comunicação entre os 9600 bps e os 1.2 ou 2.4 Mbps, e que podem estender-se por distâncias da ordem de alguns quilómetros para as velocidades mais baixas ou da ordem de algumas dezenas de metros para as velocidades mais altas, podendo, no entanto, recorrer-se à utilização de repetidores para aumentar essas distâncias. É possível acrescentar nós à rede sem ter de a reinicializar.

São geralmente redes tipo mestre-escravo ("master-slave") em que existe uma unidade, o mestre, responsável pelo controlo do acesso das outras unidades (escravos) à rede. Só o mestre pode iniciar uma troca de informação enviando para isso uma mensagem ao escravo que deve, se for caso disso, enviar uma mensagem de resposta. Não é possível um escravo iniciar uma conversação, nem é possível a troca directa de informação entre escravos. Este mecanismo de controlo de acesso permite assegurar tempos máximos de serviço. Uma desvantagem imediata resulta do facto de uma avaria no mestre inibir o funcionamento de toda a rede.

Encontram-se nesta categoria a maior parte das redes de autómatos de quase todos os fabricantes (Siemens, AEG, Telemecanique, Saia, etc.) e a rede BITBUS da Intel.

Ethernet

A Ethernet é uma rede com cerca de dez anos e já com provas dadas, existindo uma quantidade apreciável de hardware e software disponível no mercado para a maioria dos fabricantes de equipamentos informáticos. A nível industrial, se incluirmos os sistemas informáticos ligados às áreas de gestão, é a rede mais instalada.

É suportada por uma arquitectura em "bus" que utiliza normalmente cabo coaxial como suporte físico, existindo também versões para fibra óptica e para condutores de cobre. Existe na versão banda de base ("baseband") e banda de canal ou modulada ("broadband"), esta somente em cabo coaxial ou fibra óptica. Neste último caso, pode suportar simultaneamente vários canais

independentes nos quais é possível transmitir desde informação de supervisão e controlo até ficheiros de estações de CAD/CAM e sinais audio e vídeo. A velocidade de transmissão utilizada é de 10 Mbps para cabo coaxial.

O acesso à rede é múltiplo, ou seja qualquer nó pode enviar pacotes para a rede. Cada nó é identificado por um endereço o que permite que, quando um nó transmite, os outros nós possam receber o sinal. Os pacotes são iniciados por um preâmbulo onde se encontra o endereço de destino da mensagem. Podem-se enviar mensagens para todos os nós ("broadcast") ou para alguns ("multicast"), e em particular para um só.

Uma vez que todos os nós podem enviar pacotes para a rede é necessário um método de controlo do acesso ao meio. Esse método denomina-se "Carrier Sense Multiple Access / Collision Detection" (CSMA/CD) e baseia-se num mecanismo existente em todos os nós que "escuta" a rede, inibindo a transmissão se esta estiver a ser utilizada por outro nó. Caso contrário o nó envia o pacote ficando, simultaneamente, a "escutar" a rede. Se a informação recebida for diferente da enviada então é porque se verificou uma colisão com outro pacote. Neste caso, o pacote será retransmitido ao fim de um intervalo de tempo cuja duração é probabilística.

Estamos assim perante uma rede na qual, ao contrário do caso anterior, não há controlo centralizado de acesso mas sim um controlo distribuído, resultando daí que a avaria de um nó não inibe o funcionamento dos restantes nós da rede. Devido ao facto do acesso à rede ser probabilístico e de poderem ocorrer colisões, não se podem garantir tempos máximos de serviço, o que é uma desvantagem em aplicações de "tempo real" relativamente ao método mestre-escravo no qual o acesso à rede é determinístico.

MAP (Manufacturing Automation Protocol)

A arquitectura definida por este protocolo é semelhante, no que se refere à topologia e aos suportes físicos, a uma rede Ethernet utilizando cabo coaxial ou fibra óptica.

A principal diferença reside no método de acesso à rede. Tal como no caso anterior todos os nós podem transmitir para a rede, mas neste caso não o podem fazer em qualquer instante. O método de acesso, denominado "token pass", é suportado por um pacote especial, o "token", que pode ser considerado como uma chave de acesso à rede e qualquer nó só poderá transmitir quando estiver na posse desse pacote. Quando a rede é inicializada um dos nós gera o "token" e, se tiver alguma informação a transmitir, fá-lo-á. Quando acabar a transmissão, ou quando o seu tempo de retenção do "token" se esgotar, passa-o ao nó seguinte que executa as mesmas

operações, e assim sucessivamente até que o "token" regresse ao nó inicial onde tudo se repete. É de referir que o termo "nó seguinte" não se refere ao nó que está fisicamente contíguo ao actual mas sim àquele cujo endereço aparece como sendo o do "nó seguinte", resultando assim um anel lógico que interliga todos os nós activos da rede.

Com este método é eliminada uma das desvantagens da rede Ethernet, pois o acesso determinístico à rede permite garantir tempos máximos de serviço. No entanto, os tempos máximos de serviço obtidos com o MAP não são por vezes compatíveis com as exigências de tempo real de algumas aplicações devido à sobrecarga de processamento imposta pela implementação dos sete níveis do modelo OSI [3]. Para este tipo de aplicações foi desenvolvida uma rede vulgarmente conhecida por MiniMap que, utilizando o mesmo suporte físico, implementa somente os níveis um e dois do referido modelo. Conseguem-se assim tempos de serviço muito menores embora com o inconveniente de obrigar ao desenvolvimento de software para implementar, quando necessários, alguns dos serviços fornecidos pelos outros níveis do modelo OSI.

A utilização do MAP está ainda limitada a um número restrito de aplicações. Casos típicos de aplicação são soluções de automatização e informatização na indústria automóvel, na qual é necessário transferir grandes quantidades de informação de diversos tipos, como imagens vídeo, ficheiros resultantes de sistemas de CAD/CAM, etc.

O elevado custo, associado ao ainda reduzido número de produtos comercialmente disponíveis, explica a sua fraca penetração no mercado.

3.3 - Selecção da rede

Conjugando esta breve caracterização das redes disponíveis no mercado com a especificação apresentada no início deste capítulo, é possível passar à selecção da que melhor se adapta às necessidades do nosso sistema.

Qualquer das redes indicadas garante as suas características em ambientes industriais. As que apresentam melhores características de velocidade de transmissão e imunidade ao ruído são as que utilizam fibra óptica como meio de transmissão, mas são também as de custo mais elevado. A redução de preço que se consegue ao passar para o cabo coaxial ou para os condutores de cobre entrançados é acompanhada por uma redução no comprimento

máximo da rede e na velocidade de transmissão. Este problema pode porém ser ultrapassado através da utilização de repetidores. Tomando como referência a rede BITBUS, que utiliza um suporte físico RS-485, usando a velocidade de transmissão mais baixa, 62.5 Kbps, conseguem-se atingir distâncias de 1200 m sem repetidores com apenas um par entrançado ou, utilizando a velocidade de 375 Kbps, cobrir distâncias da ordem dos 900 m utilizando dois repetidores (que podem ser os próprios nós da rede) e um cabo com dois pares entrançados. Dado esta última solução cobrir perfeitamente as necessidades de transferência de informação especificadas no ponto 3.1 e ser bastante menos dispendiosa do que as alternativas, quer ao nível do preço dos cabos quer do preço do hardware de interface entre os nós e a rede, optar-se-á pela utilização de pares de condutores de cobre entrançados como suporte físico da rede.

Como foi já referido atrás, as redes mestre-escravo podem ter problemas de fiabilidade que advêm do facto de a rede ficar fora de serviço caso o mestre avarie. Contudo, se um escravo avariar isso não se reflecte no resto da rede, podendo ser retirado sem perturbar o normal funcionamento desta. A fiabilidade da rede depende pois principalmente da fiabilidade do mestre. Quer a Ethernet quer o MAP têm neste aspecto vantagens em relação às redes de campo. No entanto, no caso do sistema se basear em autómatos programáveis, componentes caracterizados por uma elevada fiabilidade em ambientes industriais, este problema é fortemente atenuado. Em casos críticos pode justificar-se, para o mestre, a duplicação do equipamento.

O custo da rede é um aspecto essencial e que pesou muito na decisão final. E quanto a este aspecto não há dúvidas em relação aos desníveis que existem entre uma rede de campo e a Ethernet, e entre esta e o MAP. Dado o desempenho das redes de campo se mostrar suficiente para as necessidades especificadas optar-se-á definitivamente por uma destas redes.

A disponibilidade de produtos de software para as redes de campo não constitui problema, porque quer os fabricantes de equipamentos que possuem redes próprias quer os que utilizam o BITBUS oferecem um conjunto de produtos que facilitam o desenvolvimento de aplicações. Entre estes incluem-se monitores que suportam funções como transferência de blocos de informação, alteração dos parâmetros da rede, etc., e bibliotecas de funções para várias linguagens de programação (C, Pascal, Basic, etc.) que podem ser utilizadas no desenvolvimento de programas customizados. No que respeita ao hardware, a escolha de uma rede

própria impõe uma limitação ao conjunto de produtos que o fabricante oferece ou venha a lançar no mercado, dado a interligação de sistemas/redes de fabricantes distintos ser uma solução impossível ou com um custo exorbitante. No caso do BITBUS o cenário é mais favorável: existem vários fabricantes com produtos para esta rede que se completam, cobrindo as necessidades de um vasto leque de aplicações. Entre esses fabricantes podemos destacar a Intel, Data Translation, Phoenix Contact, System Electronics, Analog Devices, Digital, GMF Robotics, Westinghouse, AEG, Honeywell e Toshiba.

Um outro ponto da maior relevância tem a ver com o tipo e quantidade de informação a movimentar. As redes de campo são vocacionadas para o tráfego de pacotes de pequena ou média dimensão, sendo geralmente pequena a quantidade de informação a enviar ou a receber para/de cada nó. Por exemplo, uma operação típica numa rede de autómatos será ler uma entrada digital ou analógica de uma unidade remota, processar a informação e enviá-la para uma saída, digital ou analógica, dessa mesma unidade remota ou de outra qualquer. A informação a enviar através da rede seria um bit, ou o byte no qual esse bit se encontra, no caso da entrada ou saída digitais, ou normalmente dois bytes no caso da entrada ou saída analógicas.

O comprimento, em bytes, da estrutura de informação associada a um posto de trabalho típico pode ser aproximadamente calculado em função dos comprimentos médios dos seus elementos de informação constituintes. Se admitirmos que os códigos de operador e de operação serão armazenados em ASCII, utilizando um byte para codificar cada carácter, e que necessitamos de seis caracteres para identificar cada um destes códigos, daqui resulta um comprimento de doze bytes. Se adicionarmos quatro bytes para codificar a medida de produção e um byte para o estado do posto de trabalho, essa estrutura de informação terá um comprimento típico de 17 bytes. Este comprimento pode, no entanto, variar bastante. Existem indústrias que possuem gamas de fabrico muito grandes pelo que os seus códigos de operação ou de ordem de fabrico possuem dez, quinze ou mesmo vinte caracteres. É pois necessária flexibilidade no comprimento das mensagens.

Tomando uma vez mais como referência o BITBUS o comprimento das mensagens é variável, podendo ser igual ou inferior a uma variável do sistema programável pelo utilizador e que pode tomar valores limitados apenas pela quantidade de RAM interna do microcontrolador disponível para o envio de mensagens. Um valor habitual para esta variável é 20 bytes. A adequação do

comprimento das mensagens a cada caso específico permite obviamente melhorar a eficiência da transferência de informação.

Finalmente, segue-se a análise de um ponto crucial da estratégia de desenvolvimento. Se o hardware disponível no mercado não for adequado, técnica e economicamente, ao sistema especificado haverá necessidade de o desenvolver. Para isso é crucial dispôr de informação detalhada e completa sobre todos os aspectos constituintes da rede de comunicações. Os fabricantes que usam redes próprias têm todavia uma enorme e compreensível relutância em fornecer elementos detalhados sobre as mesmas, o que torna praticamente impossível o desenvolvimento, por terceiros, de produtos para essas redes. Com o BITBUS tal não acontece, sendo possível dispôr de todos os elementos necessários ao desenvolvimento de hardware e software para a rede.

Além dos motivos apresentados ao longo deste ponto o BITBUS apresenta mais duas vantagens que advêm da utilização, nos nós da rede, do microcontrolador 8044 da Intel, a saber:

- todo o software de comunicação da rede está já disponível em firmware no microcontrolador;
- o microcontrolador dispõe de um sistema operativo multitarefa, que permite ter sete tarefas a correr simultaneamente, com um conjunto de funções como temporizações, interrupções por hardware, atrasos, etc., que o tornam uma ferramenta poderosa em aplicações onde seja necessária capacidade de processamento em tempo real nos nós da rede.

A discussão que acaba de ser apresentada justifica a escolha do BITBUS como rede utilizada para interligar a unidade central e as unidades de recolha de dados. Apresenta-se no apêndice A uma descrição mais detalhada de alguns aspectos relevantes desta rede.

CAPITULO 4

Metodologia de desenvolvimento

4. METODOLOGIA E FERRAMENTAS DE DESENVOLVIMENTO

Identificadas as funções, objectivos e arquitectura do sistema, tornou-se necessário definir uma metodologia de desenvolvimento e as ferramentas a utilizar.

4.1 - Metodologia de desenvolvimento

Como foi já realçado anteriormente, as unidades de recolha de dados são os componentes que maior peso vão ter no custo da solução final. Em consequência, foi sobre estas unidades que incidiu o maior esforço de customização. Nesse sentido foi definido um conjunto de regras estratégicas que se apresenta como metodologia de desenvolvimento:

- a partir da especificação das funções e objectivos destas unidades foi definida uma configuração mínima de hardware que, satisfazendo essa especificação e tendo como objectivo fundamental a minimização do custo, deixasse o sistema aberto a futuras expansões ou reconfigurações. No caso de existir hardware disponível comercialmente cumprindo, quer técnica quer economicamente, a especificação feita ele seria utilizado. No caso contrário o hardware teria de ser desenvolvido;
- definido o hardware passou-se ao desenvolvimento do software, sempre maximizando a utilização dos recursos disponíveis, e tendo em vista a implementação de outras funções além das especificadas desde que tal não implicasse acréscimo de hardware;
- logo que foi definido o conjunto de funções a implementar nas unidades de recolha de dados, pôde iniciar-se o desenvolvimento do software da unidade central, o qual pretendeu igualmente tirar o máximo partido dos recursos disponíveis, quer ao nível das unidades de recolha de dados quer ao nível dos serviços oferecidos pela rede BITBUS;
- o acréscimo de qualquer função acarretando um aumento no custo do hardware do sistema, esteve sempre condicionado a um estudo de viabilidade técnico-económica.

4.2 - Ferramentas de desenvolvimento

A escolha de ferramentas de desenvolvimento adequadas é, no que respeita ao desenvolvimento de software, uma questão de primordial importância.

Este é um sistema com alguma complexidade, no qual é possível identificar dois níveis - unidade central e unidades de recolha de dados - nos quais podem existir várias tarefas em execução concorrente partilhando recursos e informação. Partir da descrição genérica do sistema e chegar a um modelo adequado de representação utilizando ferramentas de desenvolvimento convencionais é pois uma tarefa difícil e sujeita a erros.

A modelização de sistemas utilizando Técnicas de Análise e Projecto Estruturado (TAPE's) pode facilitar esta tarefa. Baseando-se em duas ideias-chave, o particionamento funcional e a hierarquia, e utilizando um conjunto de ferramentas gráficas poderosas, de fácil utilização e compreensão, estas técnicas permitem acelerar todas as fases do desenvolvimento de software (do projecto ao teste e manutenção) [8],[9],[10],[11].

O desenvolvimento do modelo do sistema que suportou a fase de implementação do software recorreu à utilização de TAPE's na formulação clássica de Page-Jones e De Marco [8],[9]. A escolha desta metodologia de Projecto Orientado a Fluxos de Informação, em detrimento das alternativas, Projecto Orientado a Estruturas de Informação e Projecto Orientado a Objectos, encontra justificação através das considerações seguintes [11]:

- a descrição inicial do sistema (ver ponto 1.2) baseia-se em fluxos de informação entre o posto de trabalho e as unidades de recolha de dados, entre estas, os operadores dos postos de trabalho e a unidade central, e entre esta e os seus operadores. A utilização de uma metodologia de projecto que permita passar da especificação funcional sob a forma de fluxos de informação para a especificação do software, permite obter esta de uma forma simples e fiável;
- o Projecto Orientado a Fluxos de Informação é a metodologia com o maior espectro de áreas de aplicação, estando especialmente vocacionada para aplicações nas quais a informação é processada sequencialmente e onde não existe uma estrutura formal de informação hierarquizada. Exemplos típicos são aplicações de controlo com microprocessadores e controlo de processos;

- com a inclusão de algumas extensões, esta metodologia pode ser utilizada no projecto de aplicações de tempo real conduzidas por interrupções. Essas extensões são, resumidamente, as seguintes:
 - a) um mecanismo de representação de comunicação e sincronização entre tarefas;
 - b) uma notação para representar a dependência de estado, como por exemplo Diagramas de Transição de Estados;
 - c) uma abordagem que permita a inclusão nos Diagramas de Fluxo de Informação convencionais de particularidades dos sistemas de tempo real recorrendo, por exemplo, à utilização de diagramas onde se representem simultaneamente fluxos de informação e fluxos de controlo, fluxos de informação contínuos temporalmente, etc.
- o Projecto Orientado a Estruturas de Informação é de aplicação mais morosa e complexa, sendo vocacionado para aplicações com utilizem estruturas de informação bem definidas e hierarquizadas, como por exemplo sistemas de informação de gestão, sistemas operativos e sistemas de CAD/CAM/CAE;
- o Projecto Orientado a Objectos, que tem vindo a gozar de popularidade crescente tem, ao contrário dos anteriores, o problema da dependência de linguagens de programação específicas (ADA, MODULA, SMALLTALK, C++, Eiffel, etc.). Colocam-se aqui problemas de disponibilidade de ferramentas de desenvolvimento, e do tempo necessário para a familiarização com essas ferramentas.

No que concerne ao desenvolvimento, integração e teste de hardware e software recorreu-se a ferramentas avançadas, como um sistema de CAD-pcb para desenho de placas e um Sistema Universal de Desenvolvimento de Microcontroladores dotado de facilidades de análise lógica e de emulação dentro do circuito (ICE).

CAPITULO 5

Unidades de recolha de dados

5. UNIDADES DE RECOLHA DE DADOS

Seguindo a metodologia de desenvolvimento descrita em 4.1, e com base nas funções e objectivos pretendidos para estas unidades (ver 1.2 e 1.3), procurou-se estabelecer uma configuração mínima de hardware.

Para isso, começou-se por uma definição das funções e objectivos das unidades de recolha de dados, à qual se seguiu uma análise preliminar das tarefas a implementar.

5.1 - Funções

- Aquisição, para cada posto de trabalho, do seu código identificativo e do código do operador respectivo, e ainda do código de operação, medida de produção e estado.
- Registo horário de todas as alterações verificadas nestes parâmetros, com a excepção do valor da medida de produção o qual deve ser simplesmente actualizado ao longo do tempo.
- Afixação, para o operador, da medida de produção e do estado do posto de trabalho.
- Inserção, como escravo, numa rede BITBUS, de modo a permitir a transferência de blocos de memória com informação e/ou programas entre estas unidades e a unidade central.

5.2 - Objectivos

- Minimização do custo, dado estes serem os componentes mais numerosos do sistema.
- Garantia de grande fiabilidade na recolha e transmissão da informação.
- Manutenção da funcionabilidade mesmo em caso de falha de comunicação com a unidade central.

5.3 - Análise preliminar

A fase de análise preliminar iniciou-se com a construção de um Diagrama de Contexto (Diagrama de Fluxo de Informação de nível zero) baseado nas funções definidas em 5.1, o qual permitiu passar à definição da arquitectura interna e dos blocos constituintes das unidades de recolha de dados (figura 5.1) .

Os Diagramas de Fluxo de Informação representam o sistema como um conjunto de tarefas ou processos que recebem informações de "fontes de informação" (sensores), processam internamente essas informações e enviam os resultados para "poços de informação" (actuadores). A informação pode ser temporariamente armazenada para ser posteriormente utilizada, pela mesma ou por outra tarefa do sistema.

Podem identificar-se na figura as fontes ("sensor específico", "teclado", "rede BITBUS", etc.) e os poços ("afixador alfanumérico" e "rede BITBUS") de informação, assim como elementos de armazenamento temporário ("buffers").

Apresentam-se a seguir os pressupostos utilizados na elaboração deste diagrama:

- os códigos de operador e de operação são introduzidos pelo operador através de um teclado alfanumérico. É importante que se preveja a possibilidade da introdução destas informações utilizando um leitor de código de barras ou de cartão magnético, o que exigirá o desenvolvimento de uma interface com esses equipamentos (geralmente utilizando uma porta série). Dado estas serem soluções mais caras do que um teclado, e que em muitos casos não o substituem completamente, optou-se por não utilizar para já este tipo de equipamento tendo no entanto o cuidado de deixar o sistema aberto à sua inclusão futura;
- a medida de produção é adquirida usando impulsos de contagem gerados por um sensor, um micro-interruptor ou um gerador de impulsos rotativo ou "encoder", que são entrada para um contador de impulsos;
- o estado do posto de trabalho é automaticamente adquirido pelas unidades, sempre que tal for técnica e economicamente viável. No caso contrário, o operador pode indicar o estado quer através do teclado quer através de um interruptor rotativo;

- caso exista uma unidade de recolha de dados por posto de trabalho é fácil, na unidade central, associar um código de posto de trabalho ao endereço da unidade respectiva na rede BITBUS. Se tal não se verificar, o operador terá de introduzir esse código da mesma forma que faz com os códigos de operação e operador, i.e. através do teclado;

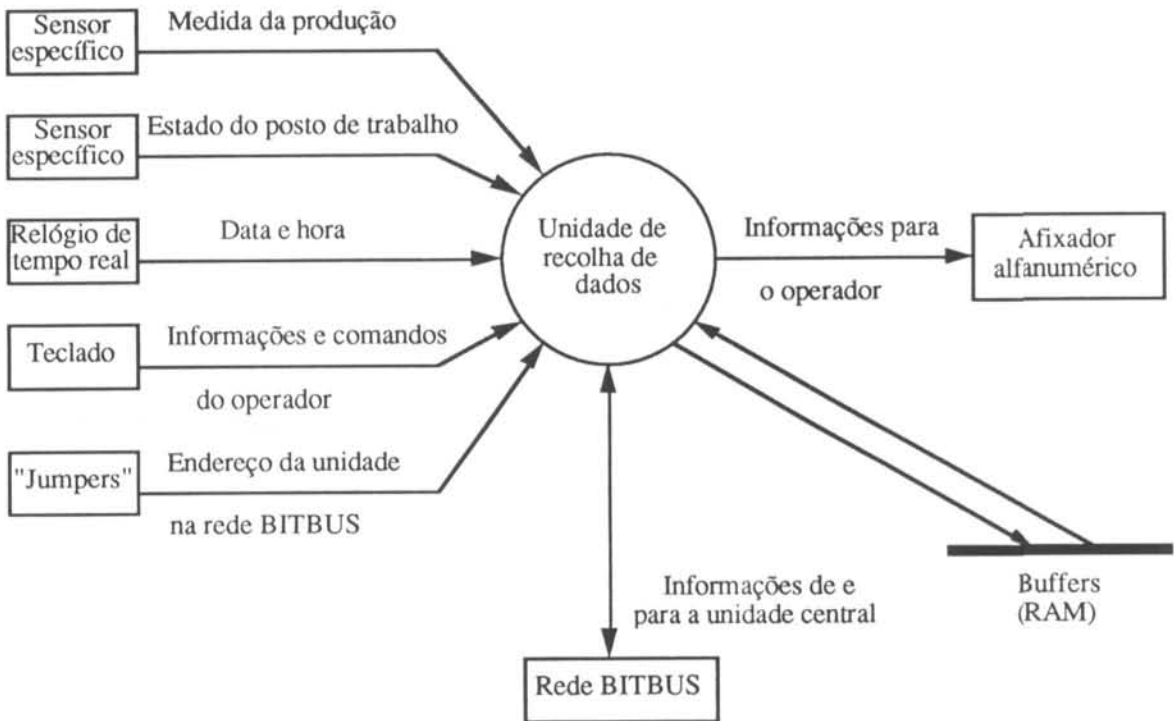


Figura 5.1 - Unidade de recolha de dados: Diagrama de Contexto

- uma vez que se pretende fazer o registo horário de todas as alterações que se verificarem no posto de trabalho, é necessária a existência de um relógio de tempo real;
- da unidade central podem chegar informações diversas, tais como valores para actualização do relógio de tempo real e pedidos de envio de informação sobre o posto de trabalho. Para a unidade central são enviadas igualmente informações sobre o posto de trabalho, erros verificados pela unidade de recolha de dados, etc. A rede BITBUS pode, portanto, ser

considerada simultaneamente como fonte e poço de informação relativamente às unidades de recolha de dados;

- o afixador alfanumérico é um poço para o qual são enviadas informações que se destinam aos operadores;
- na memória interna das unidades (RAM) são armazenadas as informações referentes ao posto de trabalho. Essas informações são escritas pela unidade de recolha de dados e lidas pela unidade central. São portanto pontos de partilha de informação, na gestão dos quais é necessário um cuidado especial para garantir a integridade da informação.

5.4 - Redefinição da arquitectura

5.4.1 - Introdução de um nível hierárquico intermédio

A análise preliminar permitiu identificar com certo detalhe alguns sub-sistemas físicos constituintes das unidades de recolha de dados. Uma vez fixados alguns elementos constituintes, foi possível fazer uma primeira iteração na definição do hardware e tirar algumas conclusões.

Este procedimento quebrou, de certa forma, o percurso de um desenvolvimento apoiado em TAPE's, comprometendo aparentemente uma das vantagens da análise estruturada - a elevada abstracção que é possível manter durante o projecto e a independência em relação à posterior implementação do sistema. Ele resultou, todavia, por um lado de alguns elementos estarem definidos à partida e por outro da necessidade de seguir uma metodologia de desenvolvimento que obriga a que, a cada passo, se analise a relação custo/desempenho das possíveis soluções alternativas e se escolha a mais favorável.

Apresentam-se seguidamente alguns pontos importantes relativamente às unidades de recolha de dados, que conduziram a uma redefinição da sua arquitectura:

- pretende-se que cada posto de trabalho disponha de uma unidade de recolha de dados. O número destas unidades pode pois variar, no caso de PME's, entre algumas dezenas e algumas centenas;
- a rede BITBUS está limitada a um número máximo de cerca de 250 nós, o que é suficiente para a maioria dos casos. Contudo, mesmo que o número de unidades de recolha de

dados não ultrapasse as 250, o tempo que a unidade central demorará a fazer a recolha da informação de todos os postos de trabalho aumentará muito para um número elevado de nós da rede;

- se, entre duas leituras sucessivas por parte da unidade central, não houver alteração das variáveis referentes ao posto de trabalho, com a excepção da medida produção, o único valor que terá de ser enviado será precisamente essa medida de produção. Esta situação é vulgar, pois geralmente um posto de trabalho mantém-se durante períodos razoáveis de tempo a executar uma determinada operação com um determinado operador. É também possível estabelecer que esse valor é representado por três ou quatro bytes. Por outro lado, as mensagens da rede BITBUS têm um cabeçalho fixo de sete bytes, independentemente da quantidade de bytes de informação que possam conter. Daqui resulta que, no cenário descrito, seriam frequentemente utilizadas mensagens com sete bytes de cabeçalho para transmitir três ou quatro bytes de informação, quando poderiam ser utilizadas para transmitir um número substancialmente superior de bytes de informação. Uma correspondência unívoca entre nós da rede e postos de trabalho pode pois determinar uma arquitectura pouco eficiente do ponto de vista da utilização da rede. Este problema pode ser ultrapassado se a informação referente a vários postos de trabalho fôr concentrada, para que possa ser enviada em blocos de maior dimensão através da rede.

O que foi apresentado nos parágrafos anteriores aponta para que, no caso de existir um número elevado de postos de trabalho, as unidades de recolha de dados sejam agrupadas. As unidades de cada conjunto estariam ligadas a uma outra unidade, que teria como função agrupar as informações dos postos de trabalho a ela ligados, e seria ela mesma o nó da rede BITBUS ligado à unidade central.

A título de exemplo, pode considerar-se o caso de um processo fabril com 100 postos de trabalho. Com a arquitectura inicial existiriam 100 unidades de recolha de dados e portanto 100 nós da rede BITBUS. Agrupando, por exemplo, as unidades de recolha de dados em 10 grupos de 10 unidades cada, existiriam 10 nós da rede BITBUS, os quais se designariam por unidades concentradoras. A cada uma dessas unidades estariam ligadas 10 unidades de recolha de dados.

Esta arquitectura resolve os problemas referidos anteriormente, por um lado diminuindo o número de nós da rede BITBUS e por outro permitindo concentrar a informação dos postos de trabalho de modo a poder ser transmitida em blocos, o que aumenta a eficiência da rede. Levanta também, no entanto, dois problemas:

- passam a existir mais 10 unidades concentradoras, o que vai aumentar o custo do sistema;
- introduz um nível hierárquico suplementar, e a consequente necessidade de considerar a ligação entre unidades de recolha de dados e unidades concentradoras.

5.4.2 - Reespecificação das unidades de recolha de dados

O problema do aumento do custo introduzido pelas unidades concentradoras pode ser ultrapassado através da reespecificação do hardware. Como foi já referido, num posto de trabalho um operador mantém-se geralmente a fazer uma determinada operação durante períodos de tempo razoáveis, pelo que a introdução pelo operador de informações através do teclado é uma tarefa executada poucas vezes por dia (tipicamente entre 2 e 10 vezes). Não será pois problemático se a introdução dos dados não for feita na unidade que se encontra no posto de trabalho mas na unidade concentradora, desde que esta não se encontre muito afastada daquele, dado ser mínima a diferença entre o tempo dispendido pelo operário num ou no outro caso.

Podem assim simplificar-se as unidades que se encontram nos postos de trabalho pois o teclado deixará de ser necessário, sendo transferido para as unidades concentradoras. Nos postos de trabalho passam a ser adquiridos unicamente o estado da máquina (automaticamente ou através de um interruptor rotativo) e a medida (contagem) da produção.

Retomando o exemplo anterior, passar-se-ia de 100 para 10 teclados alfanuméricos, com a mesma redução em todo o hardware de interface necessário. Esta simplificação do hardware das unidades de recolha de dados dos postos de trabalho deverá compensar o acréscimo resultante do aparecimento das unidades concentradoras, isto obviamente dependendo do número de postos de trabalho e do número de unidades simplificadas ligadas a cada unidade concentradora. Estas unidades simplificadas passarão a ser designadas por unidades de aquisição.

A ligação entre as unidades de aquisição e a unidade concentradora pode ser feita de várias formas:

- a) ligação ponto a ponto, através de portas série, entre a unidade concentradora e cada uma das unidades de aquisição. Esta solução tem vários pontos negativos, como por exemplo a necessidade da unidade concentradora ter uma porta série para cada unidade de aquisição, acarretando custo elevado, expansibilidade muito limitada e um acréscimo significativo na cablagem utilizada;
- b) utilização de uma sub-rede BITBUS. Neste caso, a unidade concentradora funcionaria como escravo da rede BITBUS que a liga à unidade central, e como mestre da sub-rede que a liga às unidades de aquisição. A rede BITBUS tem mecanismos para implementar este tipo de arquitectura, os quais estão referidos no Apêndice A;
- c) desenvolvimento de uma sub-rede própria. Esta solução só se justifica se permitir reduzir consideravelmente os custos do sistema, uma vez que obriga ao desenvolvimento de software de comunicação, já existente no caso do BITBUS.

Com a introdução de um segundo nível hierárquico, as tarefas a implementar nas unidades de aquisição passaram a ser muito simples: aquisição do estado do posto de trabalho e da medida de produção, afixação desses valores para o operador e envio dos mesmos para a unidade concentradora através de uma rede. Este acabou por ser um facto da maior relevância, dado o seu impacto no custo da solução. Com efeito, se praticamente qualquer microcontrolador de 8 bits pode desempenhar estas tarefas, o microcontrolador 8031 da INTEL apresenta a vantagem suplementar de ter um conjunto de instruções igual ao do 8044, o que facilita o desenvolvimento do software, com um preço cerca de um terço do deste (aproximadamente dez mil escudos mais barato). Se o número de unidades de aquisição for suficientemente elevado este factor terá grande impacto no custo final da solução.

Teve de aceitar-se como desvantagem a necessidade de desenvolver o software de comunicação. Foi contudo possível simplificar esta tarefa, tirando partido do facto dos conjuntos de instruções dos dois microcontroladores serem iguais e adoptando um subconjunto do protocolo da rede BITBUS. A capacidade de processamento das unidades de aquisição será também inferior

neste caso, embora sem comprometer as funcionalidades especificadas.

5.4.3 - Architecturas possíveis

Do que acaba de expôr-se resultam três architecturas possíveis em função do caso particular de cada aplicação:

- a) número pequeno de postos de trabalho: uma unidade de recolha de dados em cada posto de trabalho ligada a uma unidade central através de uma rede BITBUS (figura 5.2);

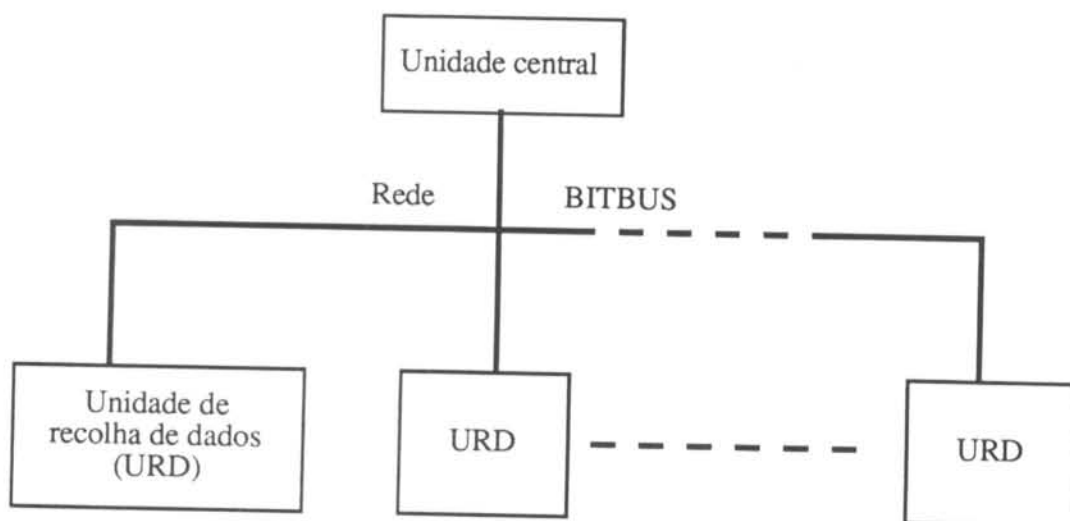


Figura 5.2

- b) número elevado de postos de trabalho, mas sem grandes exigências de processamento local: uma unidade de aquisição por posto de trabalho, baseada no microcontrolador 8031, ligada a uma unidade concentradora através de uma rede própria a desenvolver. As unidades concentradoras estão ligadas á unidade central através de uma rede BITBUS (figura 5.3);
- c) número elevado de postos de trabalho e necessidade de alguma capacidade de processamento local: arquitectura

idêntica à da alínea anterior, com a excepção de que as unidades de aquisição são também baseadas no microcontrolador 8044 e estão ligadas à unidade concentradora através de uma sub-rede BITBUS (figura 5.4). Esta solução adequa-se aos casos em que há necessidade de algum processamento local (por exemplo, monitorização da temperatura num forno, aquisição e processamento em tempo real de informação para controlo de qualidade, etc.) sendo aproveitado o facto de estar disponível uma unidade com inteligência local para implementar essas tarefas, optimizando assim os recursos disponíveis.

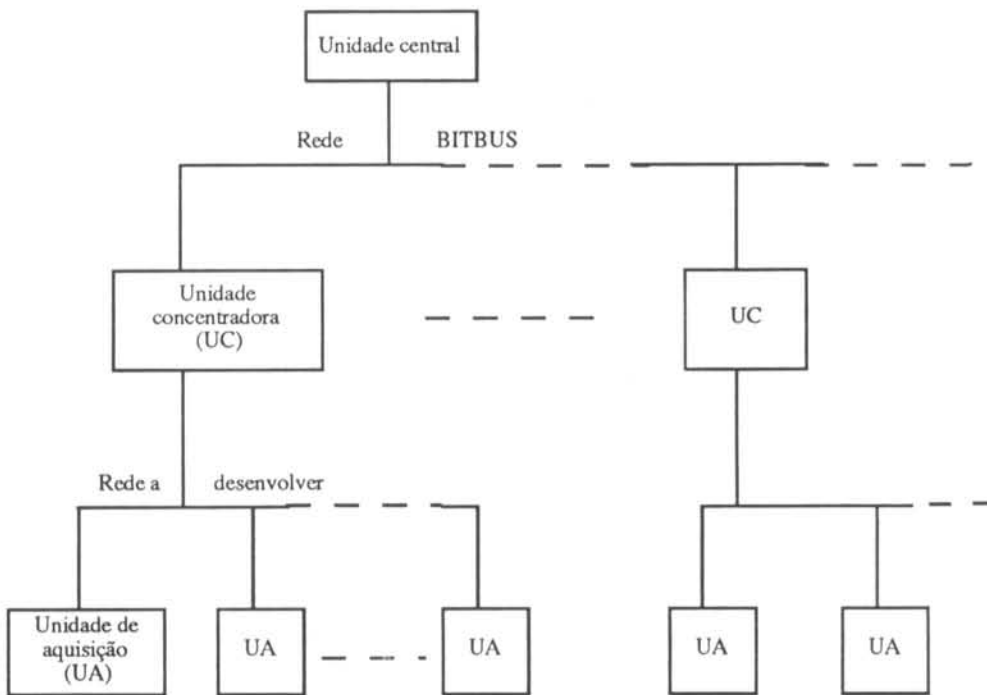


Figura 5.3

No âmbito deste trabalho optou-se pela arquitectura da alínea b) porque, apesar de obrigar a um maior esforço de desenvolvimento, é a que conduz a um mais baixo custo das unidades de aquisição.

O diagrama geral do sistema passa a ser o da figura 5.5. As unidades de recolha de dados do diagrama da figura 2.1 foram

substituídas por conjuntos constituídos por uma unidade concentradora e diversas unidades de aquisição. Com base nesta nova arquitectura passou-se à fase de análise estruturada, começando por definir as funções e objectivos das unidades de aquisição e das unidades concentradoras.

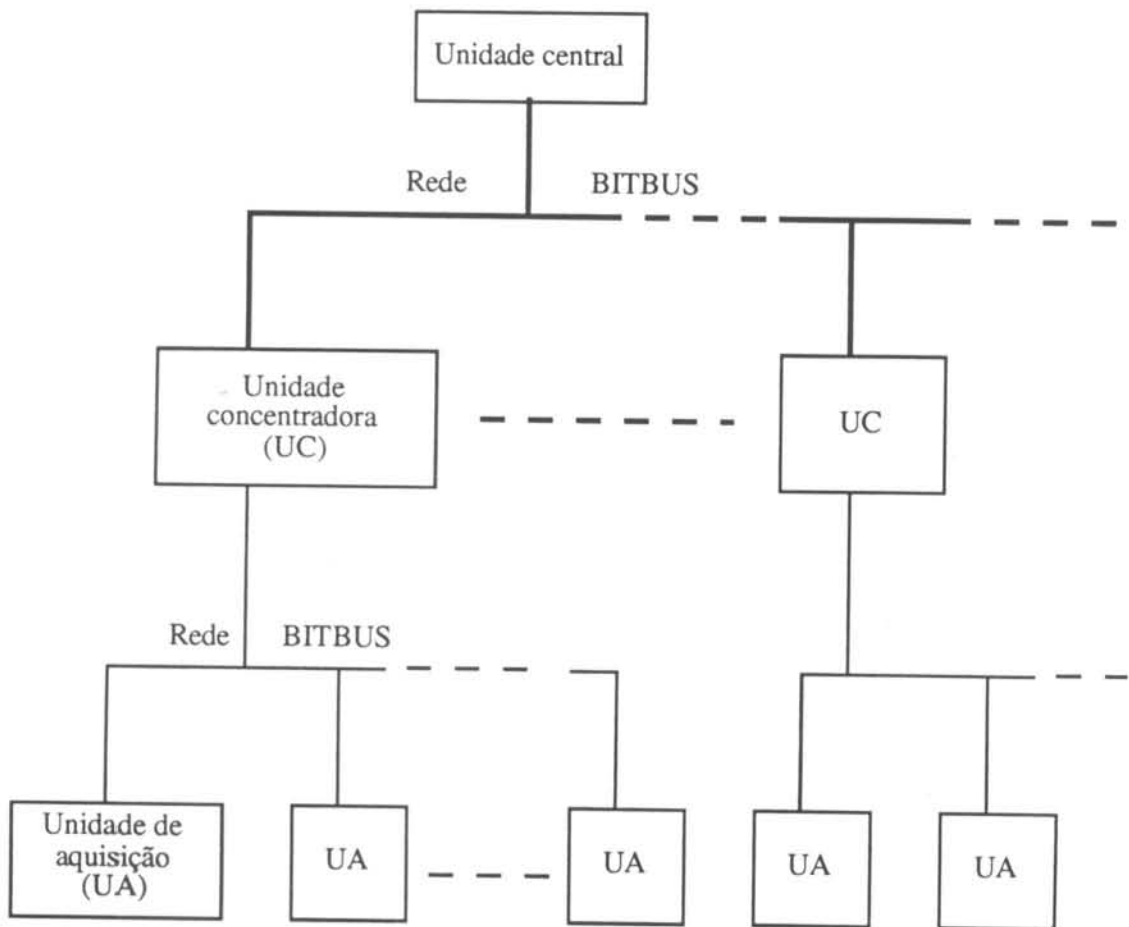


Figura 5.4

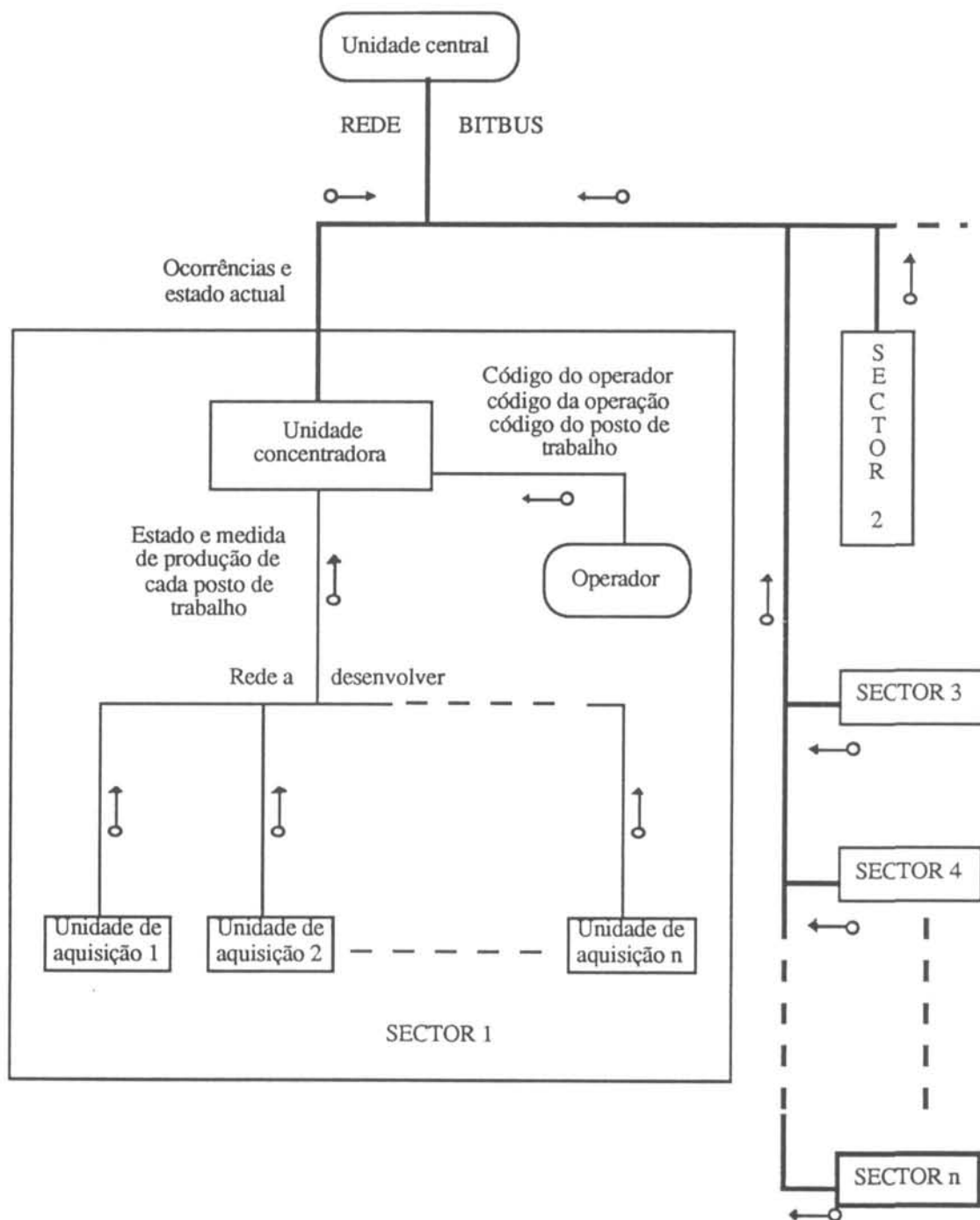


Figura 5.5 - Diagrama geral da nova arquitectura do sistema

CAPITULO 6

Unidades de aquisição

6 - UNIDADES DE AQUISIÇÃO

A definição das suas funções e dos objectivos pretendidos, precederá a aplicação de TAPE's no desenvolvimento de um modelo adequado para as unidades de aquisição.

6.1 - Funções

Na sequência da alteração de arquitectura discutida em 5.4, as funções inicialmente atribuídas às unidades de recolha de dados foram distribuídas entre as unidades de aquisição e as unidades concentradoras. Apresentam-se a seguir as funções que ficaram associadas às unidades de aquisição acrescidas de algumas outras que, embora não directamente ligadas ao controlo de produção, foram consideradas úteis à exploração do sistema:

- aquisição da medida de produção e do estado do respectivo posto de trabalho;
- geração, por software, de um contador parcial associado a um botão de reinicialização a zero ("reset"), que permita ao operador efectuar contagens parciais relacionadas com a medida de produção, e levar esse contador a zero sempre que precisar, sem no entanto alterar o valor da medida de produção. Esta nova função implica o acréscimo de um botão de "reset" e de um circuito simples de tratamento de sinal ("debounce"). Dado o baixo custo do hardware associado foi decidida a implementação desta função;
- disponibilidade de uma saída em relé que possa ser utilizada para inibir o funcionamento do posto de trabalho, caso o estado deste seja diferente de "funcionamento". O objectivo é não permitir que o operário esteja a produzir se o posto de trabalho estiver, do ponto de vista do sistema, noutro estado, por exemplo avariado ou desligado, pois nessas condições a informação contida na base de dados relativamente à produção e às médias do operador não será correcta. Para implementar esta função foi necessário acrescentar um relé e o respectivo circuito de interface com um bit de uma das portas do microcontrolador. Também neste caso foi considerado aceitável o custo suplementar inerente;

- afixação, para o operador, das contagens total e/ou parcial, do estado e do endereço do posto de trabalho;
- inserção numa rede própria que permita a transferência de blocos de informação e/ou programas entre a unidade de aquisição e uma unidade concentradora a um nível hierárquico superior.

6.2 - Objectivos

Os objectivos fundamentais tidos em conta no desenvolvimento das unidades de aquisição foram os seguintes:

- minimização do custo, dado serem os componentes mais numerosos do sistema;
- garantia de elevada fiabilidade na recolha e transmissão de informação.

6.3 - Análise Estruturada

6.3.1 - Análise de Tarefas (Diagrama de Contexto)

Foi possível, a partir da informação disponível, passar à construção de um primeiro modelo representativo dos vários estados em que o sistema se poderá encontrar, bem como das condições que provocam a transição entre eles. É assim que aparece o Diagrama de Transição de Estados da figura 6.1, neste caso o Diagrama de Contexto, no qual estão representados os cinco estados em que as unidades de aquisição se podem encontrar:

- "Fora de serviço": estado em que fica a unidade quando não alimentada;
- "Inicialização": estado que se segue ao anterior quando é fornecida alimentação à unidade, e durante o qual se efectua a inicialização e configuração do microcontrolador e periféricos. Quando estas terminam, o sistema passa para o estado de "Monitorização". Durante as inicializações todas as interrupções do microcontrolador devem estar inibidas;
- "Monitorização": estado de funcionamento normal das unidades. Nele são actualizados em RAM os valores da

medida de produção e do estado do posto de trabalho, assim como as informações enviadas para o operador. São também permitidas interrupções que desencadeiam mudanças de estado. Foram ainda implementadas outras funções definidas posteriormente, tais como a de interface com o relé de controle;

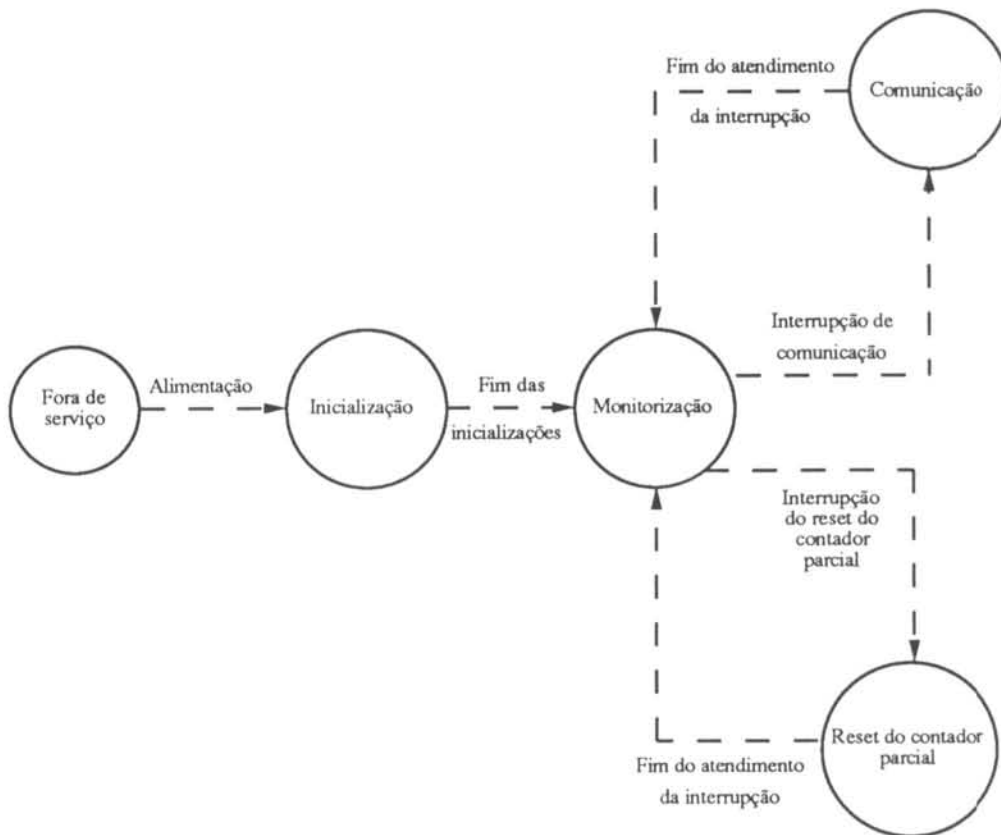


Figura 6.1 - Unidade de aquisição: Diagrama de Transição de Estados

- "Comunicação": sempre que a unidade concentradora solicitar um serviço à unidade de aquisição, a interface série desta gera uma interrupção ao microcontrolador provocando a transição para o estado "Comunicação". Neste estado será atendido o pedido enviado pela unidade concentradora, após o que o sistema retorna ao estado "Monitorização". Durante o período de comunicação as

interrupções do microcontrolador estão inibidas, para que não se verifiquem atrasos no atendimento do pedido da unidade concentradora. Daqui resulta que, durante este curto período, os valores em RAM da medida de produção e do estado do posto de trabalho se mantêm inalteradas e não são atendidos pedidos de "reset" do contador parcial. Não há, no entanto, perda de informação porque o "counter" do microcontrolador é um módulo cujo funcionamento é independente da inibição das interrupções do microcontrolador. Será, porém, inevitável um pequeno atraso na actualização na RAM dos valores do contador;

- "Reset do contador parcial": sempre que o operador pressiona o botão de "reset" do contador parcial, está a gerar uma interrupção ao microcontrolador cuja rotina de atendimento se encarrega de levar o contador parcial a zero. Durante esta rotina de atendimento as interrupções do microcontrolador estão inibidas pelo que, além das variáveis em RAM não serem actualizadas, as interrupções geradas pela interface série também o não são. Esta situação não é grave, visto ser possível prever que a duração desta rotina seja apenas de alguns microsegundos. Após o atendimento da interrupção a unidade transita de novo para o estado "Monitorização".

