USP

Campus de São Carlos

REDE LOCAL DE INSTRUMENTOS:

RÁDIO DIGITAL APLICADO À AGROPECUÁRIA

Eng. NELSON CORONA JUNIOR

Orientador: Prof. Dr. José Carlos Sartori

UNIVERSIDADE DE SÃO PAULO

ESCOLA DE ENGENHARIA DE SÃO CARLOS

**REDE LOCAL DE INSTRUMENTOS:
RÁDIO DIGITAL APLICADO
À AGROPECUÁRIA**

NELSON CORONA JUNIOR

Dissertação apresentada à Escola de Engenharia de São Carlos, da Universidade de São Paulo, como parte dos requisitos para obtenção do Título de Mestre em Engenharia Elétrica

ORIENTADOR: Prof. Dr. José Carlos Sartori

DEDALUS - Acervo - EESC

31100016697

Class.	TFSE-EESC
Cutl.	3636 n
Tombo	200/96

Eletricidade

ii

0745715

169p

Corona, Nelson Jr.

Rede local de instrumentos: Aplicação de rádio digital à agropecuária / Nelson Corona Junior. -- São Carlos, 1996.

Dissertação (Mestrado) -- Escola de Engenharia de São Carlos - Universidade de São Paulo
Orientador: José Carlos Sartori

Telemetria, Rádio, Agropecuária

FOLHA DE APROVAÇÃO

Dissertação defendida e aprovada em 14-8-1996
pela Comissão Julgadora:

Prof. Doutor JOSÉ CARLOS SARTORI - Orientador
(Escola de Engenharia de São Carlos - Universidade de São Paulo)

Prof. Doutor MARCELO BASILIO JOAQUIM
(Escola de Engenharia de São Carlos - Universidade de São Paulo)

Prof. Doutor RICARDO YASSUSHI INAMASU
(EMBRAPA - CNPDIA - São Carlos, SP)

Coordenador da área "Engenharia Elétrica"
Prof. Dr. GERALDO ROBERTO M. DA COSTA

Presidente da Comissão de Pós-Graduação
Prof. Dr. JOSÉ CARLOS A. CINTRA

Dedicatória

Aos meus pais, Nelson e Zeline, eternos incentivadores, e à
minha esposa Silmara, que com carinho e paciência ajudou-me a
finalizar este trabalho.

AGRADECIMENTOS

Ao professor José Carlos Sartori, pela orientação atenciosa e segura fornecida durante a elaboração deste trabalho e pela amizade.

Ao pesquisador Ricardo Y. Inamasu, pelo apoio e incentivo constantes, e pela amizade.

Aos colegas, professores e funcionários do Departamento de Engenharia Elétrica da EESC-USP, que contribuíram de alguma forma para o desenvolvimento deste trabalho.

Aos pesquisadores, técnicos e funcionários do Centro Nacional de Pesquisa e Desenvolvimento de Instrumentação Agropecuária, pelas colaborações e amizade.

RESUMO

A produção agropecuária é influenciada em todas as suas fases por fatores ambientais, tais como condições do solo, biológicas e atmosféricas. As fases de plantio, germinação, crescimento, floração, frutificação e colheita, bem como o armazenamento e conservação da produção, podem ter um manejo mais eficiente com o auxílio de informações obtidas a partir dos dados referentes a estes fatores. Sistemas de telemetria sem fio são obrigatórios ou convenientes em aplicações que apresentam como requisito mobilidade, flexibilidade, rapidez na instalação ou aplicações de caráter temporário. Sistemas com cabos exigem muita mão de obra para instalar, manutenção cara e demorada e, além disto, eles são difíceis de proteger contra raios, estão sujeitos à ação de roedores, depredações e acidentes como queimadas. Neste trabalho, desenvolveu-se um sistema de telemetria sem fio para obtenção de dados referentes aos fatores que influenciam a produção agropecuária. O sistema é composto por estações microprocessadas interligadas por enlaces de rádio, que compartilham o mesmo canal de radiofrequência e empregam tecnologia de redes locais de instrumentos. As principais características obtidas são: baixo consumo de energia associado a um alcance relativamente alto, boa imunidade a ruídos e facilidade de instalação.