6.3.2 - Diagramas de Fluxo de Informação (DFI's)

Com base nas funções definidas e no Diagrama de Transição de Estados (figura 6.1) seguiu-se a construção dos Diagramas de Fluxo de Informação, começando pelo DFI de ordem zero (figura 6.2). A única tarefa/processo deste DFI implementa obviamente o modelo proposto pelo Diagrama de Transição de Estados.

Podem identificar-se cinco fontes e dois poços de informação, além do canal série que será simultaneamente fonte e poço, em relação aos quais se apresentam os seguintes comentários:

- os impulsos de contagem utilizados para gerar a medida de produção são, como foi já referido, produzidos por sensores digitais, como por exemplo um micro-interruptor ou um codificador. Na figura referida está representado o primeiro caso;
- o estado do posto de trabalho é introduzido utilizando um interruptor rotativo. Em alguns casos é possível uma

detecção automática do estado (por exemplo, detectar se um tear está a produzir, se está desligado ou se parou por quebra de um fio). Na maior parte dos casos essa detecção automática é no entanto difícil, inviável ou impossível, seja técnica, seja economicamente (como é o caso de distinguir se uma determinada máquina está parada devido a uma avaria mecânica, eléctrica ou hidráulica). Perante esta situação, e para manter a generalidade, foi decidido utilizar um interruptor rotativo para indicação, pelo operador, do estado do posto de trabalho;

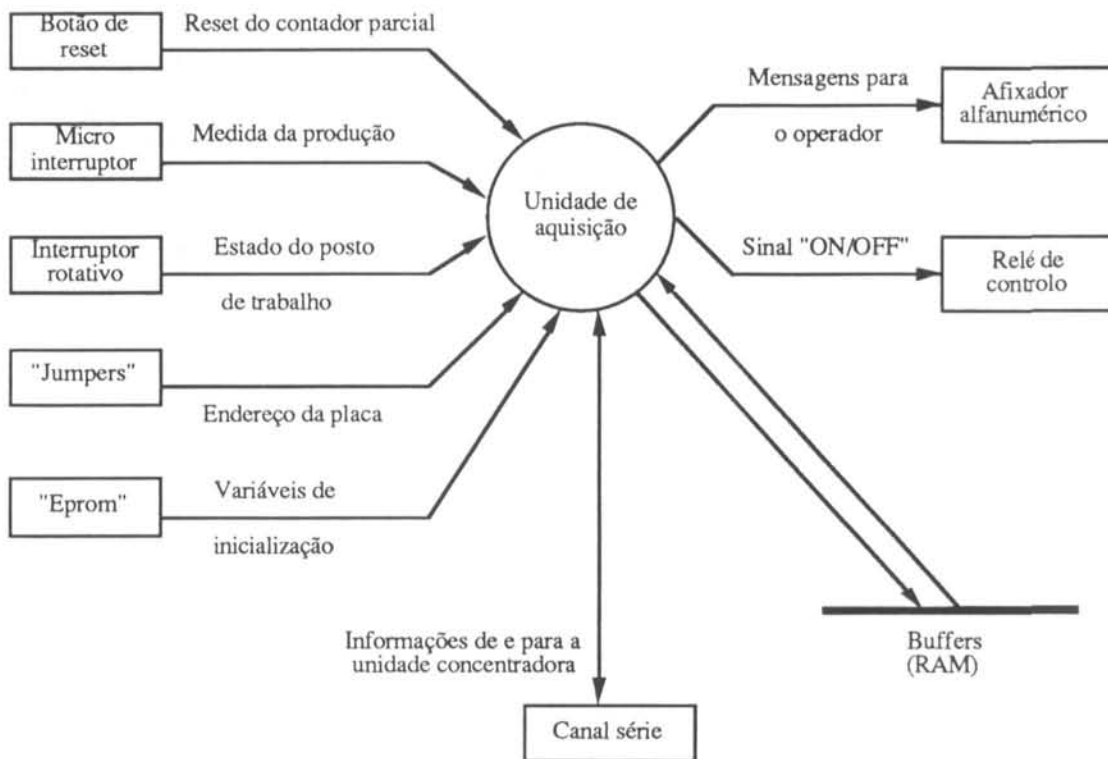


Figura 6.2 - Unidade de aquisição: DFI de ordem zero

- o endereço da unidade de aquisição na rede própria é programado através de "jumpers";
- através da rede, referida na figura por "canal série", chegam e são enviadas informações. É portanto simultaneamente uma fonte e um poço de informação;

- o conjunto das variáveis e parâmetros de inicialização (inicialização dos periféricos, etc.) e configuração da unidade estão armazenados em EPROM;
- o relé de controlo é um actuador para o qual é enviado pelo microcontrolador um sinal "on/off";
- na RAM da unidade serão implementados "buffers" para armazenamento e posterior utilização da informação.

O passo seguinte consistiu, para cada uma das tarefas/estados definidos no Diagrama de Transição de Estados, na construção de DFI's cada vez mais detalhados, i.e. de ordem superior, até se atingir o grau de detalhe julgado necessário. Consegue-se assim uma representação gráfica do sistema, contendo informação quer sobre a sua organização interna quer sobre as suas interligações com o exterior, com o pormenor necessário e mantendo um elevado grau de abstracção em relação a detalhes de implementação. Vão pois aparecer os DFI's de ordem um para os estados "Inicialização", "Monitorização", "Comunicação" e "Reset do contador parcial" e depois, se necessário, os DFI's de ordem dois para cada uma das funções/processos individuais correspondentes. Apresentam-se a seguir os DFI's de ordem um para todos os estados, e o DFI de ordem dois para o estado "Inicialização" (figuras 6.3 a 6.7). Este estado será utilizado para ilustrar a utilização desta ferramenta gráfica.

Para melhor ilustrar as transferências de informação, representa-se em cada DFI não só o processo que ele especifica como também todos os que partilham da mesma informação.

A figura 6.3 representa o DFI de ordem um referente ao estado "Inicialização". O processo de inicialização utiliza como fonte de informação a memória EPROM do sistema, onde se encontram armazenados os valores a utilizar para a inicialização e configuração do microcontrolador e dos periféricos, e um conjunto de "jumperes" onde é programado o endereço da placa na rede de comunicação. Existe um único poço de informação, o afixador alfanumérico, para onde é enviado o resultado de um conjunto de testes efectuados automaticamente pelo microcontrolador à placa e o endereço da unidade na rede de comunicações. Podem identificar-se ainda dois locais em memória RAM onde são armazenados temporariamente o resultado do autoteste e o endereço da placa. Estes valores serão posteriormente utilizados pela rotina de comunicação, tal como mostra a figura, levando à necessidade de garantir a integridade da informação partilhada. Com efeito, se uma dada tarefa estiver a actualizar a informação e

for interrompida por outra tarefa que seja utilizadora dessa informação, esta última poderá utilizar informação não válida. Neste caso isso não ocorre, porque a rotina de inicialização só escreve uma vez essas informações em RAM e durante esse período de escrita as interrupções estão ainda inibidas. A partir desse momento as referidas informações serão só lidas, pelo que manterão o seu valor inalterado até haver uma nova inicialização da unidade. O acesso mutuamente exclusivo ao "buffer" de RAM é assim garantido através de um mecanismo simples de inibição de interrupções. Apesar da sua não generalidade e de recorrer à inibição de interrupções, as curtas rotinas de acesso ao "buffer" e a simplicidade inerente tornam o mecanismo eficiente e fiável.

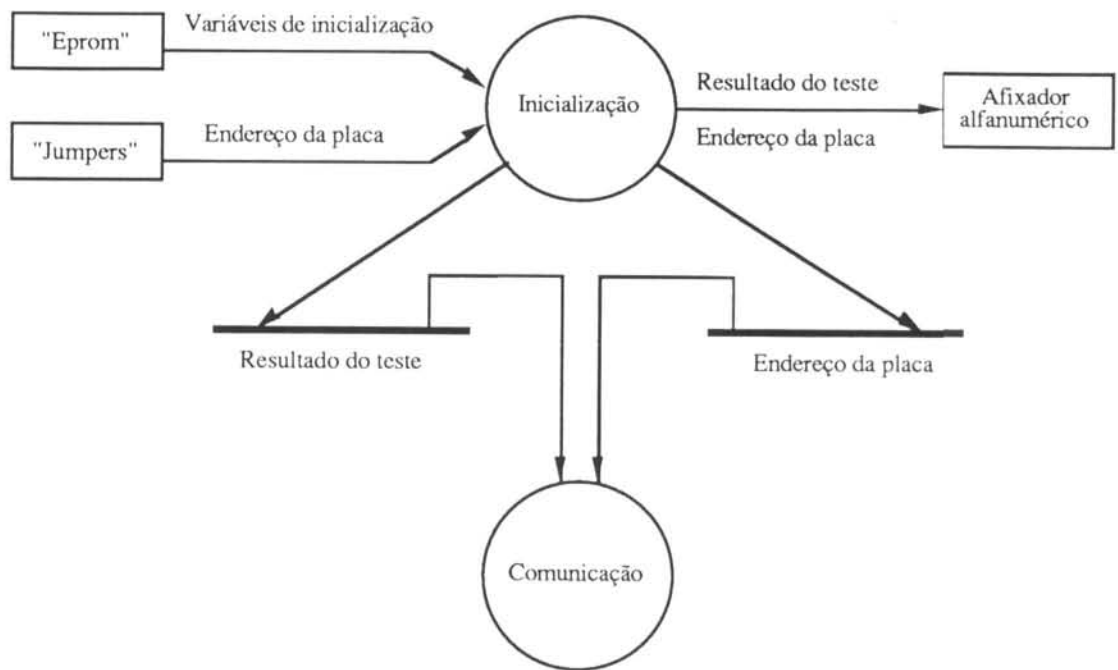


Figura 6.3 - Inicialização: DFI de ordem um

No DFI de ordem dois (figura 6.4) apresenta-se a tarefa "Inicialização" do DFI de ordem um subdividida em três tarefas: "Arranque e configuração da placa", "Teste da placa" e "Afixação do endereço da placa e do resultado do teste". Conseguiu-se assim um grau de detalhe superior, identificando os vários procedimentos em que são manipuladas as informações. Considerou-se ser fácil e

clara a passagem à fase de Projecto Estruturado a partir desta especificação, baseada em tarefas simples e bem definidas, não havendo pois necessidade de particionamento suplementar em DFI's de ordem superior. Se assim não tivesse acontecido, o método de particionamento utilizado para se passar do DFI de ordem um para o de ordem dois seria novamente aplicado, pelo que cada tarefa/processo que se pretendesse detalhar seria sucessivamente dividido em tarefas/processos mais simples até se atingir o detalhe pretendido.

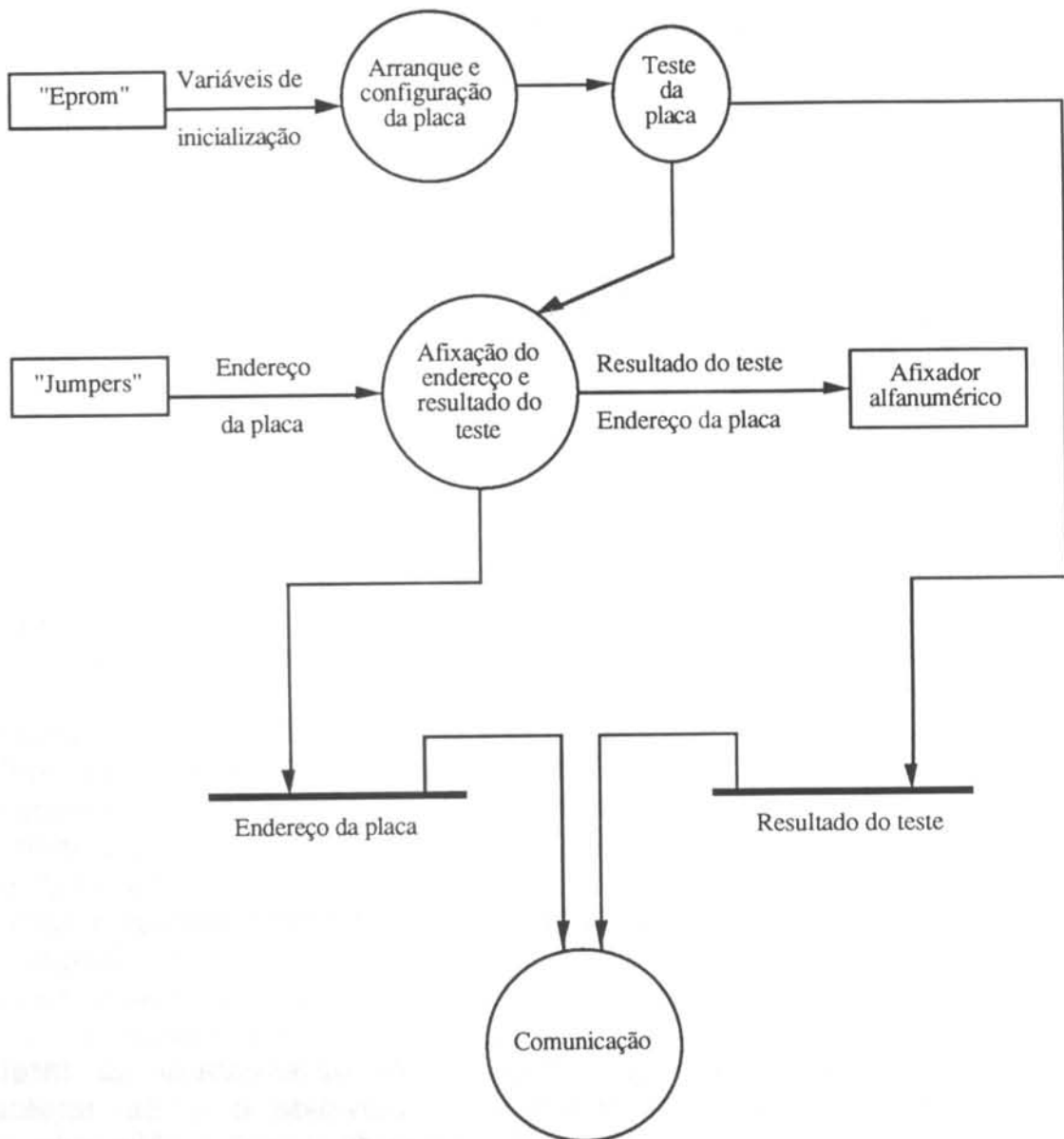


Figura 6.4 - Inicialização: DFI de ordem dois

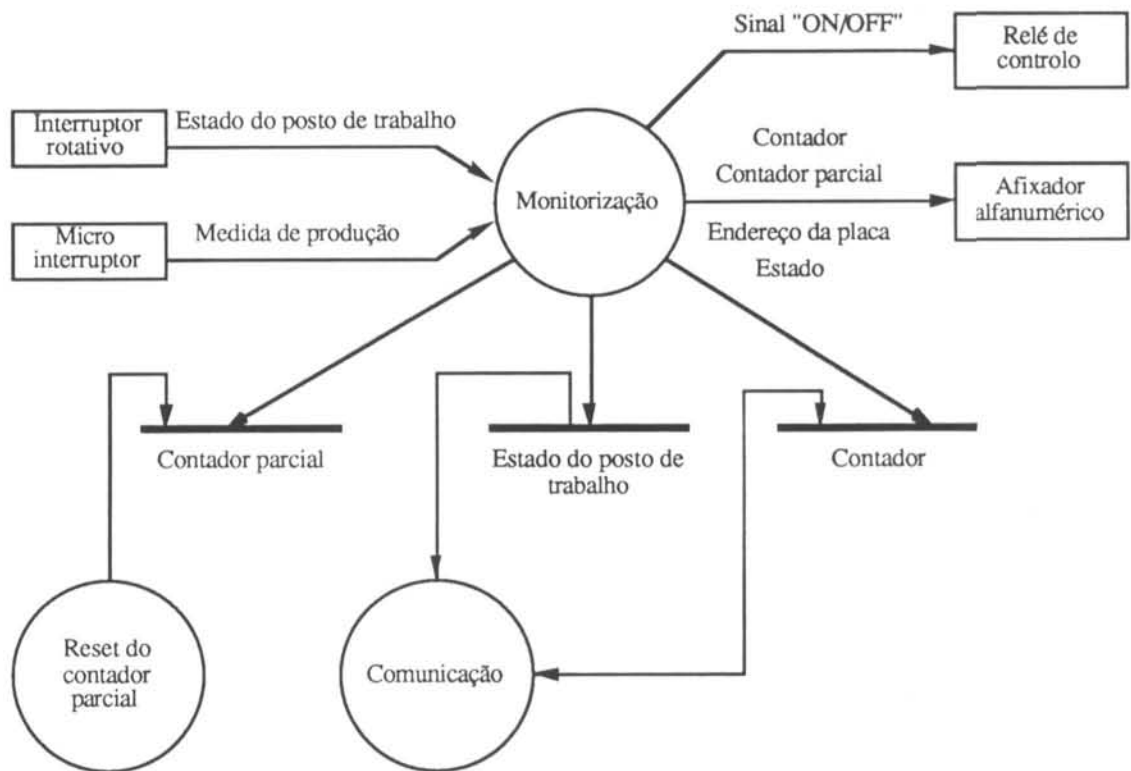


Figura 6.5 - Monitorização: DFI de ordem um

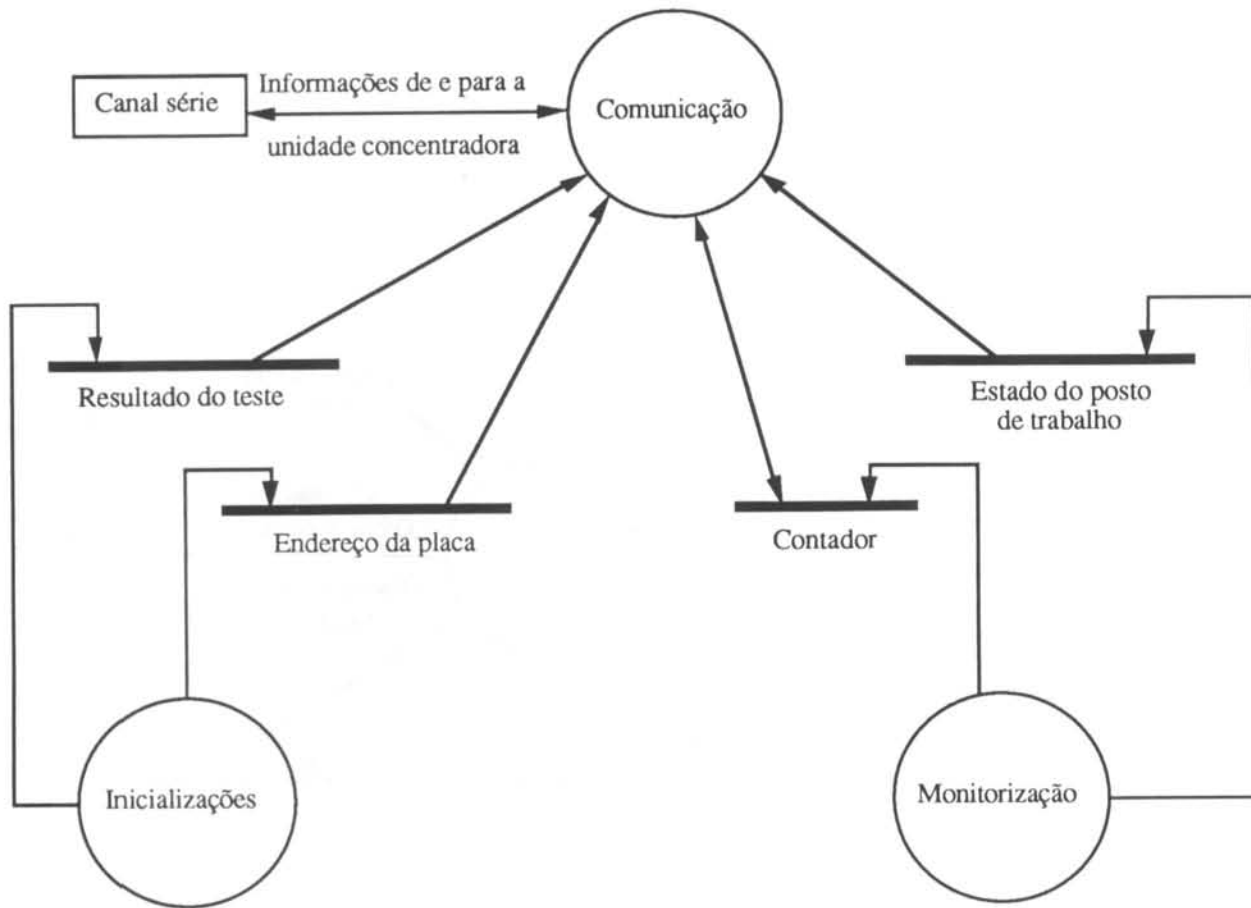


Figura 6.6 - Comunicação: DFI de ordem um

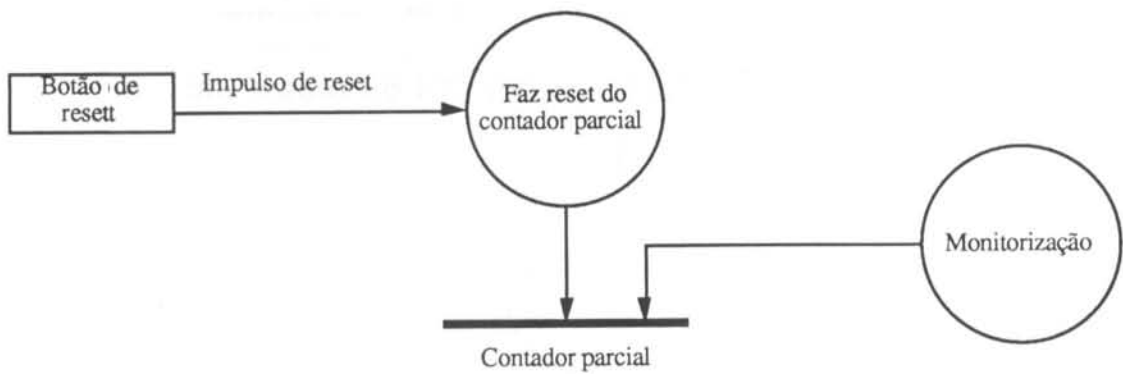


Figura 6.7 - Reset do contador parcial: DFI de ordem um

6.3.3 - Mini-especificações e Dicionário de Informação

Nas representações gráficas utilizadas figuram nomes e abreviaturas para variáveis de informação, tarefas, fontes e poços de informação, etc., cujo significado é necessário descrever por razões de inteligibilidade e rigor.

O conjunto das descrições de implementação das funções/tarefas que aparecem nas representações mais detalhadas (DFI's ou ME's de maior ordem) constituem as denominadas Mini-especificações.

O Dicionário de Informação consiste na descrição do significado de todos os nomes de variáveis, estruturas de informação, fontes e poços de informação utilizados nas representações gráficas que não sejam auto-elucidativos.

A especificação das tarefas pode recorrer a uma simples descrição, utilizar uma linguagem de especificação ou recorrer a fluxogramas, conduzindo, de uma forma ou outra, às referidas Mini-especificações [8]. Nesta especificação foram utilizados, por razões de simplicidade, fluxogramas. Na figura 6.8 está representado o fluxograma referente à especificação da tarefa "Monitorização", o qual é apresentado a título de exemplo. Os fluxogramas referentes à tarefa "Comunicação" dependem da definição dos serviços da rede própria, pelo que não foram construídos nesta fase, mas sim posteriormente. Quando as tarefas são tão simples como no caso do DFI de "Reset do contador parcial" (figura 6.7), não há necessidade de especificações posteriores.

O Dicionário de Informação, repositório das definições de todas as variáveis utilizadas, não é apresentado na sua totalidade dado o carácter auto-elucidativo da esmagadora maioria das variáveis, fontes e poços de informação utilizados. Apresenta-se todavia na figura 6.9, a título de exemplo, um pequeno extracto.

O conjunto formado pelo Diagrama de Transição de Estados, os DFI's, as Mini-especificações e o Dicionário de Informação constituem o produto final da Análise Estruturada - a Especificação Estruturada. É a partir deste conjunto de informações de especificação que se vai iniciar a fase de Projecto Estruturado.

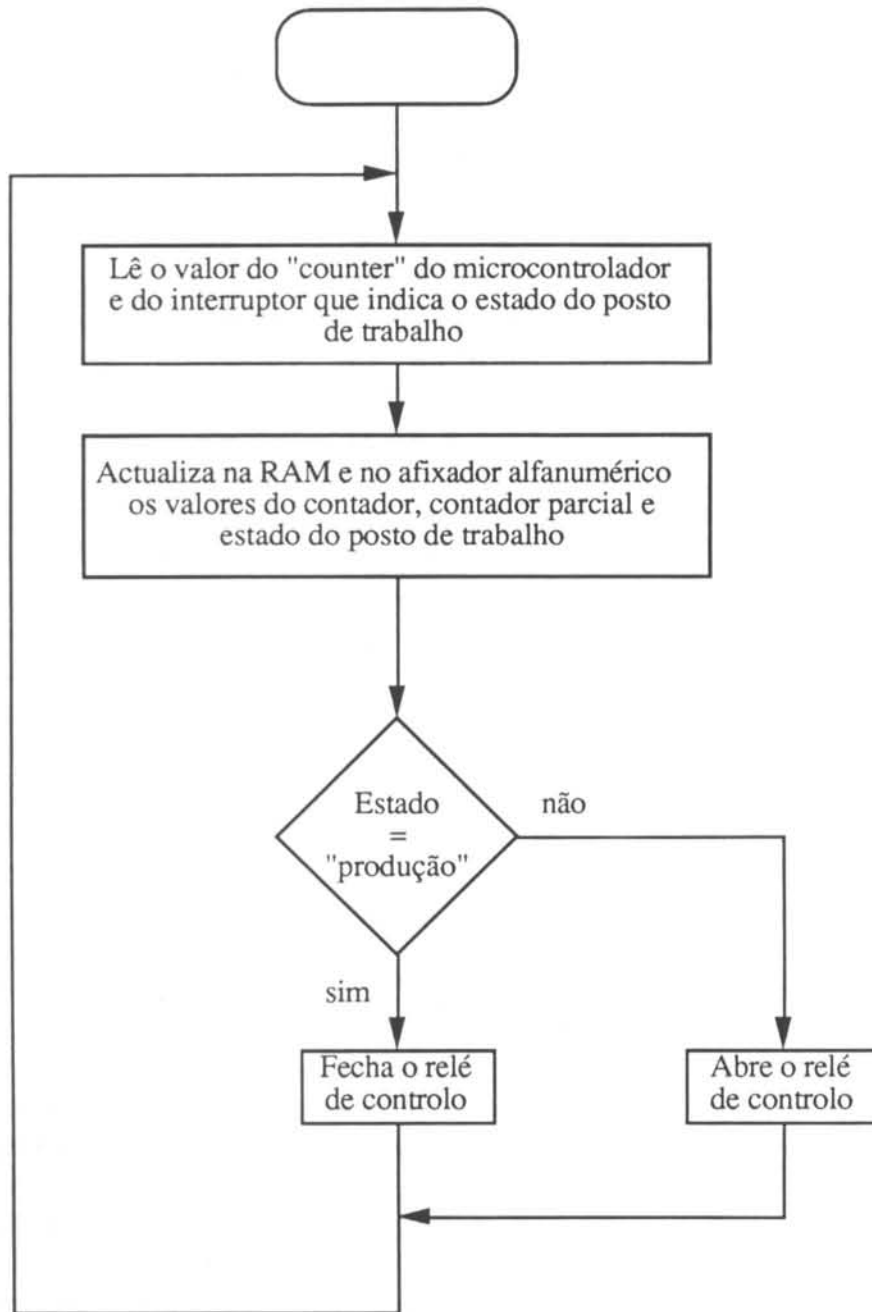


Figura 6.8 - Mini-especificação da tarefa "Monitorização"

.....
Canal série - rede de comunicação industrial própria, a desenvolver, para interligação entre uma unidade concentradora e unidades de aquisição.

.....
"Counter" - "timer/counter" interno de 16 bits do microcontrolador 8031 ou 8044.

.....
Medida de produção - valor que permita quantificar a produção realizado num posto de trabalho, como por exemplo, número de peças produzidas ou de operações efectuadas, número de metros de fio ou tecido produzido, tempo de funcionamento, etc.

.....
Sector - área fabril que contém os postos de trabalho associados a um conjunto unidade concentradora/unidades de aquisição.
.....

Figura 6.9 - Extrato do Dicionário de Informação

6.4 - Projecto Estruturado

A transição entre a Especificação Estruturada, baseada em Diagramas de Fluxo de Informação, e o modelo final do sistema é feita recorrendo a duas metodologias que são a seguir discutidas.

6.4.1 - Análise de Transacções e Análise de Transformadas

A Análise de Transacções é uma ferramenta que, a partir dos DFI's produzidos na fase de Análise Estruturada, inclui na representação do sistema transferências assíncronas de controlo, e introduz hierarquia ao mais alto nível através da criação de um executivo ou distribuidor que gere todo o sistema.

A Análise de Transformadas complementa a ferramenta anterior guiando o particionamento sucessivo de tarefas, de forma a chegar a representações cada vez mais detalhadas. Nos casos em que não há lugar à aplicação de Análise de Transacções, a Análise

de Transformadas é ainda responsável pela introdução de hierarquia.

Neste caso concreto a obtenção do modelo do sistema baseou-se fundamentalmente na Análise de Transacções, conforme se discute no ponto seguinte.

6.4.2 - Mapas de Estrutura (ME's)

Os ME's integram, numa mesma representação gráfica, relações hierárquicas e transferências de informação e de controlo, em oposição aos DFI's que contêm exclusivamente fluxos de informação e onde não existe hierarquia. O sistema é constituído por módulos independentes, cada um implementando uma tarefa/processo, caracterizado por um conjunto de entradas e saídas, por um mecanismo que transforma as entradas nas saídas e por uma lei de controlo que impõe as condições em que essa transformação se pode efectuar.

O ME de ordem zero abarca todo o sistema e consiste na representação dos módulos implementando as tarefas principais, suas fontes e poços de informação, as transferências de informação existentes e os eventos que provocam transições entre tarefas/estados. Pode representar-se a comunicação entre tarefas recorrendo à utilização de "buffers" duplos.

Seguidamente, tal como para os DFI's, mas conservando sempre uma estrutura hierarquizada, cada módulo do ME de ordem zero é dividido em sub-módulos dando origem a ME's de ordem um, e assim sucessivamente até que se atinja o desejado nível de pormenor.

Os ME's permitem por um lado fazer corresponder facilmente a cada tarefa/módulo um módulo de software, i.e. uma rotina, função, procedimento, etc., e por outro lado fornecem a estrutura hierárquica que permite articular a interligação entre módulos e implementar o gestor, caso exista, das transições assíncronas entre tarefas/estados.

A representação da comunicação entre tarefas permite ainda a identificação de regiões críticas de acesso mutuamente exclusivo a variáveis ou a estruturas de informação, por exemplo áreas de memória onde mais de uma tarefa podem ler e escrever "simultaneamente", alertando para a necessidade de implementação de mecanismos de controlo de acesso.

Na figura 6.10 está representado o ME de ordem zero. A partir dos DFI's correspondentes (ver ponto 6.3.2) identificou-se um conjunto de módulos independentes, neste caso correspondentes às tarefas definidas nos referidos DFI's. A hierarquia entre os

módulos e as condições que provocam transições entre eles são facilmente retratadas, assim como as trocas de informação existentes.

Da representação do sistema através de um DTE (figura 6.1) é fácil ver que se trata de um sistema centrado em transacções.

O gestor de transacções é o módulo responsável pela atribuição do CPU aos vários módulos, neste caso o sistema de atendimento de interrupções do microcontrolador. Não é pois necessário desenvolver software para esse fim.

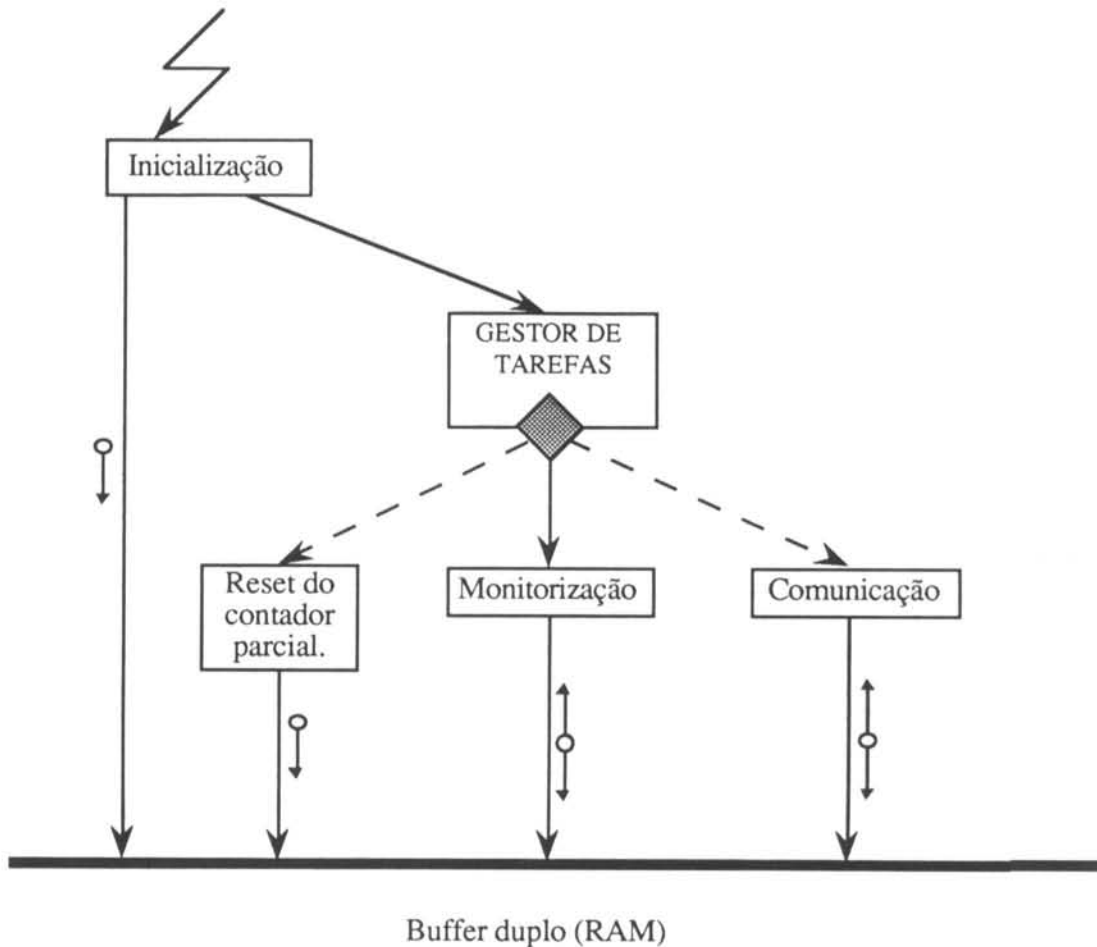


Figura 6.10 - Unidade de aquisição: ME de ordem zero

O passo seguinte foi, para cada módulo do ME de ordem zero, a construção de ME's de ordem superior, i.e., mais detalhados, até se

atingir o grau de pormenor pretendido para a representação. Surgem assim os ME's referentes aos módulos de "Inicialização" (figura 6.11), "Monitorização" (figura 6.12), "Comunicação" (figura 6.13) e "Reset do contador parcial" (figura 6.14).

A construção dos ME's termina a fase de Projecto Estruturado. O conjunto hierarquizado de ME's, juntamente com as Mini-especificações e o Dicionário de Informação já definidos para os DFI's e adequadamente complementados, constituem o produto do trabalho de Análise e Projecto Estruturados: a Especificação de Projecto Estruturado. Com as especificações resultantes do trabalho de Análise e do Projecto Estruturados foi possível passar à fase de desenvolvimento do hardware e do software, a qual será descrita no capítulo 10.

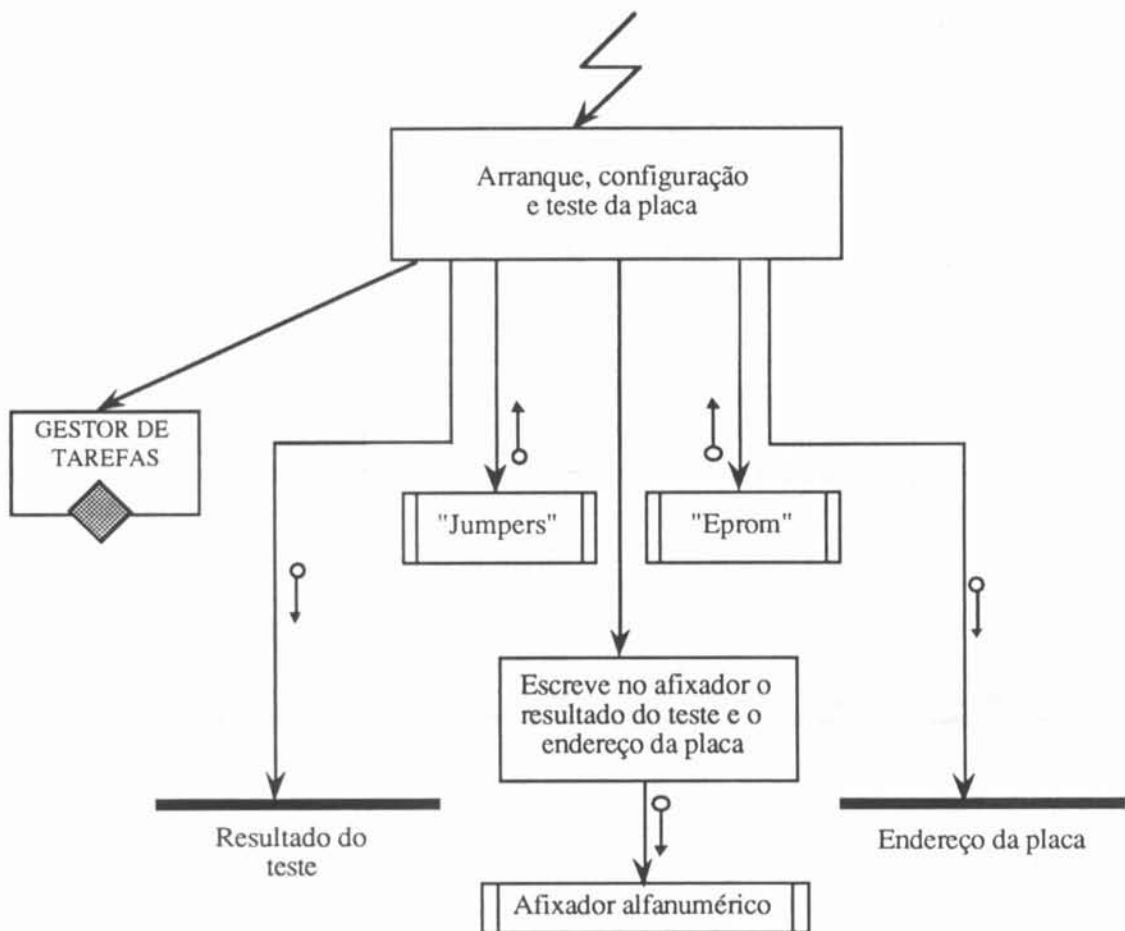


Figura 6.11 - Inicialização: ME de ordem um

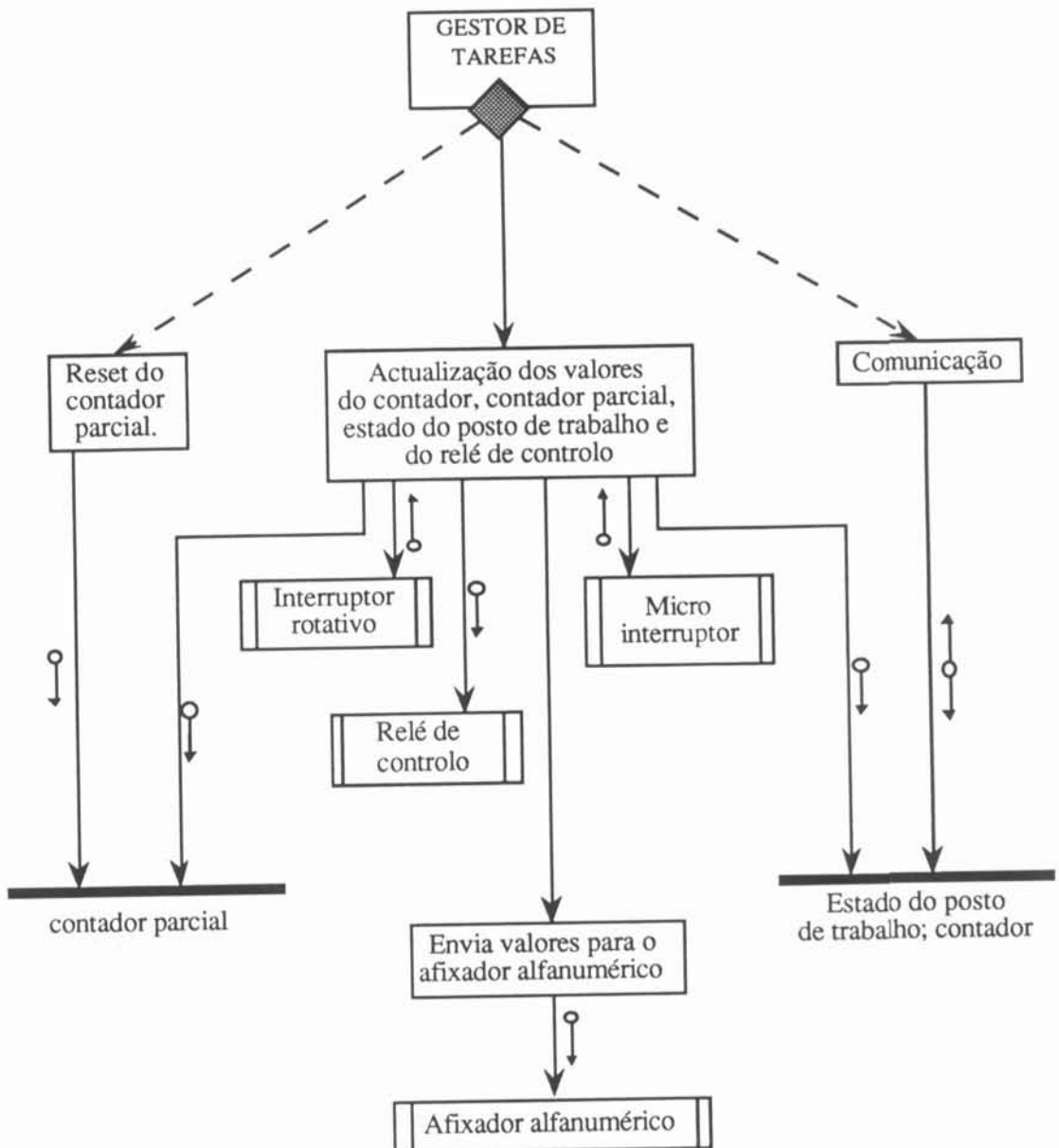


Figura 6.12 - Monitorização: ME de ordem um

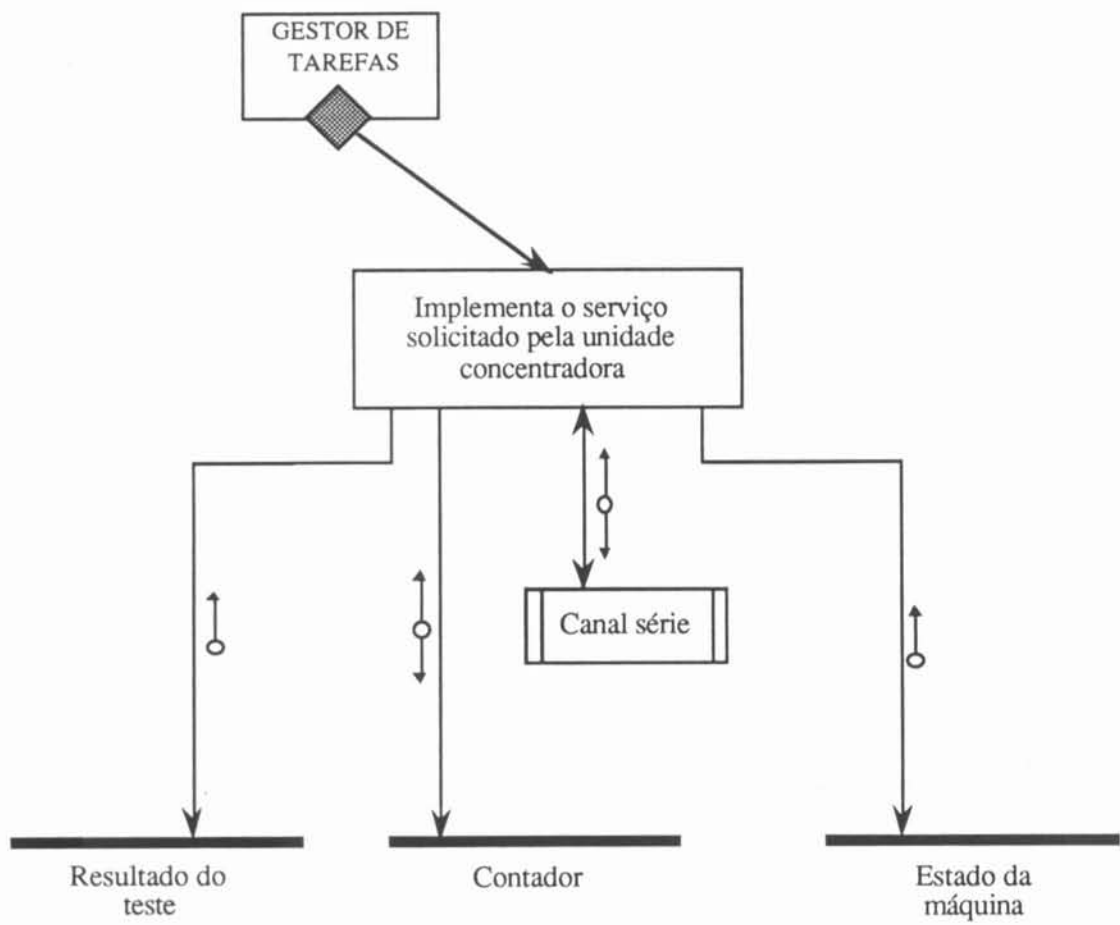


Figura 6.13 - Comunicação: ME de ordem um

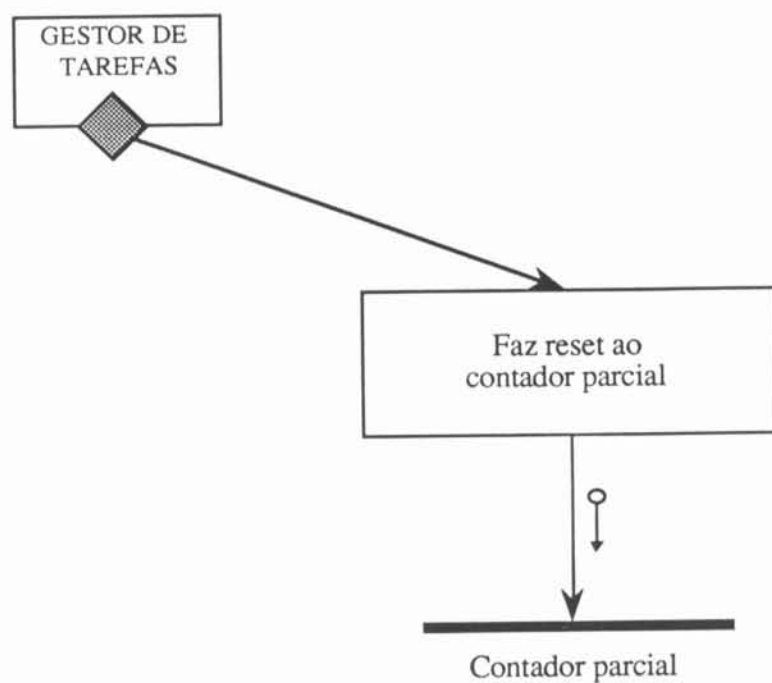


Figura 6.14 - Reset do contador parcial: ME de ordem um

CAPITULO 7

Unidades concentradoras

7 - UNIDADES CONCENTRADORAS

Tal como no capítulo anterior foram primeiro definidos as funções e objectivos das unidades concentradoras, passando-se em seguida à fase de Análise e Projecto Estruturados. No desenvolvimento das unidades concentradoras será utilizada a mesma metodologia que foi usada no desenvolvimento das unidades de aquisição.

7.1 - Funções

As funções destas unidades são as funções das unidades de recolha de dados que não ficaram associadas às unidades de aquisição, i.e.:

- aquisição, através de teclado, do código identificativo de cada posto de trabalho, e dos respectivos códigos de operador e de operação;
- inserção, como mestre, numa rede própria que permita a transferência de blocos de informação e/ou programas entre a unidade concentradora e as unidades de aquisição associadas a essa rede;
- recolha, através da rede própria, da medida de produção e do estado dos postos de trabalho associados às unidades de aquisição presentes nessa rede;
- registo horário de todas as alterações que se verificarem nas variáveis dos postos de trabalho, com a excepção do valor da medida de produção, o qual deve ser simplesmente actualizado ao longo do tempo;
- inserção, como escravo, numa rede BITBUS de modo a permitir a transferência de blocos de memória com informação e/ou programas entre estas unidades e a unidade central.

7.2 - Objectivos

Apresentam-se seguidamente os objectivos fundamentais

levados em conta no desenvolvimento das unidades concentradoras:

- como unidade concentradora e encaminhadora de informação, garantia de que não há perda ou degradação da informação veiculada através dela;
- estruturação da informação referente aos postos de trabalho em blocos, um para cada posto de trabalho, com uma formatação e localização em memória bem definidas por forma a facilitar a transferência, validação e tratamento dessa informação;
- garantia de elevada fiabilidade;
- manutenção da operacionalidade mesmo em caso de falha de comunicação com a unidade central.

7.3 - Análise Estruturada

7.3.1 - Análise de Tarefas

Da análise do conjunto de funções apresentadas em 7.1 foi possível definir para a unidade concentradora três estados possíveis. Existe um estado durante o qual esta recolhe, das unidades de aquisição, a medida de produção e o estado dos postos de trabalho, e efectua o processamento e a actualização desses valores nos blocos respectivos. Um outro estado corresponde ao diálogo com o operador, através do teclado e do afixador alfanumérico, e um terceiro estado diz respeito à gestão da comunicação com a unidade central através da rede BITBUS. Acrescentando a estes três estados os de "Fora de serviço" e de "Inicialização", cujas definições são idênticas às apresentadas em 6.3.1, obteve-se o Diagrama de Transição de Estados da figura 7.1, que se passa a detalhar:

- "Fora de serviço": estado em que fica a unidade quando lhe é cortada a alimentação;
- "Inicialização": estado que se segue ao anterior quando é fornecida alimentação à unidade, e durante o qual se efectuam as inicializações e configurações do microcontrolador e periféricos. Quando estas terminam, o sistema transita para o estado "Varrimento". Durante as

inicializações todas as interrupções do microcontrolador devem estar inibidas;

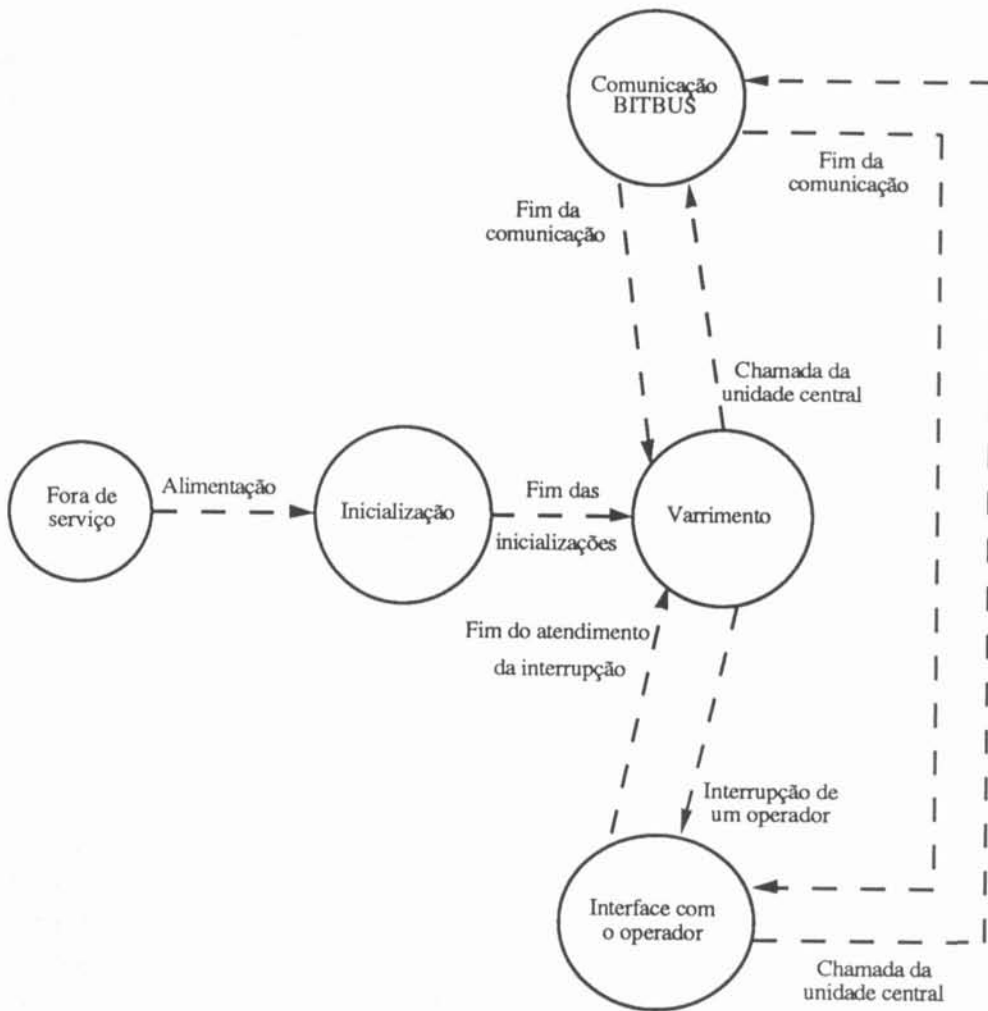


Figura 7.1 - Diagrama de Transição de Estados

- "Varrimento": neste estado a unidade concentradora recolhe de cada unidade de aquisição, através da rede própria, a medida de produção e o estado do respectivo posto de trabalho. O valor do estado recolhido é então comparado com o valor do estado anterior e, caso sejam iguais, somente o valor da medida de produção é actualizado no respectivo bloco de informação. Caso os valores sejam diferentes, são registados a data e hora actuais e o novo estado do posto de trabalho, e a medida de produção é inicializada com zero. O

passo seguinte é verificar se existe algum bloco de informação a ser enviado para ou recebido da unidade de aquisição que está a ser processada e, no caso afirmativo, essa operação é efectuada de imediato, após o que se passa à unidade seguinte. Se durante este processo se verificar algum erro, deve actualizar-se um bloco de registo de erros com um código identificativo do erro e o endereço da respectiva unidade de aquisição. Este processo é repetido ciclicamente, sendo assim efectuado um varrimento a todas as unidades de aquisição ligadas à unidade concentradora;

- "Interface com o operador": quando o operador pretende interagir com a unidade concentradora deve sinalizá-lo através do teclado. Este procedimento desencadeia a passagem para um novo estado, durante o qual o operador pode sinalizar o início ou o término de uma operação, consultar o estado de um posto de trabalho, etc. As duas primeiras são funções básicas de controlo de produção, pelo que serão descritas sucintamente. Para sinalizar o início de uma operação, o operador deve começar por indicar o código do posto de trabalho. O sistema verifica se esse posto de trabalho não está já atribuído a alguma tarefa. Caso esteja, informa o operador de tal facto e não permite que se prossiga com a inicialização. No caso contrário, pede ao operador para introduzir o seu código identificativo e o código da operação que ele vai efectuar. Em seguida são registados a data e hora actuais e o estado do posto de trabalho, e a medida de produção é inicializada com zero. Para sinalizar o término de uma operação, o operador só tem de indicar o respectivo posto de trabalho e o sistema registará a data e hora actuais, o estado e o valor da medida de produção. Assumiu-se que se durante o desenvolvimento surgisse a necessidade de implementar outras funções, elas seriam alvo de posterior caracterização;
- "Comunicação BITBUS": sempre que a unidade central solicita o envio de informações através da rede BITBUS, o sistema transita para este estado. O diálogo entre a unidade central e a unidade concentradora pode ser implementado utilizando os comandos de acesso e controlo remoto (RAC) do BITBUS, descritos no apêndice A, que já se encontram no "firmware" do microcontrolador, ou poderá ser suportado por uma tarefa especificamente desenvolvida para esse fim. No primeiro caso não será necessário desenvolver software de comunicação para a unidade concentradora.

7.3.2 - Diagramas de Fluxo de Informação

Na figura 7.2 apresenta-se o DFI de ordem zero referente às unidades concentradoras. Este DFI tem uma única tarefa, que implementa as funções necessárias ao funcionamento descrito no Diagrama de Transição de Estados. A construção deste DFI baseou-se nos seguintes pressupostos:

- o operador introduz através do teclado comandos (por exemplo, sinalização de início ou fim de operação) e informações (códigos de operador, operação e posto de trabalho, etc.) para a unidade concentradora. Tal como foi referido em 5.3, o sistema deverá ficar aberto à inclusão de um leitor de código de barras ou de cartão magnético para introdução destas informações. Para o afixador alfanumérico são enviadas mensagens contendo informações que se destinam aos operadores;



Figura 7.2 - Unidade concentradora: DFI de ordem zero

- as interfaces à rede BITBUS (canal BITBUS) e à rede própria (canal série) funcionam como fontes e poços de informação. Através da rede BITBUS chegam e são enviadas informações da e para a unidade central, enquanto que a rede própria desempenha o mesmo papel relativamente às unidades de aquisição;
- dado que se pretende fazer o registo horário das alterações verificadas nos postos de trabalho, e uma vez que a unidade concentradora deve funcionar mesmo em caso de falha da comunicação com a unidade central, impôs-se a inclusão de um relógio de tempo real;
- na RAM da unidade concentradora existe, para cada unidade de aquisição, um bloco de informação onde são armazenadas as informações referentes aos postos de trabalho para posterior leitura pela unidade central. É também criado um "buffer" para suportar a transferência de blocos de informação entre a unidade central e as unidades de aquisição, e um outro para registo de erros detectados pela unidade concentradora, quer erros detectados pela interface à rede própria, como a impossibilidade de aceder a uma unidade de aquisição, quer pela tarefa "Varrimento" quando, por exemplo, detecta um "overflow" do "buffer" de dados de uma unidade de aquisição;
- o endereço da unidade concentradora na rede BITBUS é programado através de "jumpers";
- O conjunto de variáveis e parâmetros de inicialização (inicialização dos periféricos, etc.) e configuração da unidade está armazenado em EPROM.

As figuras 7.3, 7.4, 7.6 e 7.7 apresentam os DFI's de ordem um correspondentes aos estados em que o sistema se pode encontrar, e a figura 7.5 apresenta, como exemplo, o DFI de ordem dois do estado "Varrimento".

Durante a execução de todas as tarefas representadas neste último DFI é verificada a ocorrência de erros. No caso de ser detectado algum erro, ele é registado no "buffer" respectivo. Os fluxos de informação correspondentes não estão porém representados na figura para não a sobrecarregar.

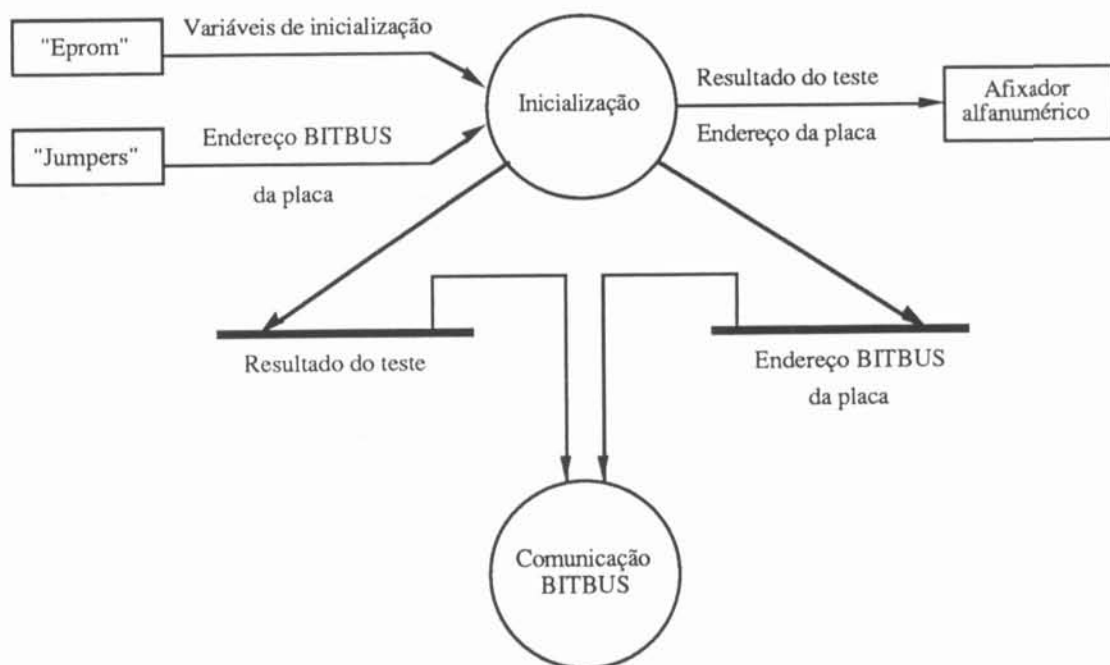


Figura 7.3 - Inicialização: DFI de ordem um

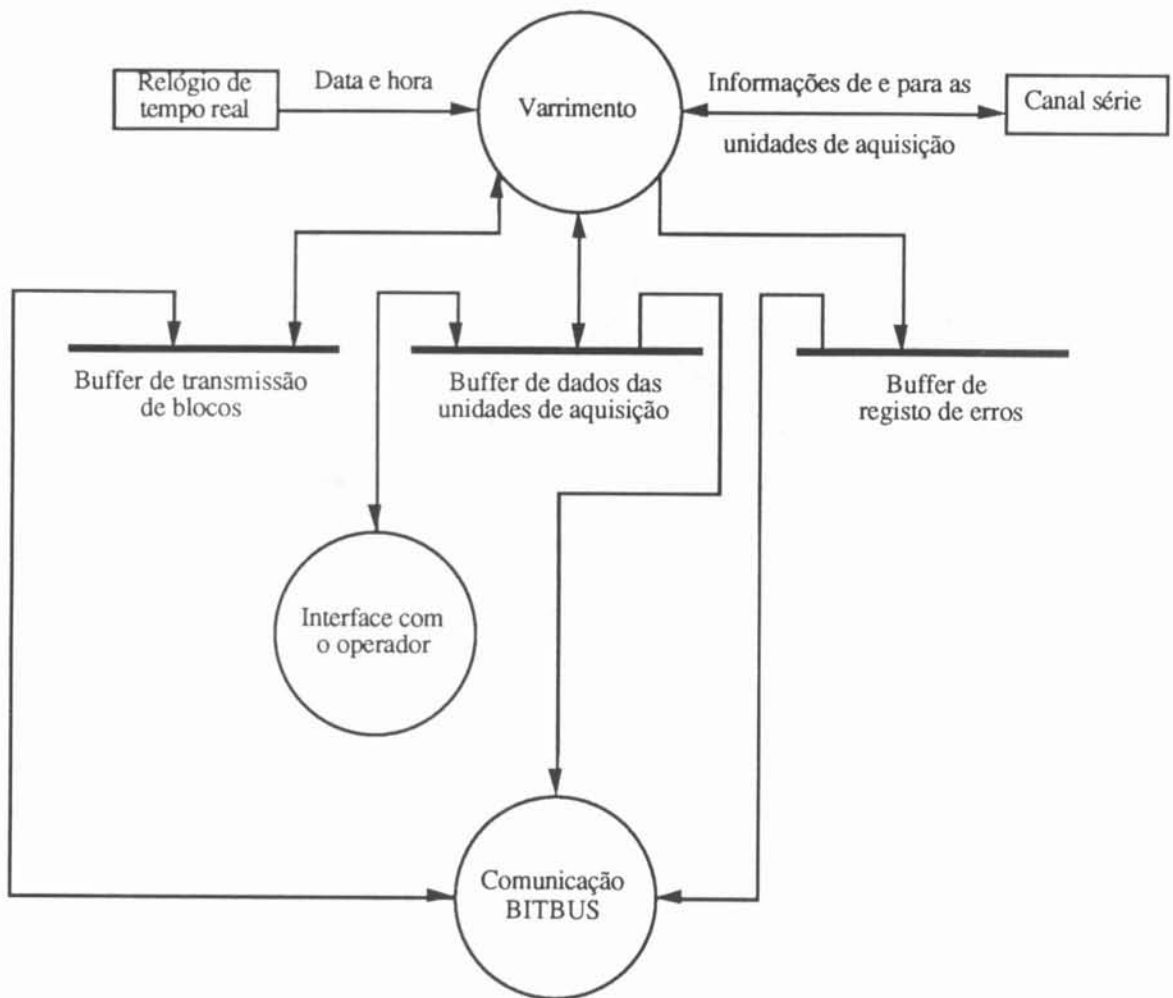


Figura 7.4 - Varrimento: DFI de ordem um

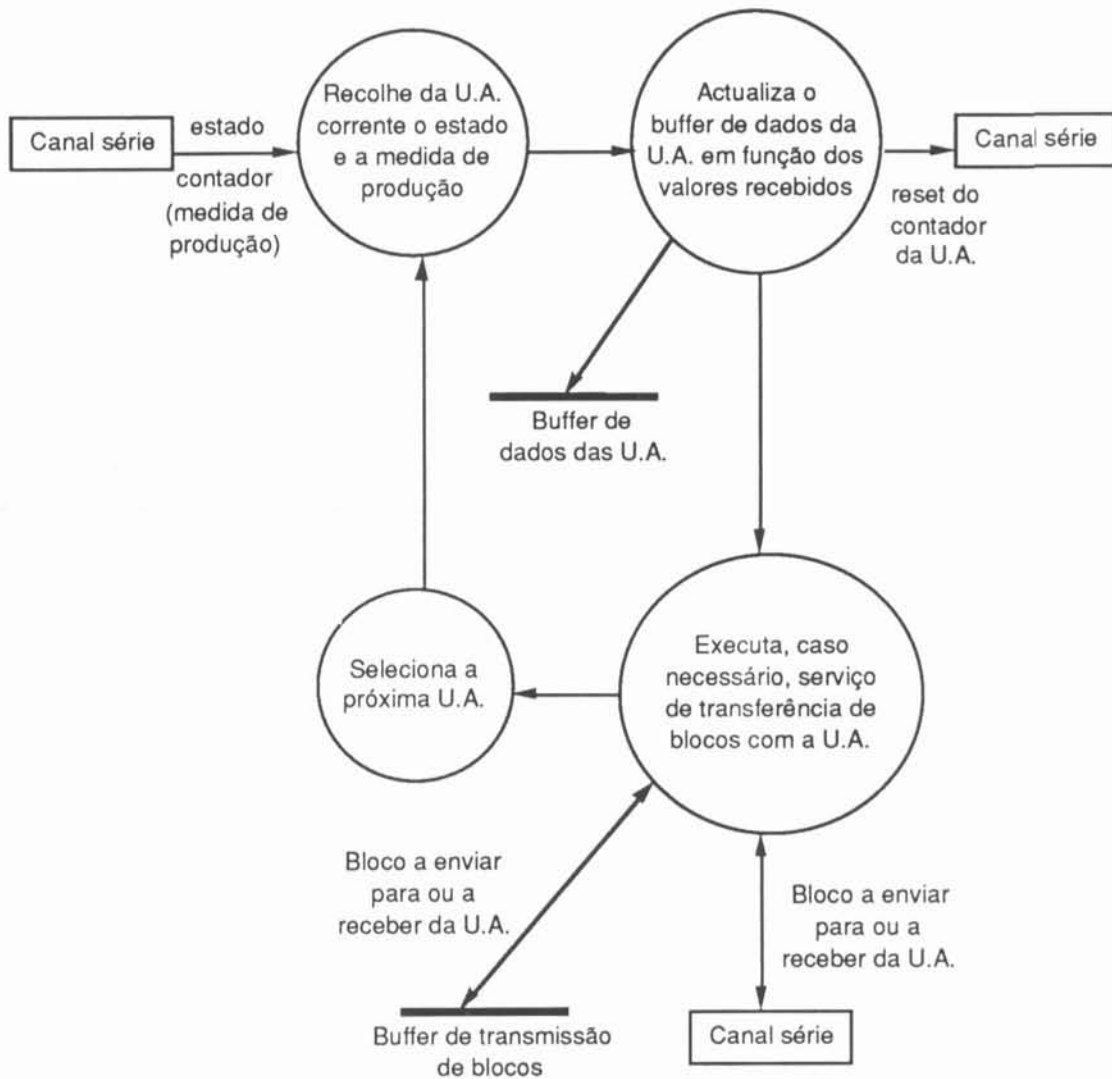


Figura 7.5 - Varrimento: DFI de ordem dois

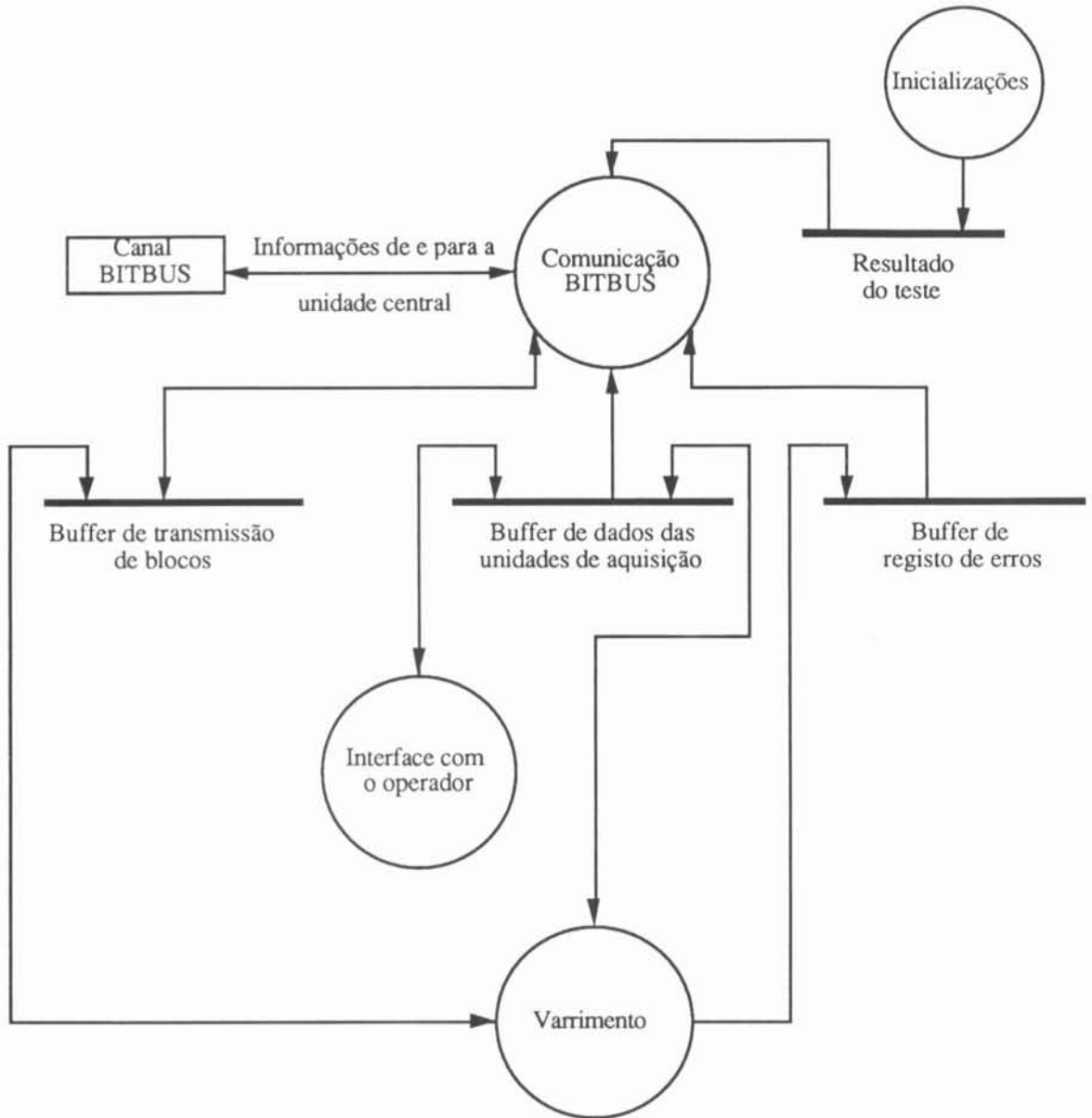


Figura 7.6 - Comunicação BITBUS: DFI de ordem um

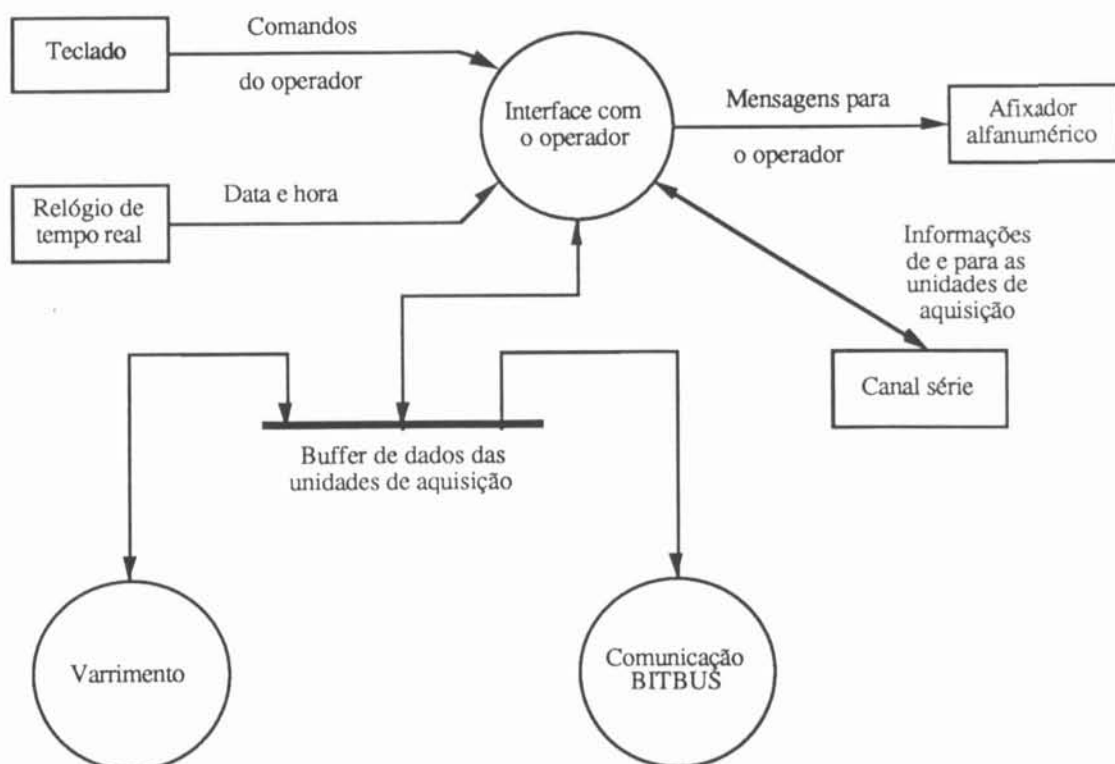


Figura 7.7 - Interface com o operador: DFI de ordem um

7.3.3 - Mini-especificações

Foram utilizados fluxogramas para descrever o funcionamento interno das tarefas dos DFI's de ordem superior. A figura 7.8 apresenta, como exemplo, o fluxograma referente à tarefa "Recolhe da U.A. corrente o estado e medida de produção", representado no DFI de ordem dois da tarefa "Varrimento".

7.4 - Projecto Estruturado

7.4.1 - Análise de Transacções e Análise de Transformadas

Tal como no caso das unidades de aquisição, a obtenção do modelo do sistema das unidades concentradoras baseou-se na aplicação de Análise de Transacções.



Figura 7.8 - Mini-especificação da tarefa "Recolhe da U.A. corrente o estado e a medida de produção"

7.4.2 - Mapas de Estrutura

A figura 7.9 apresenta o ME de ordem zero. No caso das unidades concentradoras o gestor, ou centro, de transacções é o sistema operativo do microcontrolador 8044, aliás papel que se lhe adequa perfeitamente uma vez que é um sistema multitarefa de tempo real com mecanismos próprios de atribuição do CPU em função da prioridade atribuída às tarefas. Os ME's de ordem um estão representados nas figuras 7.10, 7.11, 7.12 e 7.13.

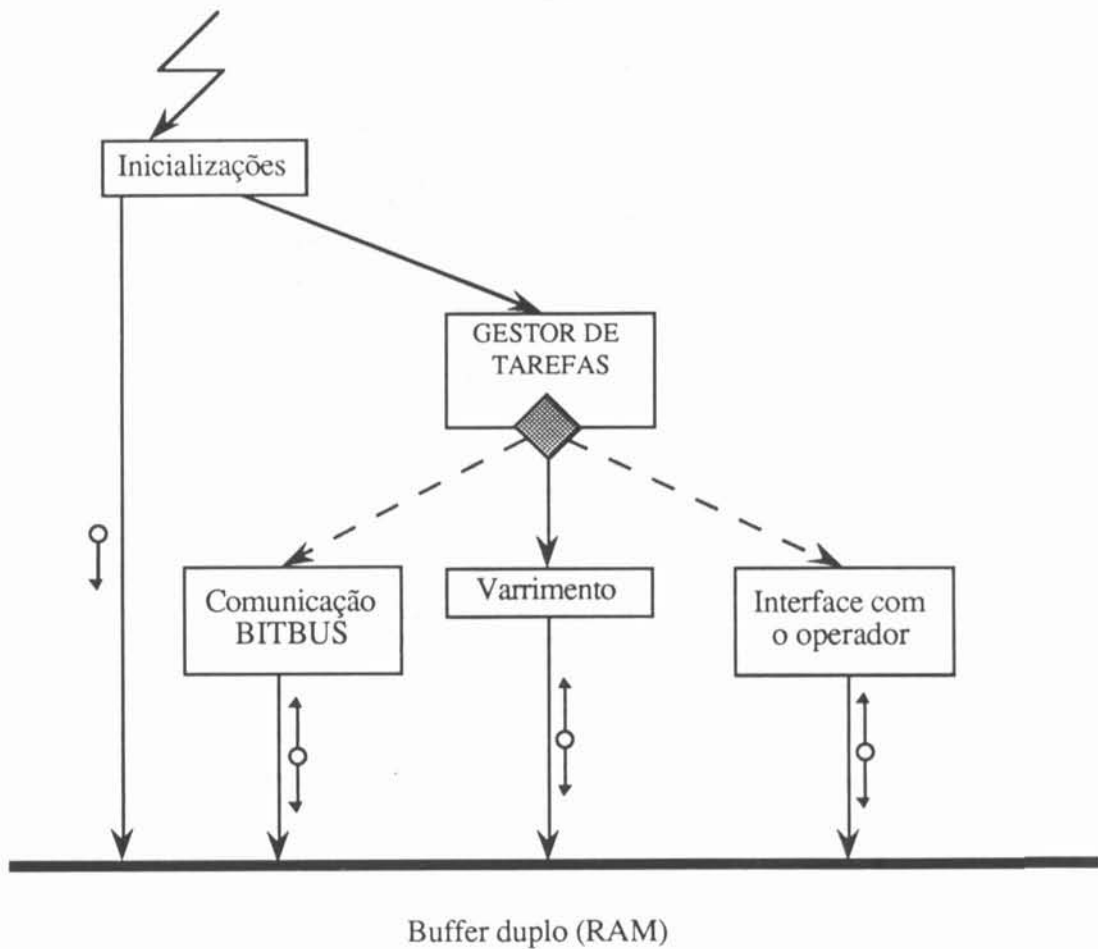


Figura 7.9 - Unidade concentradora: ME de ordem zero

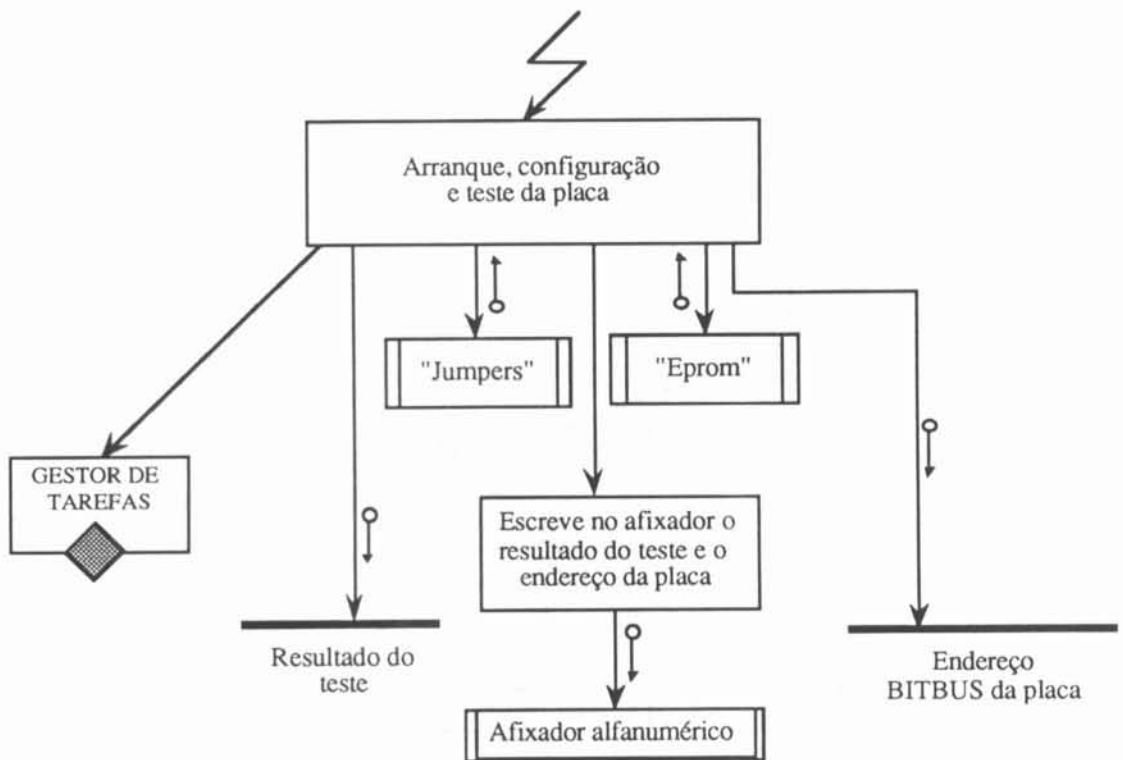


Figura 7.10 - Inicialização: ME de ordem um

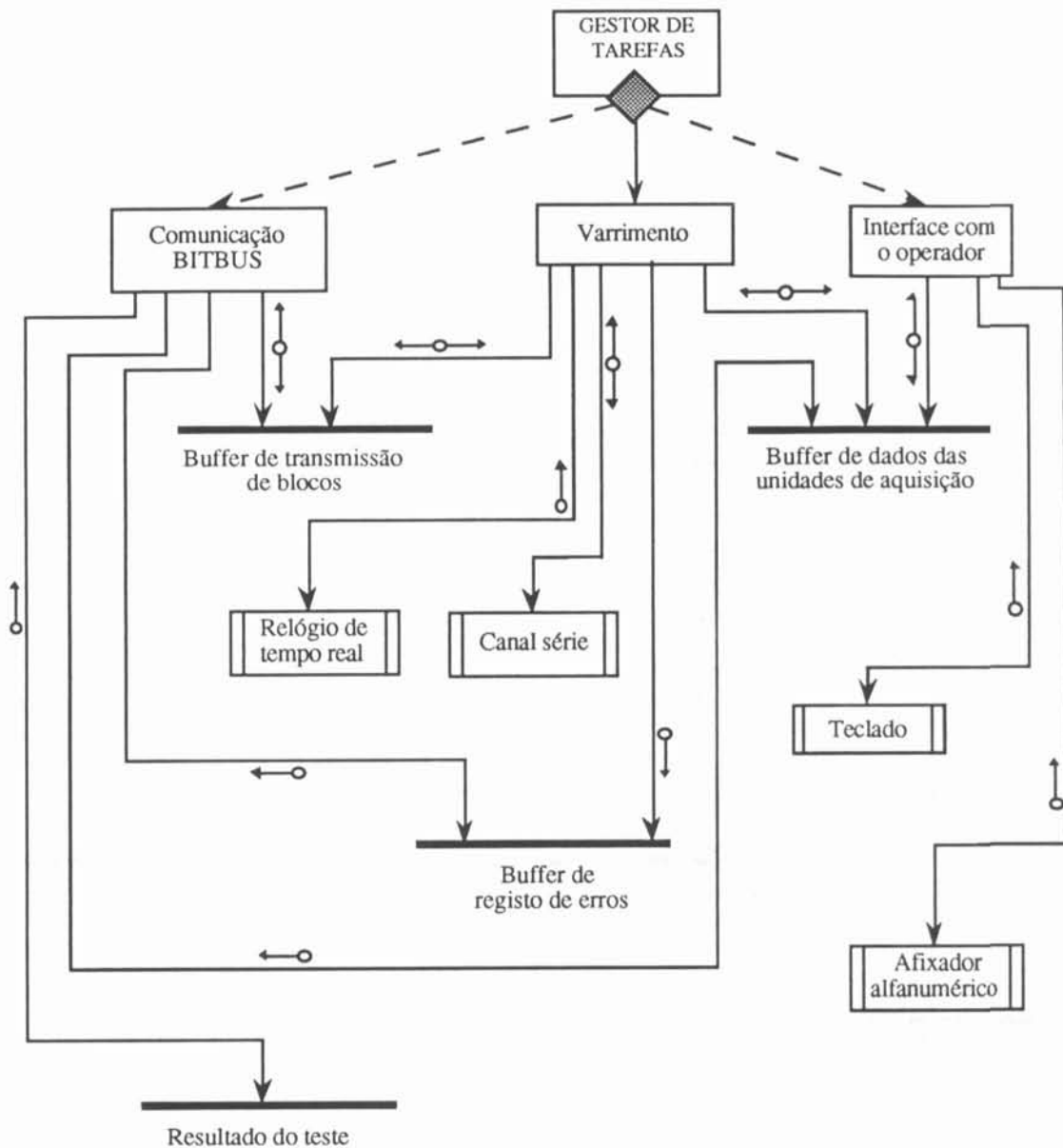


Figura 7.11 - Varrimento: ME de ordem um

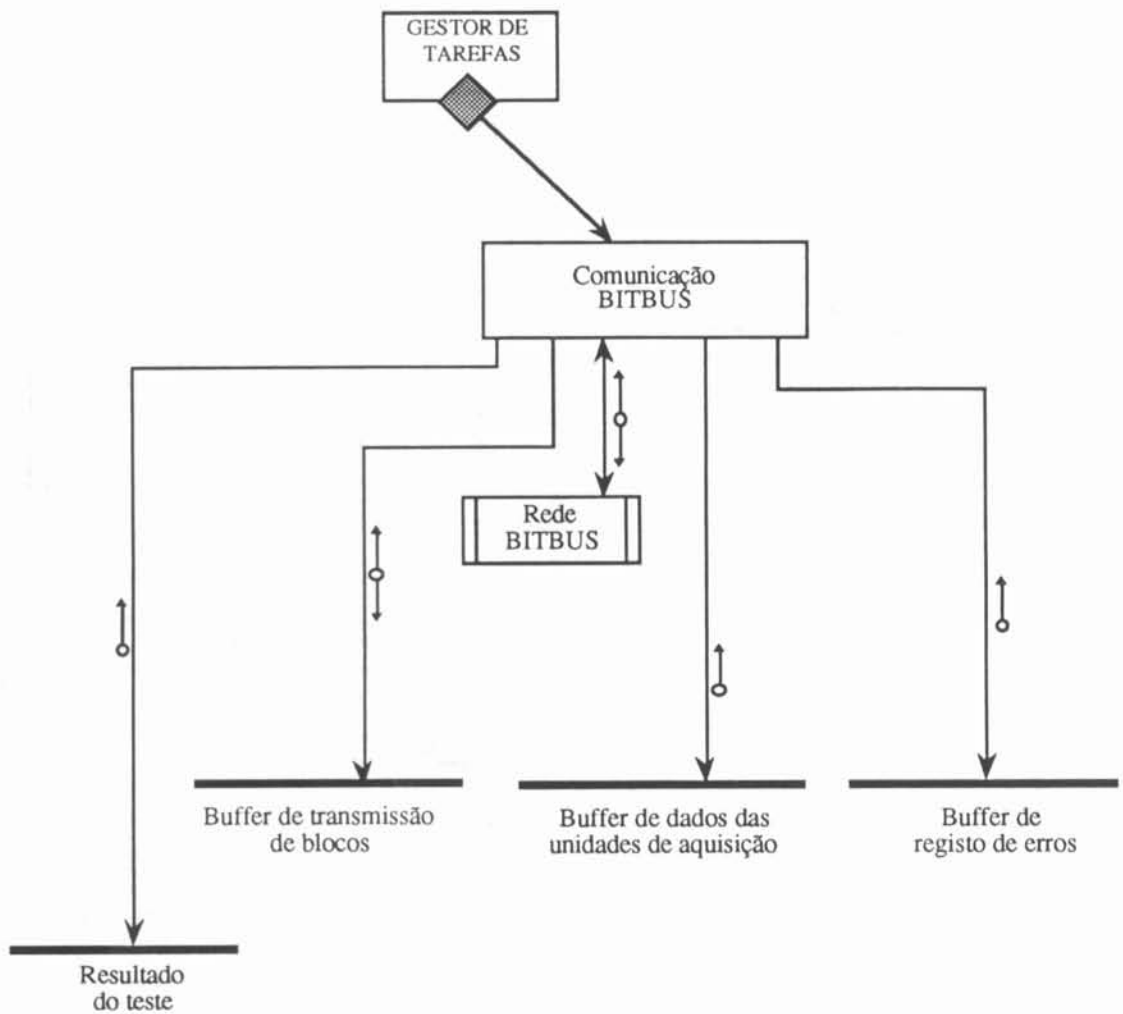


Figura 7.12 - Comunicação BITBUS: ME de ordem um

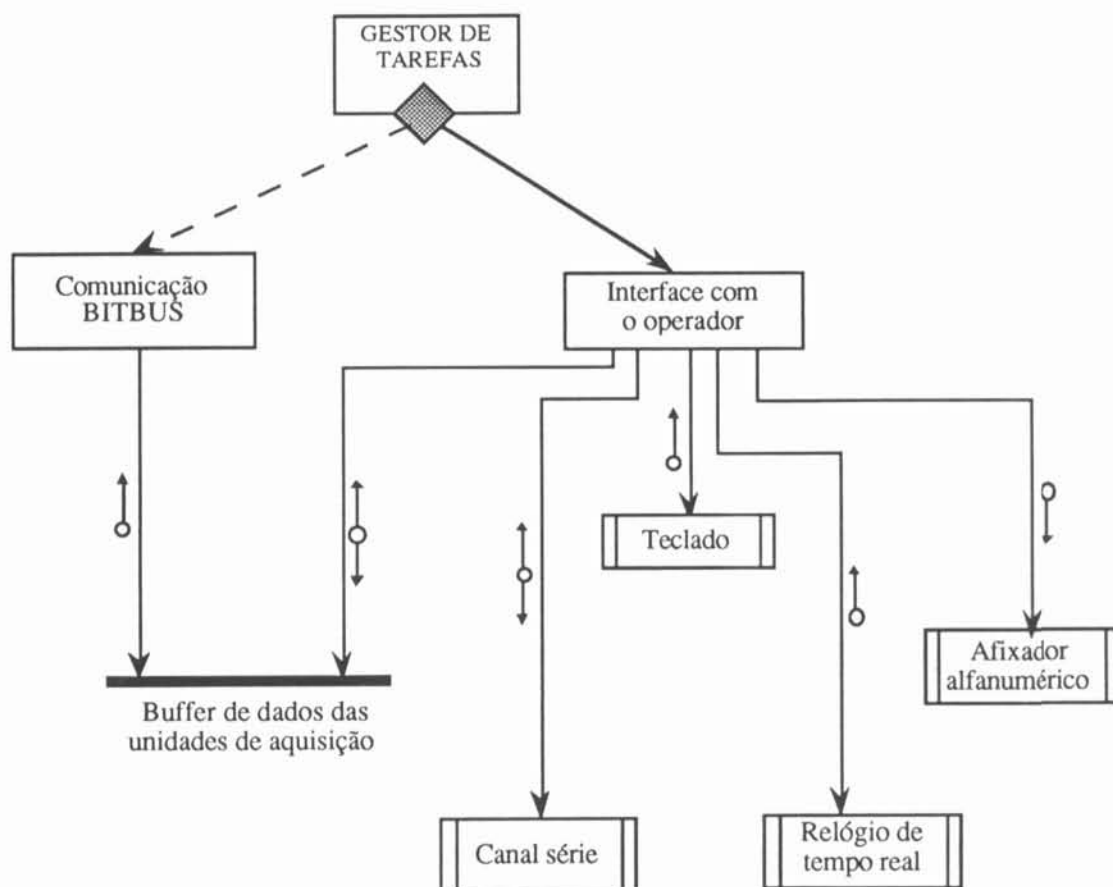


Figura 7.13 - Interface com o operador: ME de ordem um

CAPITULO 8

Rede própria

8 - REDE PROPRIA

8.1 - Introdução

A necessidade de desenvolvimento de uma rede de comunicações própria, para interligar unidades de aquisição e unidades concentradoras, foi uma das consequências da escolha da arquitectura do sistema cuja discussão se apresentou em 5.4. Foi pois iniciado o projecto de uma rede o mais simples possível, satisfazendo a especificação apresentada no ponto seguinte.

Tal como foi referido no capítulo 5, poderia ser utilizado, como protocolo da rede própria, um subconjunto do software de comunicações da rede BITBUS, tirando assim partido dos conjuntos de instruções dos dois microcontroladores serem iguais. A arquitectura da rede é pois semelhante à da rede BITBUS. Nos parágrafos seguintes definir-se-ão as características particulares da rede implementada e, em resultado dessa definição, os vários níveis dessa rede, sendo discutidas as semelhanças e diferenças em relação ao BITBUS. A descrição que se segue tem como base a estrutura em níveis definida pelo modelo OSI [3] e pressupõe a implementação, a exemplo do que acontece no BITBUS, dos níveis um (físico), dois (ligação lógica) e sete (aplicação).

8.2 - Características

Apresentam-se seguidamente as características pretendidas para a rede própria desenvolvida, algumas das quais são semelhantes às características definidas, em 3.1, para a rede de ligação entre a unidade central e as unidades de recolha de dados, i.e.:

- especificação de rede industrial, ou seja garantia das suas características mesmo em ambientes adversos;
- elevada fiabilidade;
- baixo custo de instalação e manutenção;
- adequação ao tráfico de blocos de informação de pequena dimensão;
- arquitectura o mais semelhante possível à da rede BITBUS;

- implementação de um conjunto de serviços adequado às necessidades de troca de informação entre as unidades concentradoras e de aquisição;
- minimização do tempo de ocupação dos microcontroladores.

8.3 - Serviços a implementar na rede (nível aplicação)

A rede BITBUS inclui um conjunto de serviços de acesso e controlo remoto aos nós (RAC), que cobrem a maioria das operações que o mestre da rede pode efectuar nesses nós. A definição dos serviços de comunicação necessários entre a unidade concentradora e as unidades de aquisição, permitiu decidir da inclusão na rede própria de todos os serviços considerados no BITBUS, de apenas um subconjunto desses serviços e/ou da implementação de alguns serviços específicos.

O desenvolvimento efectuado até este ponto permitiu concluir que a operação de comunicação mais frequentemente executada entre a unidade concentradora e as unidades de aquisição seria a recolha da medida de produção e do estado do posto de trabalho respectivo. A unidade concentradora poderia também fazer a reinicialização do contador das unidades de aquisição, e ainda enviar para ou receber das unidades de aquisição blocos genéricos de memória.

Todas estas operações poderiam ser implementadas utilizando a última, porque quer o contador quer o estado do posto de trabalho estão registados em memória RAM. No entanto, um mecanismo desse tipo implicaria que, conjuntamente com um código identificativo do serviço pretendido, a unidade concentradora tivesse de enviar sempre os endereços onde essas variáveis se encontram. Sendo possível definir um endereço fixo para a localização em memória destas variáveis, e uma vez que estas vão ser operações repetidas a uma cadência elevada, houve vantagem em definir um serviço correspondente ao envio, pela unidade de aquisição, do respectivo contador e estado e um outro que efectua reinicialização do respectivo contador. Neste caso, a unidade concentradora só tem de enviar para a unidade de aquisição pretendida o código identificativo do serviço pretendido, o que resulta numa maior eficiência. Qualquer outra transferência de informação utiliza o serviço de envio ou recepção de blocos genéricos.

O BITBUS tem serviços para implementar esta última função de transferência de blocos de memória, designados por "Upload

Memory" e "Download Memory" (apêndice A). Foi pois utilizada uma estrutura semelhante para implementar este serviço na rede própria cuja descrição é feita a seguir. A estrutura dos outros dois serviços foi definida em função da sua descrição já apresentada anteriormente.

Segue-se uma breve descrição dos serviços existentes na rede própria:

- "Recebe Status": quando a unidade concentradora solicita este serviço, a unidade de aquisição envia-lhe o valor do contador (medida de produção) e do estado respectivos. A mensagem enviada pela unidade concentradora tem uma estrutura semelhante à apresentada na figura 8.1 (comando), sendo também apresentada na figura 8.1 (resposta) a estrutura correspondente à mensagem enviada pela unidade de aquisição. O campo "comando" é utilizado para enviar o código identificativo do serviço, e o campo "resposta" para enviar um código de confirmação de execução do serviço ou, em caso de erro, um código identificativo desse erro. Note-se que as estruturas apresentadas são acrescidas de informação complementar, nomeadamente endereços, comprimento da mensagem, detecção e correcção de erros de transmissão, etc., a qual é gerada nos outros níveis da rede (níveis dois e um).

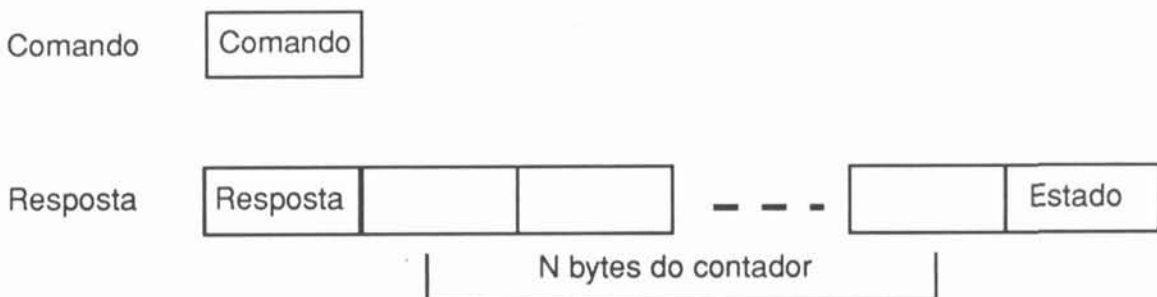


Figura 8.1 - Serviço "Recebe status"

- "Reset do contador": a estrutura deste serviço é muito semelhante à do anterior, com a diferença de que, neste caso, a unidade de aquisição não envia informação para a

unidade concentradora devendo, no entanto, enviar a confirmação da execução do serviço solicitado (figura 8.2).

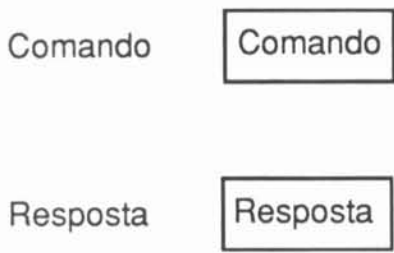


Figura 8.2 - Serviço "Reset do contador"

- "Envia bloco": neste caso a unidade concentradora envia o código do serviço, o endereço da primeira posição onde pretende escrever, o comprimento do bloco e o conjunto de valores a escrever. No BITBUS esta operação de escrita na memória é seguida de uma operação de leitura das posições escritas, para confirmação do sucesso da operação. Este mecanismo de verificação está também implementado nesta rede. A resposta enviada pela unidade de aquisição, confirmando ou não a execução do serviço, é igual à utilizada no serviço "Reset do contador". Na figura 8.3 está representada a estrutura das mensagens deste serviço.

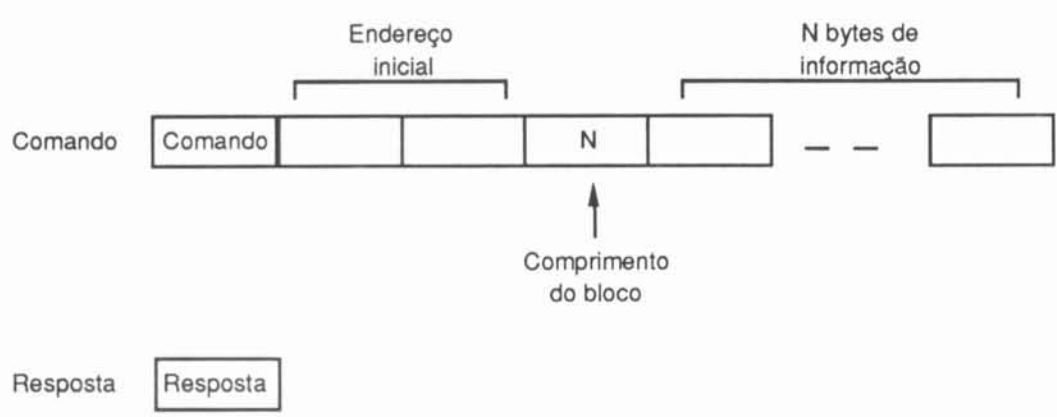


Figura 8.3 - Serviço "Envia bloco"

- "Recebe bloco": a unidade concentradora solicita este serviço enviando o respectivo código, o endereço da primeira posição e o comprimento do bloco que pretende ler (figura 8.4). A unidade de aquisição responde com uma estrutura que contém o código "resposta" seguido do bloco solicitado.

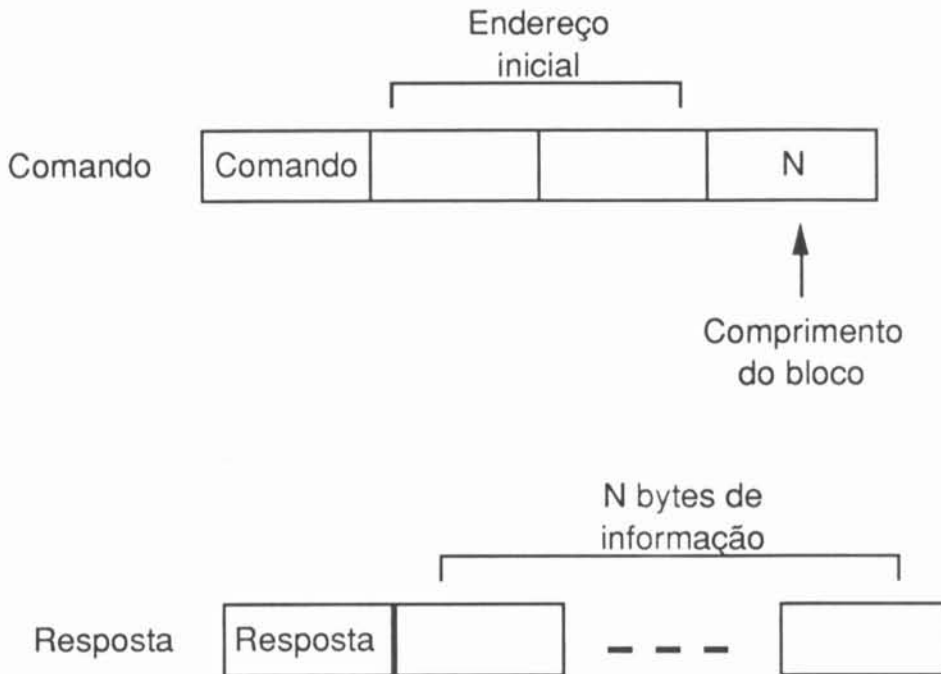


Figura 8.4 - Serviço "Recebe bloco"

Existe pois uma grande semelhança entre esta e a estrutura das mensagens do BITBUS, com a exceção do segundo byte (MT,SE,DE e TR) e do quarto byte (SOURCE TASK / DESTINATION TASK), que não se aplicam neste caso (apêndice A).

Este conjunto de quatro mensagens cobre todas as necessidades de comunicação entre a unidade concentradora e as unidades de aquisição, identificadas até esta fase do desenvolvimento, pelo que não foram implementadas outras. Se futuramente for sentida a necessidade de um conjunto de serviços mais completo será extremamente fácil acrescentá-los aos existentes.

Os códigos identificativos de cada um dos serviços atrás descritas são os que constam da figura 8.5.

1 - Comandos:

Nome do serviço	Código
"Recebe Status"	01H
"Reinicialização do Contador"	02H
"Envia Bloco"	03H
"Recebe Bloco"	04H

2 - Respostas:

Resposta	Código
"Serviço Executado"	00H
"Erro na Execução"	(diferente de 00H)

Figura 8.5 - Códigos identificativos utilizados nos serviços da rede própria

Quando a Resposta não for igual a zero, isso significa que se verificou um erro na execução do serviço. Se houver necessidade de identificar o tipo de erro verificado, para futuro processamento e/ou tentativa de recuperação, este campo poderá ser utilizado para codificar vários tipos de erro.

8.4 - Protocolo de ligação (nível de ligação lógica)

A simplicidade da comunicação descrita, associada a vários factores que serão em seguida discutidos, permitiu simplificar o protocolo de ligação exigível para a rede própria, relativamente ao utilizado no BITBUS.

Esta simplificação baseia-se na consideração de quatro factores fundamentais:

- No caso da rede BITBUS o protocolo de ligação tem de ter em conta que, quer no mestre quer nos escravos, existe um microcontrolador com um sistema operativo multitarefa, no qual podem existir tarefas, criadas pelos utilizadores, que comunicam entre si. Além dessas tarefas existe ainda a tarefa que implementa os serviços de Acesso e Controlo Remotos aos nós (RAC), denominada TASK 0 (apêndice B), a qual utiliza o mesmo formato de mensagem. Por outro lado, existem quatro possíveis origens ou destinos para a informação: outra tarefa no mesmo microcontrolador ou na sua extensão e outra tarefa noutra nó da rede ou na sua extensão. Para gerir o tráfego de todos estes pacotes de informação são necessários mecanismos que provocam um "overhead" significativo do sistema relativamente à quantidade de informação útil enviada. Tal não se verifica no caso da rede própria desenvolvida. O protocolo implementado é com efeito um protocolo simples do tipo "mestre solicita um serviço ao escravo e aguarda que, dentro de um intervalo de tempo a definir, o escravo responda ao pedido". Além disso, no CPU de cada unidade de aquisição (8031) não existe sistema operativo multitarefa, e no da unidade concentradora (8044) existe uma única tarefa suportando a comunicação.
- Uma vez que a quantidade de informação que transita na rede é, na maior parte dos casos, muito pequena optou-se por um método assíncrono de comunicação. Este facto, conjugado com a simplicidade do protocolo, levou a dispensar a necessidade da implementação de tramas de sincronização e supervisão, que existem no caso do BITBUS (apêndice A), passando a existir unicamente tramas de informação.
- A fiabilidade da transmissão da informação através da rede requer especial atenção. Para este efeito, é utilizado um mecanismo de detecção de erro igual ao do BITBUS, baseado em códigos cíclicos de detecção de erros, e cuja descrição se apresenta no apêndice A. Em caso de erro a trama é ignorada, o serviço solicitado é anulado e poderá eventualmente ser repetido.

- O primeiro e último campo de uma trama BITBUS (apêndice A) são campos limitadores constituídos por uma sequência fixa (01111110). Estes limitadores são utilizados, pelo bloco de interface à rede do 8044 (SIU), para identificar o início e o fim da trama. Como no caso da rede própria esse bloco não existe, estes limitadores não são necessários.

Pelo exposto nos parágrafos anteriores pode concluir-se que é possível utilizar um protocolo bastante mais simples do que o BITBUS. A informação acrescentada às estruturas definidas no ponto 8.3 foi unicamente:

- a) um byte com o endereço da unidade de aquisição. No caso de envio de um pacote pela unidade concentradora, este byte será interpretado como endereço de destino. No caso de ser uma resposta de uma unidade de aquisição, será utilizado pela unidade concentradora para verificar se a resposta teve origem na unidade de aquisição correcta. É conveniente que este seja o primeiro campo da trama porque, analisando-o, cada unidade de aquisição pode verificar se a trama se destina a ela e, caso isso não se verifique, pode abandonar de imediato a recepção da mesma trama;
- b) um byte indicando o comprimento da trama, necessário dado terem deixado de existir os limitadores respectivos. Para limitar o tempo de ocupação da rede por cada unidade de aquisição o comprimento máximo permitido para os pacotes deve ser reduzido. Este problema só se coloca na transmissão de blocos de informação, pelo que esse valor deve ser ajustado em função das necessidades específicas de cada aplicação. O comprimento da trama ficou limitado a um máximo de 256 bytes, pelo facto de se usar um byte para o codificar. Este byte deve seguir-se ao do endereço. Nos serviços apresentados no ponto 8.3, onde é definido um comprimento ("envia bloco" e "recebe bloco"), esta estrutura foi modificada dado ser redundante a inclusão de dois comprimentos. Nesses dois casos, o comprimento do bloco de informação é igual ao comprimento em bytes da trama subtraído do número de bytes que não são de informação, cujo número é conhecido. Este mecanismo é, aliás, o utilizado pelo BITBUS;
- c) dois bytes para detecção de erros de transmissão. São, como no BITBUS, os últimos dois bytes. As iniciais utilizadas na figura resultam do termo inglês para

"sequência de verificação da trama" (Frame Check Sequence).

Na figura 8.6 apresenta-se a estrutura genérica de uma trama, indicando o número de bytes de cada campo. Na figura 8.7 estão representadas, a título de exemplo, as tramas enviada e recebida pela unidade concentradora quando solicita o serviço "Recebe Status".



Figura 8.6 - Estrutura de uma trama

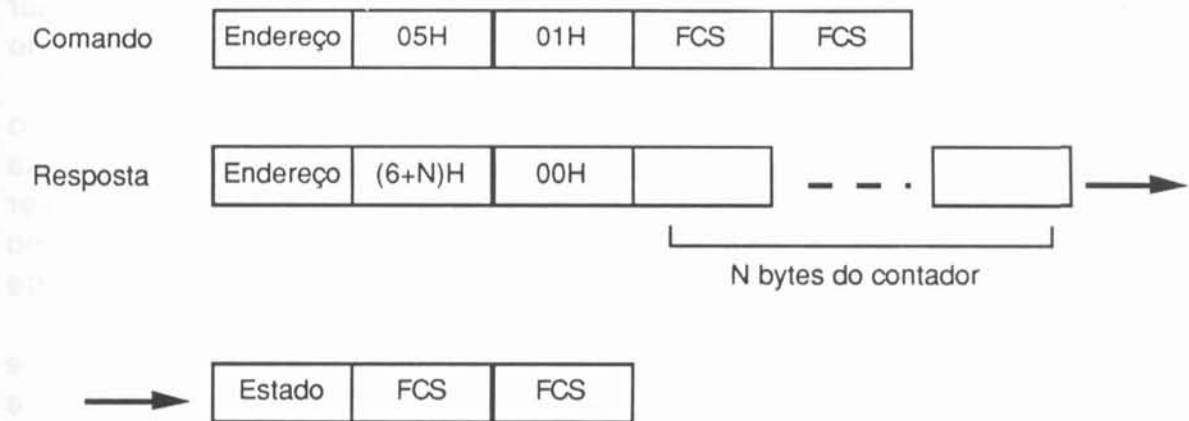


Figura 8.7 - Serviço "Recebe status": Tramas enviada e recebida

8.5 - Nível físico

As especificações eléctricas utilizadas pelo BITBUS adequam-se perfeitamente às características da rede própria, pelo que

foram seguidas. Foi no entanto possível, devido à especificidade deste caso, simplificar um pouco essa interface eléctrica.

A área a cobrir por uma rede deste tipo é muito menor do que no caso do BITBUS, uma vez que ela se limita a um sector da planta fabril. No ponto 8.4 especificou-se que se utilizará um método assíncrono de comunicação devido à quantidade de informação a enviar pela rede ser pequena, e porque isso permite simplificar o protocolo. Sendo pequeno o volume de informação a transmitir podem utilizar-se taxas de transmissão mais baixas do que os 62.5 Kbps especificados no BITBUS como velocidade mínima. Daqui resultam duas vantagens:

- é possível cobrir maiores distâncias com a mesma rede, ou cobrir a mesma distância sem recorrer a repetidores;
- diminui-se a probabilidade de ocorrência de erros de transmissão.

Tomando a opção de não implementar a função de repetidor nas unidades de aquisição, consegue-se diminuir o custo do sistema e o custo da cablagem, uma vez que se passa a utilizar cabo de um par entrançado de condutores em lugar de dois pares, como no caso do BITBUS.

Um ponto importante a considerar aqui é o facto de, quer no 8031, quer na interface do 8044 à rede própria, não existir uma unidade de interface série do tipo da que existe para o BITBUS (ver apêndice C). Foi portanto necessário implementar um mecanismo que desempenhasse essas funções com um mínimo de processamento adicional para o CPU dos microcontroladores.

Abordar-se-á em seguida o desenvolvimento da interface entre a rede e os microcontroladores da unidade concentradora e das unidades de aquisição.

O microcontrolador 8031 dispõe de uma porta série programável, que pode ser utilizada na ligação à rede. É possível configurá-la para os modos de comunicação mais usuais, quer no que respeita às taxas de transmissão, quer no controlo de paridade, e ainda para gerar interrupções ao microcontrolador no caso de recepção de um byte. No caso das unidades concentradoras, a porta série do 8044 está já ocupada pela interface à rede BITBUS, pelo que será necessário incluir nestas unidades uma USART (Universal Synchronous/Asynchronous Receiver Transmitter) para implementar a interface entre o microcontrolador e a rede própria.

O objectivo da inclusão desta interface série é o de aliviar o mais possível o CPU dos microcontroladores. Uma das formas de se

conseguir este objectivo é implementar um mecanismo de interrupção do microcontrolador no caso de recepção de um byte na porta série, garantindo que ele só será interrompido se a informação lhe for destinada. Isto permite ultrapassar o problema da utilização de uma topologia em "bus", na qual quando uma unidade qualquer envia algum byte para a rede todas as outras unidades o vão receber.

Todas as informações que chegam à porta série da unidade concentradora são-lhe destinadas, devido à filosofia mestre/escravo da rede, pelo que poderemos associar a USART a uma linha de interrupção do microcontrolador 8044. No caso das unidades de aquisição não foi evidentemente possível aplicar este mecanismo. O ideal seria que o microcontrolador de uma unidade de aquisição só fosse interrompido para receber uma trama ele destinada. Mas, com efeito, estando activo o mecanismo de geração de interrupções da porta série, os microcontroladores de todas as unidades de aquisição serão interrompidos sempre que for enviado um byte para a rede, quer pela unidade concentradora, quer pelas outras unidades de aquisição. A porta série do 8031 permite, no entanto, implementar um outro modo de gerar interrupções ao CPU. O seu bit de paridade pode ser programado para gerar ou não uma interrupção em função do seu valor. Neste caso, não poderá ser utilizado este bit como parte de um mecanismo de verificação de erro de transmissão mas, como já se implementou um outro, este facto não constituirá problema.

Este mecanismo, que permite que se minimize a ocupação dos CPU's das unidades de aquisição, consta do seguinte:

- a unidade concentradora, ao enviar o primeiro byte da trama, que corresponde ao endereço de destino, envia o respectivo bit de paridade com o valor que provoca interrupção dos microcontroladores das unidades de aquisição;
- todos os microcontroladores 8031 são interrompidos, e as respectivas rotinas de atendimento de interrupção devem começar por verificar se o endereço recebido corresponde ao seu. No caso afirmativo, a rotina deverá continuar a receber os restantes bytes da trama e implementar o serviço solicitado. Caso contrário, isso significa que a trama não se destina à respectiva unidade de aquisição, pelo que se poderá abandonar de imediato a rotina de atendimento da interrupção;
- todos os outros bytes da trama enviada pela unidade concentradora devem levar o bit de paridade complementado.

Desta forma os microcontroladores das outras unidades de aquisição não serão posteriormente interrompidos;

- da mesma forma, os bytes da trama de resposta, enviada pela unidade de aquisição endereçada para a unidade concentradora, deverão levar o bit de paridade complementado para não interromper os microcontroladores das outras unidades de aquisição.

Consegue-se assim atingir o objectivo pretendido, com um mecanismo muito simples e sem acréscimo de hardware nas unidades de aquisição.

CAPITULO 9

Unidade central

9 - UNIDADE CENTRAL

O papel da unidade central é recolher das unidades concentradoras, através da rede BITBUS, as informações referentes aos postos de trabalho, armazená-las num sistema gestor de base de dados e permitir consulta e processamento posteriores. Apresentam-se a seguir as funções e objectivos desta unidade.

9.1 - Funções

As funções especificadas para a unidade central foram resumidamente as seguintes:

- desempenhar o papel de unidade mestre da rede BITBUS;
- recolher, de uma forma transparente para o utilizador, as informações armazenadas nas unidades concentradoras referentes aos postos de trabalho monitorizados pelo sistema. Essa recolha deve ser feita em intervalos de tempo definidos pelo gestor do sistema, ou sempre que um utilizador pretenda uma actualização da informação disponível centralizadamente;
- processar as informações recolhidas e proceder ao seu armazenamento em base de dados;
- permitir ao utilizador a consulta das informações referentes aos postos de trabalho. Essa consulta pode fornecer os valores das variáveis dos postos de trabalho no momento da consulta ou dizer respeito a um intervalo de tempo a definir pelo utilizador;
- implementar funções de alarme e registo de erros;
- possibilitar a interface com outros programas, como por exemplo aplicações de planeamento e controlo de produção, gestão de stocks, controlo horário, salários, etc., de forma a permitir a essas aplicações utilizar a informação existente sobre os postos de trabalho.

9.2 - Objectivos

Os objectivos fundamentais colocados para o desenvolvimento/especificação da unidade central foram:

- oferecer ao utilizador uma interface fácil e simples, tanto para a configuração e introdução de comandos, como para a visualização das informações dos postos de trabalho;
- baixo custo, sendo utilizado sempre que possível um PC-AT compatível;
- robustez e fiabilidade;
- dispôr de alimentação não interruptível.

9.3 - Análise estruturada

9.3.1 - Diagrama de Transição de Estados

Na unidade central existe uma estrutura de informação fundamental, o núcleo da base de dados, constituído por um conjunto de tabelas contendo as informações referentes aos postos de trabalho. Tornou-se pois necessário desenvolver um conjunto de módulos de software para implementar as funções de actualização, processamento e consulta da informação armazenada nesse núcleo central, assim como de configuração do sistema. O DTE da figura 9.1 ilustra a interligação entre as diferentes tarefas/estados do sistema. Segue-se uma breve descrição dessas tarefas, assim como das condições de transição entre elas.

- "Inicialização": o sistema entra neste estado quando lhe é fornecida alimentação. São efectuadas as inicializações da unidade central, nomeadamente arranque do computador e inicialização do software de gestão de base de dados e da rede BITBUS.
- "Opções": terminadas as inicializações o sistema transita para este estado, no qual fica a aguardar que o utilizador solicite uma das opções disponíveis (Configuração, Monitorização, Históricos, Erros e Actualização), ou que seja despoletada uma actualização temporizada. A selecção pelo utilizador de uma dada opção provoca a transição do sistema para o estado com o mesmo nome.

- "Configuração": neste estado o utilizador pode, desde que tenha permissão para tal, alterar os valores das variáveis do sistema, inserir, modificar e apagar linhas das tabelas da base de dados, etc.

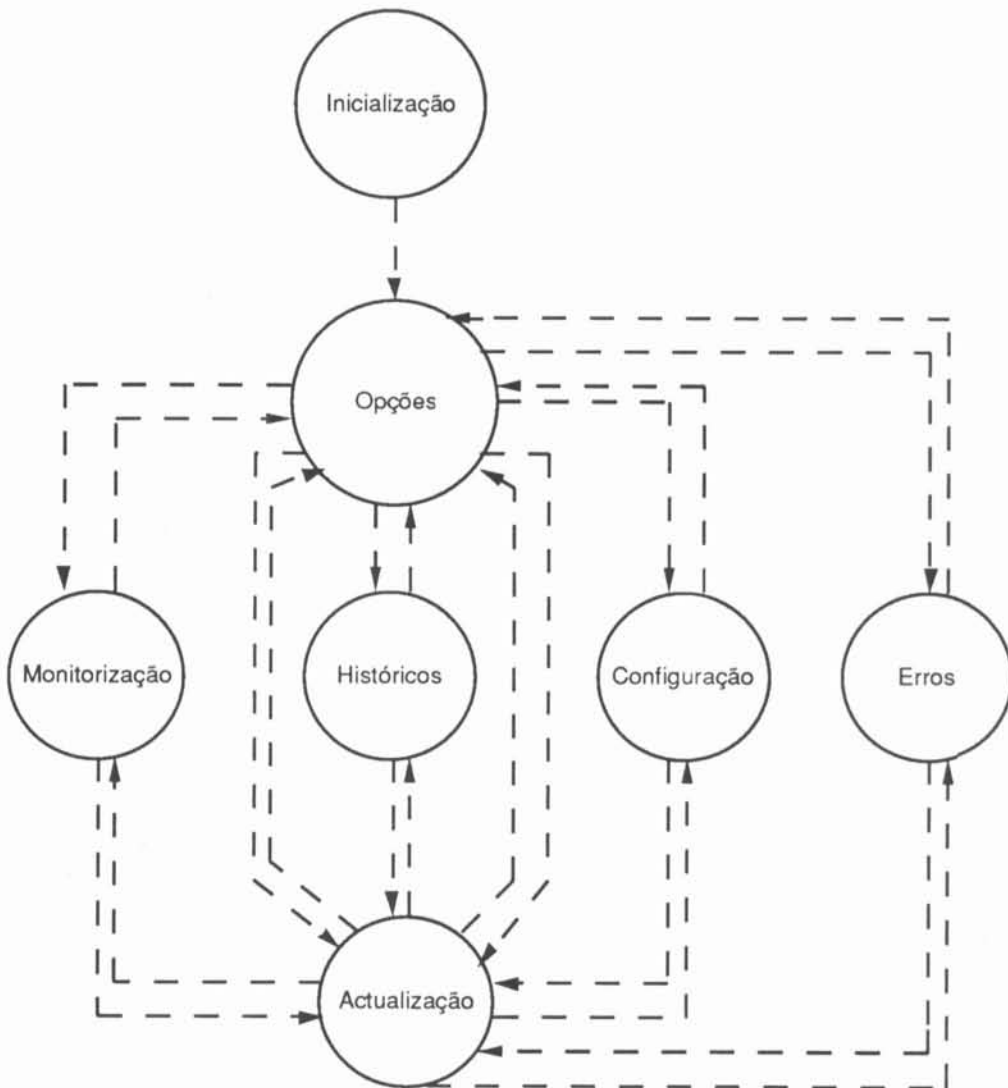


Figura 9.1 - Unidade central: Diagrama de Transição de Estados

- "Monitorização": estado no qual o utilizador é informado sobre o valor actual das variáveis referentes a um determinado posto de trabalho, como por exemplo a

identificação do operador que ocupa esse posto de trabalho, a ordem de fabrico ou operação que está a ser executada, o estado e a produção actuais e a média efectuada.

- "Históricos": neste estado o utilizador tem acesso a informação do mesmo tipo da fornecida no estado anterior, referindo-se todavia não aos valores actuais mas a um intervalo de tempo por ele definido.
- "Erros": quando o utilizador é avisado da ocorrência de erros no sistema deve, logo que possível, seleccionar esta opção, na qual terá acesso a informação sobre os erros ocorridos, nomeadamente origem, tipo, data e hora de ocorrência, etc.
- "Actualização": estado correspondente à recolha, processamento e armazenamento na base de dados das informações recolhidas dos postos de trabalho através da rede BITBUS. O sistema pode passar para este estado, quer por solicitação do utilizador, a partir do estado "Opções", quer automaticamente, a intervalos de tempo previamente definidos, a partir de qualquer dos estados possíveis excepto "Inicialização".

9.3.2. - Estrutura Básica da Informação na Base de Dados

Apesar de, no caso específico do software da unidade central, o Projecto Orientado a Fluxos de Informação não ser a metodologia de projecto mais apropriada, foi ainda a metodologia aqui seguida por razões de homogeneidade do trabalho.

Antes mesmo de ser iniciada a construção dos DFI's foi definida uma primeira estrutura para a informação a armazenar na base de dados. Esta primeira iteração poderia ser posteriormente modificada se o desenvolvimento do software a isso conduzisse.

Essa estrutura de informação agrupa, de uma forma organizada, as informações referentes aos postos de trabalho já especificadas no capítulo 1.

Os postos de trabalho são caracterizados por um conjunto de variáveis, nomeadamente as relativas ao controlo de produção já definidas em 1.2, acrescidas de outras necessárias para as complementar. Estas **variáveis** que a seguir se definem, agrupadas de forma coerente em variáveis estruturadas, vão

constituir os registos da base de dados:

- Operadores: caracterizados pelo **nome** e por um **código**;
- Operações: caracterizadas por um **nome**, um **código** e um **tempo médio de execução**. O tempo médio de execução é utilizado para verificar as médias dos operadores e também no planeamento da produção;
- Postos de trabalho: caracterizados por um **nome**, um **código**, o **endereço da unidade de aquisição** correspondente e o **endereço da unidade concentradora** onde esta está ligada;
- Erros: caracterizados por um **nome** e por um **código**.

Além destas estruturas de informação há necessidade de mais duas: uma para o registo das ocorrências, do ponto de vista de controlo de produção, nos postos de trabalho e outra para o registo dos erros que eventualmente se verifiquem durante o funcionamento do sistema. Surgem assim as seguintes estruturas:

- Produção: caracterizada por um **código de posto de trabalho**, um **código de operação**, um **código de operador**, **produção**, **estado do posto de trabalho**, **data** e **hora de início da operação** e **duração da operação**;
- Registo de erros: caracterizado por um **código do erro**, um **código do posto de trabalho**, a **data** e a **hora** da ocorrência do erro e uma variável booleana, **erro-ok**, que indica se o erro foi ou não reconhecido por um utilizador.

9.3.3 - Diagramas de Fluxo de Informação

No DFI da figura 9.2 a unidade central está representada através de um processo único denominado, "Programa de Controlo de Produção", que inclui o conjunto de funções especificadas em 9.3.1 e corresponde ao DTE da figura 9.1. Este processo recebe informações do teclado, da base de dados (Estrutura de Informação) e, através da rede BITBUS, das unidades concentradoras. Essas informações são depois processadas, em função da opção escolhida, e podem ser enviadas para o monitor, para a impressora ou para um ficheiro de interface com aplicativos, que se constituem assim como poços de informação

do sistema. O processamento pode incluir também a actualização da informação na base de dados e/ou o envio de informações para as unidades concentradoras, pelo que a estrutura de informação e a rede BITBUS serão simultaneamente fontes e poços de informação.

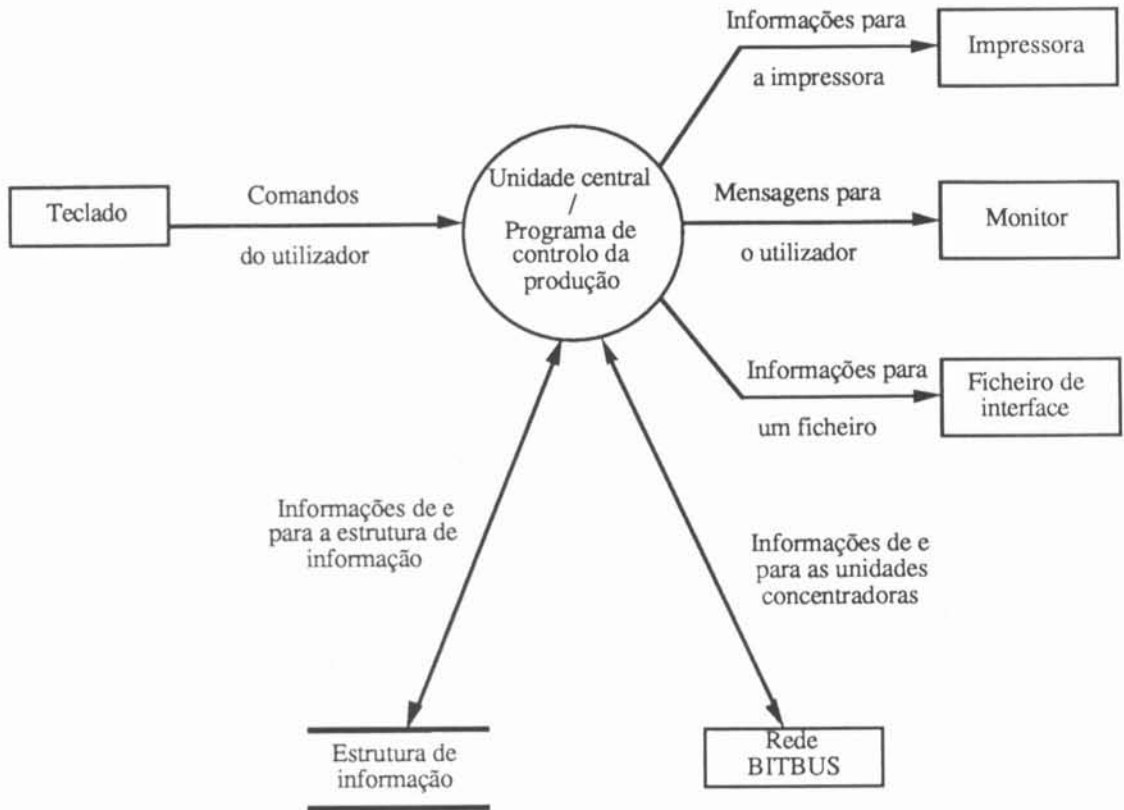


Figura 9.2 - Unidade central: DFI de ordem zero

Nas figuras 9.3 a 9.7 apresentam-se os DFI's de ordem um para cada uma das tarefas definidas em 9.3.1. As tarefas "Inicialização", "Monitorização", "Históricos", "Configuração" e "Erros" utilizam campos da base de dados que vão ser acedidos simultaneamente pela tarefa "Actualização" pelo que, na fase de implementação, deverão ser tomadas medidas para garantir a integridade da informação. É importante realçar na tarefa "Actualização" a possibilidade de, estando em curso a actualização da informação de um posto de trabalho na sequência de uma solicitação do utilizador, o mecanismo de actualização automática

da informação poder ser activado por ter terminado o intervalo de tempo entre actualizações. Neste caso a tarefa "Actualização" é de novo activada, desta vez para actualizar a informação de todos os postos de trabalho. Este é outro caso que deve ser considerado com cuidado na fase de implementação, para que não se verifiquem conflitos na actualização da informação.

Para ilustrar a localização das zonas críticas, os DFI's das figuras 9.5 a 9.7 incluem não só as tarefas que representam como também as que com elas partilham estruturas de informação

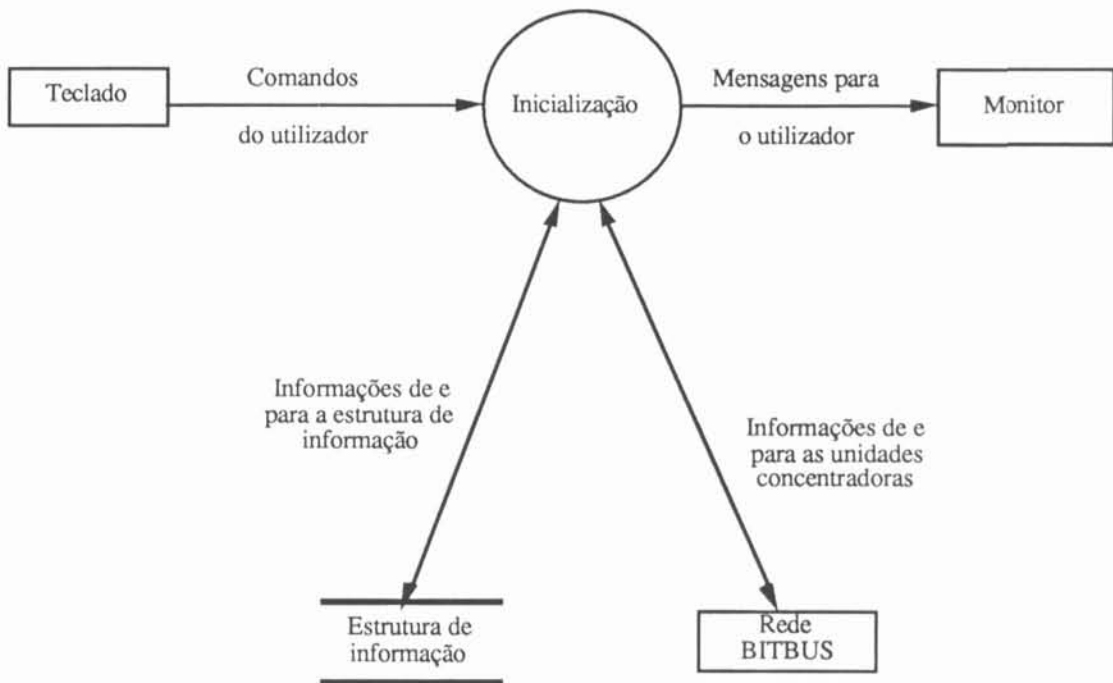


Figura 9.3 - Inicialização: DFI de ordem um

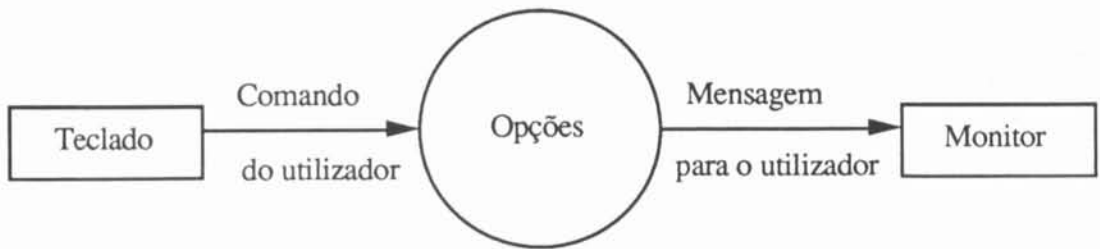


Figura 9.4 - Opções: DFI de ordem um

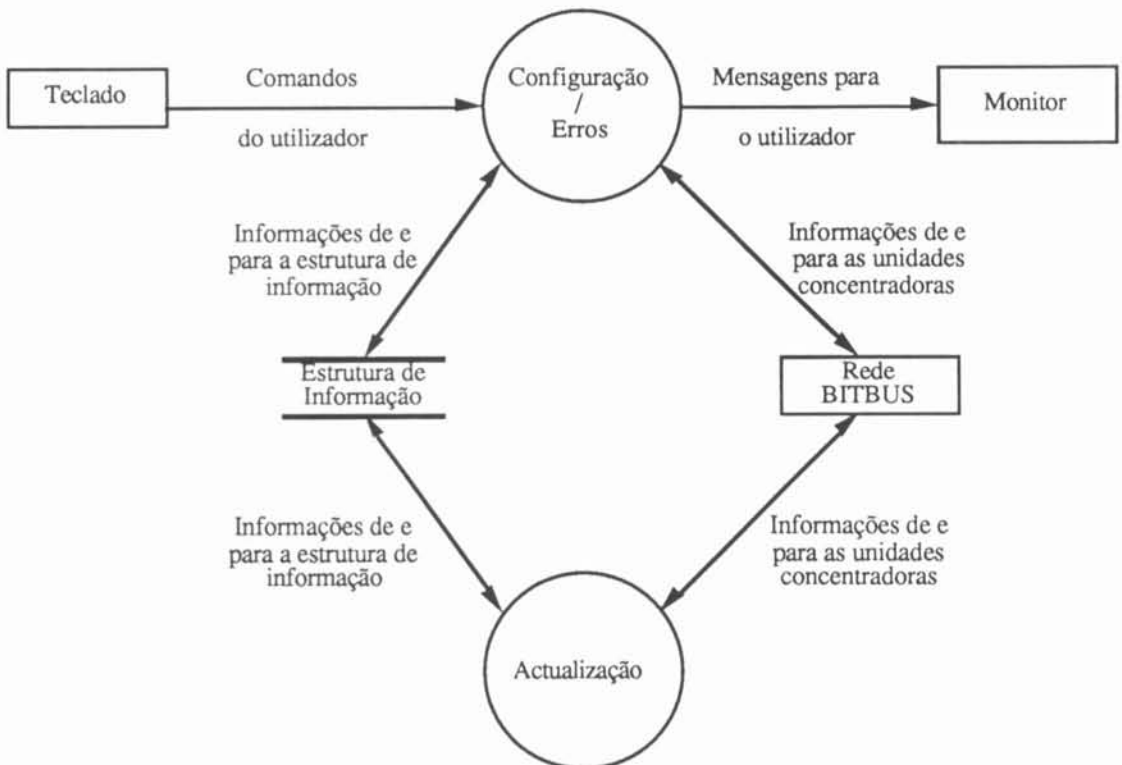


Figura 9.5 - Configuração/Erros: DFI de ordem um

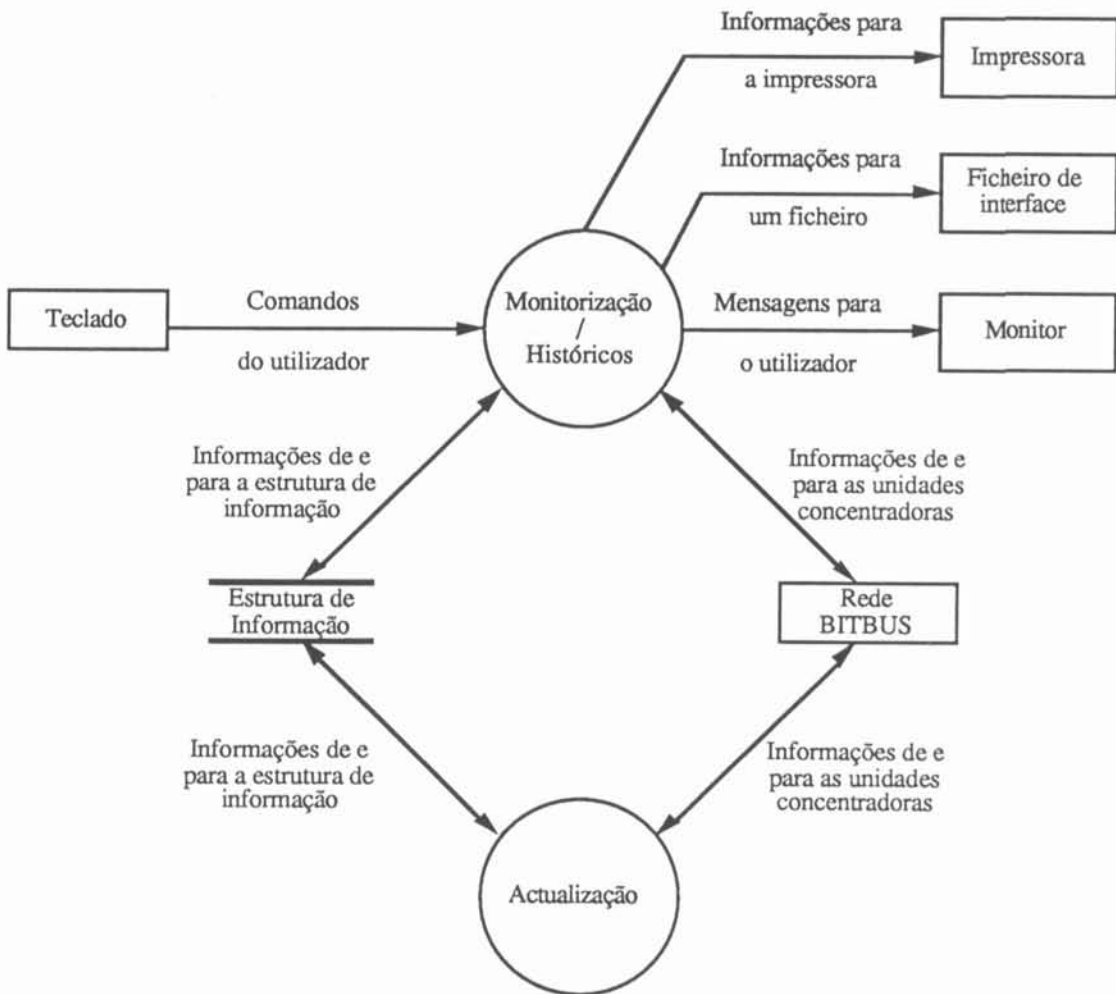


Figura 9.6 - Monitorização/Históricos: DFI de ordem um

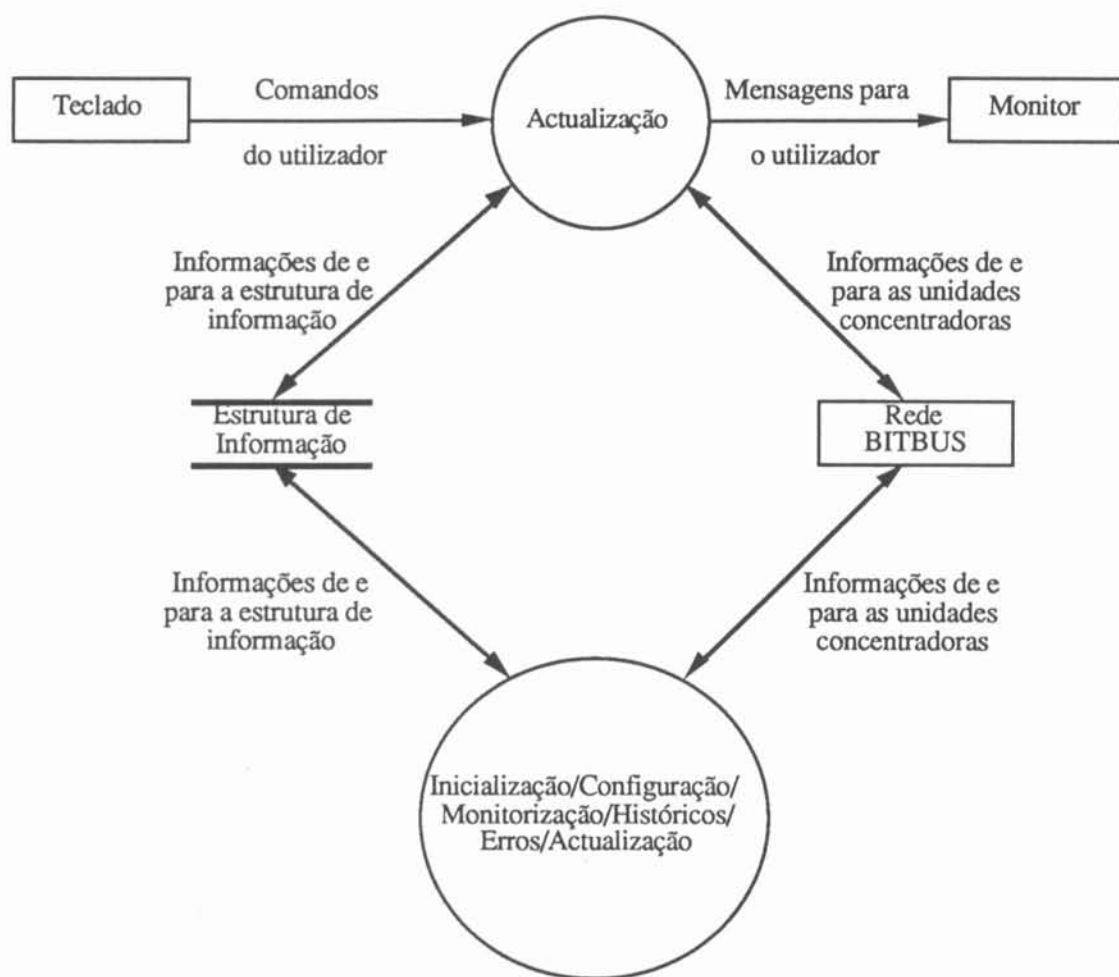


Figura 9.7 - Atualização: DFI de ordem um

9.3.4 - Mini-especificações e Dicionário de Informação

A tarefa "Atualização" é, sem dúvida, uma tarefa crucial. Dado ser responsável pela transferência de informação entre a rede BITBUS e a base de dados, dela dependem significativamente o desempenho e as potencialidades do software da unidade central. As restantes tarefas são típicas de aplicações de processamento de informação desenvolvidas sobre bases de dados.

Apresenta-se pois, na figura 9.8, a Mini-especificação da tarefa "Atualização" através de um fluxograma.

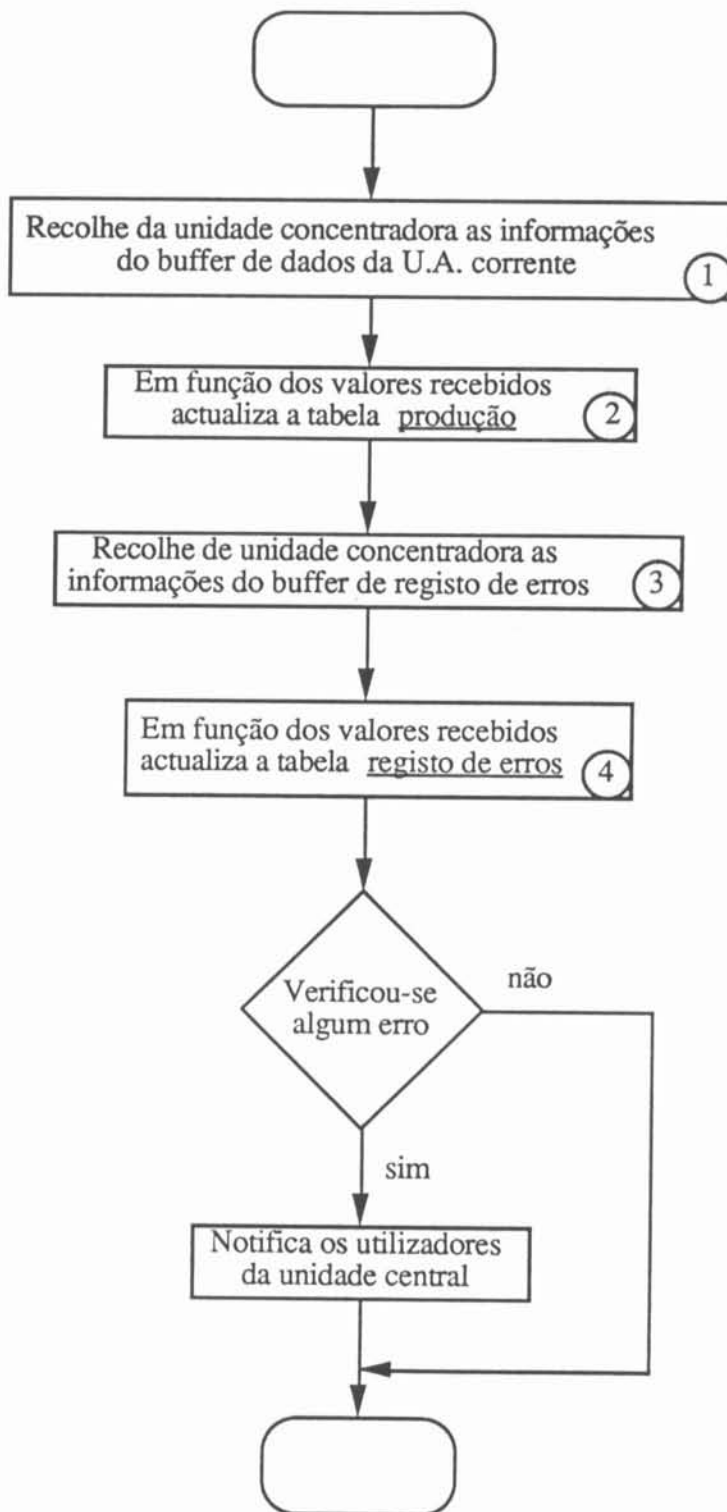


Figura 9.8 - Mini-especificação da tarefa "Atualização"

Este fluxograma não é mais do que uma descrição dos procedimentos a implementar, apresentando-os como um conjunto de sub-tarefas ligadas por uma determinada sequência. As sub-tarefas 1 e 3 correspondem a uma leitura, por parte da unidade central através da rede BITBUS, dos "buffers" de dados das unidades de aquisição e de registo de erros. As sub-tarefas 2 e 4 constituem a escrita, nas tabelas da base de dados, das informações recolhidas. O fluxograma apresentado inclui os passos necessários para processar a informação referente a um posto de trabalho. A tarefa "Actualização" deverá receber, como parâmetro, o endereço do posto de trabalho a processar ou, no caso de se pretender receber as informações referentes a todos os postos de trabalho existentes, um código especial.

As Mini-especificações das restantes tarefas e o Dicionário de Informação, contendo as descrições de todas as variáveis, complementam o conjunto de DFI's obtido.

9.4 - Projecto Estruturado

9.4.1 - Mapas de Estrutura

A Análise de Transacções foi, mais uma vez, a ferramenta utilizada para a passagem da fase de Análise à fase de Projecto Estruturado.

No caso da unidade central os Mapas de Estrutura apenas representam os vários módulos, as respectivas ligações de informação e de controlo e a hierarquia constituída. Para que a representação gráfica não resulte muito complexa não se representam as fontes e poços de informação nem os locais de armazenamento temporário. Essa informação pode, no entanto, ser obtida através dos DFI's correspondentes.

A figura 9.9 apresenta o ME de ordem zero correspondente ao "Programa de Controlo de Produção". Nas figuras seguintes estão representados os ME's de ordem um para os módulos "Monitorização" e "Históricos", e o ME de ordem dois para o módulo "Seleção e visualização da informação".

Está pois concluído o modelo do software da unidade central, cuja implementação será discutida no capítulo 10.

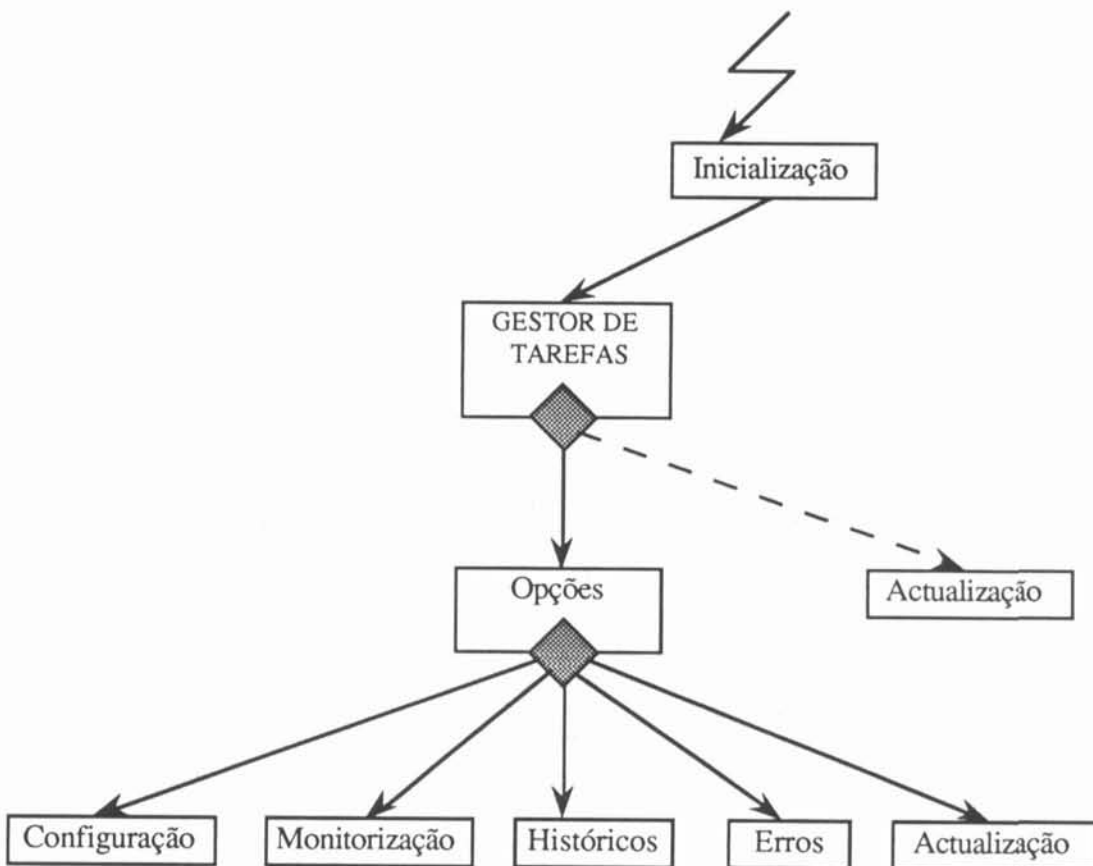


Figura 9.9 - Unidade central: ME de ordem zero

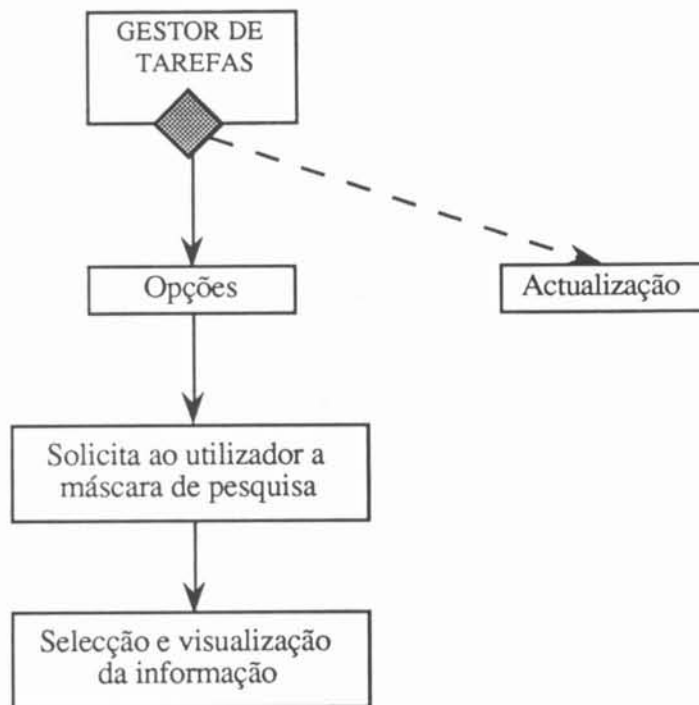


Figura 9.10 - "Monitorização: ME de ordem um

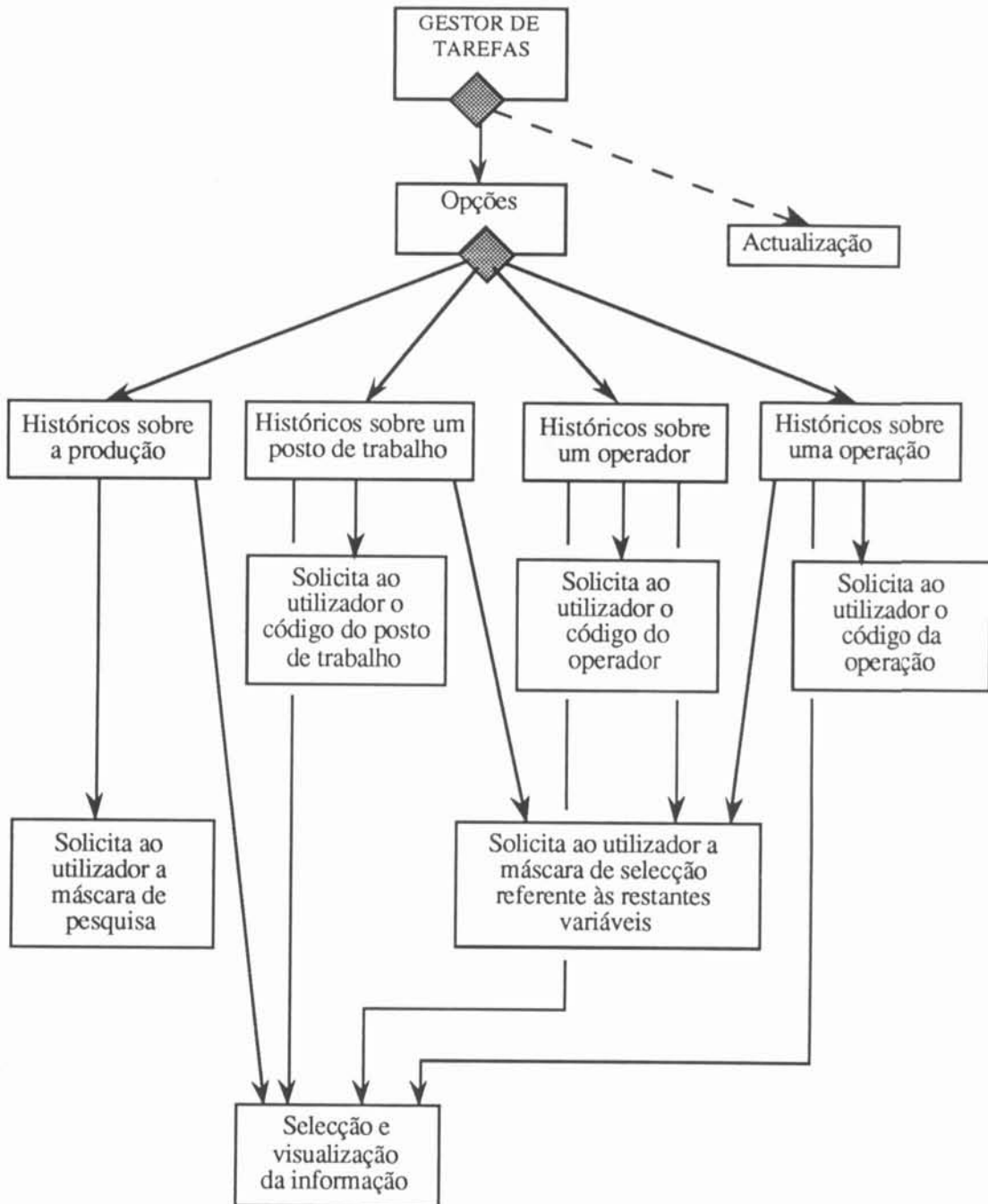


Figura 9.11 - "Históricos": ME de ordem um

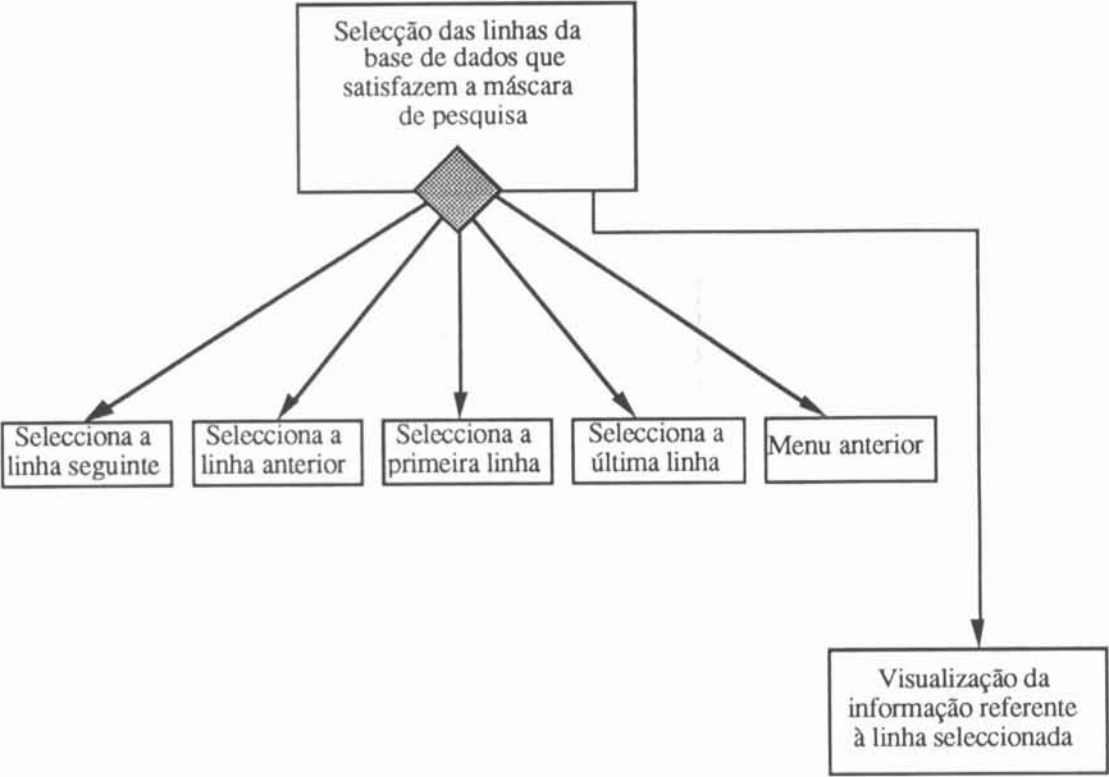


Figura 9.12 - "Seleção e visualização da informação: ME de ordem dois

CAPITULO 10

Implementação e testes

10 - IMPLEMENTAÇÃO E TESTES

10.1 - Implementação

Com base nos modelos desenvolvidos para os vários componentes do sistema, iniciou-se a fase de implementação. Essa implementação, que teve como ponto de partida as Especificações de Projecto Estruturado obtidas nos capítulos 6,7 e 9, desenrolou-se de acordo com um conjunto de regras que se apresentam seguidamente:

- era objectivo primordial a obtenção de um protótipo laboratorial que permitisse avaliar, por um lado tanto o desempenho do sistema como um todo, como o de cada um dos seus componentes (unidade central, unidades concentradoras, unidades de aquisição e redes de comunicação), e por outro lado o custo final da solução proposta;
- existindo hardware e/ou software disponível comercialmente, cumprindo as exigências técnico-económicas especificadas para implementar partes do sistema, ele seria utilizado;
- o desenvolvimento de hardware teria como objectivo a sua reutilização, sendo portanto orientado para a obtenção de módulos o mais independentes possível e facilmente expansíveis e reconfiguráveis;
- a exemplo do hardware, também o software deveria ser reutilizável, sendo natural, dado serem iguais os conjuntos de instruções dos dois microcontroladores, o desenvolvimento de uma biblioteca de funções comuns;
- terminada a fase de testes laboratoriais com resultados satisfatórios, seria proposto o desenvolvimento de um protótipo industrial para realização de testes em condições reais e eventual comercialização.

10.1.1 - Unidades de aquisição

10.1.1.1 - Hardware

Face à especificação respectiva decidiu-se subdividir o hardware das unidades de aquisição em cinco blocos distintos, nomeadamente:

- 1 - Núcleo central: incluindo microcontrolador, memória de programa e memória de dados, e descodificação.
- 2 - Interface com o operador: constituída por um afixador alfanumérico, um interruptor rotativo para indicação do estado do posto de trabalho e um botão de "reset" do contador parcial.
- 3 - Interface com sinais digitais de entrada e saída: como sinal de entrada existem os impulsos relativos à medida de produção, e como saída um sinal "on/off" para controlo de um relé.
- 4 - Interface com a rede própria: interface entre a porta série do microcontrolador 8031 e a rede própria definida no capítulo 8.
- 5 - Alimentação: bloco de conversão da tensão de alimentação para os níveis de tensão necessários.

Apresentam-se seguidamente diagramas simplificados correspondentes à implementação de cada um destes blocos.

Núcleo central

Na figura 10.1 estão representados os elementos que constituem o núcleo central. Podem identificar-se o microcontrolador (pode ser utilizado qualquer microcontrolador da família 8051 da Intel), uma LATCH para armazenamento das oito linhas de endereço menos significativas, uma PAL que implementa a descodificação, um integrado para memória de programa e outro para memória de dados, e um bloco de "jumpers" para configuração das memórias. Os mapas de colocação dos "jumpers" podem ser consultados no Apêndice D.

Interface com o operador

Na figura 10.2 são apresentadas as interfaces entre o microcontrolador e o afixador alfanumérico, o botão de "reset" do contador parcial e o interruptor rotativo. O interruptor rotativo utilizado dispõe de doze saídas, correspondentes a doze possíveis posições/estados do posto de trabalho. É utilizada uma PAL para converter essas doze linhas num código binário que constitui a entrada para o microcontrolador. Se for utilizado um interruptor rotativo com uma saída já codificada, em binário ou BCD, pode dispensar-se a utilização da PAL.

Interface com sinais de entrada e saída

A figura 10.3 ilustra a interface com a entrada de contagem, constituída por um circuito de "debounce", necessário para evitar eventuais problemas de ruído causados pela comutação de interruptores mecânicos, e por um bloco de isolamento óptico. A saída deste bloco é ligada à entrada do "Timer/Counter 0" do microcontrolador. Está representada na mesma figura a interface com o relé de controlo constituída simplesmente por um bloco de isolamento óptico.

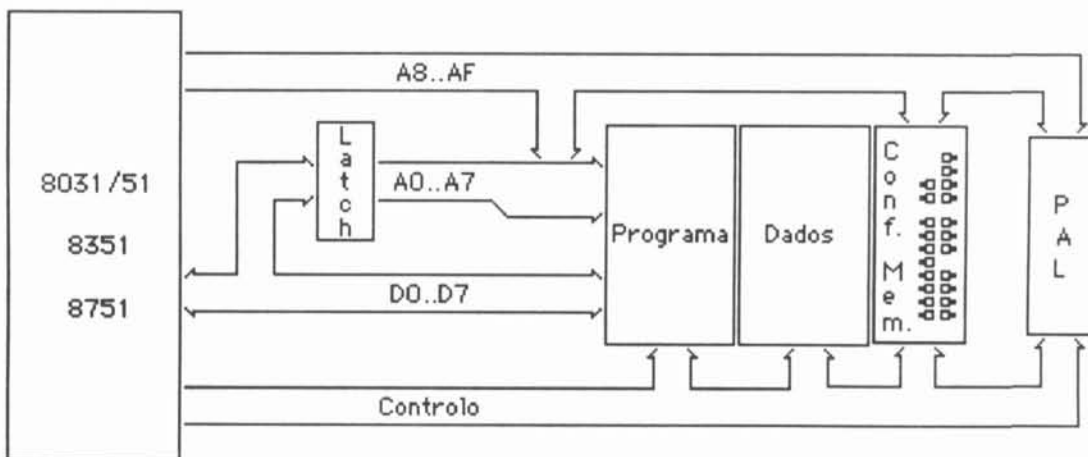


Figura 10.1 - Núcleo central da U.A.: diagrama de blocos

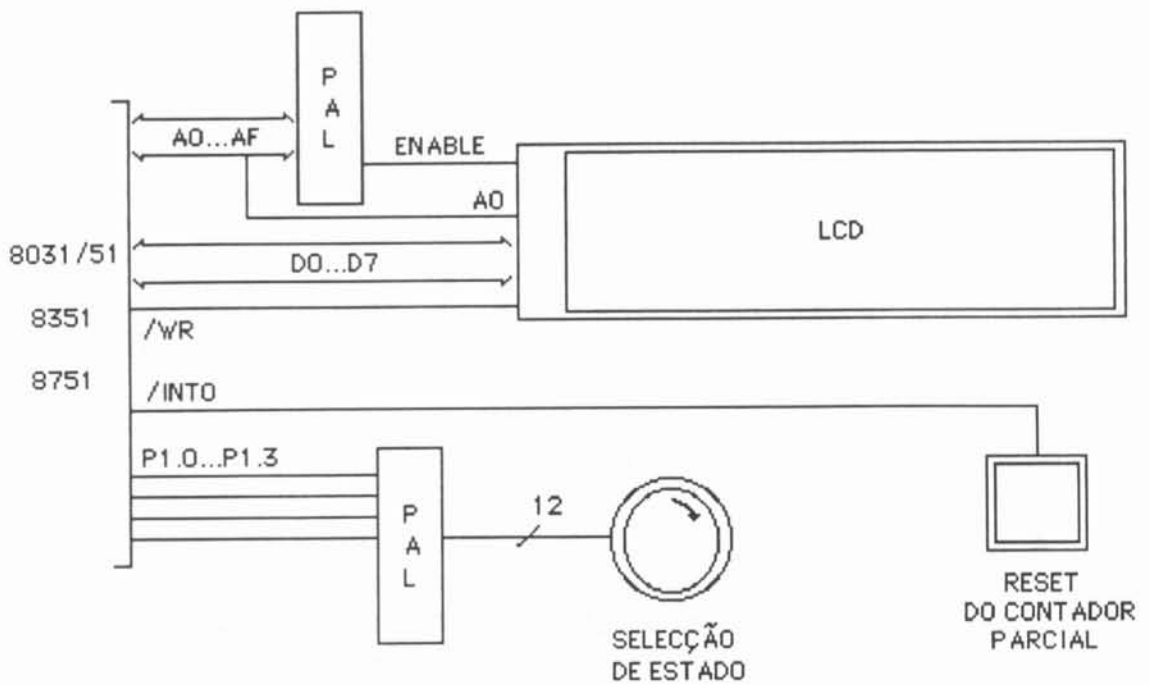


Figura 10.2 - Interface com o operador da U.A.: diagrama de blocos

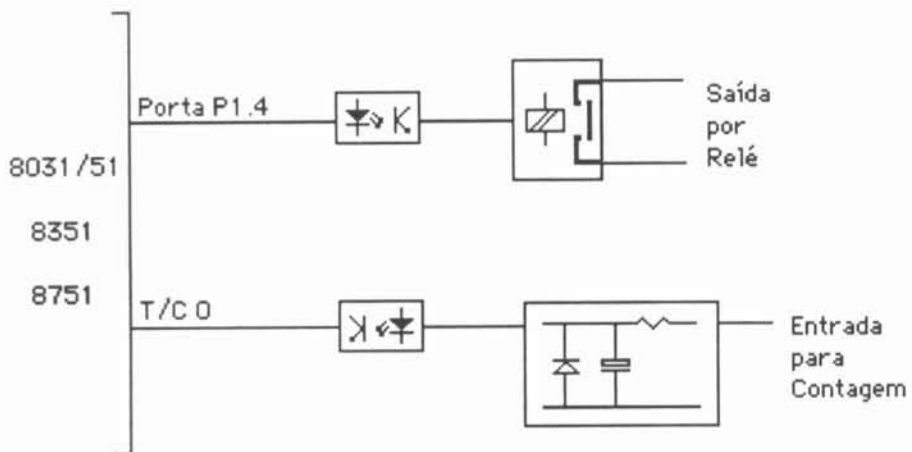


Figura 10.3 - Interface com sinais de entrada e saída da U.A.: diagrama de blocos

Interface com a rede própria

Esta interface é apresentada na figura 10.4. Os sinais de "transmit" e "receive" do microcontrolador estão ligados, através de um bloco de isolamento óptico, a um "transceiver" que se encarrega de proporcionar os níveis eléctricos exigidos pelo standard RS485. A gestão do sentido de circulação da informação no "transceiver" é realizada utilizando um bit do microcontrolador. A alimentação fornecida aos dois lados do bloco de isolamento óptico está isolada uma da outra, tendo sido utilizado para esse efeito um conversor DC/DC. Faz ainda parte desta interface um bloco de "jumpers" utilizado para programar o endereço da unidade de aquisição na rede própria. A programação deste bloco pode ser consultada no Apêndice D.

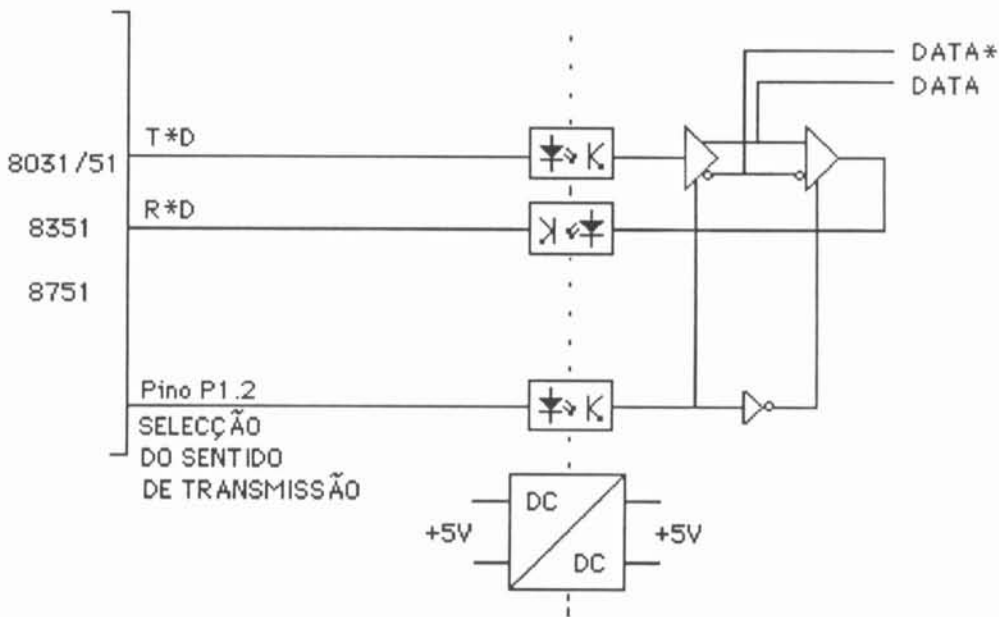


Figura 10.4 - Interface com a rede própria da U.A.:diagrama de blocos

Alimentação

No protótipo laboratorial desenvolvido utilizou-se uma fonte de alimentação comutada com entrada de 220V AC e saídas de +5V, +12V e -12V DC. O seu diagrama está representado na figura 10.5



Figura 10.5 - Alimentação da U.A.: diagrama de blocos

10.1.1.2 - Software

O software das unidades de aquisição foi desenvolvido em linguagem ASSEMBLY 51, por um lado porque as rotinas a implementar são relativamente simples e por outro lado porque esta linguagem garante uma maior portabilidade entre o 8031 e o 8044, do que por exemplo o PLM 51. Foi integralmente implementado o modelo desenvolvido no capítulo 6.

Foram desenvolvidas várias rotinas de uso comum, por exemplo para enviar um carácter para o afixador alfanumérico, realizar operações aritméticas com números representados com mais de um byte, enviar e receber caracteres pela porta série, etc., que foram posteriormente compiladas e "linkadas" numa biblioteca. Este conjunto de rotinas pôde depois ser reutilizado aquando do desenvolvimento do software das unidades concentradoras, e poderá vir a ser útil na implementação de outros sistemas.

A taxa de transmissão utilizada pela rede própria foi fixada em 9600 bps. Quando forem efectuados testes do sistema em ambiente industrial serão experimentadas outras taxas de transmissão mais elevadas. Se se verificar que não há degradação das características da rede própria, a taxa agora fixada poderá ser alterada.

Foi também definido que o valor do contador (medida de produção) será representado utilizando três bytes, o que corresponde a um valor decimal máximo de $4,294 \times 10^9$, e que o contador parcial utilizará dois bytes. A gama de contagem do contador parcial foi fixada entre 0 e 50.000, e atingido este valor é feita a sua re-inicialização a zero.

A figura 10.6 apresenta um exemplo das mensagens enviadas para o afixador alfanumérico, onde é possível identificar-se o número da unidade de aquisição, o estado do posto de trabalho (neste caso, "Produção") e que desde o início da operação se realizaram 15801 unidades de produção, apresentando o contador parcial um valor de 528 unidades de produção.

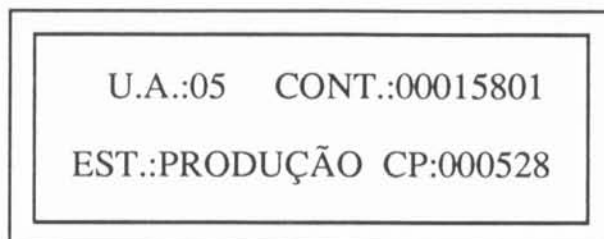


Figura 10.6 - Mensagens enviadas para o afixador alfanumérico da U.A.

10.1.2 - Unidades concentradoras

10.1.2.1 - Hardware

O hardware das unidades concentradoras foi dividido em seis blocos:

- 1 - Núcleo central: constituído pelo microcontrolador 8044, memória de programa e memória de dados e descodificação.
- 2 - Interface com o operador: incluindo o afixador alfanumérico e o teclado.
- 3 - Interface com a rede BITBUS: bloco de interface entre a porta série do 8044 e a rede BITBUS.
- 4 - Interface com a rede própria: entre o 8044 e uma USART e entre esta e a rede própria (ver 8.5).
- 5 - Interface com o relógio de tempo real.

- 6 - Alimentação: bloco de conversão da tensão de alimentação para os níveis de tensão necessários.

Núcleo central

O núcleo central baseado no 8044 (figura 10.7) tem uma arquitectura muito semelhante à do núcleo baseado no 8031. A diferença fundamental é o aparecimento de um bloco de configuração do modo de funcionamento do nó BITBUS, conforme é especificado pela Intel nas suas cartas que utilizam o 8044 [12]. As tabelas de colocação dos "jumpers" para a configuração do nó BITBUS e das memórias são apresentadas no Apêndice D.

Interface com o operador

A interface com o teclado é implementada utilizando um integrado dedicado, uma PPI (figura 10.8). O operador avisa o microcontrolador que pretende dialogar com o sistema, através de uma tecla especial que actua sobre a linha /INT0 do 8044. A interface com o LCD é igual à utilizada nas unidades de aquisição.

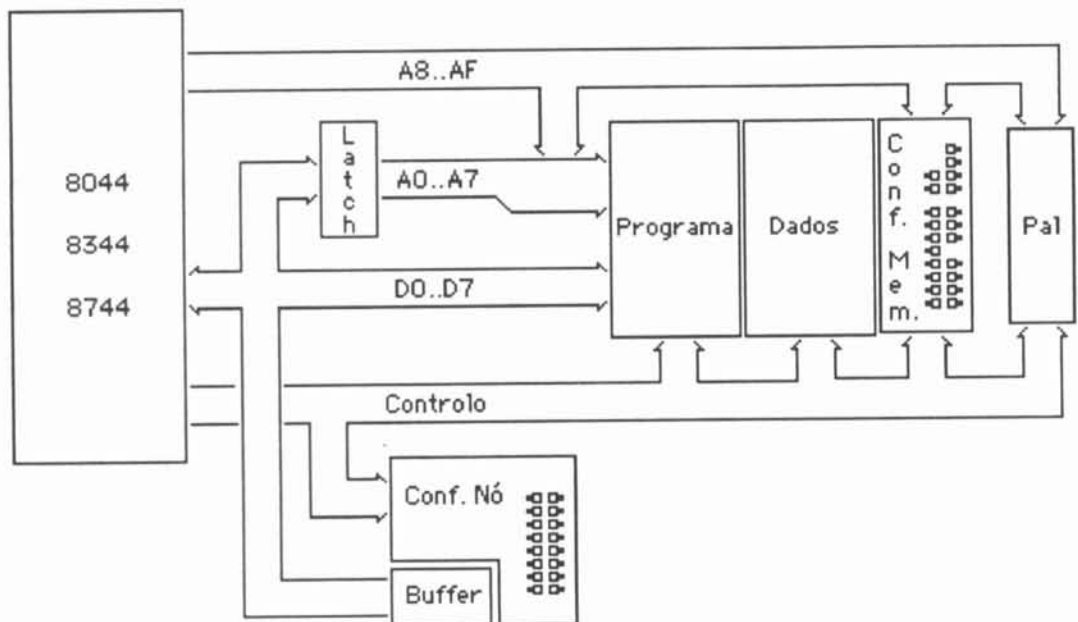


Figura 10.7 - Núcleo central da U.C.: diagrama de blocos

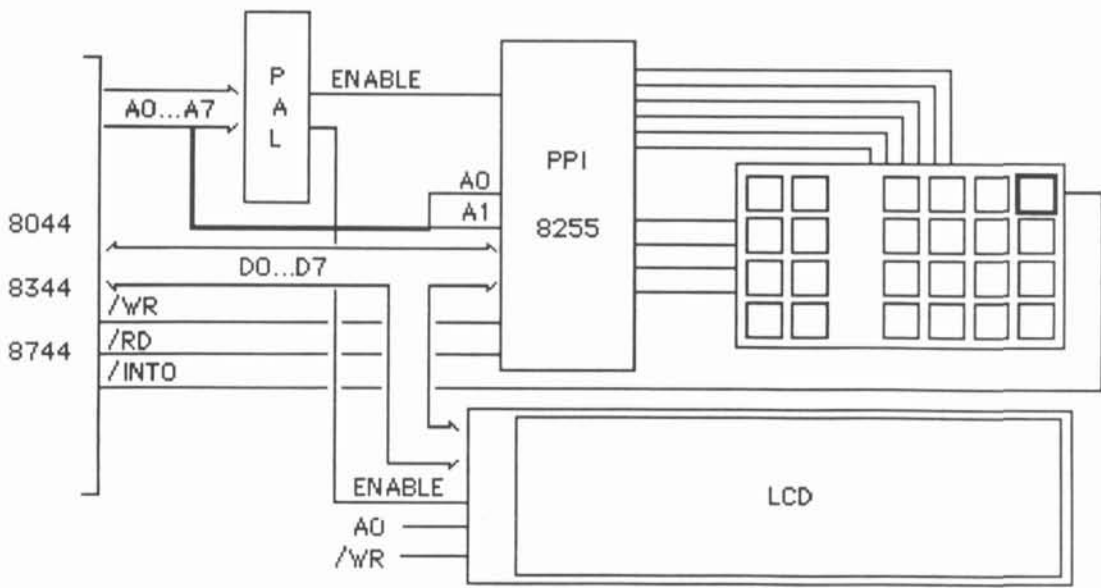


Figura 10.8 - Interface com o operador da U.C.: diagrama de blocos

Interface com a rede BITBUS

Esta interface é semelhante à interface com a rede própria existente nas unidades de aquisição. A principal diferença reside no facto de, no caso do 8044, a gestão do sentido de circulação da informação ser feita automaticamente pelo firmware, enquanto que no 8031 essa gestão teve de ser implementada pelo software de comunicação desenvolvido. A figura 10.9 apresenta o hardware de interface com a rede BITBUS.

Interface com a rede própria

Tal como foi especificado no ponto 8.5, foi utilizada uma USART para implementar a interface entre o 8044 e a rede própria (figura 10.10). A troca de informação entre o microcontrolador e a USART é suportada por interrupções, permitindo ao CPU executar outras tarefas durante os tempos de transmissão e de espera. A entrada de interrupção utilizada é a INT1. No caso desta interface não existe obviamente a necessidade de atribuição de endereço.

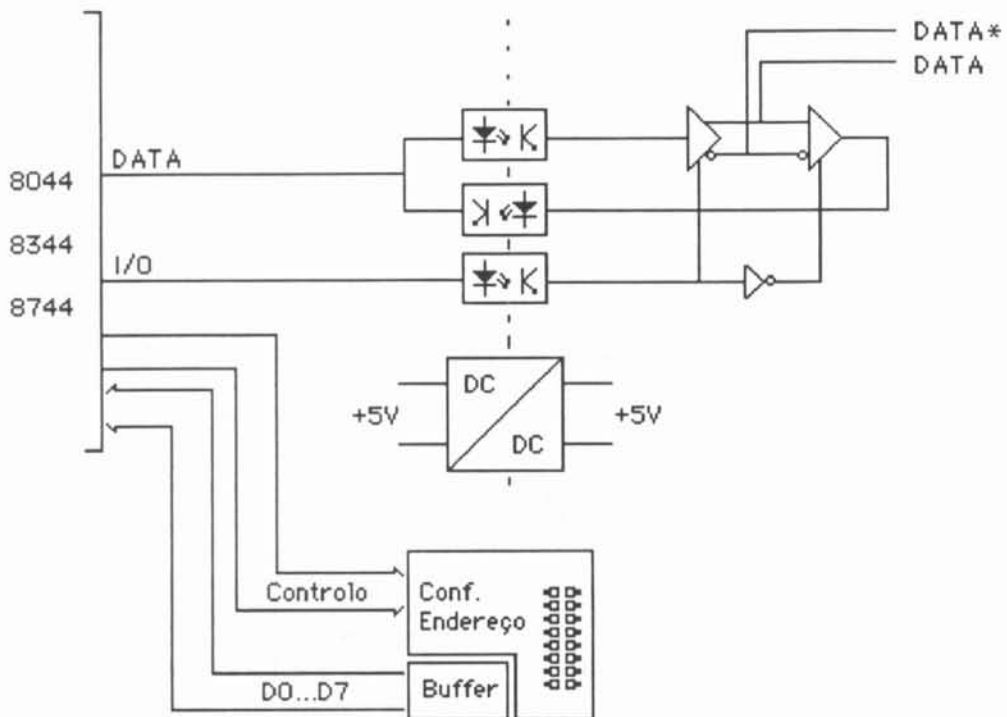


Figura 10.9 - Interface com a rede BITBUS da U.C.: diagrama de blocos

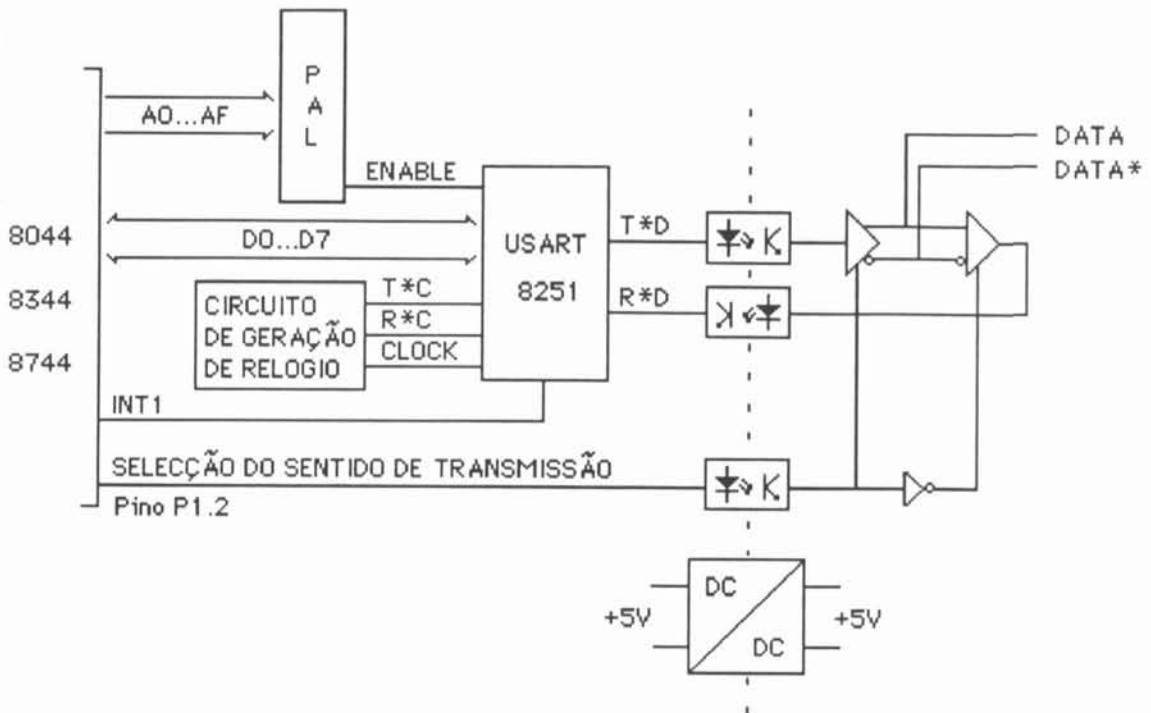


Figura 10.10 - Interface com a rede própria da U.C.: diagrama de blocos

Interface com o relógio de tempo real

O relógio de tempo real (RTC) é visto pelo microcontrolador como um conjunto de posições consecutivas de memória onde pode ler e escrever. O acesso pode ser feito por interrupções ou por sondagem ("polling"). A entrada de interrupção utilizada é a /INT0 (figura 10.11). Embora esta linha de interrupção esteja já a ser utilizada pelo teclado, não se verificam conflitos de identificação da fonte da interrupção graças à utilização do seguinte mecanismo:

- o RTC tem um bit interno que é colocado no nível lógico "1" pelo seu hardware quando pretende fazer uma interrupção ao microcontrolador, ou seja em simultâneo com a activação da linha /IRQ, e é complementado por software ao ser atendido o pedido de interrupção ("set by hardware/cleared by software");

- a rotina de atendimento correspondente à linha /INT0 lê esse bit do RTC. Se o valor lido for "1", então a fonte de interrupção é o RTC. No caso contrário, a fonte de interrupção é o teclado;
- em caso de simultaneidade, é primeiro atendido o pedido de interrupção gerado pelo RTC, dada a sua prioridade naturalmente mais elevada, após o que é atendido o teclado. A rotina de atendimento do RTC é aliás executada em poucos microsegundos.

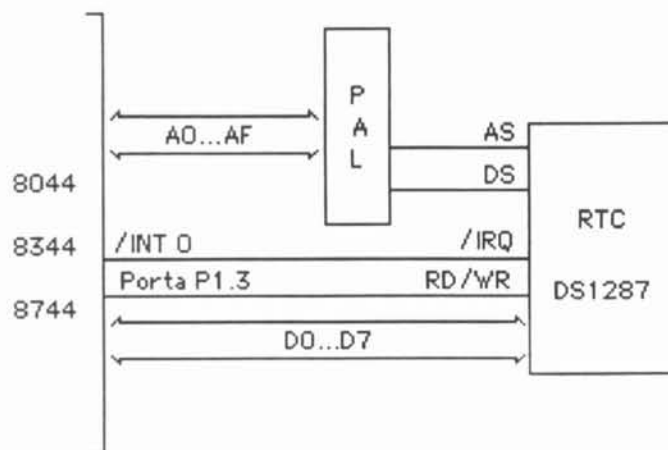


Figura 10.11 - Interface com o relógio de tempo real da U.C.:
diagrama de blocos

Alimentação

O bloco de alimentação é igual ao já descrito para as unidades de aquisição.

10.1.2.2 - Software

10.1.2.2.1 - Implementação

A implementação do software das unidades concentradoras foi feita em linguagem ASSEMBLY 51, obedecendo à especificação

feita no capítulo 7. Esta especificação foi todavia enriquecida com módulos de validação de comandos do operador, de detecção e correcção de erros, etc., os quais foram omitidos nesta descrição para a não alongar demasiadamente.

O módulo "Varrimento" foi implementado como uma tarefa do sistema operativo do 8044, o IDCX51 (Apêndice B), assim como a tarefa "Interface com o operador". Não foi desenvolvido software na unidade concentradora para o módulo "Comunicação BITBUS". Para implementar as funções correspondentes a este módulo foram utilizados os comandos RAC do BITBUS (Apêndice A).

A figura 10.12 apresenta, a título de exemplo, uma sequência mostrando as mensagens afixadas pela interface com o operador durante uma sinalização de início de operação. Após o operador ter sinalizado que pretende iniciar uma operação, o sistema pede-lhe que introduza o código do posto de trabalho. Depois de introduzido este código, o sistema verifica se esse posto de trabalho está ou não já alocado a outra operação. Em caso afirmativo, o operador é avisado e a operação é abortada. Caso contrário, o sistema pede ao operador que introduza o seu código e o código da operação que vai efectuar. Essa informação é processada pela unidade concentradora e o operador é informado de que a operação de sinalização de início de operação terminou com sucesso.

É descrita a seguir a implementação dos "buffers" de dados das unidades de aquisição, de registo de erros e de transmissão de blocos referidos no capítulo 7.

"Buffer" de dados das unidades de aquisição

Este "buffer" está subdividido num conjunto de outros "buffers" de igual dimensão, cada um referente a uma unidade de aquisição presente na rede própria, onde são registadas todas as ocorrências verificadas. A estrutura desses "buffers", representada na figura 10.13, consiste num cabeçalho que contém dois apontadores, um de leitura e um de escrita, seguido de um bloco de bytes onde é armazenada a informação. A estrutura descrita funciona como uma pilha FIFO ("first in - first out") e o acesso é feito rotativamente, ou seja, quando se atinge o fim da pilha passa-se automaticamente para o início. O comprimento destes "buffers" foi fixado em 256 bytes, incluindo o cabeçalho, por razões de facilidade de endereçamento.

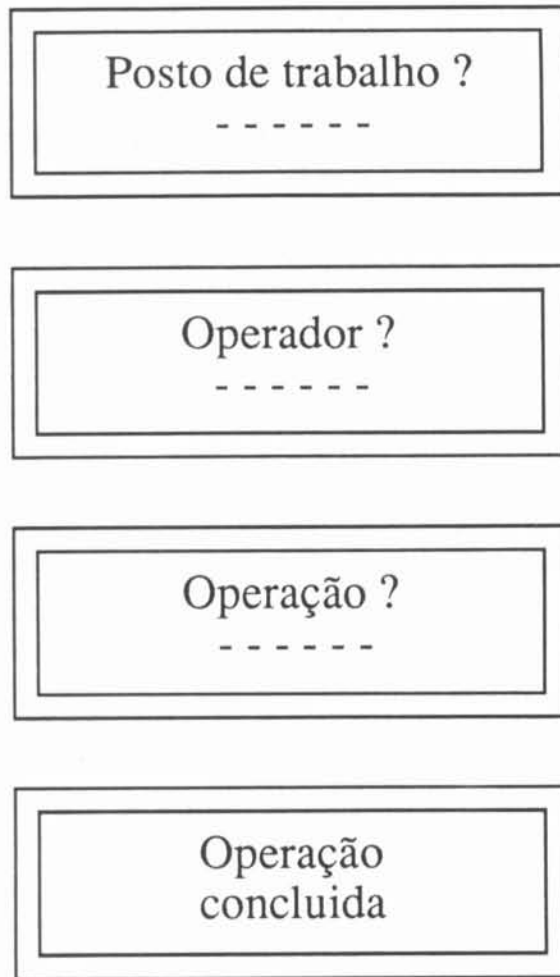


Figura 10.12 - Mensagens enviadas para o afixador alfanumérico da U.C. durante uma sinalização de início de operação

"Buffer" de registo de erros

A estrutura deste "buffer" é igual à dos "buffers" das unidades de aquisição descritos no ponto anterior.

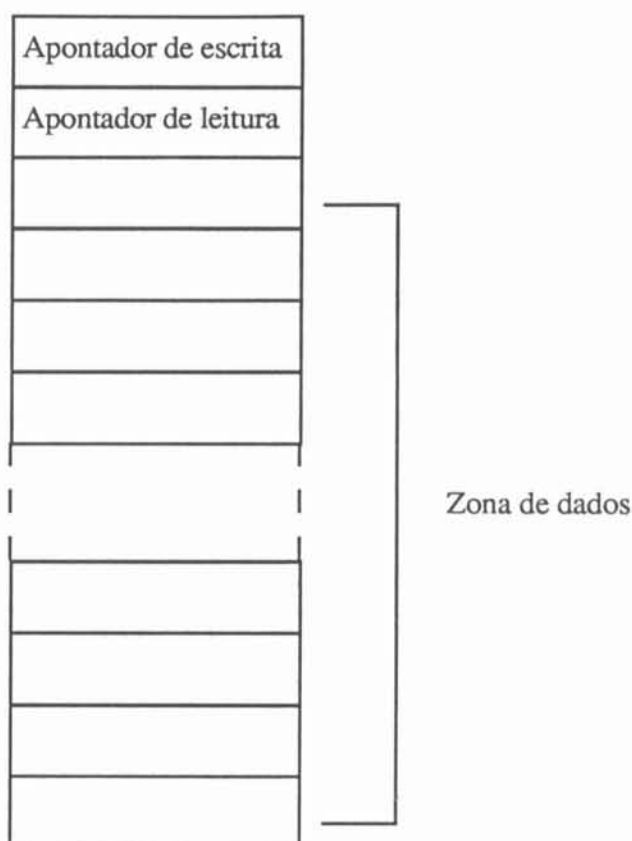


Figura 10.13 - "Buffer" de dados de uma unidade de aquisição

"Buffer" de transmissão de blocos

A estrutura deste "buffer", apresentada na figura 10.14, consiste basicamente num cabeçalho constituído por um byte de controlo de acesso, um byte de identificação da unidade de aquisição para a qual se pretende enviar ou da qual se pretende receber um bloco de memória, dois bytes para representar o endereço inicial do bloco e um byte para indicar o seu comprimento.

O conteúdo do byte de controlo de acesso está também representado na mesma figura.

10.1.2.2.2 - Integridade da informação

Os "buffers" descritos no ponto anterior são locais onde, tal como foi identificado na fase de Análise Estruturada das unidades concentradoras, a informação é partilhada por vários módulos. Tornou-se pois necessário implementar mecanismos que garantam a integridade da informação neles armazenada.

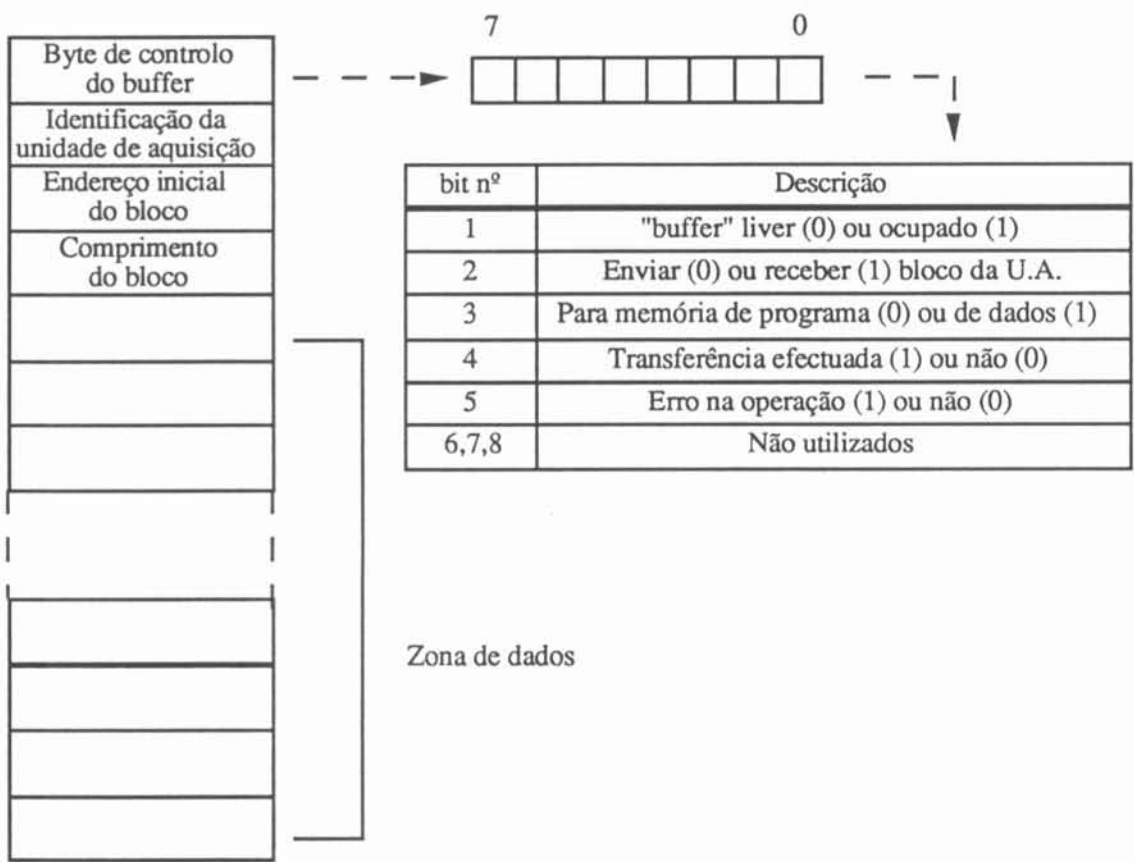


Figura 10.14 - "Buffer" de transmissão de blocos

No caso dos "buffers" das unidades de aquisição e do "buffer" de registo de erros, a unidade concentradora escreve informações nos "buffers", as quais são posteriormente lidas pela unidade central. Uma situação na qual pode existir perda da integridade da informação será a leitura, pela unidade central, da informação de um "buffer" de uma unidade de aquisição a meio de um

procedimento de sinalização, por parte de um operador, de início ou fim de operação.

Esta situação foi salvaguardada utilizando, nas operações de escrita nos "buffers", cópias do apontador de escrita em lugar do próprio apontador. No início dos módulos que contêm operações de escrita nos "buffers" é criada uma cópia do apontador de escrita. É essa cópia que é utilizada para endereçar as operações de escrita, e o seu valor é incrementado por cada byte inserido. Terminada a operação, o apontador de escrita é actualizado com o valor da sua cópia ou seja, somado do número de bytes que foram escritos pelo módulo.

Desta forma a unidade central, como utiliza o apontador de escrita como limite do bloco a ler, só "vê" a informação que acabou de ser escrita no "buffer" quando estiver terminada a execução do módulo de escrita ou, por outras palavras, quando o apontador de escrita for actualizado com o valor da sua cópia.

Quer a unidade central quer a unidade concentradora, podem ler e escrever informação no "buffer" de transmissão de blocos. Para gerir estas operações de leitura e escrita foram criadas nos cabeçalhos respectivos um conjunto de variáveis, cujo funcionamento se descreve a seguir (referindo a figura 10.14) para o caso do envio de um bloco de informação para uma unidade de aquisição:

- a unidade central, utilizando a rede BITBUS, testa o bit 1 para verificar se o "buffer" está livre. No caso afirmativo a unidade central complementa este bit. Neste caso não é necessário implementar mecanismos do tipo "test and set", uma vez que só a unidade central vai executar esta operação.
- seguidamente a unidade central escreve no bit 2 o valor lógico "1", no bit 3 "0" ou "1" conforme o bloco se destine à memória de dados ou de programa, no bit 4 e 5 "0", e escreve no cabeçalho do "buffer" o endereço da unidade de aquisição a que se destina o bloco, o endereço inicial e o seu comprimento;
- a unidade concentradora encarrega-se de executar o serviço, transferindo o bloco para a unidade de aquisição. Após a execução do serviço, a unidade concentradora complementa o bit 4 e escreve no bit 5 o valor lógico "1" ou "0" conforme se tenha ou não verificado um erro durante o processo;

- finalmente, a unidade central, que testa ciclicamente o valor do bit 4, detecta que a transferência foi executada quando o seu valor for "1". É ainda testado o bit 5 para verificar a existência de algum erro, e seguidamente o bit 1 é complementado, libertando assim o "buffer" para receber um novo pedido, por parte da unidade central, de transferência de um bloco.

10.1.3 - Unidade central

10.1.3.1 - Hardware

O hardware da unidade central é constituído por um computador PC 386 com "slots" de expansão AT compatíveis, um dos quais é utilizado para ligação da placa de interface à rede BITBUS. Estas placas são comercializadas por vários fabricantes, como a Intel, Data Translation e Phoenix. Dado o seu custo ser relativamente baixo quando comparado com o dos restantes componentes do sistema, e dado o considerável esforço de desenvolvimento de um tal produto, a aquisição da placa foi a opção natural.

O computador utilizado é um PC 386 AT-compatível com 4 Mbytes de RAM, 160 Mbytes de disco duro, uma unidade de banda magnética de 160 Mbytes e uma unidade de diskettes de 5 1/4 - 1.2 Mbytes. A placa de interface à rede BITBUS é a PCX 344 da Intel. Foi ainda utilizado hardware (carta WD8003E da Western Digital) e software (SCO TCP/IP) de ligação à Ethernet, prevendo a ligação a outros sistemas informáticos, de modo a permitir a partilha da informação.

Este equipamento acomoda o sistema gestor da base de dados, onde são armazenadas as informações referentes aos postos de trabalho, e é o mestre da rede BITBUS.

Finalmente, dado o seu papel crítico de mestre da rede, foi incluída no hardware da unidade central uma fonte de alimentação não interruptível.

10.1.3.2 - Software

10.1.3.2.1 - O sistema operativo

Se é certo que o sistema operativo MS-DOS é ainda o mais utilizado na área da gestão industrial, não é menos certo que, com o decréscimo do custo dos computadores e com a necessidade

crescente de sistemas multiutilizador e multitarefa, as máquinas 386 utilizando o sistema operativo XENIX ou UNIX têm vindo a implantar-se como a solução "standard" para os próximos anos.

A escolha recaiu sobre o sistema operativo SCO XENIX 386 pelas seguintes razões:

- o XENIX é um sistema operativo menos exigente do que o UNIX, no que diz respeito à RAM e ao espaço em disco necessários, resultando daí uma configuração mais económica;
- o software desenvolvido em ambiente SCO XENIX 386 é portátil para SCO UNIX 386, não sendo o inverso, contudo, verdadeiro.

Quando o número de postos de trabalho a controlar for pequeno, e/ou quando existirem já instalados sistemas informáticos de gestão utilizando MS-DOS, poderá ser vantajoso utilizar esse sistema operativo como suporte para o software de controlo de produção. O software de controlo de produção deverá portanto ser portátil, ou facilmente convertível, para este sistema operativo.

10.1.3.2.2 - O sistema gestor de base de dados

Existe disponível comercialmente uma grande variedade de sistemas gestores de base de dados, dos quais se destacam, pelo número de sistemas instalados e pelo número e diversidade de aplicações que suportam, o DBASE III/IV e CLIPPER (versão compilável), o INFORMIX e o ORACLE. Após uma comparação das características técnico-económicas destes sistemas foi preferido o INFORMIX, pelas razões que a seguir se apontam:

- só existe DBASE para MS-DOS, o que inviabiliza a sua utilização face à opção feita relativamente ao sistema operativo;
- o INFORMIX dispõe, além do SQL "standard", de uma linguagem de programação de quarta geração, o 4GL, particularmente vocacionada para facilitar o desenvolvimento de aplicações com base de dados. Existe ainda um conjunto de ferramentas de ajuda ao desenvolvimento de software, como o RDS ("Rapid Development System"), que é um programa que cria um

ambiente de desenvolvimento conduzido por menus cobrindo todas as fases do desenvolvimento de software, um "debugger" interactivo e bibliotecas de funções SQL que podem ser incluídas nas linguagens de programação mais vulgares, como PASCAL, "C", FORTRAN, etc.;

- existem versões do INFORMIX e de todos os seus produtos para cerca de 250 combinações máquina/sistema operativo, nomeadamente PC-AT compatíveis com MS-DOS, XENIX e UNIX, estações de trabalho SUN ou HP com UNIX e estações de trabalho Digital com UNIX ou VMS, etc;
- as aplicações desenvolvidas em INFORMIX-4GL-RDS são 100 por cento portáteis, bastando para tal adquirir o "runtime" para a máquina em questão. Se as aplicações forem desenvolvidas em INFORMIX-4GL, é necessário também adquirir o compilador para a nova máquina e recompilar o código fonte;
- existia já alguma experiência de desenvolvimento de aplicações em INFORMIX no grupo de trabalho onde este projecto está inserido, pelo que o esforço inicial de aprendizagem da linguagem seria menor do que, por exemplo, no caso do ORACLE;
- houve algumas dificuldades em obter informações sobre o ORACLE e pelo que foi possível averiguar, quer através de documentação, quer através de utilizadores deste sistema gestor de base de dados, o ORACLE parece sobretudo vocacionado para aplicações de médio e grande porte não sendo muito apropriado para ser utilizado em máquinas do tipo PC compatível;
- na altura em que foi feita esta análise, o custo do software necessário para fazer desenvolvimento em ORACLE era, para a mesma configuração, cerca de três vezes superior ao do software necessário para desenvolver em INFORMIX .

10.1.3.2.3 - O programa de controlo de produção

O "programa de controlo de produção", cuja implementação seguidamente se apresenta, não pretende propor soluções de processamento de informação ao nível do planeamento e controlo da produção, nem da interligação com outras áreas da gestão

industrial. Por um lado existem já disponíveis no mercado pacotes de software que implementam essas funções, e por outro essa área sai fora do âmbito do presente trabalho.

O principal objectivo deste programa é:

- recolher e armazenar, numa base de dados, as informações relativas aos postos de trabalho;
- permitir o acesso à informação por outros pacotes de software (o que pode ser feito, por exemplo, utilizando os produtos INFORMIX ESQL, ou recorrendo a ficheiros para a transferência da informação);
- dispôr de módulos de software que permitam demonstrar as potencialidades de sistemas deste tipo como geradores de informações relevantes, não só para o controlo da produção mas também para outras áreas da gestão fabril, como o planeamento, controlo horário, custos, etc.

Pelos motivos que acabam de ser apresentados, os módulos "Actualização", "Monitorização" e "Históricos" foram alvo de especial atenção.

Os módulos de software que constituem o programa de controlo de produção, já apresentados no capítulo nove, podem ser agrupados em duas categorias: os que são típicos de aplicações de base de dados e os que, além da interface com a base de dados, incluem uma interface com a rede BITBUS. Dentro da primeira categoria incluem-se os módulos "Monitorização", "Históricos", "Configuração" e "Erros", enquanto que os módulos "Actualização" e "Inicialização" cabem na segunda.

Os módulos "Opções", "Monitorização", "Históricos", "Erros" e "Configuração" foram desenvolvidos em 4GL porque, tal como foi referido no ponto anterior, esta linguagem é especialmente vocacionada para implementar interfaces entre o operador e a base de dados vocacionadas para a consulta e/ou alteração da informação. Os módulos "Inicialização" e "Actualização" foram desenvolvidos em "C", utilizando as bibliotecas de funções SQL para essa linguagem, o ESQL/C, por dois motivos fundamentais:

- já existiam, desenvolvidos em "C", os "drivers" para XENIX que implementam os comandos RAC do BITBUS e a interface à placa mestre da rede, o que exigia que o software de

interface à rede tivesse de ser desenvolvido nessa linguagem;

- pretendia-se que o módulo "Actualização" fosse efectivamente um programa independente do programa constituído pelos restantes módulos, para que pudesse ser activado pelo sistema operativo a intervalos de tempo a definir pelo gestor do sistema. Para tal ele deveria por um lado implementar a interface com a rede BITBUS, para recolher as informações das unidades concentradoras, e por outro armazenar essas informações na base de dados utilizando as funções do ESQL/C.

Nos DFI's apresentados no capítulo nove pode identificar-se um conjunto de campos da base de dados onde a informação é partilhada por várias tarefas podendo, em alguns casos, esses campos ser lidos e escritos concorrentemente por duas tarefas. Não foi todavia necessário implementar mecanismos de acesso mutuamente exclusivos para garantir a integridade da informação, visto que tais mecanismos são já implementados pelo sistema gestor de base de dados. Com efeito, dado que o INFORMIX foi desenvolvido para funcionar em ambientes multiutilizador e multitarefa, inclui um conjunto de funções que permitem controlar o acesso a blocos de informação, como por exemplo as que permitem inibir ou autorizar o acesso a campos ou a linhas (conjuntos de campos) das tabelas da base de dados. Este mecanismo, extremamente útil e eficaz, foi usado no módulo "Actualização" da forma que a seguir, resumidamente e a título de exemplo, se ilustra :

- antes do módulo "Actualização" escrever na tabela produção uma nova linha correspondente a uma operação que foi iniciada, ou actualizar a variável **produção** referente a uma operação já existente, ele inibe o acesso a essa linha tornando-a "invisível" a todos os outros módulos;
- em seguida actualiza a tabela produção em função das informações recebidas através da rede industrial;
- terminada a actualização da tabela, permite novamente o acesso à linha, ficando a informação disponível para ser utilizada por todos os módulos.

É assim possível resolver o problema das zonas críticas de informação sem necessidade de desenvolver mecanismos que

garantam a exclusão mútua no acesso a recursos partilhados, como por exemplo semáforos, eventos ou monitores.

Apresentam-se a seguir, como exemplo, os menus gerados pelo programa quando é seleccionada a opção de "Monitorização". Não é apresentada a descrição da implementação dos restantes módulos dado o seu grande número e o facto de serem típicos de aplicações com bases de dados.

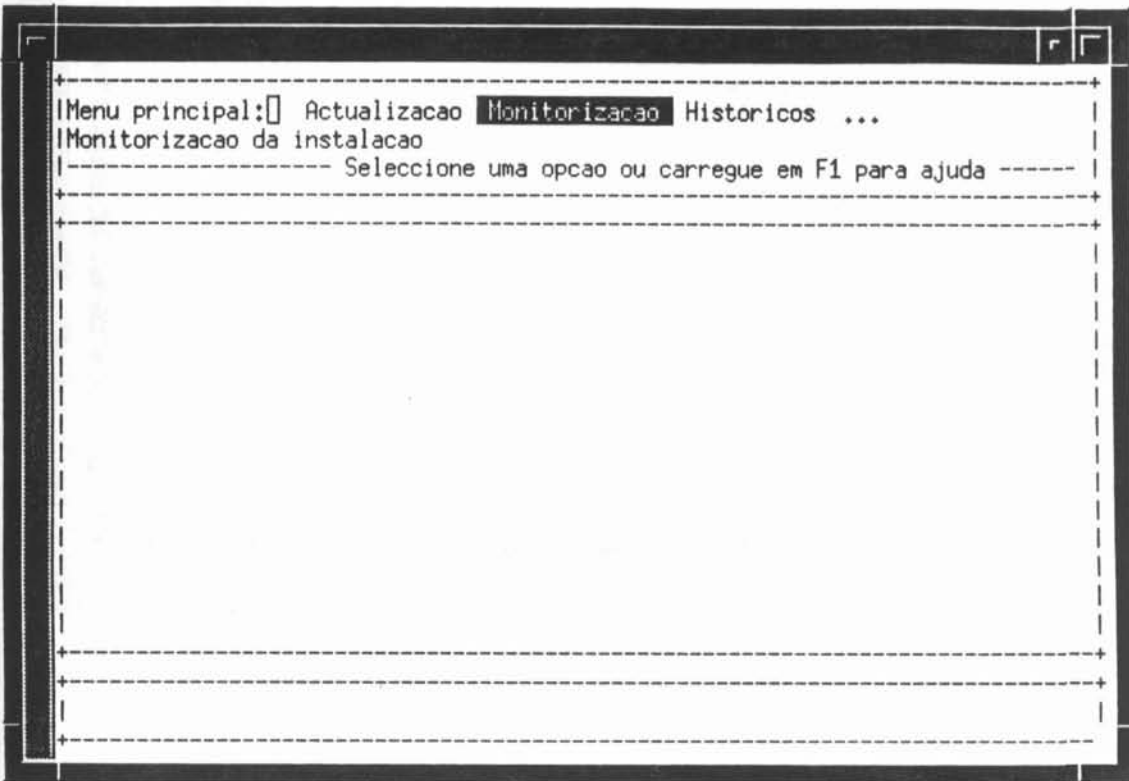


Figura 10.15 - Selecção da opção "Monitorização"

CRITERIO DE SELECCAO (ESC para aceitar)									
Maquinas[]		Operadores[]		Operacoes[]					
LINHAS SELECCIONADAS									
Operador:Codigo[]		Nome[]							
Maquina:Codigo[]		Nome[]							
Operacao:Codigo[]		Nome[]							
Data[]		Estado []				Media Real []			
Hora[]		Producao[]				Media Teorica[]			
Operador:Codigo[]		Nome[]							
Maquina:Codigo[]		Nome[]							
Operacao:Codigo[]		Nome[]							
Data[]		Estado []				Media Real []			
Hora[]		Producao[]				Media Teorica[]			

Figura 10.16 - Introdu  o da m  scara de selec  o

```

+-----+
|
|
|CRITERIO DE SELECCAO (ESC para aceitar)
|
+-----+
|Maquinas[torn*      ] Operadores[          ] Operacoes[[]      ]
|-----LINHAS SELECCIONADAS-----
|IOperador:Codigo[    ] Nome[                                ]
|IMaquina:Codigo[     ] Nome[                                ]
|IOperacao:Codigo[    ] Nome[                                ]
|IData[              ] Estado [          ] Media Real [        ]
|IHora[              ] Producao[          ] Media Teorica[      ]
|
|IOperador:Codigo[    ] Nome[                                ]
|IMaquina:Codigo[     ] Nome[                                ]
|IOperacao:Codigo[    ] Nome[                                ]
|IData[              ] Estado [          ] Media Real [        ]
|IHora[              ] Producao[          ] Media Teorica[      ]
|
+-----+
|
|
+-----+

```

Figura 10.17 - Máscara de selecção definida: máquinas cujo código comece por "torn", todos os operadores e todas as operações

Utilize as setas do cursor para ver os varios registos
Pressione ESC para terminar

-----LINHAS SELECCIONADAS-----

Operador:Codigo[000000003]	Nome[Raul Serafim]	
Maquina:Codigo[torno00001]	Nome[torno mecanico 3m.]	
Operacao:Codigo[torno00001]	Nome[torneamento 3m.]	
Data[10/12/1990]	Estado [manutencao]	Media Real [0.00]
Hora[15:30:00]	Producao[]	Media Teorica[60]
Operador:Codigo[]	Nome[]	
Maquina:Codigo[]	Nome[]	
Operacao:Codigo[]	Nome[]	
Data[]	Estado []	Media Real []
Hora[]	Producao[]	Media Teorica[]

Figura 10.18 - Visualização da informação referente às linhas seleccionadas

```

+-----+
|Utilize as setas do cursor para ver os varios registos|
|Pressione ESC para terminar                           |
+-----+
|Maquinas[torn*   ] Operadores[           ] Operacoes[           ]|
|-----LINHAS SELECCIONADAS-----|
|Operador:Codigo[0000000003] Nome[Raul Serafim           ]|
|Maquina: Codigo[torno00001] Nome[torno mecanico 3m.    ]|
|Operacao:Codigo[torno00001] Nome[torneamento 3m.      ]|
|Data[10/12/1990] Estado [manutencao                    ] Media Real [ 0,00]|
|Hora[15:30:00]  Producao[                               ] Media Teorica[ 60]|
|-----|
|Operador:Codigo[           ] Nome[                     ]|
|Maquina: Codigo[           ] Nome[                     ]|
|Operacao:Codigo[           ] Nome[                     ]|
|Data[           ] Estado [           ] Media Real [           ]|
|Hora[           ] Producao[           ] Media Teorica[           ]|
|-----|
|Quer fazer mais alguma seleccao (s/n)?                |
+-----+

```

Figura 10.19 - Opção para nova selecção ou regresso ao menu anterior

10.2 - Testes

Foram realizados em primeiro lugar testes estáticos, i.e. de avaliação analítica da documentação, e posteriormente testes dinâmicos com o objectivo de garantir a correcção do software. Além dos testes efectuados durante a implementação estruturada dos diversos modelos, a metodologia de teste de software incluiu testes de percursos lógicos, testes de valores limites e excepcionais, tentativas de acessos não autorizados, simulações do funcionamento real do sistema, etc.

Foi ainda efectuado um conjunto específico de testes com o objectivo de procurar avaliar o desempenho, como sistema de tempo real, do sistema desenvolvido.

Esses testes incidiram sobre um conjunto de módulos com grande importância no desempenho e no dimensionamento do

sistema, nomeadamente no que respeita ao número máximo de unidades de aquisição para cada unidade concentradora, à memória RAM necessária na unidade concentradora por cada unidade de aquisição, etc. Como exemplo ilustrativo da necessidade dessa informação, pode considerar-se o caso do módulo "Varrimento". Uma vez que as unidades de aquisição não dispõem de relógio de tempo real, a precisão do registo horário das ocorrências vai depender do tempo necessário para a realização do ciclo de varrimento às unidades de aquisição "penduradas" numa dada unidade concentradora. É pois importante conhecer o tempo médio associado ao varrimento a cada unidade de aquisição, de modo a poder fixar, em função do erro horário máximo admissível, o número máximo de unidades de aquisição por unidade concentradora.

Apresentam-se a seguir alguns desses testes e os resultados obtidos.

10.2.1 - Unidades de aquisição

- Tempo de execução de cada ciclo do módulo "Monitorização": aproximadamente 200 microsegundos.
- Tempo de execução do módulo "Reset do contador parcial": 12 microsegundos.

10.2.2 - Rede própria

Tempo de execução dos serviços oferecidos pela rede própria:

- "Recebe status": 1,5 milisegundos.
- "Reset do contador": 1,2 milisegundos.
- "Envia bloco" (bloco de 100 bytes de informação): 13 milisegundos.
- "Recebe bloco" (bloco de 100 bytes de informação): 13 milisegundos.

É importante notar que a principal componente dos tempos apresentados é o tempo de transmissão. Se forem utilizadas taxas de transmissão mais elevadas na rede própria estes tempos sofrerão reduções significativas.

10.2.3 - Unidades concentradoras

O módulo "Varrimento", sem dúvida o que mais impacto tem no desempenho das unidades concentradoras, é constituído por um bloco de software, descrito no capítulo sete, executado ciclicamente para cada unidade de aquisição presente na rede própria. O tempo de varrimento de todas as unidades de aquisição vai pois depender do seu número e do tipo de operações a serem efectuadas em cada uma delas.

O teste realizado constou da realização de 50 ciclos de varrimento, espaçados de dois segundos, a duas unidades de aquisição, durante os quais se alteraram os estados dos postos de trabalho. O tempo médio de execução do varrimento a cada unidade de aquisição foi de aproximadamente 6.2 milisegundos.

10.2.4 - Unidade central

Foi realizado um conjunto de testes para avaliar o desempenho da rede BITBUS na execução de operações de transferência de blocos de informação. Para isso, foram utilizados pequenos programas em "C", desenvolvidos sobre os "drivers" do BITBUS, e recorreu-se à função "time" do XENIX, cuja saída é o tempo de sistema e o tempo de CPU gastos durante execução de um programa que lhe é passado como parâmetro. Assim utilizou-se um programa de teste que envia um bloco de mil bytes para um nó da rede, e um outro que executa a tarefa inversa.

Os resultados obtidos foram os seguintes:

- programa "envia"

tempo de ocupação do CPU = 1.2 segundos

tempo de sistema = 4 segundos

- programa "recebe"

tempo de ocupação do CPU = 1.2 segundos

tempo de sistema = 3.9 segundos

Em seguida foi calculado o tempo médio de processamento, por parte da unidade central, da informação recolhida pelo módulo "Actualização".

Tal como no caso da módulo "Varrimento", apresentado no ponto anterior, executaram-se 50 ciclos de actualização da informação referente a quatro unidades de aquisição, distribuídas

IMPLEMENTAÇÃO E TESTES

idades concentradoras. As actualizações foram feitas a intervalos de quinze minutos, durante os quais se alteraram os dados dos postos de trabalho e se sinalizaram inícios e fins de operações na unidade concentradora.

O tempo médio de sistema assim obtido para a actualização da base de dados de cada unidade foi de 0.4 segundos.

O tempo médio de execução de um ciclo de actualização da base de dados vai variar de caso para caso pois depende, entre outros, do tamanho do fabrico da empresa, do comprimento dos códigos de barras para identificar as operações, os operadores e os postos de trabalho e do tempo que decorre entre ciclos de actualização. Se as ordens de execução das ordens de fabrico forem pequenos, haverá muitas sinalizações de início e fim de operação, logo a quantidade de informação a ser processada será grande. De qualquer modo, uma vez caracterizados estes parâmetros, e com base nos resultados obtidos nos testes, poder-se-á estimar um tempo médio por ciclo. Esse tempo médio poderá depois ser ajustado para obter o tempo pretendido alterando o tempo entre ciclos.

CAPITULO 11

Conclusões e evolução futura

11 - CONCLUSÕES E EVOLUÇÃO FUTURA

Terminada a fase de implementação e testes do protótipo laboratorial do sistema, é possível retirar do trabalho desenvolvido algumas conclusões e perspectivar a evolução futura de um sistema deste tipo.

11.1 - Conclusões

1. A utilização de Técnicas de Análise e Projecto Estruturado na especificação, projecto e implementação do sistema mostrou-se extremamente útil e vantajosa. Por comparação com desenvolvimentos efectuados anteriormente, nos quais não foram utilizadas estas técnicas, foi possível verificar que o esforço desenvolvido na construção dos DTE's, DFI's e ME's é largamente compensado pela redução do tempo e do esforço dispendidos na implementação e teste do sistema. A identificação dos fluxos de informação, o particionamento do sistema em módulos simples, a representação conjunta dos fluxos de informação, da estrutura hierárquica e das condições de transição entre módulos do sistema, e a identificação de regiões e módulos críticos, permitem acelerar a fase de implementação e possibilitam uma identificação precoce das possíveis fontes de erros.

2. Com o desenvolvimento de uma rede própria, além de se conseguir uma redução apreciável do custo final do sistema sem degradação do seu desempenho, criou-se uma ferramenta para o desenvolvimento e teste de novas soluções para redes industriais de baixo custo no que diz respeito a taxas de transmissão, técnicas de detecção e correcção de erros de transmissão, comprimentos máximos permitidos para as mensagens, etc. É possível ajustar todos estes parâmetros por forma a maximizar o desempenho global.

3. A implementação efectuada teve como objectivo primordial, para além do baixo custo, a maximização da reutilização do hardware e do software desenvolvidos, mesmo à custa de algum acréscimo do custo.

O hardware e software foram desenvolvidos de forma modular, para que pudessem ser facilmente reconfiguráveis e adaptáveis a novas exigências. Daqui resultaram alguns subsistemas, que poderão ser utilizados em futuras variantes de implementações do

sistema, diminuindo muito o esforço de desenvolvimento correspondente. Esses subsistemas são os seguintes:

- carta de núcleo central para o microcontrolador 8044: inclui o microcontrolador, suportes para vários tipos de memórias de programa e de dados (programáveis através de "jumpers"), decodificação, interface para a rede BITBUS e ligação a um barramento de expansão com todos os sinais necessários para a utilização de placas de interface customizadas (linhas de dados, endereços, decodificação, interrupção, "timers" e contadores, etc.). Esta unidade é 100 por cento compatível com a especificação do BITBUS;
- carta de interface com o operador: contém interfaces para um teclado de 24 teclas, para um afixador alfanumérico do tipo LCD, para um relógio de tempo real e para uma porta série;
- carta de núcleo central para o microcontrolador 8031: inclui o microcontrolador, suportes programáveis para memórias de programa e de dados, decodificação e um conector de expansão com todos os sinais do microcontrolador e de decodificação;
- biblioteca de funções de uso corrente para os microcontroladores 8044 e 8031. Entre elas incluem-se funções para configuração e operação do teclado, LCD, relógio de tempo real e porta série.

4. O desenvolvimento de um protótipo industrial, com vista a uma possível comercialização do sistema, tem objectivos diferentes dos expostos anteriormente. Neste caso, será vital diminuir o custo do hardware e entrar em conta com outros aspectos de industrialização, pelo que é natural que o projecto do hardware das placas seja alterado. Se, por exemplo, não for necessário prevêêr a expansão das unidades de aquisição, deixará de haver necessidade de interface ao barramento de expansão. Por outro lado, em vez do hardware estar distribuído por duas placas (o núcleo central do 8031 e a placa com o hardware de interface ao operador, ao posto de trabalho e à rede própria) será possível agrupá-lo todo numa só placa. Conseguem-se assim reduções possivelmente significativas no custo de cada unidade de aquisição, e com relevância no custo final do sistema.

5. O custo, para pequenas quantidades, dos componentes de uma unidade de aquisição situa-se entre os vinte e os trinta e cinco mil escudos, em função do tipo de alimentação pretendido. O primeiro valor refere-se a uma alimentação a 12V ou 24V DC, e o segundo a 220V AC. Este custo deve ser acrescido do custo do respectivo invólucro, e diminuído do desconto que se obtém ao comprar os componentes em grandes quantidades, e que é, em muitos casos, apreciável. A mesma análise feita em relação às unidades concentradoras, resulta num custo que se situa entre os trinta e cinco e os cinquenta mil escudos. Considerando o caso de termos cem postos de trabalho divididos em dez sectores, o custo das unidades de aquisição e concentradoras necessárias para os informatizar, para o caso de alimentação a 24V DC, ronda os dois milhões e trezentos e cinquenta mil escudos. Se compararmos este preço com o de uma configuração equivalente, utilizando hardware importado disponível no mercado nacional proveniente de dois fabricantes distintos, poderemos constatar que existe uma poupança de cerca de 60% num caso e de 71% no outro. Neste estudo não foi incluído o preço dos cabos nem da unidade central, uma vez que são semelhantes para qualquer dos casos.

Ao efectuar uma análise de custos é necessário considerar também o custo do desenvolvimento do sistema. O esforço de desenvolvimento aplicado neste trabalho foi avaliado em cerca de 18 homens/mês. No entanto este esforço será dividido por várias áreas, como desenvolvimento de hardware e software para microcontroladores de oito bits, projecto e desenho de placas em circuito impresso, redes industriais, bases de dados, etc., o que obriga a uma atenção especial na quantificação do custo do homem/mês.

O custo correspondente ao esforço de desenvolvimento terá, no entanto, um peso secundário dado que é diluído por um número bastante elevado de componentes/sistemas.

6. É de notar que, mesmo incluindo o preço da unidade central, da cablagem e do software, o custo total do sistema de controlo da produção representa, na maior parte dos casos, uma percentagem muito pequena do custo dos equipamentos de produção a informatizar.

Consegue-se assim obter um sistema de baixo custo, sem dúvida acessível à grande maioria das PME's portuguesas.

11.2 - Evolução futura

A evolução dos preços dos componentes poderá vir a influenciar quer a arquitectura do sistema quer a das várias unidades que o integram. O preço do microcontrolador 8044, por exemplo, decresceu mais de 60% desde o início deste projecto, enquanto que o do microcontrolador 8031 decresceu cerca de 90%. Poder-se-ão desenvolver num futuro próximo unidades mais sofisticadas, capazes de implementar outras funções, a preços iguais ou inferiores aos das actuais.

A evolução futura de sistemas deste tipo está sobretudo dependente da evolução técnico-económica das empresas às quais é dirigido. Ao aumentarem os seus volumes de investimento, e com a componente mão-de-obra cada vez mais importante no custo final do produto e a exigência de tempos de fabrico e prazos de entrega cada vez mais curtos, as empresas sentirão cada vez mais a importância das questões de planeamento e controlo da produção, levando a investimentos em sistemas cada vez mais sofisticados e tendo como meta um cenário vulgarmente conhecido como CIM ("Computer Integrated Manufacturing").

É importante prespективar desde já esse futuro próximo, para que as empresas possam dispôr de produtos que lhes permitam responder aos desafios que lhes vão surgir.

Cientes da existência de um mercado e da falta de soluções adequadas, as Comunidades Europeias lançaram diversos projectos de I&D, no âmbito do programa ESPRIT, nestas áreas, como por exemplo o CIDAM, FLEXPLAN, CNMA, FIELDBUS, SHOP-CONTROL, etc. [13], contemplando as áreas de recolha de informação fabril, bases de dados em ambiente industrial, planeamento e controlo da produção, redes industriais, etc.

Será pois de primordial importância acompanhar as evoluções que se verificarem nestas áreas e desenvolver, ou implementar por integração, sistemas que as contemplem, adaptando-os todavia às realidades da indústria portuguesa.

Do nosso conhecimento da indústria no momento actual parece-nos que a aceitação de sistemas deste tipo está ainda fortemente dependente do seu custo, dado o empresário o comparar, quase exclusivamente, com as reduções de mão de obra directa associadas à sua introdução, e não valorizar convenientemente outros factores, como a fiabilidade e a disponibilidade da informação em tempo útil para tomada de decisões, etc.

Esta perspectiva é, de certa forma, compreensível, se atendermos que na maioria das PME's estão actualmente em curso esforços de reorganização e de renovação de equipamentos produtivos, factores vitais para assegurar a competitividade no dia-a-dia, sendo a integração do ponto de vista da informação fabril ainda uma visão futurista.

Não obstante, os contactos em curso para instalação do sistema em condições reais de teste têm sido francamente encorajadores.

Referências e bibliografia

REFERENCIAS

- [1] BASTOS, J. M. (1989)
Configuração, Planeamento e Controlo em Sistemas de
Produção Discreta.
Tese de Doutoramento, FEUP
- [2] BANDIN, M. (1990)
Manufacturing Systems Analysis with Application to
Production Scheduling
Yourdon Press
- [3] RANKY, P. G. (1990)
Computer Networks for World Class CIM Systems
CIMware Limited
- [4] STALLINGS, W. (1985)
Data and Computer Communications
Macmillan, Inc.
- [5] NEVES, N.; RIBEIRO, D.; MARQUES, P.; FERNANDES, M.
(1988)
DCM - Sistemas de Controlo Distribuidos
Trabalho realizado no âmbito da cadeira de Projecto do 5º
Ano do Curso de Engenharia Electrotécnica e dos
Computadores da FEUP
- [6] RANKY, P. G. (1990)
Flexible Manufacturing Cells and Systems in CIM
CIMware Limited
- [7] MENDONÇA, J. M. (1986)
Microcomputer On-Line Control of Wind Energy Conversion
Systems
Tese de Doutoramento, Imperial College, University of
London
- [8] PAGE-JONES, M. (1980)
The Practical Guide to Structured System Design
Yourdon Press
- [9] DE MARCO, T. (1979)
Structured Analysis and System Specification
Yourdon Press

- [10] YOURDON, E., CONSTANTINE, L. (1978)
Structured Design
Yourdon Press
- [11] PRESSMAN, R.S. (1987)
Software Engineering - A Practitioner's Approach
McGraw-Hill International Editions
- [12] iPCX 344 Intelligent BITBUS Interface Board User's Guide
(1986)
Intel Corporation
- [13] ESPRIT-CIM Project Synopses (1990)
Directorate General XIII - CEC
- [14] BITBUS Toolbox Manual (1986)
Intel Corporation
- [15] BERTSEKAS, D.; GALLAGER, R. (1987)
Data Networks
Prentice-Hall International Editions
- [16] 8-Bit Embedded Controller Handbook (1989)
Intel Corporation

BIBLIOGRAFIA

- Guide to Using the Distributed Control Modules
Intel Corporation (1984)
- Informix 4GL: User's and Reference Manual
- Informix ESQL/C: User's and Reference Manual
- Informix Rapid Development System: User's and Reference Manual
- Informix Interactive Debugger: User's and Reference Manual
Informix Software, Inc. (1987)
- SCO XENIX 386 Operating System: User's and Reference Manual
- SCO XENIX 386 Development System: User's and Reference Manual
The Santa Cruz Operation, Inc. (1987)

Com o objectivo de conhecer os produtos disponíveis no mercado, de alguma forma relacionados com o desenvolvimento efectuado, foram também utilizados como bibliografia, ao longo dos dois últimos anos, vários números dos seguintes jornais e revistas:

- Electronic Product News;
- Industrial Product News;
- Electronic Design;
- Control Engineering

APENDICES

APENDICE A

Descrição da arquitectura BITBUS

APENDICE A - DESCRIÇÃO DA ARQUITECTURA BITBUS [5] [14]

A.1 - Introdução

A Intel apresentou em 1984 a arquitectura BITBUS, baseada no protocolo SDLC/HDLC, para preencher a lacuna existente ao nível dos "standards" de conexão entre microcontroladores.

O protocolo de comunicação BITBUS destina-se a criar um barramento série de alto desempenho, fácil de utilizar e que seja transparente para o programador de aplicações.

Baseia-se na adopção de um sistema hierárquico a um ou dois níveis, que permite reduzir a complexidade do protocolo necessário bem como o tempo de processamento associado. Na estrutura de um só nível, o sistema BITBUS baseia-se na existência de um órgão central denominado mestre, que controla um ou vários sistemas pendurados na rede, escravos, que por si só não têm capacidade de tomar iniciativa de qualquer acção, limitando-se a responder às ordens enviadas pelo mestre (figura A.1).



Figura A.1: Estrutura hierárquica a um nível

No sistema multi-nível existe também um único mestre que controla todo o sistema, sendo no entanto possível que, a um nível mais baixo do que o escravo, existam um ou mais mestres a controlarem, cada um por si só, outras redes BITBUS (figura A.2).

Este tipo de ligação tem muito interesse, especialmente na interconexão de sistemas com velocidades diferentes.

Na verdade, o sistema multinível consiste em múltiplos sistemas BITBUS ligados entre si.

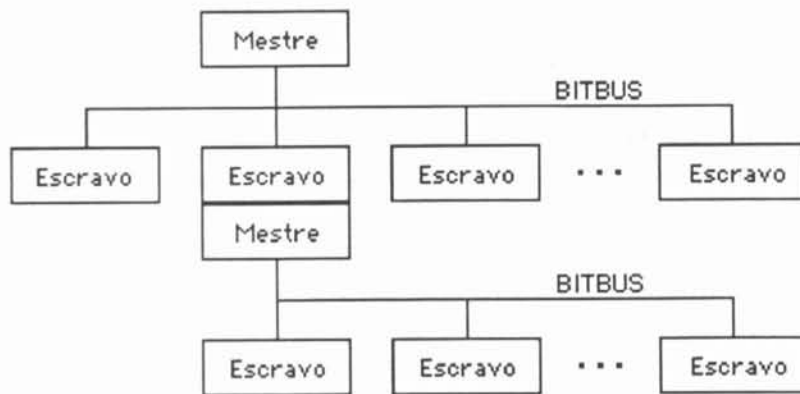


Figura A.2: Estrutura hierárquica a dois níveis.

A.2 - Nível físico

A.2.1 - Elementos constituintes

O BITBUS é constituído por uma série de nós nos quais estão penduradas as unidades compatíveis com este barramento. São quatro os tipos de elementos que podem estar inseridos na rede BITBUS: mestre, escravo, repetidor e extensão.

Mestre (master) - controla todas as operações no barramento BITBUS. Existe um só mestre por cada sistema BITBUS. Só o mestre pode iniciar um processo de transferência de informação. O mestre pode ser estacionário num nó ou então o atributo de mestre pode ser transferido na rede através de um sistema de token. É no entanto desaconselhado, do ponto de vista de fiabilidade, a troca de mestre. Esta troca deve porém ser realizada quando detectada uma avaria no mestre activo, especialmente se todos os potenciais mestres estiverem entre dois repetidores, separados por um pequeno comprimento de cabo. O atributo de mestre não pode ser passado por repetidores.

Esravo (slave) - são elementos que por si só não podem iniciar qualquer acção no barramento, limitando-se a responder às chamadas do mestre. Esta estrutura simplifica significativamente o protocolo de comunicação necessário.

Repetidor - trata-se de um elemento usado na regeneração dos sinais do BITBUS, permitindo desse modo aumentar o comprimento e o número de nós. O repetidor só pode ser usado no modo de operação auto (self-clocked) que adiante será discutido com maior detalhe.

Extensão - é basicamente um processador secundário num nó. A principal utilidade deste elemento reside no facto de ele permitir a comunicação entre vários sistemas do tipo BITBUS, ou entre um sistema tipo BITBUS e outro qualquer sistema. Dado sair do âmbito deste trabalho, não nos debruçaremos com mais profundidade sobre este tipo de elemento.

A.2.2 - Modos de operação do sistema

O BITBUS pode funcionar em dois modos distintos:

- modo síncrono (synchronous mode);
- modo auto (self clocked mode).

Estes dois modos de funcionamento diferem essencialmente na velocidade de transmissão, número máximo de nós admissíveis e número de condutores necessários.

Modo síncrono - este tipo de funcionamento permite elevado desempenho, embora a uma distância relativamente curta. De facto, são permitidas velocidades até 2,4 Mb/s a uma distância máxima de 30 metros, sendo o número máximo de nós admissíveis de 28.

Este modo de operação utiliza dois sinais diferentes, o que implica a utilização de dois pares de fios. O primeiro é utilizado para transmissão de informação (DATA, DATA*), e o segundo é utilizado para o sinal de relógio de sincronismo (DCLK, DCLK*). Neste modo não é possível a utilização de repetidores, pois estes não têm capacidade de regenerar o sinal de relógio.

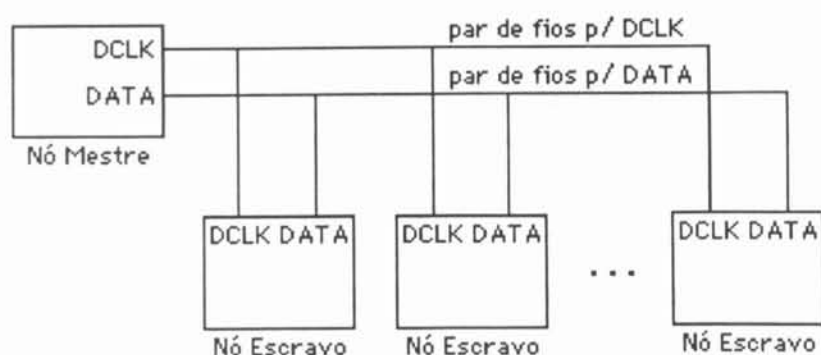


Figura A.3: Esquema de ligação no Modo Síncrono.

Modo auto - com este modo de operação é possível ter até um máximo de 250 nós recorrendo ao uso de repetidores, sendo no máximo admissíveis 10 repetidores. Entre cada repetidor podemos ter até 28 nós.

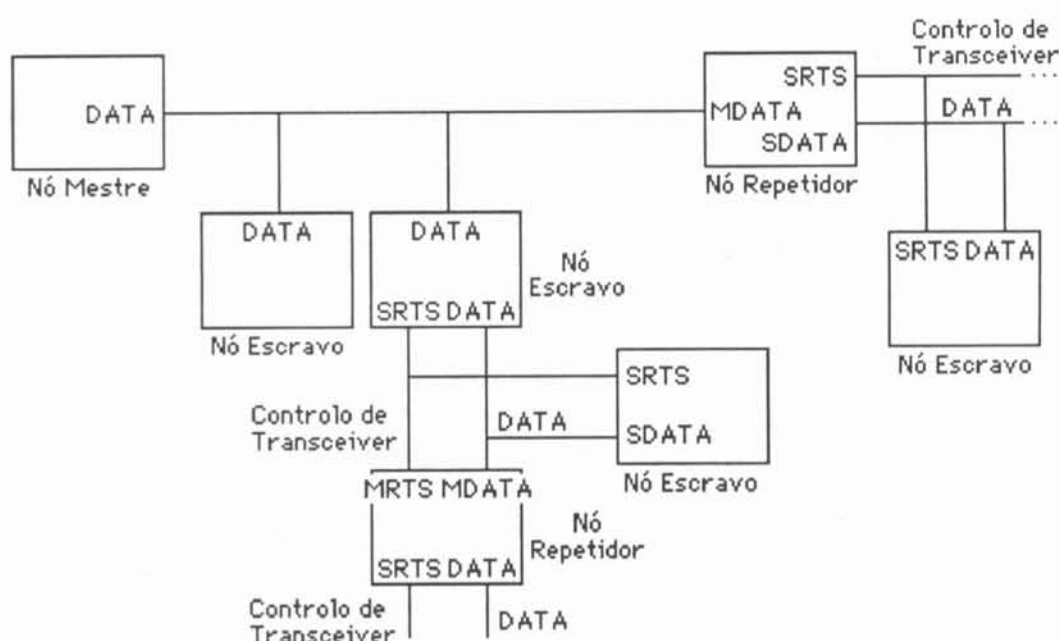


Figura A.4: Esquema de ligação no Modo Auto.

DESCRIÇÃO DA ARQUITECTURA BITBUS

Existem duas velocidades "standard" para este modo de funcionamento: 375 e 62.5 Kbytes/seg. À velocidade mais elevada a distância máxima é de 300 metros admitindo o uso de dois repetidores, enquanto que a 62,5 Kbytes/seg. é possível a ligação de até 10 repetidores e uma distância total de barramento de 2000 metros.

Este tipo de comunicação utiliza um par de fios (sinal diferencial) para informação (DATA, DATA*), necessitando de mais um par de fios para controlar os repetidores, caso existam.

O quadro A.1 apresenta uma comparação sumária entre os dois modos de operação.

	Velocidade Kb/s	Distância max. entre repetidores	Número de nós entre repetidores	Número max. de repetidores
Synchronous	2400	30/100	28	0
Self Clocked	375	300/1000	28	2
	62.5	1200/4000	28	10

Quadro A.1: Compromisso entre velocidade e distância.

A.2.3 - Especificações eléctricas

As especificações eléctricas do BITBUS requerem a definição de técnicas de codificação de informação e dos parâmetros DC/AC da interface lógica e do cabo de interligação. Vejamos resumidamente essas especificações.

A.2.3.1 - Técnicas de codificação

Como vimos, todos os sinais do BITBUS são diferenciais. A convenção usada pelo fabricante para designar o sinal diferencial é

por exemplo: SINAL e SINAL*. Se SINAL está a um potencial mais elevado do que SINAL*, o nível lógico é 1 (um), se SINAL está a um potencial inferior ao do SINAL*, então o nível lógico é 0 (zero).

Como atrás se refere, o BITBUS permite dois modos de operação, que usam códigos de informação diferentes. De seguida passaremos a descrevê-los em separado.

A.2.3.2 - Modo síncrono

No modo síncrono (synchronous mode) a informação é amostrada na transição ascendente do sinal relógio e a mudança de informação ocorre na transição descendente, como mostra a figura seguinte.

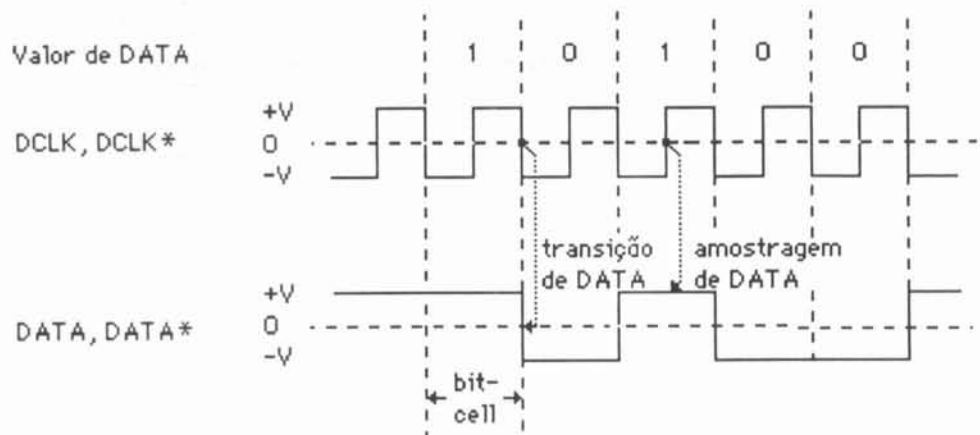


Figura A.5: Codificação de dados no modo síncrono.

A.2.3.3 - Modo auto (Self Clocked Mode)

Este modo de operação utiliza o "standard" de codificação NRZI (Non-return to zero inverted), que combina a informação e o sinal de relógio em apenas um par de sinais diferenciais. O nível lógico 0 é representado por uma mudança de polaridade do par de sinais, no bit-cell (duração de um bit, que corresponde a um período de relógio) precedente. O nível lógico 1 é representado pela não ocorrência de mudança de polaridade no bit-cell precedente. A figura seguinte esclarece melhor o que acabamos de dizer.

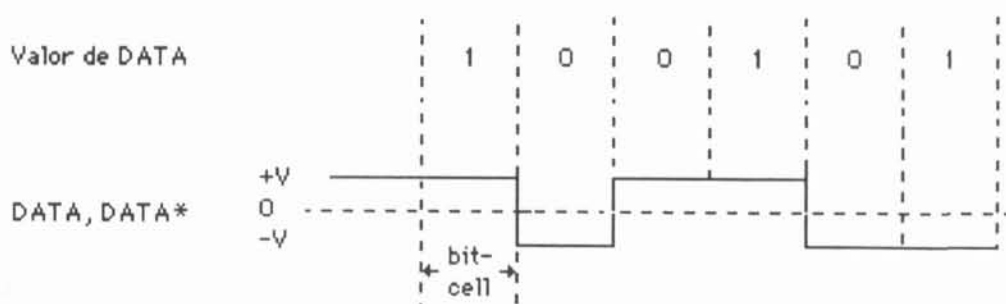


Figura A.6: Codificação de dados no modo Auto.

O modo auto especifica que o receptor obtém o sinal de relógio a partir da cadência da informação (data bit stream). Para o efeito, a interconexão BITBUS providencia um "phase locked loop" digital (DPLL). A sincronização dos receptores DPLL é garantida pela transmissão de uma "pré-trama" de sincronismo (PFS) antes do envio de qualquer trama. A PFS consiste no envio de um mínimo de oito zeros, isto é transições, por forma a sincronizar mestre com escravo.

A.2.4 - Especificações DC/AC

As especificações DC do BITBUS são baseadas no "standard" eléctrico RS485.

A especificação da unidade de carga "standard" é usada com base na definição de carga apresentada por um nó do BITBUS. A figura seguinte fornece a relação I/V de uma unidade de carga.

As especificações DC do transmissor e receptor, as características do cabo de ligação e as terminações do mesmo devem obdecer ao "standard" RS485.

Por não ser o objectivo primordial deste trabalho, informações detalhadas sobre este assunto, nomeadamente, especificações AC e testes dos parâmetros a respeitar, poderão ser consultados no catálogo da Intel "Distributed Control Modules Databook".

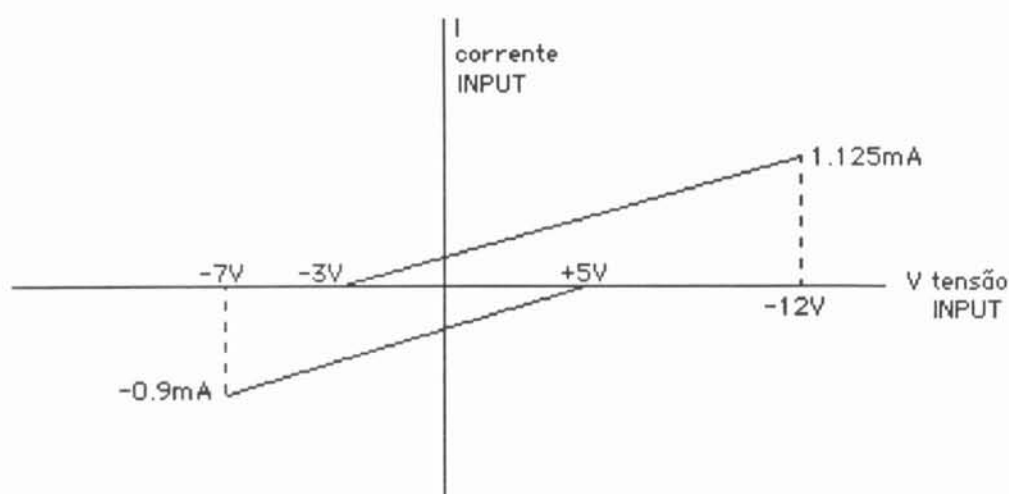


Figura A.7: Unidade de carga "standard".

A.3 - Protocolo de ligação (Data Link Protocol)

Segundo a hierarquia mestre-escravo atrás descrita, o BITBUS baseia-se no protocolo de ligação SDLC (Synchronous Data Link Control) da IBM.

O mestre tem sempre conhecimento, quer do seu próprio estado quer do estado do escravo. Por outro lado, nenhuma operação do escravo requer o conhecimento do estado do mestre pelo que o seu estado não passa na rede.

Modos do escravo

O escravo está sempre num dos dois modos de funcionamento seguintes:

- Modo de desligado (NDM - Normal Disconnect Mode);
- Modo de resposta (NRM - Normal Response Mode).

Um escravo está no modo NDM após um "reset" local ou quando detecta um erro irrecuperável. Uma vez neste modo, o escravo espera um comando específico do mestre para entrar no modo NRM. No modo NRM o escravo está sincronizado com o mestre.

Sequências de contagem

No modo NRM, o mestre e o escravo usam sequências de contagem para garantir que não se perde nem se duplica nenhuma trama.

Cada escravo guarda duas sequências de contagem de três bits, designadas por Ns (number sent) e Nr (number received). O mestre guarda o correspondente par de sequências de cada escravo.

Ns indica o número da próxima mensagem à espera de reconhecimento, e Nr indica o número da próxima mensagem esperada.

Após receber uma trama de informação, o receptor deve comparar o seu contador Nr com o contador Ns do emissor (enviado no campo de controlo da trama de informação, como veremos no capítulo seguinte). Se estiverem iguais a mensagem será aceite, caso contrário poderão ocorrer duas situações distintas: se Ns for diferente de Nr ou Nr-1 o erro será irrecuperável e nesta situação o mestre deverá resincronizar com o escravo. Se $Ns = Nr - 1$ o nó receptor poderá ainda aceitá-la, mediante algumas medidas, por exemplo pedir uma repetição da trama.

A.3.1 - Formato de uma trama (Frame)

Todas as tramas usam o formato que é apresentado na figura seguinte, que indica também o número de bytes de cada campo.

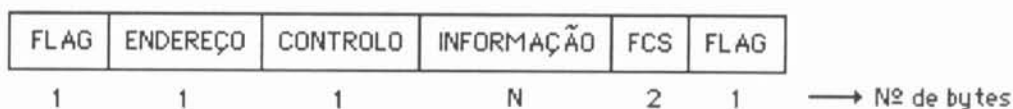


Figura A.8: formato de uma trama.

A.3.1.1 - Campo de limitação de trama (Flag Fields)

O primeiro e último campo de uma trama, designado por FLAG, delimita a trama. Esta sequência ocorre apenas no início ou no fim

de cada trama. O uso do bit zero a inserir ou a apagar garante a unicidade da sequência 01111110 na trama.

A.3.1.2 - Campo de endereço (Address Field)

Contém o endereço do escravo envolvido na transferência. Os valores de endereço zero e 251 a 255 são reservados pela Intel. Este é um campo obrigatório.

A.3.1.3 - Campo de controlo (Control Field)

Tem apenas um byte, e define a operação a realizar pela trama. É usado como comando, e para efectuar mudança de estado entre mestre e escravo. Comporta três tipos de operação:

- sincronização (tramas não numeradas);
- supervisão;
- transferência de mensagem.

A.3.1.3.1 - Tramas não numeradas (Unnumbered Frames)

São tramas usadas para sincronizar escravos com o mestre cujo formato apresentamos na figura seguinte.

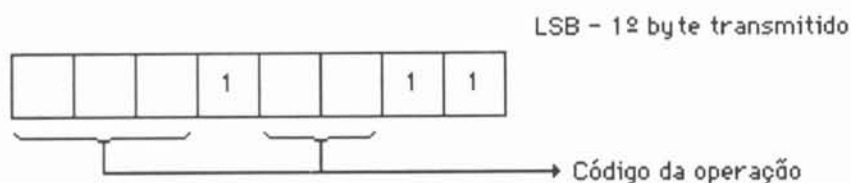


Figura A.9: Campo de uma trama não numerada.

As operações permitidas neste tipo de trama são:

Comandos:

DISC - desliga (Disconnect) - 53H;

SNRM - coloca no modo de resposta (Set Normal Response Mode) - 93H.

Respostas:

FRMR - Rejeição de trama (Frame reject) - 97H;

UA - Reconhecimento de trama não numerada (Unnumbered acknowledge) - 73H.

O comando SNRM é enviado pelo mestre para sincronizar um escravo. Se um escravo está em NDM, este comando obriga o escravo a passar para NRM. Se o escravo está em NRM, este comando é inválido.

O escravo envia para o mestre FRMR quando detecta um campo de controlo inválido. Depois de receber este comando, o mestre deve iniciar a resincronização com o escravo.

UA é a resposta usada pelo escravo a confirmar a recepção de uma trama não numerada válida (DISC ou SNRM).

O comando DISC é enviado pelo mestre a um escravo, para iniciar a resincronização. Este comando coloca ou mantém o escravo no modo NDM. É usado pelo mestre quando detecta a necessidade de resincronizar, ou em resposta a uma FRMR de um escravo. Quando o mestre recebe UA em resposta a DISC fica a conhecer o estado do escravo endereçado, isto é, que está em NDM.

A.3.1.3.2. - Tramas de supervisão (Supervisory Operation)

Neste tipo de trama o escravo terá que estar sempre em NRM. O BITBUS suporta duas operações de supervisão:

RR - receptor preparado (receiver ready);

RNR - receptor não preparado (receiver not ready).

O formato do campo de controlo é o que se apresenta na figura seguinte.

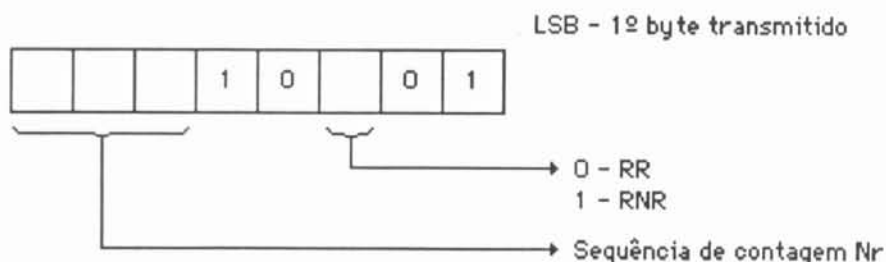


Figura A.10: Campo de uma trama de supervisão.

O mestre usa a trama de supervisão RR para pedir a um escravo (poll) uma trama de informação. Para pedir ao escravo o seu estado usa RNR, daí que o escravo não responde a RNR com uma trama de informação.

O escravo usa RR para confirmar uma recepção válida da anterior trama de informação ou de supervisão, quando a mensagem não está disponível. Usa RNR para indicar que a última trama foi reconhecida, contudo não tem "buffer" disponível para a armazenar. Neste caso o mestre deverá retransmitir a trama.

Ambas as tramas, RR e RNR, usam a sequência de contagem Nr. O receptor de uma trama de supervisão deve comparar este valor com o seu Ns para verificar se está correcta.

A.3.1.3.3 - Tramas de informação (Information Frames)

A transferência de mensagens no BITBUS processa-se nas tramas de informação. O campo de controlo é o que se apresenta na figura A.11.

A.3.1.3.3.1 - Campo de informação (Information Field)

Este campo transporta a mensagem na rede BITBUS. É usado apenas em tramas de informação.

DESCRIÇÃO DA ARQUITECTURA BITBUS

As tramas de informação têm no seu campo de informação uma "estrutura de mensagem" (protocolo) como a que a figura seguinte ilustra, e que passamos a descrever.

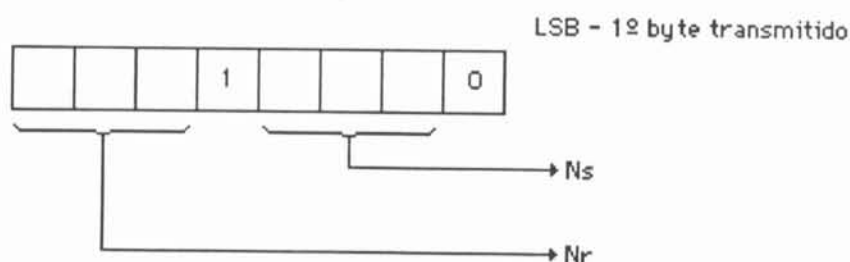


Figura A.11: Campo de uma trama de informação.

Comprimento (length) - Especifica o número total de bytes da mensagem. O comprimento da mensagem é igual ao nº de bytes de informação mais sete (dois bytes reservados pela Intel, acrescidos dos cinco bytes dos restantes campos). Deste modo este campo de oito bits pode conter valores entre sete e 255. Todas as implementações do BITBUS, mesmo as mais elementares, suportam um campo de dados até 13 bytes, o que implica 20 bytes de comprimento no campo de informação.

MT (tipo de mensagem) - Este campo é usado para especificar se a mensagem é uma ordem ou uma resposta. O mestre envia sempre ordens colocando este bit num nível lógico 0. O escravo envia sempre respostas colocando este bit num nível lógico 1.

SE (extensão fonte) - Este campo indica se a fonte da ordem, ou o destino da resposta, é o mestre ou a extensão a ele associada. Quando com um nível lógico 1 indica que se trata da extensão, e o nível lógico 0 que se trata do mestre. Este bit não muda entre ordem e correspondente resposta.

DE (extensão destino) - Indica se o destino da ordem ou a origem da resposta, é o escravo ou se é a extensão a ele associada. Quando com um nível lógico 1 indica que se trata da extensão, e o nível lógico 0 que se trata de um escravo. Do mesmo modo do que

no campo anterior este bit não muda entre ordem e correspondente resposta.



Figura A.12: Formato da mensagem (campo de informação).

TR (pista) - Este campo é usado para efectuar o controlo de mensagem no mestre ou no escravo em certas implementações.

Campo de reserva - A Intel reserva os quatro bits deste campo para futuras alterações. Devem ser colocados a zero aquando do envio de alguma mensagem.

Endereço do nó - Endereço do nó de destino da mensagem.

Tarefa fonte (source task) - Identifica a tarefa gerada pelo mestre ou sua extensão.

Tarefa destino (destination task) - Identifica a tarefa do receptor de ordens, escravo ou extensão.

Comando / resposta - Este campo é usado pelas tarefas e pelo protocolo de mensagem. Em condições normais este campo é usado

únicamente pelas tarefas. O protocolo de mensagem usa este campo para indicar os erros.

Dados - Transporta a mensagem. A implementação mais elementar deverá poder manusear até um máximo de 13 bytes de dados, que poderão ser estendidos até um máximo de 248 bytes.

A.3.1.4 - Verificação de sequência da trama (FCS)

Este campo (Frame Check Sequence - FCS), providencia o mais baixo nível de detecção de erros no BITBUS. O transmissor gera e envia este campo, utilizando a especificação "16 bit cyclic redundancy check (CRC)" - "standard CRC - CCITT polinomial $X^{16} + X^{12} + X^5 + 1$ " [15]. Se o receptor ao verificar o FCS detecta um CRC incorrecto ignora a trama. É um campo obrigatório.

A.4 - Acesso e controlo remotos - RAC

O RAC (REMOTE ACCESS AND CONTROL), define um conjunto de comandos e respostas de alto nível, para implementar funções/serviços com os nós escravos.

Vimos já no ponto anterior qual o formato de cada mensagem, ao qual obedece também uma mensagem do tipo RAC. Vejamos, com mais detalhe, alguns dos campos de mensagem incluídos no RAC.

- Tarefa destino (destination task) - define qual a tarefa destino que, no caso do RAC, é a tarefa 0. Se o nó escravo não suporta as funções/serviços do tipo RAC (task 0), a tarefa 0 pode ser usada para outra função.

- Comando resposta (comand/response) - este campo é usado para definir qual o tipo de ordem que está a ser transmitida. Os vários tipos de comandos e respostas suportados pelo RAC são especificados mais adiante.

- Dados (data) - é um campo determinante para a execução do comando RAC. Consoante o tipo de comando a ser executado o comprimento deste campo varia, podendo ir de zero bytes até um máximo especificado pela configuração da rede em causa.

A.4.1 - Comandos RAC

O quadro seguinte apresenta o conjunto de comandos disponíveis do tipo RAC.

COMANDO	ACESSO	CONTOLO	VALOR
reset escravo		✓	00H
cria tarefa		✓	01H
apaga tarefa		✓	02H
le FUNCTION ID		✓	03H
protege RAC		✓	04H
le I/O	✓		05H
escreve I/O	✓		06H
le bloco de I/O	✓		07H
le bloco de memória	✓		08H
escreve bloco de memória	✓		09H
operação lógica OR com I/O	✓		0AH
operação lógica AND com I/O	✓		0BH
operação lógica XOR com I/O	✓		0CH
le STATUS	✓		0DH
escreve STATUS	✓		0EH
reservado			0FH...0BFH
para uso do utilizador			0COH...0FFH

Quadro A.2: Comandos da RAC.

Existe uma área, de 0FH a 0BFH, reservada pela INTEL para uso próprio, e existe ainda uma outra área, de 0CH a 0FFH, destinada a comandos a serem definidos pelo utilizador.

Estas áreas não estão no entanto estandardizadas podendo, por conseguinte, variar.

Vejamos então com mais detalhe cada um dos comandos definidos:

"reset" slave - este comando provoca o "reset" de um escravo (ou extensão) e sua consequente inicialização. O campo de dados deste comando é nulo. Este é o único comando que não implica o envio de uma resposta após recepção válida do comando. O único caso em que temos uma mensagem de resposta é aquando da existência de um erro. Caso contrário, e visto o

escravo ter começado um processo de "reset", torna-se evidente a impossibilidade de termos uma mensagem de resposta.

Create task - este comando cria uma tarefa a ser executada no escravo. O campo de dados deste comando inclui um apontador de endereço, que não é mais do que o endereço de memória onde a tarefa deverá ser colocada, cujo comprimento é variável dependendo do hardware do escravo em causa. Após o atendimento deste comando, é enviada uma resposta ao mestre com 00H no campo de resposta, mas com o campo de dados inalterado, ou seja a resposta é dada e no campo de dados segue o mesmo endereço que antes tinha sido transmitido.

Delete task - este comando provoca a desactivação indefinida de uma tarefa a ser executada pelo escravo, e cujo número é incluído no comando. Neste caso, o campo de dados é unicamente de um byte, contendo o número da tarefa que se pretende desactivar. O comprimento total da mensagem é portanto de oito bytes. Após a execução desta tarefa, é dada resposta ao mestre com 00H no campo de resposta e com o campo de dados inalterado.

Get fuction ID'S - a este comando o escravo responde fornecendo ao mestre uma "lista" com o código identificativo das tarefas que estão a correr. O comando contém um espaço de dados vazio, que é utilizado pelo escravo para enviar os ID'S dessas tarefas.

RAC Protect - este comando permite colocar o escravo em dois modos distintos de funcionamento para respostas a comandos do tipo RAC: inibido (disabled) ou activo (enabled). Quando inibido, o escravo só reconhece as tarefas RAC se estas forem do tipo comando, não respondendo às tarefas do tipo acesso. Após um "reset", as funções do tipo RAC estão activadas. Este comando tem um comprimento total de oito bytes, sendo o comprimento do campo de dados de um só byte. Quando este byte for 00H as funções do tipo RAC são activadas; quando este byte for 01H as funções são inibidas. Qualquer outro

código deverá ser evitado, pois pode originar um resultado indeterminado. Após cumprida esta tarefa, a mensagem de retorno ao mestre tem o código 00H no campo de resposta, não sendo alterado o campo de dados.

Comando I/O - este conjunto de comandos permite o acesso até cerca de 250 portas I/O por cada escravo, ou extensão. Dentro deste conjunto de comandos distinguem-se três tipos essenciais: leitura, escrita e operações lógicas. O campo de dados é composto, neste caso, por um ou mais pares de bytes, dependendo do número de "acções" que se deseja implementar. Por cada par de bytes, o primeiro representa o endereço da porta com a qual pretendemos trabalhar, e o segundo byte contém a "máscara" com a qual se pretende realizar a operação. A figura seguinte dá-nos o formato do campo de informação da trama enviada quando deste comando.

Read I/O - o escravo responde a este comando lendo a(s) porta(s) especificada(s) no campo de dados. Como resposta o escravo envia o conteúdo da(s) porta(s) lida(s).

Write I/O - a este comando o escravo responde escrevendo na(s) porta(s) especificada(s) no campo de dados os valores que lhe são enviados.

Update I/O - este comando é também um comando de escrita, só que difere do comando anterior pelo facto de, após a escrita nas portas indicadas, o escravo ir ler essas portas e enviar o resultado da leitura ao mestre.

Operações lógicas - a este comando o escravo responde lendo a porta especificada, após o que realiza a operação indicada pelo comando, cuja máscara está no campo de dados. Depois de realizada a operação lógica, o escravo escreve o resultado na porta em questão, e em seguida relê a porta e envia o seu conteúdo ao mestre. As operações possíveis são "bit set", "bit reset" (cleared) e o "bit toggled". Consegue-se fazer o "bit set" realizando o OR I/O com a máscara indicada em data. Para o "reset (cleared)" o processo

DESCRIÇÃO DA ARQUITECTURA BITBUS

é o mesmo, só que a operação realizada é o AND I/O com zero. Finalmente, para o "toggled" realiza-se o XOR I/O com um nível lógico 1.

COMPRIMENTO				
MT	SE	DE	TR	RESERVADA(4 bits)
ENDEREÇO DO NO				
TAREFA FONTE			TAREFA DESTINO	
COMANDO/RESPOSTA				
ENDEREÇO DA PORTA 1				
BYTE DE DADOS 1				
ENDEREÇO DA PORTA 2				
BYTE DE DADOS 2				
ENDEREÇO DA PORTA 3				
BYTE DE DADOS 3				
ENDEREÇO DA PORTA 4				
BYTE DE DADOS 4				
ENDEREÇO DA PORTA 5				
BYTE DE DADOS 5				
ENDEREÇO DA PORTA 6				
BYTE DE DADOS 6				

Figura A.13: Formato do campo de informação dos comandos de I/O.

Upload Memory - Este comando causa a leitura de uma zona de memória do escravo (ou extensão), sendo o endereço da primeira posição que se pretende ler transmitido

nos dois primeiros bytes do campo de dados e, dependendo o número de bytes que se pretende ler, do comprimento do campo de dados sendo:

N° de bytes a ler = comprimento total do campo de dados - 2 bytes

em que as consecutivas posições de memória são determinadas por incrementos da primeira posição. A figura A.13 mostra o formato do comando.

Download Memory - trata-se de um comando de escrita em posições de memória do escravo, de características iguais ao anterior, sendo os dados que se pretendem escrever especificados no campo de dados por todos os bytes denominados na figura A.13 por byte de dados.

Status Command - estes comandos permitem ler ou escrever numa zona específica de 256 bytes de memória do escravo (ou extensão). Estas instruções permitem criar um banco de dados entre o mestre e o escravo. O formato da trama a ser enviada pelo mestre é ilustrada na figura A.14.

Read Status - este comando provoca a leitura de uma zona específica da memória cujo endereço a ler está colocado no campo de dados, sendo o byte imediatamente a seguir usado para transferir para o mestre a posição que se pretende ler.

Write Status - este comando realiza a mesma tarefa do que o anterior, só que permite escrever nas posições de memória em causa.

A.4.2 - Respostas RAC

Em geral o escravo responde a um comando do tipo RAC com uma mensagem ao mestre. Os únicos casos em que isso não acontece são quando o comando enviado for "Reset slave", ou quando ocorre uma condição de erro. Assim, aquando da resposta ao mestre, o RAC contém um conjunto de respostas que assinalam certos tipos de erros quando estes ocorrem. O quadro seguinte apresenta o conjunto de respostas do tipo RAC.

COMPRIMENTO				
MT	SE	DE	TR	RESERVADA(4 bits)
ENDEREÇO DO NO				
TAREFA FONTE			TAREFA DESTINO	
COMANDO/RESPOSTA				
APONTADOR DE ENDEREÇOS(byte alto)				
APONTADOR DE ENDEREÇOS(byte baixo)				
BYTE DE DADOS 1				
BYTE DE DADOS 2				
BYTE DE DADOS 3				
BYTE DE DADOS 4				
BYTE DE DADOS 5				
BYTE DE DADOS 6				
BYTE DE DADOS 7				
BYTE DE DADOS 8				
BYTE DE DADOS 9				
BYTE DE DADOS 10				
BYTE DE DADOS 11				

Figura A.13: Formato do campo de informação para comandos de memória.

COMPRIMENTO				
MT	SE	DE	TR	RESERVADA(4 bits)
ENDEREÇO DO NO				
TAREFA FONTE			TAREFA DESTINO	
COMANDO/RESPOSTA				
ENDEREÇO 1				
BYTE DE DADOS				
ENDEREÇO 2				
BYTE DE DADOS 2				
ENDEREÇO 3				
BYTE DE DADOS 3				
ENDEREÇO 4				
BYTE DE DADOS 4				
ENDEREÇO 5				
BYTE DE DADOS 5				
ENDEREÇO 6				
BYTE DE DADOS 6				

Figura A.14: Formato do campo de informação para comandos STATUS.

RESPONSE	VALOR
Tarefa inexistente	80H
OVERFLOW da tarefa	81H
OVERFLOW do banco de registo	82H
duplicação de FUNCTION ID	83H
falta de buffers na memória	84H
protecção da RAC	95H
comando de RAC desconhecido	96H

Quadro A.3: Respostas da RAC.

- No Task - é a resposta enviada ao mestre quando a ordem transmitida por este é de desactivar uma tarefa que não existe no escravo.

- Task Overflow - o escravo envia ao mestre este código quando o mestre dá ordem de criar uma nova tarefa, mas ele não tem capacidade de suportar mais tarefas.

- Register Bank Overflow - resposta enviada quando o mestre manda criar uma nova tarefa, e esta não pode ser criada por falta de espaço no banco de registos do escravo.

- Duplicate Function ID - é enviada quando o mestre manda criar uma função (FUNCTION ID) que já tinha sido criada.

- No "buffers" - resposta quando a criação de uma nova tarefa não é possível por falta de recursos de memória.

- RAC Protect - é enviado quando, em resposta a um comando de acesso RAC, estas funções estão inibidas (disabled).

- Unknown RAC Comand - resposta gerada quando o comando RAC recebido pelo escravo, não é reconhecido como fazendo parte do conjunto de comandos possíveis.

APENDICE B

Descrição do IDCX51

APENDICE B - DESCRIÇÃO DO IDCX51 [5] [14]

B.1 - Introdução

O IDCX51 DISTRIBUTED CONTROL EXECUTIVE é um núcleo que suporta várias tarefas ("multitasking") em tempo real. Este sistema corre nos microcontroladores da família 51 da INTEL nas versões Romless, Rom ou Eprom. O IDCX51 providencia todos os serviços necessários em sistemas de controlo múltiplos e em tempo real.

Com este executivo as funções de controlo são representadas como blocos de programa que se chamam tarefas (TASK's). Os sistemas baseados no IDCX51 suportam no máximo oito tarefas.

As componentes principais no IDCX51 podem ser representadas na figura B.1.

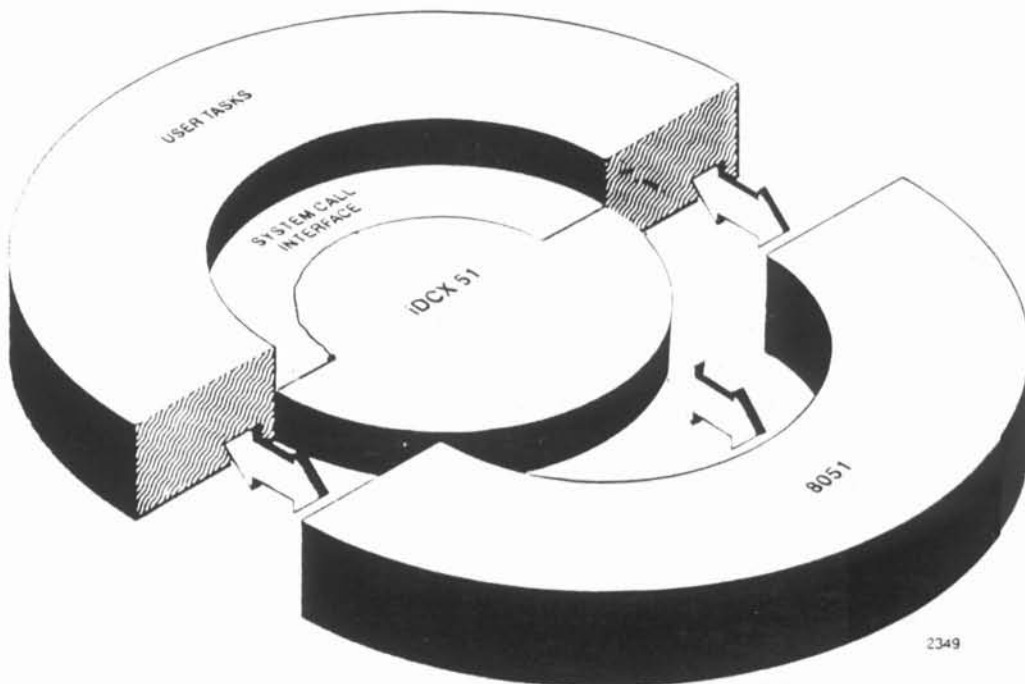


Figura B.1: Interligação de blocos no IDCX51

B.2 - Sistemas de tempo real

Um sistema de tempo real deve responder em tempo útil aos acontecimentos externos que ocorrem assincronamente. Para coordenar e executar as funções que respondem a estes acontecimentos, o IDCX51 suporta interrupções, comunicação entre tarefas, gestão de memória interna, serviços de temporização e um sistema de priorização das tarefas.

O IDCX51 sincroniza e cataloga as tarefas, e permite correr várias tarefas em tempo real. No entanto, só uma tarefa pode ser executada, i.e. ter a posse do CPU, em cada instante. A multiplexagem das várias tarefas é que permite gerar a pseudo-concorrência, e admitir que todas as tarefas estão a ser executadas simultaneamente.

B.3 - Características do IDCX51

As características do IDCX51 são:

- elevado desempenho;
- capacidade de suportar oito tarefas, a correr com quatro níveis de prioridade;
- dimensão reduzida: apenas ocupa 2.2 Kbytes;
- suporte de mensagens BITBUS;
- permite a transferência de mensagens entre tarefas situadas em nós locais ou remotos;
- compatível com o executivo IDCX86;
- configuração simples;
- gestão dinâmica da memória interna;
- suporte de interrupções;
- suporte de software em PLM 51.

B.4 - Serviços do IDCX51

Existem cinco categorias de serviços do IDCX51:

- gestão de tarefas;
- comunicação entre tarefas;
- gestão de memória interna;
- suporte de interrupções;
- serviços de temporização.

B.4.1 - Gestão de tarefas

Uma das funções mais importantes do IDCX51 é supervisionar a execução das tarefas que constituem o suporte de uma determinada aplicação. Os serviços do IDCX51 que permitem a gestão das tarefas são as seguintes:

- Criar e ordenar as tarefas do utilizador, estaticamente, na inicialização, ou dinamicamente, usando a chamada ao sistema RQ\$CREATE\$TASK;
- Desactivar uma tarefa, usando a chamada ao sistema RQ\$DELETE\$TASK;
- Obter a identificação de função (function ID) da tarefa em curso, usando a chamada ao sistema RQ\$GET\$FUNCTION\$ID;
- Suspende a execução de uma tarefa, usando a chamada ao sistema RQ\$WAIT.

B.4.2 - Comunicação entre tarefas

O IDCX51 possui serviços que permitem a comunicação entre tarefas, de modo a permitir que estas troquem informação através de mensagens. Usando estes serviços, as tarefas podem sincronizar as suas funções e partilhar o sistema com outras tarefas.

Os serviços que permitem a comunicação entre tarefas são os seguintes:

- Alocar "buffers" nas zonas de memória em que as tarefas colocam as mensagens antes de as enviar, usando a chamada ao sistema RQ\$ALLOCATE;
- Enviar mensagens para a tarefa destino, usando chamada ao sistema RQ\$SEND\$MESSAGE;
- Provocar a paragem de uma tarefa até receber uma mensagem, usando a chamada ao sistema RQ\$WAIT;
- Devolver um "buffer" alocado previamente ao SBP (system buffer pool) usando a chamada ao sistema RQ\$DEALLOCATE.

B.4.3 - Gestão da memória interna

O IDCX51 faz a gestão da memória RAM interna, o que permite

otimizar a sua utilização. Os serviços que permitem fazer a gestão de memória interna são:

- Obter o endereço de um bloco de memória interna disponível, com um comprimento especificado, usando a chamada ao sistema RQ\$GET\$MEM;
- Devolver um bloco de memória com um comprimento especificado ao SYSTEM MEMORY POOL, usando a chamada ao sistema RQ\$RELEASE\$MEM;
- Disponibilizar a gestão de memória na inicialização e em funcionamento.

B.4.3.1 - SMP - System Memory Pool

O IDCX51 faz a gestão da memória interna através da existência da SMP. A SMP é uma lista de blocos de memória de tamanho variável e comprimento par, constituída a partir da memória interna depois de terem sido satisfeitas as necessidades do microcontrolador. O tamanho da SMP é definido pelo utilizador durante a configuração do sistema.

B.4.3.2 - Gestão da memória na inicialização

Durante a inicialização do IDCX51, a SMP é construída e modificada. Primeiro, o executivo cria a SMP e depois, à medida que são criadas as tarefas na inicialização, o sistema adquire áreas da memória para estas tarefas a partir do topo da SMP. Se é encontrado um IDD (Initial Data Descriptor) na cadeia dos ITD (Initial Task Descriptor) e o utilizador reserva um bloco de memória, este espaço é alocado a partir do início da SMP. Finalmente é criado o SBP (System Buffer Pool).

O SBP é criado quando o executivo divide a SMP no maior número de "buffers", com comprimento especificado pela constante do sistema RQ\$SYSBUFSIZE, que for possível. A figura B.2 ilustra este processo.

B.4.3.3 - Gestão dinâmica de memória

Depois da inicialização, o sistema implementa a gestão da memória interna, usando as chamadas ao sistema RQ\$GET\$MEM e RQ\$RELEASE\$MEM. Como à medida que são adicionadas novas tarefas os recursos da memória ficam mais escassos, o IDCX51,

para solucionar isto, devolve automaticamente o SBP para o SMP através de sucessivas chamadas ao sistema RQ\$RELEASE\$MEM.

Em seguida, o executivo tenta adquirir espaço de memória através de RQ\$GET\$MEM. Uma vez obtida, a SBP é reconstruída a partir dessa memória disponível.

Quando uma tarefa é desactivada, o espaço por ela ocupado é devolvido à SMP. Este processo pode ser observado na figura B.3.

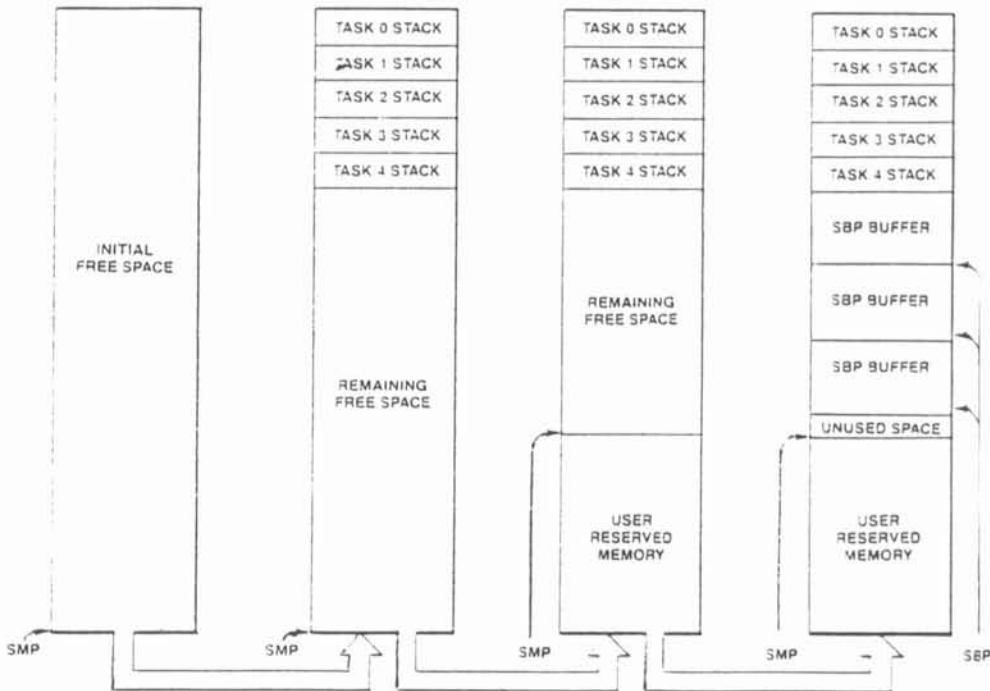


Figura B.2: Gestão da memória interna durante a inicialização.

B.4.4 - Suporte de interrupção

O IDCX51 controla as interrupções mascarando-as de acordo com a prioridade das tarefas. Os serviços que permitem suportar as interrupções são:

- Atribuir uma interrupção a uma determinada tarefa durante a configuração. Isto é especificado no ITD (Initial Task Descriptor);
- Desactivar interrupções através de chamadas ao sistema RQ\$DISABLE\$INTERRUPT;

- Reactivar interrupções previamente desactivadas através de chamadas ao sistema RQ\$ENABLE\$INTERRUPT;
- Sincronizar tarefas com interrupções através de chamadas ao sistema RQ\$WAIT.

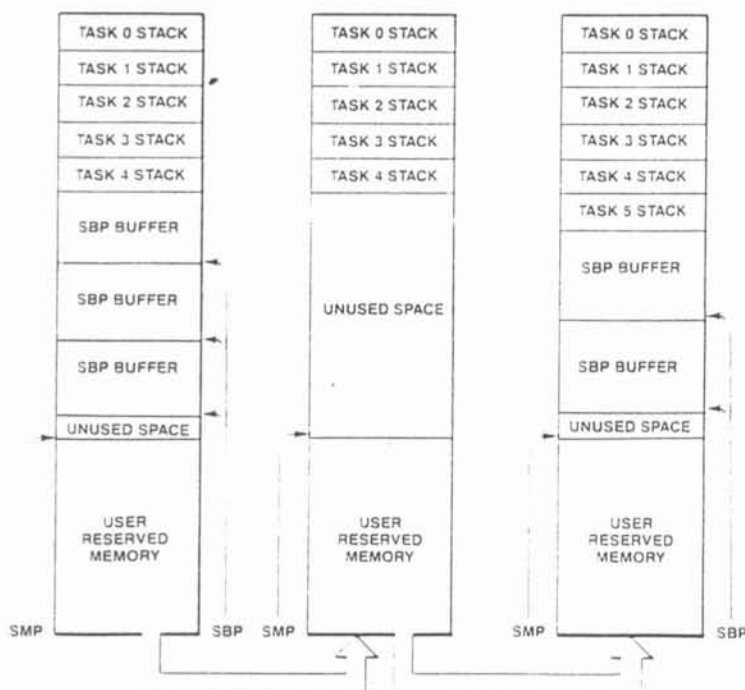


Figura B.3: Gestão da memória interna durante a acção de desactivação de uma tarefa.

B.4.4.1 - Modelo de interrupções do IDCX51

Qualquer das cinco fontes de interrupções podem interromper o processador quando o INTERRUPT ENABLE BIT apropriado no registo IE (Interrupt enable) e o GLOBAL ENABLE estão, ambos activados com valor lógico "1".

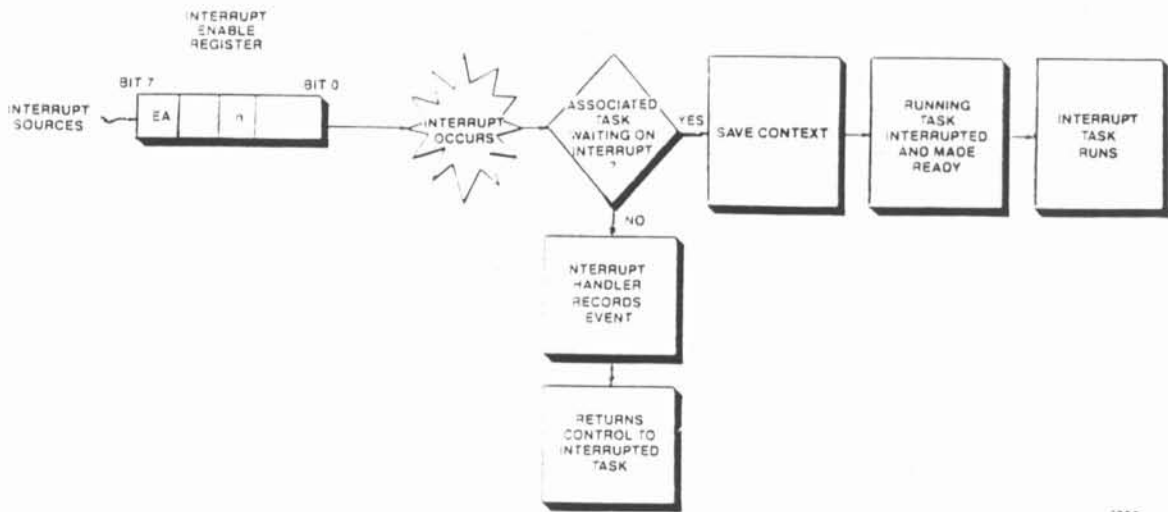
Quando o processador é interrompido, o controlo é transferido para o gestor de interrupções do IDCX51 que faz o seguinte:

- Se a tarefa associada à interrupção está à espera de uma interrupção, então o contexto da tarefa em curso é guardado e a tarefa que estava a correr é interrompida e posta no

estado READY. A tarefa que estava à espera da interrupção passa para o estado RUNNING;

- Se a tarefa associada com a interrupção não está à espera de nenhuma interrupção, então a ocorrência de uma interrupção é registada e a tarefa que foi interrompida volta ao estado original.

Isto pode ser observado na figura B.4.



2356

Figura B.4: Modelo de interrupções para o IDCX51.

B.4.4.2 - Associação de interrupções a tarefas

As interrupções são associadas às tarefas durante a configuração do sistema.

Quando se definem os ITD's para as tarefas, são atribuídas a cada uma as suas interrupções específicas.

B.4.4.3 - Máscara de interrupções

Para permitir ou inibir uma ou mais interrupções, de modo a atender às necessidades do sistema, utiliza-se uma máscara de interrupções.

O IDCX51 tem dois tipos de máscara de interrupções:

- SYSTEM INITIALIZATION MASKING (permanente);
- USER MASKING (temporário).

B.4.5 - Serviços de temporização

O IDCX51 usa o TIMER 0 do microcontrolador para criar o sinal de relógio para os TIMEOUT's e intervalos de tempo. Os intervalos de tempo permitem que as tarefas possam executar funções baseadas em intervalos de tempo bem definidos. Os TIMEOUT's permitem a suspensão de uma tarefa durante um período de tempo, ou limitar o tempo de espera da tarefa por um acontecimento. Os serviços que permitem realizar estas funções são:

- Atribuir um intervalo de tempo e um sinal de chamada quando o evento ocorre nesse intervalo: RQ\$SET\$INTERVAL;
- Esperar por um acontecimento ou por um TIMEOUT durante um intervalo de tempo: RQ\$WAIT.

B.5 - Constituição do sistema IDCX51

Um sistema IDCX51 é constituído pelo IDCX51 EXECUTIVE e por um conjunto de tarefas especificadas pelo utilizador. A criação de um sistema envolve a definição das constantes do sistema, das tarefas do utilizador e a linkagem destas com o executivo.

B.5.1 - Inicialização do sistema

A sequência de inicialização do IDCX51 pode ser vista na figura B.5.

Depois do reset, o executivo inicializa as suas estruturas e tabelas de dados, formando a SMP (system memory pool). Em seguida, é processada a cadeia de ITDs (initial task descriptor). No processamento desta cadeia, os atributos de cada tarefa são colocados nas tabelas internas do executivo e cada uma das tarefas é colocada na lista de tarefas READY por ordem de prioridade. O espaço de memória para cada tarefa é alocado a partir do topo da SMP.

Terminada a inicialização das tarefas, o executivo coloca o STATUS na variável RQ\$INIT\$STATUS e procura um IDD (initial data descriptor) válido no local especificado pela constante RQ\$RAM\$ID.

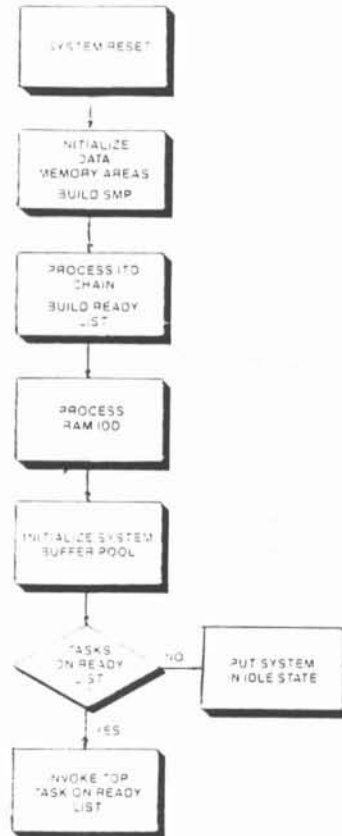


Figura B.5: Processo de inicialização.

Se é encontrado um IDD, o sistema é reconfigurado de acordo com o conteúdo deste. Depois, a tabela de máscaras de interrupções do executivo é actualizada. Em seguida, a SBP (System Buffer Pool), onde os "buffers" das mensagens são colocados, é inicializada. Finalmente, a primeira tarefa da lista de tarefas READY é invocada. Se não existirem tarefas na lista, o sistema fica parado (Idle State), o que indica a existência de erro na inicialização.

Após a inicialização, os registos do microcontrolador ficam com os valores apresentados na figura B.6.

Registro	Alterado?	Comentários
PC	YES	Inicializado pela primeira TASK
ACC	YES	Sem significado
B	YES	Sem significado
PSW	YES	Inicializado pela primeira TASK
SP	YES	Inicializado pela primeira TASK
DPTR	YES	Sem significado
P0	NO	
P1	NO	
P2	NO	
P3	NO	
IP	NO	
IE	YES	Mascara de Interrupção
TMOD	NO	Bit0 e Bit4 usados pelo System Clock
TCON	NO	
T2CON	NO	Registro do 8052
TH0	NO	Usado pelo System Clock
TLO	NO	Usado pelo System Clock
TH1	NO	
TL1	NO	
TH2	NO	Registro do 8052
TL2	NO	Registro do 8052
RCAP2H	NO	Registro do 8052
RCAP2L	NO	Registro do 8052
SCON	NO	
SBUF	NO	
PCON	NO	

Figura B.6: Estado dos registros do μ C8044 após a inicialização.

B.5.2 - Configuração do IDCX51

B.5.2.1 - O ITD - Initial Task Descriptor

O ITD é uma estrutura de dados do IDCX51 que especifica os atributos de cada tarefa do sistema. O ITD é definido pelo utilizador no momento da configuração.

Durante a inicialização, o executivo cria uma cadeia de ITD's no sistema. O executivo usa esta cadeia na inicialização das tarefas durante o arranque do sistema. A estrutura do ITD é ilustrada na figura B.7.

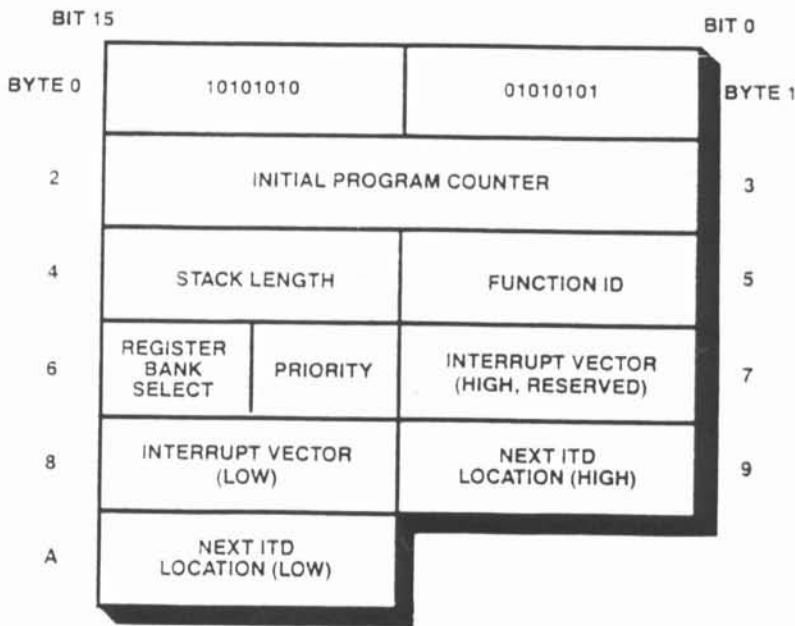


Figura B.7: Estrutura do ITD (Initial Task Descriptor)

Os campos do ITD são os seguintes:

Pattern: 1010101001010101 (B) = AA55 H

Valor que identifica um bloco como um ITD válido.

Initial Program Counter: Endereço do início do código da tarefa, relativamente ao início do ITD. Calcula-se fazendo ITD-START.

Stack Length: Número de bytes requeridos pela STACK da tarefa. Pode ter no mínimo dois bytes.

Funtion ID: Valor atribuído á tarefa que reflecte a função que está a executar. O valor para as tarefas do utilizador pode variar entre 80H e FEH.

Register Bank Select: Associa um dos quatro bancos de registos do 8051 á tarefa. Os valores possíveis são:

Bit 7 - Sempre com valor lógico zero;

Bit 6 - Se tem valor lógico zero, o IDCX51 escolhe o banco de registos e os bits 4 e 5 são ignorados. Se

tem valor lógico um, os bits 4 e 5 especificam qual dos quatro bancos de registos (0-3) está associado à tarefa.

Priority: Especifica a prioridade da tarefa. Este valor pode ir de 1H a 4H.

Interrupt Vector: Vector de oito bits que especifica quais as interrupções associadas à tarefa. O significado dos bits pode ser visto na figura B.8.

Next ITD: Endereço do próximo ITD. Calcula-se fazendo ITD-Next ITD.

Bits	Interrupt Source
7	reserved
6	reserved
5	Timer 2 (8052 only)
4	Internal Serial Port 1
3	Timer 1
2	External Request 1
1	reserved (Timer 0 used by system)
0	External Request 0

Figura B.8: Codificação das interrupções relativas a uma dada tarefa.

B.5.2.2 - O IDD - Initial Data Descriptor

O IDD é uma estrutura de dados do IDCX51 que permite a redefinição de algumas constantes e variáveis do sistema. São elas:

System Clock Unit (RQ\$CLOCK\$TICK);

System Clock Priority (RQ\$CLOCK\$PRIORITY);
 System Fixed Length Buffer Size (RQ\$SYS\$BUF\$SIZE);
 User Reserved Memory in System Memory Pool (SMP).

Os IDD são linkados na cadeia dos ITD durante a inicialização. Um outro IDD pode ser definido e a sua posição é definida por RQ\$RAM\$ID. Este é processado quando a cadeia ITD/IDD termina. Subsequentes IDD's, quando processados, redifinem prévios IDD's. O formato do IDD pode ser visto na figura B.9.

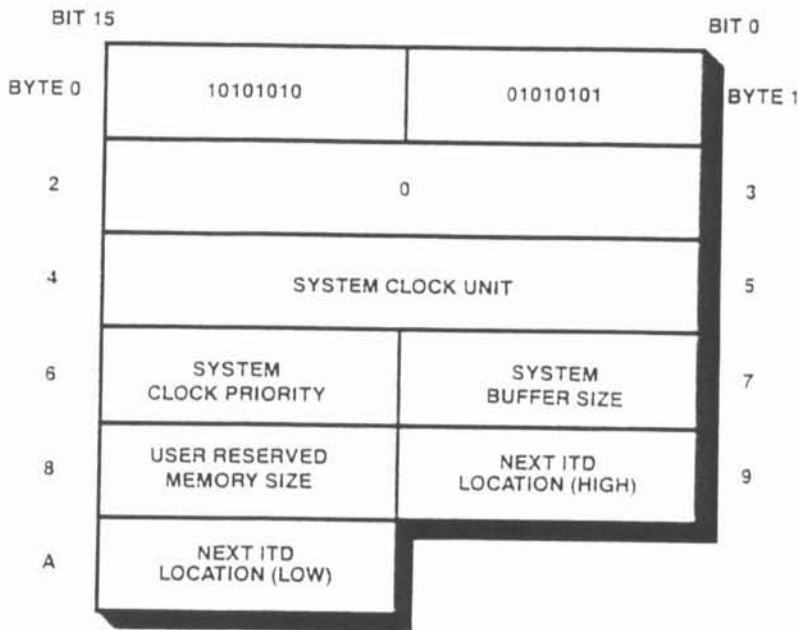


Figura B.9: Estrutura do IDD (Initial Data Descriptor).

Os campos do IDD são os seguintes:

Pattern: 1010101001010101 = AA55 H
 Identifica um bloco como um IDD válido;

0: Identifica este bloco como sendo um IDD;

System Clock Unit: Novo valor do valor do relógio do sistema;

System Clock Priority: Nova prioridade do relógio do sistema;

System Buffer Size: Novo valor do comprimento dos "buffers" que estão na SBP;

User Memory Size: Comprimento da memória reservada pelo utilizador a partir da SMP.

Next ITD/IDD: Localização do próximo ITD ou IDD.

B.6 - Memória do sistema

A memória do sistema (interna e externa, de dados e de programa) é dividida pelo IDCX51 de acordo com os mapas representados nas figuras B.10, B.11 e B.12.



Figura B.10: Memória de Dados externa.



Figura B.11: Memória de Programa.

B.7 - Definição da tarefa

Uma tarefa é um bloco de programa que executa uma função específica. Começa num único ponto e é executada em ciclo contínuo. Todas as tarefas correm independentemente, o que permite a partilha de programa entre tarefas e a resposta assíncrona a acontecimentos externos.

Uma tarefa possui as seguintes elementos:

- Area de memória;
- Registos;
- "Program Counter".

Os seus atributos são:

- Um ITD (initial task descriptor);
- Uma prioridade;
- Uma identificação (TID - task identification) e uma função (FID - function identification);

- Um sinal de relógio para intervalos de tempo e TIMEOUT's;
- Um conjunto de interrupções a ela associados;
- Um estado: ASLEEP, READY ou RUNNING.

B.7.1 - Estado das tarefas

As tarefas num sistema IDCX51 podem estar em um de três estados:

- ASLEEP
- READY
- RUNNING

Estado ASLEEP: uma tarefa está no estado ASLEEP quando está à espera de um acontecimento, depois de ter executado um RQ\$WAIT SYSTEM CALL.

Estado READY: uma tarefa está no estado READY quando está pronta a correr. Quando uma tarefa está no estado RUNNING, só pode passar ao estado ASLEEP ou ser desactivada. O executivo escolhe então a tarefa que está no topo da lista de tarefas READY para passar ao estado RUNNING.

Estado RUNNING: é o estado em que se encontra a tarefa que está nesse momento a ser executada pelo processador.

Estado IDLE: se não há tarefas do utilizador no estado RUNNING ou READY, o executivo coloca o sistema no estado IDLE.

A figura B.13 ilustra as transições entre os estados de diferentes tarefas que a seguir se descrevem.

1. Ocorrendo uma chamada a RQ\$WAIT por um acontecimento, a tarefa que está no estado RUNNING passa ao estado ASLEEP.
2. Quando ocorre um evento e uma tarefa associada, que está no estado ASLEEP, tem prioridade maior que a da tarefa que está a correr. A tarefa que está no estado ASLEEP passa ao estado RUNNING e a tarefa que estava a correr passa ao estado READY.
3. Quando ocorre um evento e uma tarefa associada que está no estado ASLEEP tem prioridade menor ou igual á prioridade da tarefa que está no estado RUNNING, a tarefa que está no estado ASLEEP passa ao estado READY.

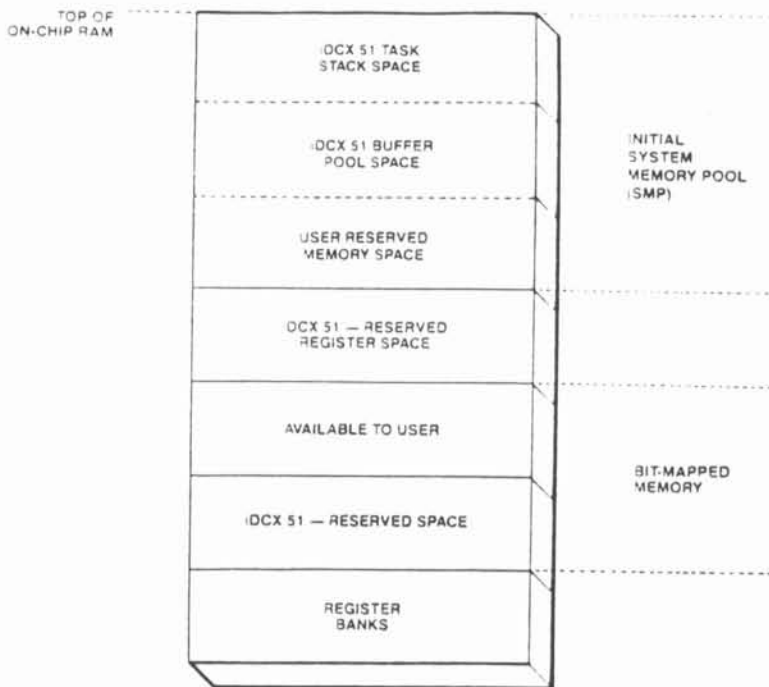


Figura B.12: Memória de Dados interna.

4. Quando a tarefa que está no estado RUNNING é desactivada ou passa a esperar por um acontecimento, a tarefa que estava no estado READY passa ao estado RUNNING.
5. Quando uma tarefa com prioridade maior do que a da tarefa que está a correr fica no estado READY, essa tarefa é posta a correr e a tarefa que estava a correr fica no estado READY.
6. Não há tarefas a correr ou no estado READY.

B.8 - FUNCTION ID's

FUNCTION ID é um byte que identifica a tarefa independentemente da sua TASK ID. A FUNCTION ID é especificada no ITD.

A diferença entre FUNCTION ID e TASK ID Function ID é a primeira identificar as tarefas de um modo absoluto, enquanto que a segunda as identifica de um modo relativo.

A TASK ID é atribuída pelo executivo e pode tomar valores de 0 a 7. A FUNCTION ID é atribuída pelo utilizador. Os seus valores são dados pela figura B.14.

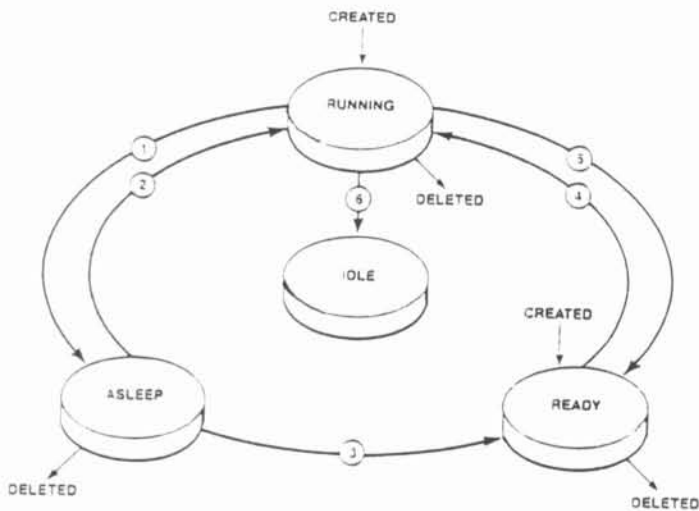


Figura B.13: Estados possíveis para uma tarefa.

Value(s)	Meaning
00H	No task (error indicator)
01H – 7FH	Reserved
80H – FEH	User assignable
FFH	Task with no function ID assigned

Figura B.14: Valores que a Task ID pode tomar.

B.9 - Definição da mensagem

Existem dois tipos de mensagens:

- ORDER: mensagens que instruem a tarefa de recepção para executar uma acção;
- REPLY: mensagens que respondem a mensagens de ordem previamente recebidas.

O formato de uma mensagem do IDCX51 pode ser vista na figura B.15.

B.10 - Chamadas ao sistema IDCX51

B.10.1 - Definição

Um SYSTEM CALL é um procedimento que permite aceder aos serviços do IDCX51.

Os SYSTEM CALL do IDCX51 permitem o acesso aos seguintes serviços:

- Iniciar uma tarefa dinamicamente;
- Terminar uma tarefa dinamicamente;
- Enviar mensagens para outras tarefas;
- Alocar "buffers" de mensagens;
- Desalocar "buffers" de mensagens;
- Esperar pela chegada de uma mensagem;
- Gerir a SMP (System memory pool) interna;
- Activar interrupções;
- Desactivar interrupções;
- Esperar pela ocorrência de uma interrupção;
- Estabelecer um intervalo de tempo;
- Esperar pelo fim do intervalo de tempo;
- Determinar a FUNCTION ID de todas as tarefas do sistema.

Os parâmetros são passados ao sistema, carregando previamente estes valores em registos específicos. O conteúdo original destes registos é destruído depois de executado procedimento.

A convenção para formar IDCX51 SYSTEM CALLS é carregar os parâmetros de entrada nos registos Acc, B e DPTR. Os parâmetros de saída, se forem bytes, vêm no registo Acc e se forem palavras vêm nos registos R6 e R7.

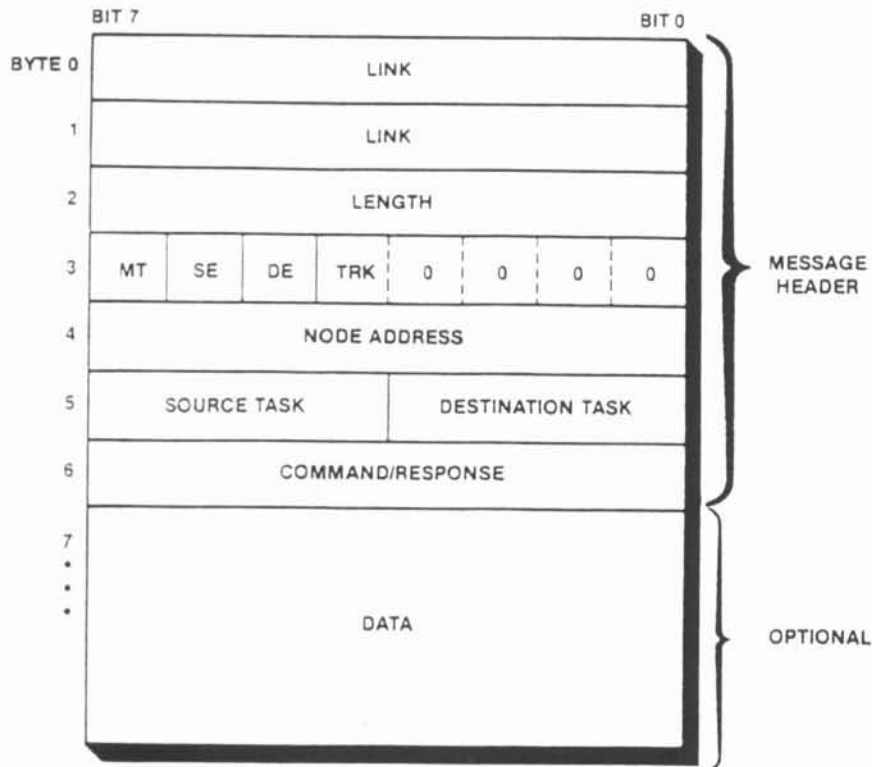


Figura B.15: Formato de uma mensagem em IDCX51/BITBUS.

B.10.2 - SYSTEM CALL DICTIONARY

O conjunto dos SYSTEM CALLS (chamadas ao sistema operativo) pode ser visto na figura B.16.

B.11 - Variáveis acessíveis ao utilizador

As variáveis do sistema são variáveis que o IDCX51 coloca acessíveis às tarefas do utilizador para obter informação sobre o sistema ou para o modificar. Existem dois tipos de variáveis:

- 1 - Variáveis Run-Time: reflectem as condições da parte de programa que está a correr nesse momento;
- 2 - Variáveis de diagnóstico de Development-Time: reflectem as condições da parte de programa que está a correr, e são

usadas durante o desenvolvimento de aplicações. Não são acessíveis directamente pelas aplicações.

O mapa com estas variáveis pode ser visto nas figuras B.17. e B.18. A lista destas variáveis pode ser vista na figura B.19.

Call Name	Description
RQ\$ALLOCATE	Obtain a message buffer from the system buffer pool.
RQ\$CREATE\$TASK	Create and schedule a new task.
RQ\$DEALLOCATE	Return a message buffer to the system buffer pool.
RQ\$DELETE\$TASK	Delete the specified task from the system.
RQ\$DISABLE\$INTERRUPT	Temporarily disable an interrupt.
RQ\$ENABLE\$INTERRUPT	Re-enable an interrupt previously disabled by RQ\$DISABLE\$INTERRUPT.
RQ\$GET\$FUNCTION\$IDS	Obtain the function IDs of tasks currently in the system.
RQ\$GET\$MEM	Get address of continuous block of System Memory Pool (SMP) memory of specified length.
RQ\$RELEASE\$MEM	Return block of memory of specified length back to the SMP
RQ\$SEND\$MESSAGE	Send a message to the specified task.
RQ\$SET\$INTERVAL	Establish a time interval.
RQ\$WAIT	Wait for an interrupt, message, interval, or timeout event.

Figura B.16: Lista de chamadas ao Sistema operativo.

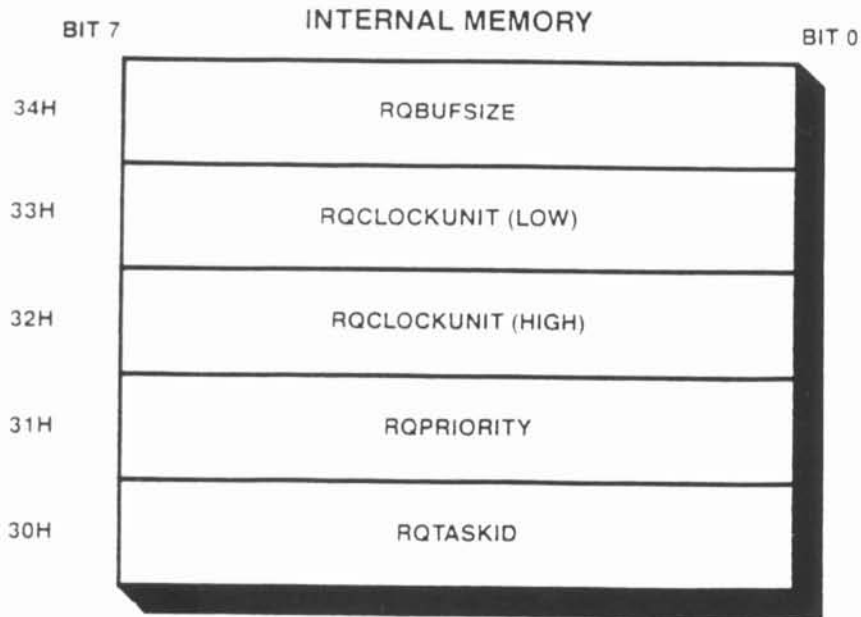


Figura B.17: Mapa da memória interna para as variáveis do Sistema Operativo.

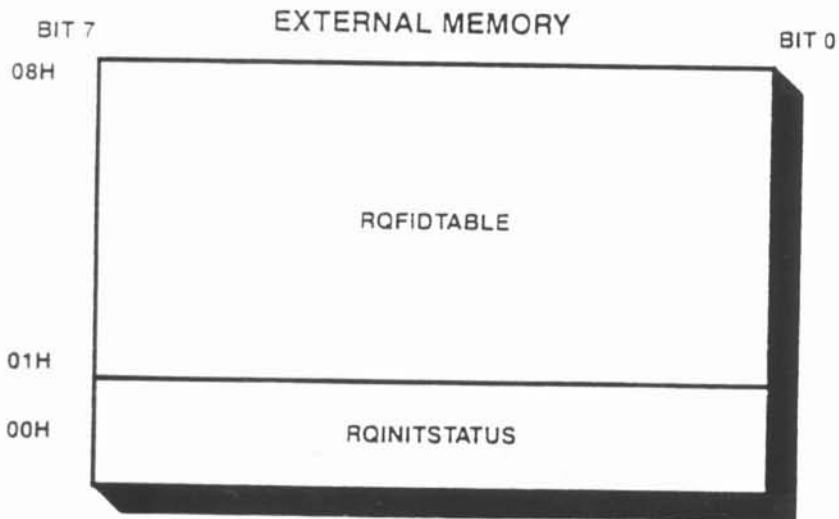


Figura B.18: Mapa da memória externa para as variáveis do Sistema Operativo.

Run—Time Variables			
Variable	Type	Address	Description
RQTASKID	BYTE	30H Internal	Contains the task ID of the running task.
RQCLOCKUNIT	WORD	32H Internal	Specifies the unit of time for the system clock.
RQBUFSIZE	BYTE	34H Internal	Indicates length of fixed length buffers in SBP

Development—Time Diagnostic Variables			
Variable	Type	Address	Description
RQTASKID	BYTE	30H Internal	Contains the task ID of the running task.
RQPRIORITY	BYTE	31H Internal	Contains the priority of the running task, or zero if the system is idle.
RQINITSTATUS	BYTE	00H External	Specifies the reason that initialization terminated.
RQFIDTABLE	BYTE*8	01H—08H External	Specifies function IDs for all tasks, indexed by task ID.

Figura B.19: Descrição de variáveis do Sistema Operativo.

APENDICE C

Descrição do microcontrolador 8044

APENDICE C - DESCRIÇÃO DO μ C8044 [5] [16]

C.1 - Introdução

O microcontrolador 8044 junta as características do microcontrolador de 8 bits 8051 com as de um controlador inteligente de comunicação série, o SIU, que implementa o HDLC. Ambos tem acesso a uma RAM de dupla entrada. O SIU trabalha concorrentemente com o CPU 51, dado que possui inteligência própria, libertando o CPU das várias funções de comunicação. O CPU pode pois ser usado para o controlo de periféricos ou para o controlo de processos externos, permitindo a um único integrado implementar quer processamento quer controlo distribuído.

C.2 - Características

Basicamente este microcontrolador pode ser caracterizado por:

- 4 Kbytes de memória de programa (para as versões 8044 e 8744);
- 256 bytes de memória de dados interna;
- 32 linhas de entrada/saída compatíveis com níveis TTL;
- 2 Timer/Event Counter de 16 bits cada;
- 5 fontes de interrupção:
 - SIU;
 - Timer/Counter's 0,1;
 - Interrupções 0,1.
- 2 níveis de prioridade de interrupção;
- 1 canal série em FULL DUPLEX;
- processador booleano;
- circuito interno de oscilador e relógio;
- compatibilidade com as famílias MCS.80 e MCS.85 para expansão de memória e I/O;
- tempo de execução por instrução reduzido a 1 μ s para 60% das instruções (o resto será da ordem dos 2 μ s, excepto a divisão e a multiplicação que consomem 4 μ s).

A estrutura interna do μ C8044 pode ser vista de uma forma esquemática na figura C.1.

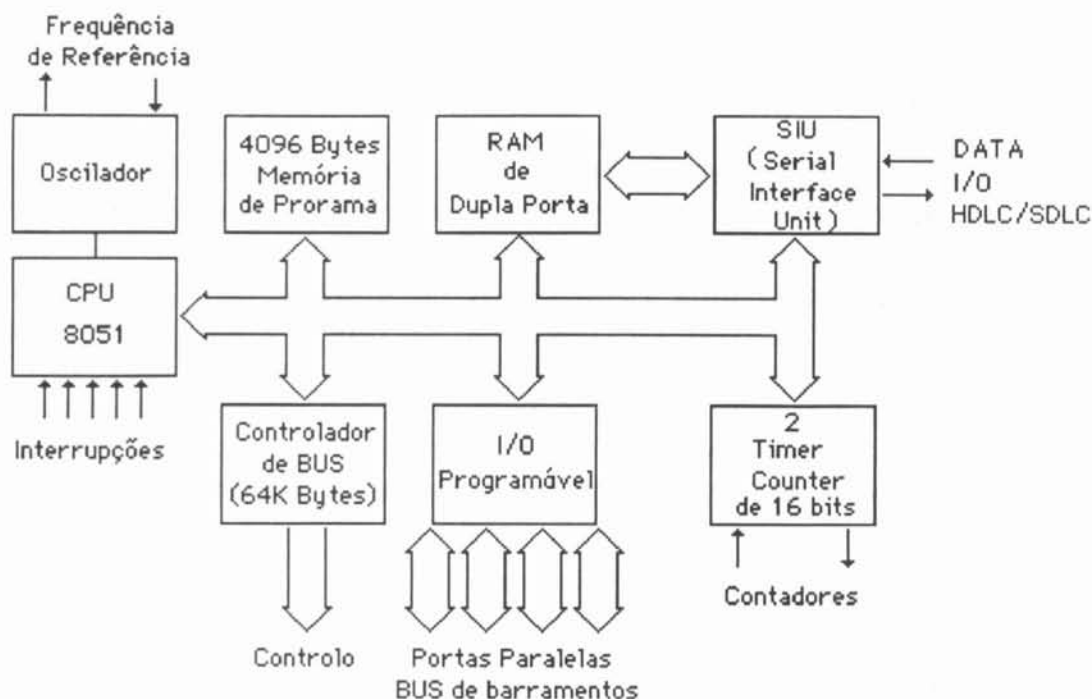


Figura C.1: Estrutura interna do μ C8044.

C.3 - Descrição do PINOUT

O microcontrolador 8044 por ser apresentado na forma de um integrado de 40 pinos, na apresentação DIP, ou com 44 pinos, na apresentação PLCC, e por ter inúmeros sinais de controlo e várias portas I/O (4 portas), obriga a que a maioria dos pinos sejam utilizados por vários sinais multiplexados conforma ilustra a figura C.2.

Segue-se uma breve descrição desta utilização do PINOUT com referência à figura C.2.

Port 0: porta bidireccional de 8 bits, multiplexada com as linhas de endereço mais baixas, multiplexadas por sua vez com as linhas de dados quando o μ C8044 acede à memória externa.

Port 1: porta de 8 bits quasi-bidireccional, que só apresenta sinais multiplexados para o /RTS (P1.6) e para o /CTS (P1.7) quando em modo de NON LOOP MODE TWO. Quando

	P1.0	1	40	VCC	
	P1.1	2	39	P0.0	AD0
	P1.2	3	38	P0.1	AD1
	P1.3	4	37	P0.2	AD2
	P1.4	5	36	P0.3	AD3
	P1.5	6	35	P0.4	AD4
	$\overline{RTS}$ P1.6	7	34	P0.5	AD5
	$\overline{CTS}$ P1.7	8	INTEL 33	P0.6	AD6
	RST VPD	9	32	P0.7	AD7
I/O	RxD P3.0	10	8044 31	EA	VPP
DATA	TxD P3.1	11	8344 30	ALE	$\overline{PROG}$
	INT0 P3.2	12	8744 29	$\overline{PSEN}$	
	INT1 P3.3	13	28	P2.7	A15
	TO P3.4	14	27	P2.6	A14
SCLK	T1 P3.5	15	26	P2.5	A13
	$\overline{WR}$ P3.6	16	25	P2.4	A12
	$\overline{RD}$ P3.7	17	24	P2.3	A11
	XTAL2	18	23	P2.2	A10
	XTAL1	19	22	P2.1	A9
	VSS	20	21	P2.0	A8

Figura C.2: PINOUT do $\mu C8044$

/RTS tem um valor lógico "0" indica que o $\mu C8044$ está pronto a transmitir. Quando /CTS tem um valor lógico "0" indica que o $\mu C8044$ está pronto a receber.

Port 2: porta de 8 bits, quasi-bidireccional, que está multiplexada com as linhas de endereços mais altas quando o $\mu C8044$ acede à memória externa.

Port 3: porta de 8 bits, quasi-bidireccional, que contém multiplexados os sinais de interrupção (/INT0 e /INT1), sinais dos TIMER's, sinais de escrita e leitura (/WR e /RD), e os sinais respeitantes à comunicação série (I/O-RxD, DATA-TxD e SCLK-T1). O primeiro sinal, controla o sentido da informação no pino P3.1 quando em configuração de POINT-TO-POINT ou em MULTI-POINT, e serve como recepção de DATA em configuração de anel ou em modo de diagnóstico. O segundo sinal, contém a informação que transita na linha e que pode ser de INPUT ou OUTPUT se o $\mu C8044$ estiver em configuração de POINT-TO-POINT ou

MULTI-POINT; caso contrário, na configuração em anel, este pino destina-se só à transmissão. O terceiro sinal, tanto pode ser uma entrada para o TIMER 1 como pode ser a entrada de um sinal de CLOCK série.

ALE/PROG: ADDRESS LATCH ENABLE é um sinal que é activado em cada 6 períodos de oscilador, excepto no acesso à memória externa. Este sinal permite o fixar dos endereços para a memória externa. O sinal /PROG é utilizado para a recepção por parte do μ C8044 do programa que será gravado na EPROM interna na versão μ C8744.

/PSEN: PROGRAM STORE ENABLE é um sinal que é activado em cada 6 períodos de oscilador, excepto quando no acesso à memória de dados externa. Este sinal permite o acesso à memória de programa externa pelo barramento de dados durante o FETCH das instruções. Podemos dizer que funciona como um sinal de leitura para a memória de programa externa.

/EA/VPP: quando tem um valor lógico "1", o μ C8044 entende como os primeiros 4K de memória de programa, os 4K que possui em ROM. Se tiver um valor lógico "0", o μ C8044 entende como os primeiros 4K de memória de programa os 4K que estão na memória de programa externa. Este pino também pode receber os 21V como tensão de programação da EPROM interna na versão μ C8744.

C.4 - Descrição funcional

C.4.1 - Memória de dados interna

O espaço de memória de dados é constituído por 256 bytes (0-255). As 32 primeiras posições são ocupadas por 4 bancos de 8 registos cada. Temos também 128 bits de endereçamento directo colocados nas posições 32 a 47. Ver figuras C.3 e C.4.

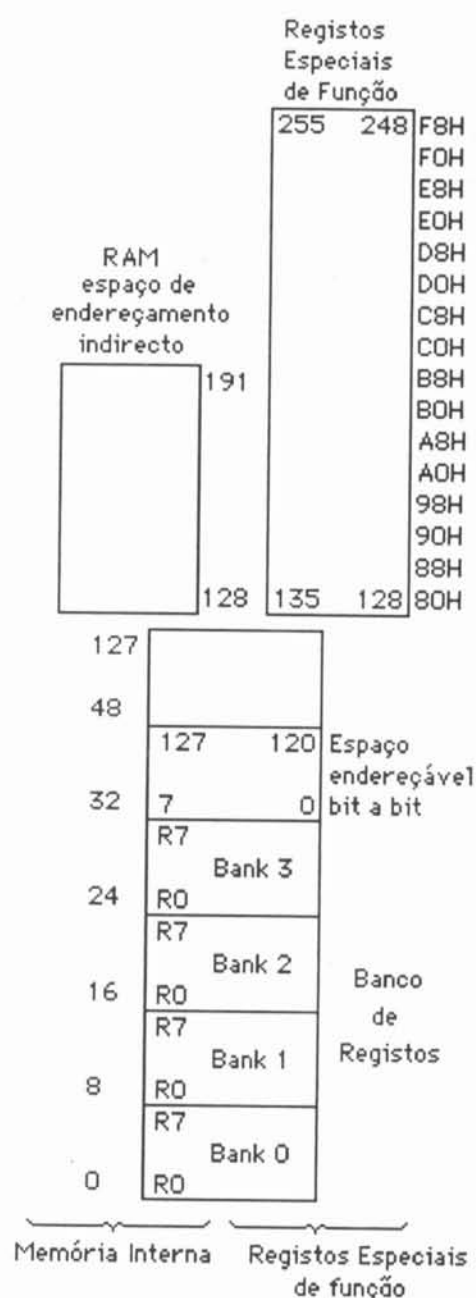


Figura C.3: Estrutura da memória interna do µC8044.

F8								
F0	B							
E8								
E0	ACC							
D8	NSNR	SIUST	TCB	TBL	TBS	FIFO	FIFO	FIFO
D0	PSW							
C8	STS	SMD	RCB	STAD	RFL	RBS	RBL	DMA CNT
C0								
B8	IP							
B0	P3							
A8	IE							
A0	P2							
98								
90	P1							
88	TCON	TMOD	TLO	TL1	TH0	TH1		
80	P0	SP	DPL	DPH				
		1	2	3	4	5	6	7

Figura C.4: Localização na memória interna dos registos do μ C8044.

C.4.2 - TIMERS/COUNTERS

Os dois COUNTERS que o μ C8044 possui podem ser utilizados na medida de intervalos de tempo, largura de impulsos, contagem de eventos, geração precisa de pedidos periódicos de interrupção e geração de sinal de relógio para a comunicação série. Os TIMERS são conduzidos internamente a 1/12MHz, e externamente podem ser conduzidos com uma frequência até 500KHz.

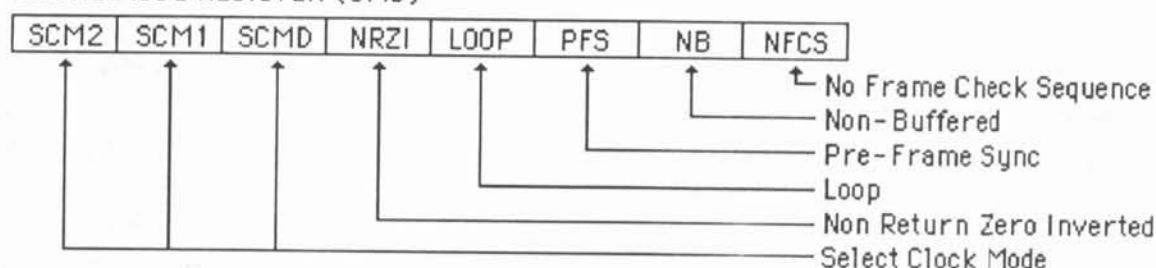
C.4.3 - Sistema de interrupções

No μ C8044 podem aparecer pedidos de interrupção devido a cinco motivos diferentes e serem atendidos segundo a sua prioridade, que pode ter dois níveis diferentes. Estas cinco fontes de interrupção são: SIU, TIMER 0, TIMER 1, INT 0 e INT 1. As interrupções de origem externa, INT 0 e INT 1, tanto podem ser activas ao nível como ao bordo descendente.

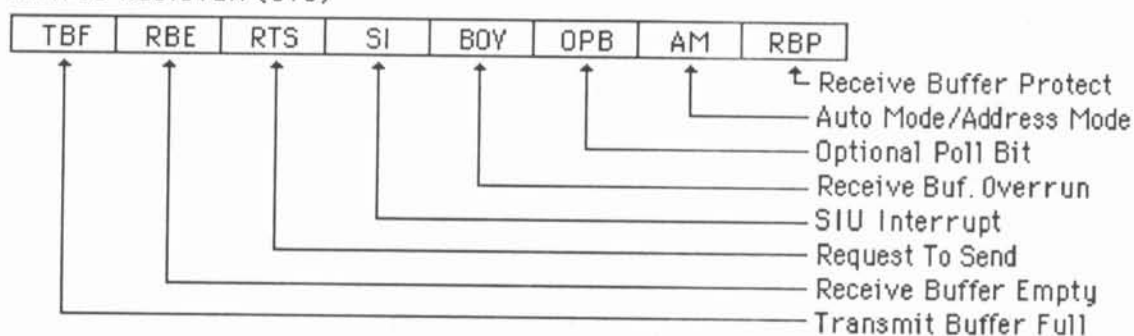
C.4.4 - SERIAL INTERFACE UNIT (SIU)

Esta unidade é utilizada no μ C8044 para a realização da comunicação série em HDLC/SDLC. Como em certas ocasiões é vantajoso que seja o CPU a controlar a recepção e emissão no canal série, o SIU está preparado para ser inicializado quer em Auto-mode como em Flexible (ou Non-auto) mode. No primeiro caso é o SIU que executa o controlo da comunicação deixando o CPU livre dessa função, ao contrário do que acontece no segundo caso em que é o CPU que controla toda a comunicação. Para uma correcta inicialização do SIU existem três registos de controlo cuja estrutura se apresenta na figura C.5.

SERIAL MODE REGISTER (SMD)



STATUS REGISTER (STS)



SEND COUNT RECEIVE COUNT REGISTER (NSNR)

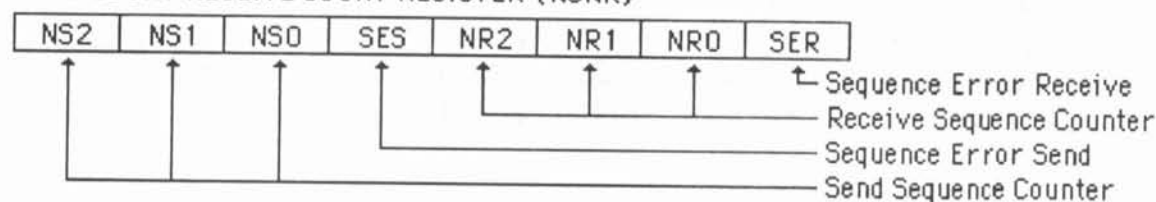


Figura C.5: Registos de controlo do SIU.

C.4.4.1 - AUTO-MODE

Quando neste modo o SIU implementa através de hardware uma parte do protocolo de SDLC, conseguindo responder a várias tramas em SDLC sem ser necessária a intervenção do CPU. As vantagens deste modo de funcionamento são:

- é necessário menos software para a implementação do escravo;
- a execução das respostas em hardware é muito mais rápida do que se for executada por software.

Uma desvantagem que apresenta o Auto-mode está na impossibilidade de um módulo a funcionar neste modo poder ser utilizado como mestre da rede.

Para a transmissão em Auto-mode, o CPU deve carregar os registos TRANSMIT INFORMATION "BUFFER", TRANSMIT BUFFER START, TRANSMITER BUFFER LENGHT e activar o bit TRANSMIT BUFFER FULL. O SIU automaticamente responderá a um pedido, enviando a trama de informação com o bit P/F do campo de controlo com o valor lógico "1".

Quando o SIU receber a informação de que a trama de informação enviada foi aceite automaticamente, incrementa o campo NS0,1,2 no registo NSNR e interrompe o CPU. Se depois da emissão da trama de informação o CPU não reconhecer essa informação, o SIU retransmitirá a trama de informação .

Para a recepção em Auto-mode o CPU deve carregar os registos RECEIVE BUFFER START, RECEIVE BUFFER LENGTH, fazer o reset do bit RECEIVE BUFFER PROTECT e activar o bit RECEIVE BUFFER EMPTY. Se nesta situação o SIU for acedido pela linha série e o bit TBF indicar que o TRANSMIT BUFFER está vazio, um comando de RECEIVE READY é respondido automaticamente. Se a informação então recebida for considerada válida, o SIU incrementará o campo NR0,1,2 no registo NSNR e interromperá o CPU .

Quando em Auto Mode o SIU pode reconhecer e responder, sem que o CPU intervenha, aos seguintes comandos:

- I (information);
- RR (receive ready);
- RNR (receive not ready);
- REJ (reject);
- UP (unnumbered poll).

Por outro lado o SIU pode gerar as seguintes respostas, também sem a intervenção do CPU:

- I (information);
- RR (receive ready);
- RNR (receive not ready).

Quando o bit RECEIVE BUFFER EMPTY (RBE) indica que o RECEIVE BUFFER está vazio a recepção é activada; caso contrário a recepção é desactiva.

C.4.4.2 - FLEXIBLE MODE

Neste modo de funcionamento todas as comunicações são controladas pelo CPU. É o CPU que codifica e descodifica o campo de controlo, e gere os reconhecimentos ou não das mensagens enviadas ou recebidas. Neste modo o módulo pode funcionar tanto como Master como Slave.

Para a transmissão em Flexible-mode, o CPU deve carregar o "buffer" que conterá a informação a transmitir, os registos TRANSMIT BUFFER LENGTH, o TRANSMIT CONTROL BYTE e fazer o set dos bits TBF e RTS. O SIU executa, depois de efectuar a transmissão, um pedido de interrupção ao CPU indicando que completou essa transmissão.

Para a recepção em Flexible-mode o CPU deve carregar os registos RECEIVE BUFFER START, RECEIVE BUFFER LENGTH, fazer o reset do bit RECEIVE BUFFER PROTECT e fazer o set do bit RECEIVE BUFFER EMPTY. Na recepção o CPU guarda o campo de controlo da trama recebida no registo RCB e guarda a informação no RECEIVE "BUFFER". Se o CRC não apresentar erro o SIU interromperá o CPU indicando que foi recebida uma trama válida. Se houver erro de CRC então não será gerada a interrupção por parte do SIU, e este pedirá uma retransmissão da trama.

O registo RECEIVE FIELD LENGTH contém o número de bytes contidos no campo de informação da trama recebida .

C.4.4.3 - Opções no formato da trama

Além do formato standard da trama SDLC, o μ C8044 pode suportar vários tipos de tramas, como se mostra na figura C.6.

C.4.4.4 - Registos de controlo e de estado

No μ C8044 existem três registos de controlo e de estado:

- SERIAL MODE REGISTER (SMD - C9H);
- STATUS/COMMAND REGISTER (STS - C8H);
- SEND/RECEIVE COUNT REGISTER (NSNR - D8H).

Opção de Trama	NFCS	NB	AM	Formato da Trama					
SDLC Standard Modo Non-Auto	0	0	0	F	A	C	I	FCS	F
SDLC Standard Modo Auto	0	0	1	F	A	C	I	FCS	F
Modo Non-Buffered Modo Non-Auto	0	1	0	F	A	I	FCS	F	
Modo Non-Addressed Modo Non-Auto	0	1	1	F	I	FCS	F		
Sem campo de FCS Modo Non-Auto	1	0	0	F	A	C	I	F	
Sem campo de FCS Modo Auto	1	0	1	F	A	C	I	F	
Sem campo de FCS Modo Non-Buffered Modo Non-Auto	1	1	0	F	A	I	F		
Sem campo de FCS Modo Non-Addressed Modo Non-Auto	1	1	1	F	I	F			

Bits:

AM - Modo Addressed/Modo Auto

NB - Modo Non-buffered

NFCS - Modo sem campo de FCS

Legenda:

F = Flag (01111110)

A= Campo de Endereço

C= Campo de Control

I= Campo de Informação

FCS= Campo de verificação da trama

Figura C.6: Formatos da trama em SDLC.

Para assegurar que, aquando do power-up, o SIU esteja em estado neutro, i.e. nem recebe nem transmite, o sistema faz o reset destes registos após o RESET.

C.4.4.4.1 - SERIAL MODE REGISTER

O SMD selecciona os modos de funcionamento do SIU. Este registo pode ser lido ou escrito pelo CPU, mas o SIU só o pode ler.

Para evitar complicações no acesso ao SMD por parte do CPU e SIU, o primeiro só faz o acesso quando os bits RBE e RTS têm ambos valor lógico "0". Normalmente o SMD só é acedido durante a inicialização. Para uma melhor compreensão ver figuras C.7 e C.8.

BIT #	Nome da Flag	Descrição
SMD.0	NFCS	Trama sem campo de FCS
SMD.1	NB	Modo Non-Buffered Trama sem campo de controlo
SMD.2	PFS	Modo de trama de pré-sincronismo
SMD.3	LOOP	Configuração de Loop
SMD.4	NRZI	Opção de código de NRZI
SMD.5	SCM0	Modo de Clock - Bit 0
SMD.6	SCM1	Modo de Clock - Bit 1
SMD.7	SCM2	Modo de Clock - Bit 2

Figura C.7: SMD - Serial Mode Register - Descrição dos bits.

C.4.4.4.2 - STATUS/COMMAND REGISTER

No STS selecciona-se o modo como o CPU vai controlar o SIU e é onde o CPU vai analisar o estado em que se encontra o SIU. O SIU pode ler o STS e alterar algum dos seus bits conforme o estado em que se encontre (ver figura C.9).

Os bits SCM descodificam as seguintes hipóteses

SCM			Modo de Clock	Baud Rate
2	1	0		
0	0	0	Clock externo	0 - 2.4M
0	0	1	reservado	
0	1	0	Clock interno	244 - 62.5K
0	1	1	Reservado	
1	0	0	Clock interno	0 - 375K
1	0	1	Clock interno	0 - 187.5K
1	1	0	Clock interno	375K
1	1	1	Clock interno	187.5K

Figura C.8: SMD - Serial Mode Register - Configuração de taxas de transmissão.

C.4.4.4.3 - SEND/RECEIVE COUNT REGISTER

O NSNR contém o número de sequências recebidas e emitidas além de dois indicadores de erro como se pode ver na figura C.10.

Este registo pode ser lido ou escrito quer pelo SIU quer pelo CPU.

C.4.4.5 - Registos de parâmetros

Existem oito registos de parâmetros usados em relação ao SIU. Qualquer um destes registos pode ser lido ou escrito pelo CPU. Os registos RFL e RCB normalmente são inicializados pelo SIU.

C.4.4.5.1 - STATION ADDRESS REGISTER

O STAD (CEH) indica qual o endereço da estação em que está inicializado. Para evitar conflitos entre o acesso ao registo pelo SIU e pelo CPU, o CPU só deve aceder ao STAD quando o SIU estiver

inactivo, que é o mesmo que dizer que teremos de ter RTS com valor lógico "0" e RBE com valor lógico "0". Normalmente o STAD é acedido durante a inicialização.

BIT #	Nome da Flag	Descrição
STS.0	RBP	Protecção do buffer de recepção
STS.1	AM	Modo Auto/ Modo Addressed
STS.2	OPB	Optional Poll
STS.3	BOV	Ocorrência de Overrun no buffer de recepção
STS.4	SI	Interrupção pelo SIU
STS.5	RTS	Request To Send
STS.6	RBE	Buffer de recepção vazio
STS.7	TBF	Buffer de emissão cheio

Figura C.9: STS - Status Command Register.

C.4.4.5.2 - TRANSMIT BUFFER START ADDRESS REGISTER

O TBS (DCH) indica o endereço de início do "buffer" que irá ser transmitido. O CPU só deve aceder ao TBS quando o SIU não estiver a transmitir ou seja TBF terá ter valor lógico "0".

C.4.4.5.3 - TRANSMIT BUFFER LENGTH REGISTER

O TBL (DBH) contém o comprimento, em bytes, do campo de informação a ser transmitido. Este registo pode conter "00H". O CPU só deve aceder ao TBL quando o SIU não estiver a transmitir ou seja TBF ter valor lógico "0".

Para a transmissão e para a recepção não são permitidos "buffers" circulares e será activada uma flag de fim de "buffer"

cada vez que o "buffer", ou parte dele, atingir um comprimento que corresponda ao endereço BFH da memória.

BIT #	Nome da Flag	Descrição
NSNR.0	SER	Erro na sequência de recepção: NS(P) ≠ NR(S)
NSNR.1	NRO	Contador de sequência de recepção Bit 0
NSNR.2	NR1	Contador de sequência de recepção Bit 1
NSNR.3	NR2	Contador de sequência de recepção Bit 2
NSNR.4	SES	Erro na sequência de emissão NP(P) ≠ NS(S) e NP(P) ≠ NS(S)+1
NSNR.5	NSO	Contador de sequência de emissão Bit 0
NSNR.6	NS1	Contador de sequência de emissão Bit 1
NSNR.7	NS2	Contador de sequência de emissão Bit 2

Figura C.10: NSNR - Send/Receive Count Register.

C.4.4.5.4 - TRANSMIT CONTROL BYTE REGISTER

O TCB (DAH) contém o byte que será carregado no campo de controlo da mensagem a ser transmitida se tivermos o μ C8044 em Non-Auto Mode. O CPU só deve aceder ao TCB quando o SIU não estiver a transmitir, ou seja TBF ter valor lógico "0".

C.4.4.5.5 - RECEIVE BUFFER START ADDRESS REGISTER

O RBS (CCH) indica onde é que começará a ser guardado o "buffer" recebido na linha série. O CPU só deve aceder ao RBS

quando o SIU não estiver a receber uma trama, ou seja quando RBE tiver valor lógico "0".

C.4.4.5.6 - RECEIVE BUFFER LENGTH REGISTER

O RBL (CBH) contém o comprimento, em bytes, do "buffer" onde poderá ser guardada a trama recebida. Este registo pode conter "00H". O CPU só deve aceder ao RBL quando o SIU não estiver a receber uma trama, ou seja RBE ter valor lógico "0".

C.4.4.5.7 - RECEIVE FIELD LENGTH REGISTER

O RFL (CDH) contém o comprimento, em bytes, do campo de informação presente na trama recebida. Este registo deverá ser carregado pelo SIU e poderá ser "00H". O CPU só deve aceder ao RFL quando o SIU não estiver a receber uma trama, ou seja RBE ter valor lógico "0".

C.4.4.5.8 - RECEIVE CONTROL BYTE REGISTER

O RCB (CAH) contém o campo de controlo da trama recebida. Este registo deve ser carregado pelo SIU. O CPU só deve aceder ao RCB quando o SIU não estiver a receber uma trama, ou seja RBE ter valor lógico "0".

C.4.5 - Conjunto de instruções para o μ C8044

Instruction Set Summary

Mnemonic		Description	Byte	Cycle
Arithmetic operations				
ADD	A,Rn	Add register to accumulator	1	1
ADD	A,direct	Add direct byte to accumulator	2	1
ADD	A,@Ri	Add indirect RAM to accumulator	1	1
ADD	A,#data	Add immediate data to accumulator	2	1
ADDC	A,Rn	Add register to accumulator with carry flag	1	1
ADDC	A,direct	Add direct byte to A with carry flag	2	1
ADDC	A,@Ri	Add indirect RAM to A with carry flag	1	1
ADDC	A,#data	Add immediate data to A with carry flag	2	1
SUBB	A,Rn	Subtract register from A with borrow	1	1
SUBB	A,direct	Subtract direct byte from A with borrow	2	1
SUBB	A,@Ri	Subtract indirect RAM from A with borrow	1	1
SUBB	A,#data	Subtract immediate data from A with borrow	2	1
INC	A	Increment accumulator	1	1
INC	Rn	Increment register	1	1
INC	direct	Increment direct byte	2	1
INC	@Ri	Increment indirect RAM	1	1
DEC	A	Decrement accumulator	1	1
DEC	Rn	Decrement register	1	1
DEC	direct	Decrement direct byte	2	1
DEC	@Ri	Decrement indirect RAM	1	1
INC	DPTR	Increment data pointer	1	2
MUL	AB	Multiply A and B	1	4
DIV	AB	Divide A by B	1	4
DA	A	Decimal adjust accumulator	1	1
Logical operations				
ANL	A,Rn	AND register to accumulator	1	1
ANL	A,direct	AND direct byte to accumulator	2	1
ANL	A,@Ri	AND indirect RAM to accumulator	1	1
ANL	A,#data	AND immediate data to accumulator	2	1
ANL	direct, A	AND accumulator to direct byte	2	1

Instruction Set Summary (cont'd)

Mnemonic		Description	Byte	Cycle
Logical operations (cont'd)				
ANL	direct, #data	AND immediate data to direct byte	3	2
ORL	A, Rn	OR register to accumulator	1	1
ORL	A, direct	OR direct byte to accumulator	2	1
ORL	A, @Ri	OR indirect RAM to accumulator	1	1
ORL	A, #data	OR immediate data to accumulator	2	1
ORL	direct, A	OR accumulator to direct byte	2	1
ORL	direct, #data	OR immediate data to direct byte	3	2
XRL	A, Rn	Exclusive OR register to accumulator	1	1
XRL	A, direct	Exclusive OR direct byte to accumulator	2	1
XRL	A, @Ri	Exclusive OR indirect RAM to accumulator	1	1
XRL	A, #data	Exclusive OR immediate data to accumulator	2	1
XRL	direct, A	Exclusive OR accumulator to direct byte	2	1
XRL	direct, #data	Exclusive OR immediate data to direct byte	3	2
CLR	A	Clear accumulator	1	1
CPL	A	Complement accumulator	1	1
RL	A	Rotate accumulator left	1	1
RLC	A	Rotate A left through carry flag	1	1
RR	A	Rotate accumulator right	1	1
RRC	A	Rotate accumulator right through carry flag	1	1
SWAP	A	Swap nibbles within the accumulator	1	1

Data transfer

MOV	A, Rn	Move register to accumulator	1	1
MOV	A, direct*)	Move direct byte to accumulator	2	1
MOV	A, @Ri	Move indirect RAM to accumulator	1	1
MOV	A, #data	Move immediate data to accumulator	2	1
MOV	Rn, A	Move accumulator to register	1	1
MOV	Rn, direct	Move direct byte to register	2	2
MOV	Rn, #data	Move immediate data to register	2	1
MOV	direct, A	Move accumulator to direct byte	2	1
MOV	direct, Rn	Move register to direct byte	2	2
MOV	direct, direct	Move direct byte to direct byte	3	2

*) MOV A, ACC is not a valid instruction

Instruction Set Summary (cont'd)

Mnemonic	Description	Byte	Cycle
----------	-------------	------	-------

Data transfer (cont'd)

MOV	direct.@R	Move indirect RAM to direct byte	2	2
MOV	direct.#data	Move immediate data to direct byte	3	2
MOV	@Ri,A	Move accumulator to indirect RAM	1	1
MOV	@Ri,direct	Move direct byte to indirect RAM	2	2
MOV	@Ri,#data	Move immediate data to indirect RAM	2	1
MOV	DPTR,#data 16	Load data pointer with a 16-bit constant	3	2
MOVC	A.@A+DPTR	Move code byte relative to DPTR to accumulator	1	2
MOVC	A.@A+PC	Move code byte relative to PC to accumulator	1	2
MOVX	A.@Ri	Move external RAM (8-bit addr.) to accumulator	1	2
MOVX	A.@DPTR	Move external RAM (16-bit addr.) to accumulator	1	2
MOVX	@Ri,A	Move A to external RAM (8-bit addr.)	1	2
MOVX	@DPTR,A	Move A to external RAM (16-bit addr.)	1	2
PUSH	direct	Push direct byte onto stack	2	2
POP	direct	Pop direct byte from stack	2	2
XCH	A,Rn	Exchange register with accumulator	1	1
XCH	A,direct	Exchange direct byte with accumulator	2	1
XCH	A.@Ri	Exchange indirect RAM with accumulator	1	1
XCHD	A.@Ri	Exchange low-order digit indirect RAM with A	1	1

Boolean variable manipulation

CLR	C	Clear carry flag	1	1
CLR	bit	Clear direct bit	2	1
SETB	C	Set carry flag	1	1
SETB	bit	Set direct bit	2	1
CPL	C	Complement carry flag	1	1
CPL	bit	Complement direct bit	2	1
ANL	C,bit	AND direct bit to carry flag	2	2
ANL	C,.bit	AND complement of direct bit to carry	2	2
ORL	C,bit	OR direct bit to carry flag	2	2
ORL	C,.bit	OR complement of direct bit to carry	2	2
MOV	C,bit	Move direct bit to carry flag	2	1
MOV	bit,C	Move carry flag to direct bit	2	2

DESCRIÇÃO DO MICROCONTROLADOR 8044

Instruction Set Summary (cont'd)

Mnemonic		Description	Byte	Cycle
Program and machine control				
ACALL	addr 11	Absolute subroutine call	2	2
LCALL	addr 16	Long subroutine call	3	2
RET		Return from subroutine	1	2
RETI		Return from interrupt	1	2
AJMP	addr 11	Absolute jump	2	2
LJMP	addr 16	Long jump	3	2
SJMP	rel	Short jump (relative addr.)	2	2
JMP	@A-DPTR	Jump indirect relative to the DPTR	1	2
JZ	rel	Jump if accumulator is zero	2	2
JNZ	rel	Jump if accumulator is not zero	2	2
JC	rel	Jump if carry flag is set	2	2
JNC	rel	Jump if carry flag is not set	2	2
JB	bit rel	Jump if direct bit is set	3	2
JNB	bit rel	Jump if direct bit is not set	3	2
JBC	bit rel	Jump if direct bit is set and clear bit	3	2
CJNE	A direct rel	Compare direct byte to A and jump if not equal	3	2
CJNE	A #data rel	Comp. immed. to A and jump if not equal	3	2
CJNE	Rn #data rel	Comp. immed. to reg. and jump if not equal	3	2
CJNE	@Ri #data rel	Comp. immed. to ind. and jump if not equal	3	2
DJNZ	Rn rel	Decrement register and jump if not zero	2	2
DJNZ	direct rel	Decrement direct and jump if not zero	3	2
NOP		No operation	1	1

Notes on data addressing modes:

- Rn - Working register R0 - R7
- direct - 128 internal RAM locations, any I/O port, control or status register
- @Ri - Indirect internal or external RAM location addressed by register R0 or R1
- #data - 8-bit constant included in instruction
- #data 16 - 16-bit constant included as bytes 2 and 3 of instruction
- bit - 128 software flags, any I/O pin, control or status bit
- A - Accumulator

Notes on program addressing modes

- addr 16 - Destination address for LCALL and LJMP may be anywhere within the 64 Kbyte program memory address space.
- addr 11 - Destination address for ACALL and AJMP will be within the same 2 Kbyte page of program memory as the first byte of the following instruction.
- rel - SJMP and all conditional jumps include an 8-bit offset byte. Range is +127/-128 bytes relative to first byte of the following instruction.

All mnemonics copyrighted © Intel Corporation 1980

APENDICE D

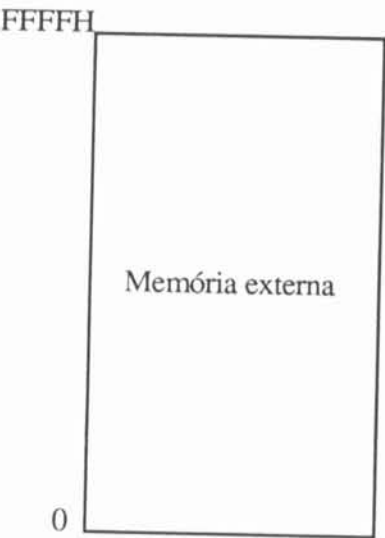
Pormenores fisicos de implementação

APENDICE D - PORMENORES FISICOS DE IMPLEMENTAÇÃO

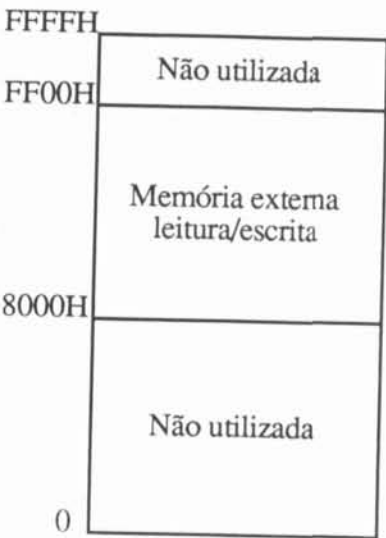
D.1 - Unidades de aquisição

D.1.1 - Mapa de memória

D.1.1.1 - Memória de programa

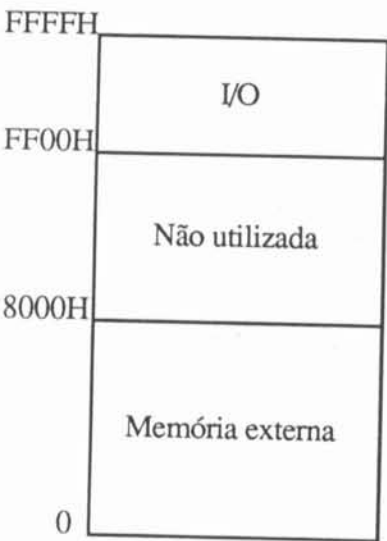


Opção A



Opção B

D.1.1.2 - Memória de dados



D.1.2 - Programação dos "jumpers"

D.1.2.1 - Memória de programa e de dados

Configuração Eprom :

J1 ☐ ☐ J2
J3 ☐ ☐ J4
J5 ☐ ☐ J6
J7 ☐
J8 ☐ ☐ J9
J10 ☐ ☐ J11
J12 ☐ ☐ J13
J14 ☐ ☐ J15

		
EPROM	4K	J11,J13 ; J12,J14
	8K	J1,J2 ; J8,J10 ; J9,J11
	16K	J1,J2 ; J8,J10 ; J9,J11 ; J14,J15
	32K	J1,J2 ; J7,J8 ; J11,J13 ; J14,J15
	64K	J3,J5 ; J7,J9 ; J11,J13 ; J14,J15

J47 ☐ ☐ J48

J47,J48	0 Opção A
	1 Opção B

Configuração Ram :

J32 ☐ ☐ J33
J34 ☐ ☐ J35
J36 ☐ ☐ J37
J38 ☐
J39 ☐ ☐ J40
J41 ☐ ☐ J42
J43 ☐ ☐ J44
J45 ☐ ☐ J46

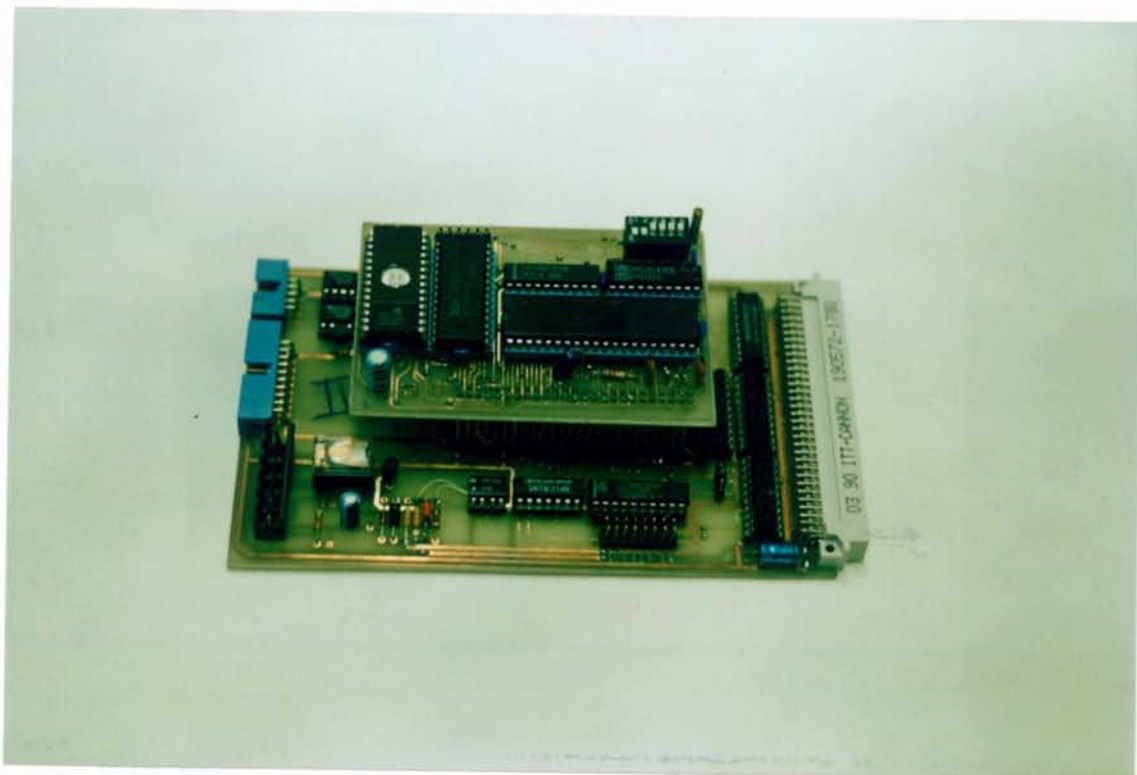
		
RAM	2K	J42,J44 ; J43,J45
	4K	J39,J40 ; J42,J44
	8K	J39,J40 ; J42,J44
	32K	J36,J38 ; J39,J40 ; J42,J44 ; J45,J46

D.1.2.2 - Endereço da unidade de aquisição na rede própria

J16 ☐ ☐ J17
J18 ☐ ☐ J19
J20 ☐ ☐ J21
J22 ☐ ☐ J23
J24 ☐ ☐ J25
J26 ☐ ☐ J27
J28 ☐ ☐ J29
J30 ☐ ☐ J31

	1
	0
J16,J17 : LSB J30,J31 : MSB	

D.1.3 - Placa das unidades de aquisição



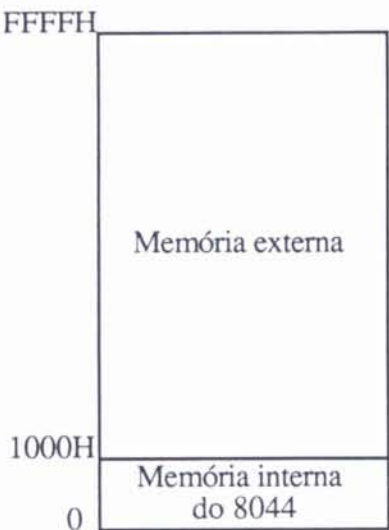
D.1.4 - Vista de uma unidades de aquisição



D.2 - Unidades concentradoras

D.2.1 - Mapa de memória

D.2.1.1 - Memória de programa

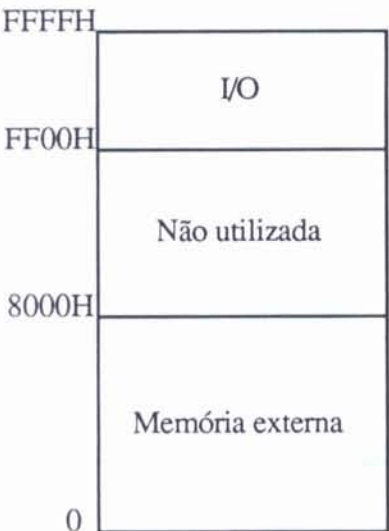


Opção A



Opção B

D.2.1.2 - Memória de dados



D.2.2 - Programação dos "jumpers"

D.2.2.1 - Memória de programa e de dados

Configuração Eprom:

J1 ☐ ☐ J2
J3 ☐ ☐ J4
J5 ☐ ☐ J6
J7 ☐
J8 ☐ ☐ J9
J10 ☐ ☐ J11
J12 ☐ ☐ J13
J14 ☐ ☐ J15

		
EPROM	4K	J11,J13 ; J12,J14
	8K	J1,J2 ; J8,J10 ; J9,J11
	16K	J1,J2 ; J8,J10 ; J9,J11 ; J14,J15
	32K	J1,J2 ; J7,J8 ; J11,J13 ; J14,J15
	64K	J3,J5 ; J7,J9 ; J11,J13 ; J14,J15

Configuração Ram:

J32 ☐ ☐ J33
J34 ☐ ☐ J35
J36 ☐ ☐ J37
J38 ☐
J39 ☐ ☐ J40
J41 ☐ ☐ J42
J43 ☐ ☐ J44
J45 ☐ ☐ J46

		
RAM	2K	J42,J44 ; J43,J45
	4K	J39,J40 ; J42,J44
	8K	J39,J40 ; J42,J44
	32K	J36,J38 ; J39,J40 ; J42,J44 ; J45,J46

D.2.2.2 - Configuração do modo de funcionamento do nó BITBUS

J16 ☐ ☐ J17
J18 ☐ ☐ J19
J20 ☐ ☐ J21
J22 ☐ ☐ J23
J24 ☐ ☐ J25
J26 ☐ ☐ J27
J28 ☐ ☐ J29
J30 ☐ ☐ J31

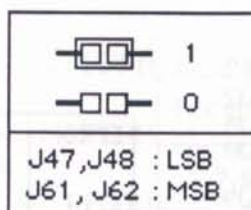
J16,J17 : LSB
J30,J31 : MSB

 1
 0

J16,J17	00 Sincronos
J18,J19	01 Self Clocked 373 Kb/seg
	10 reservado
	11 Self Clocked 62.5 Kb/seg
J20,J21	reservado
J22,J23	0 ROM interna
	1 ROM externa
J24,J25	0 Opção A
	1 Opção B
J26,J27	1 Com FIFO
	0 Sem FIFO
J28,J29	1 FIFO por DMA
	0 FIFO por interrupção
J30,J31	reservado

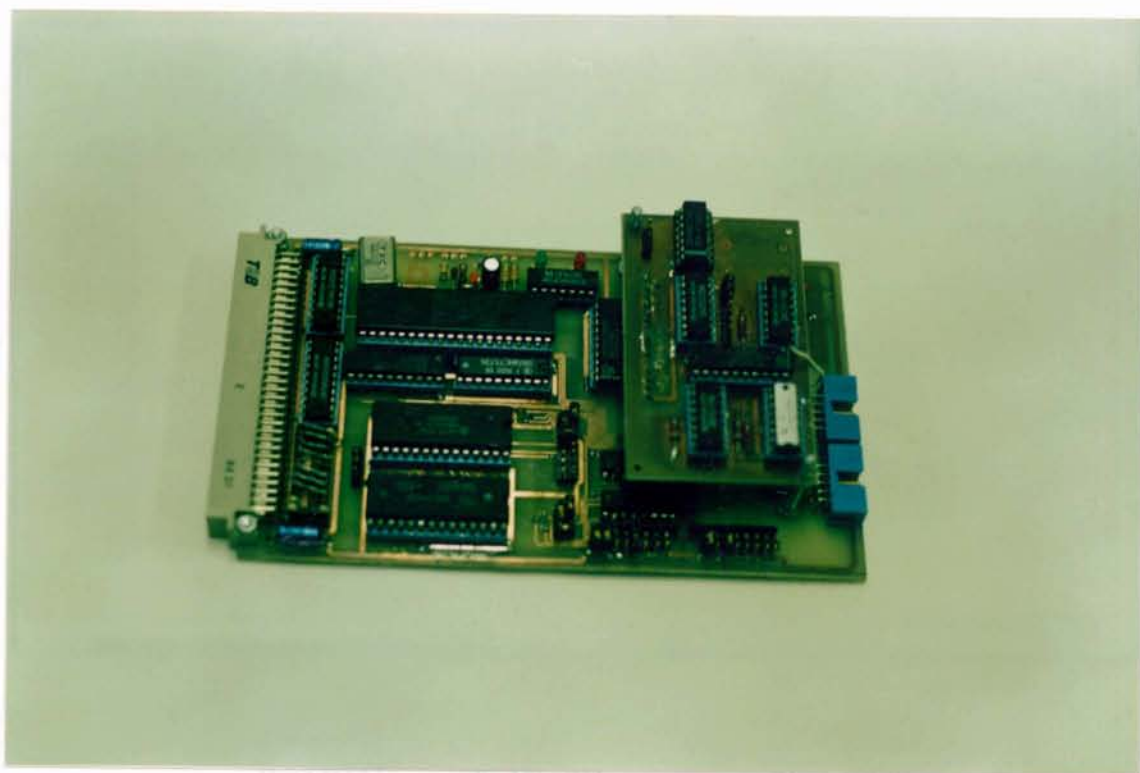
D.2.2.3 - Endereço da unidade concentradora na rede BITBUS

J47	□	□	J48
J49	□	□	J50
J51	□	□	J52
J53	□	□	J54
J55	□	□	J56
J57	□	□	J58
J59	□	□	J60
J61	□	□	J62

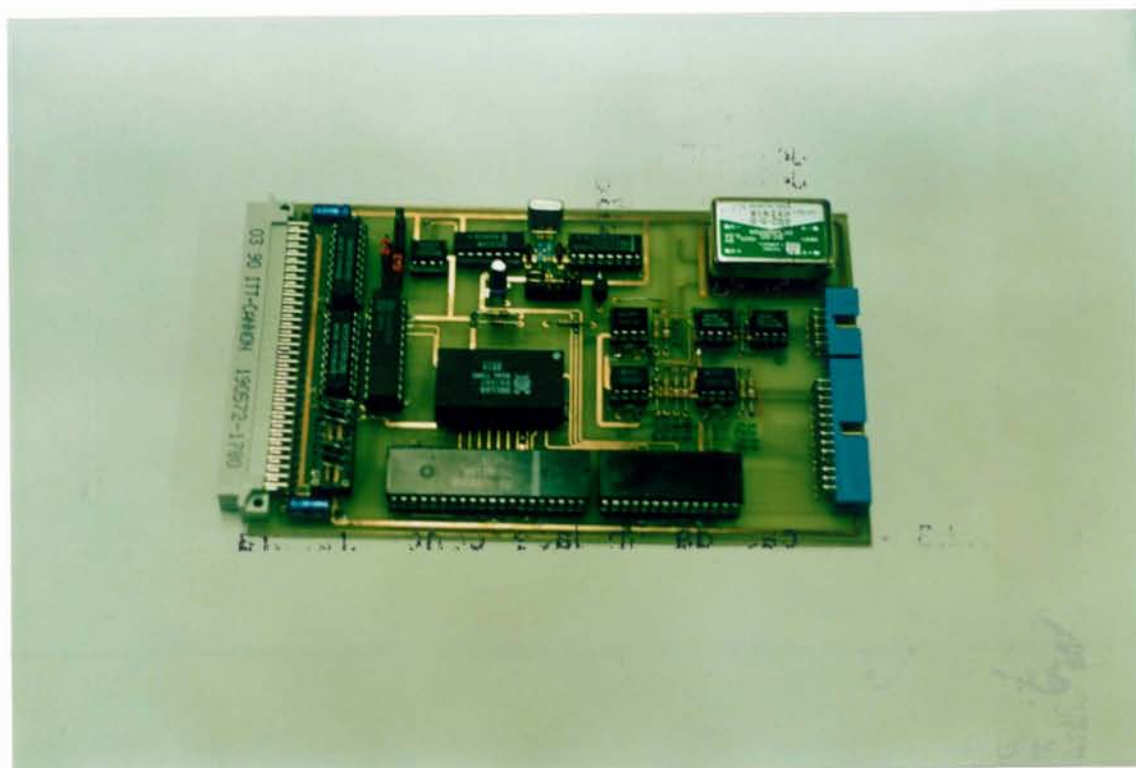


D.2.3 - Placas da unidade concentradora

D.2.3.1 - Carta de CPU 8044



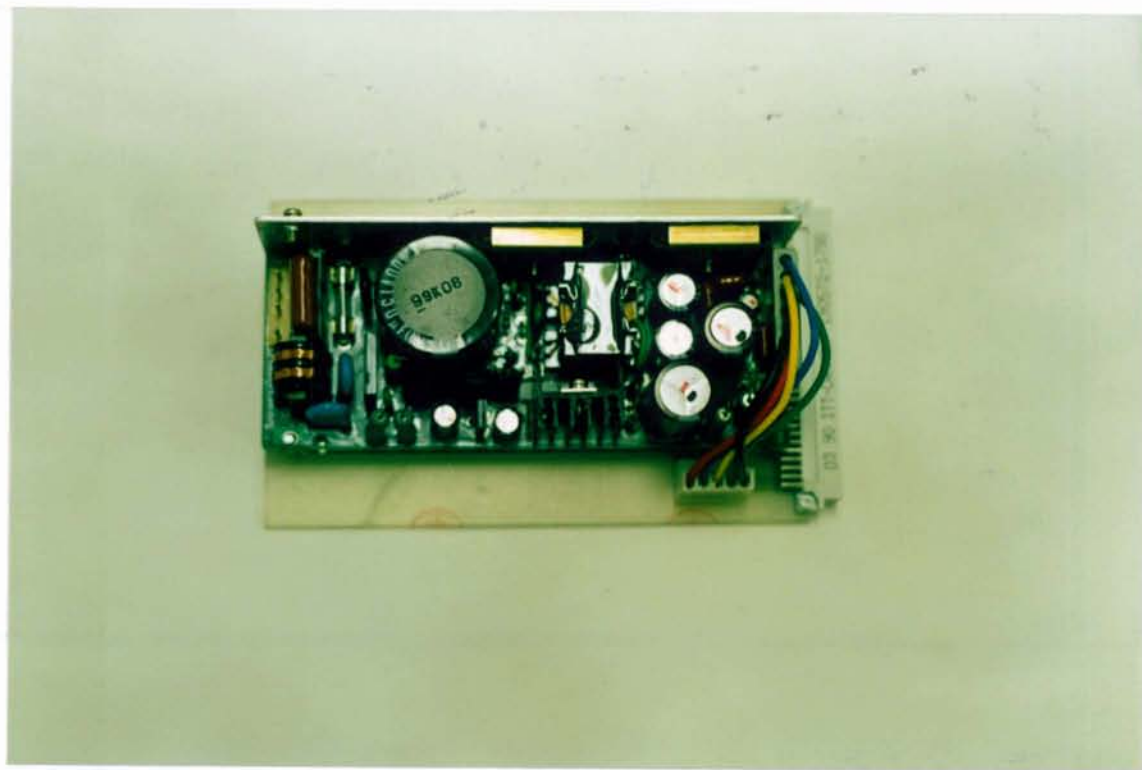
D.2.3.2 - Carta de interfaces



D.2.4 - Vista de uma unidade concentradora



D.3 - Carta de alimentações





FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

BIBLIOTECA





FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

BIBLIOTECA



000006480