ABSTRACT

The agricultural production is influenced in all of its phases by environmental factors such as weather, soil and biological conditions. The planting, germination, growing, flowering, fructification and harvest phases, as well as warehousing and maintenance of production, can be handled in a more effective way using information gathered from data related to those factors. The wireless telemetry systems are mandatory or suitable in applications where mobility, flexibility, and fast or temporary installation are main requirements. Wire systems demand too much workmanship for installation, expensive and time consuming maintenance besides that they are difficult to protect against lightning, rodents, spoiling and mishap such as fire in a wood. The main goal of this work is a project for wireless telemetry system in order to provide data related to states in the agricultural production. The system is made of microprocessed stations interconnected by radio links, sharing same radiofrequency channel and using local networks technology for instruments. The main advantages are: low energy consumption related to a relatively long path, good immunity to noise and easy installation.

Sumário

1. INTRODUÇÃO.....	1
1.1 AGROPECUÁRIA E OS FATORES AMBIENTAIS	1
1.2 AGROPECUÁRIA E TELEMETRIA	2
1.3 TELEMETRIA SEM FIOS.....	3
1.4 INSTRUMENTAÇÃO MICROPROCESSADA.....	4
1.5 TELEMETRIA COMPUTADORIZADA.....	5
1.6 TELEMETRIA COMPUTADORIZADA APLICADA À AGROPECUÁRIA	6
1.7 ESTAÇÕES DE DADOS.....	6
1.8 REDES INDUSTRIAIS DE INSTRUMENTOS.....	8
1.9 AGROPECUÁRIA E AS REDES INDUSTRIAIS DE INSTRUMENTOS	10
2. PROPOSIÇÃO	11
3. REVISÃO DA LITERATURA.....	13
3.1 PADRÃO 4-20MA	13
3.2 COMUNICAÇÃO DIGITAL.....	14
3.2.1 Migração da comunicação analógica para a digital ...	15
3.2.2 Comunicação digital e informações secundárias.....	15
3.3 A PADRONIZAÇÃO DAS REDES INDUSTRIAIS DE INSTRUMENTOS	16
3.3.1 A padronização de redes industriais para o chão de <i>fábrica</i>	17
3.3.2 Dificuldades da padronização.....	18
3.3.3 O que esperar da padronização	19

3.4 MEIOS FÍSICOS E MECANISMOS DE TRANSMISSÃO	19
3.4.1 <i>Laser e infravermelho</i>	20
3.4.2 <i>Rádio</i>	21
3.5 LEGISLAÇÃO SOBRE RADIOTRANSMISSÃO.....	23
3.5.1 <i>Legislação sobre radiotransmissão no brasil</i>	24
3.5.2 <i>Legislação em outros países</i>	24
3.5.3 <i>A legislação brasileira e a parte prática deste trabalho</i>	25
3.6 PROTEÇÃO CONTRA SURTOS DE TENSÃO	26
3.6.1 <i>Equipamentos de campo e os surtos de tensão</i>	27
3.6.2 <i>Severidade dos surtos</i>	27
3.6.3 <i>Aterramento</i>	28
3.6.4 <i>Centelhadores a gás</i>	29
3.6.5 <i>Varistores</i>	30
3.6.6 <i>Supressores de transientes de silício</i>	30
3.6.7 <i>Proteção com múltiplos estágios</i>	30
3.7 PROTEÇÃO CONTRA TEMPERATURA E UMIDADE	31
3.7.1 <i>Proteção contra umidade</i>	31
3.7.2 <i>Proteção contra temperatura</i>	32
3.8 VELOCIDADE DE TRANSMISSÃO	32
3.9 TRABALHOS DESENVOLVIDOS NA ÁREA	33
4. DESENVOLVIMENTO.....	40
4.1 ESTAÇÃO DE DADOS.....	40
4.1.1 <i>Configuração da estação de dados</i>	41
4.1.2 <i>Transceptor de rádio</i>	47

4.1.3 Instrumento microprocessado.....	60
4.2 PROTOCOLO DE COMUNICAÇÃO.....	75
4.2.1 Modelo OSI.....	77
4.2.2 Modelo OSI simplificado.....	77
4.2.3 Topologia da rede.....	78
4.2.4 Camada física.....	80
4.2.5 Camada de enlace de dados.....	86
4.2.6 Camada de aplicação.....	91
4.2.7 Rotina principal.....	91
4.3 TESTE DOS PROTÓTIPOS.....	95
4.3.1 Procedimento de teste.....	96
5. RESULTADO E DISCUSSÃO.....	98
5.1 DISPOSITIVO DE COMUNICAÇÃO VIA RÁDIO.....	98
5.1.1 Alcance.....	98
5.1.2 Velocidade de transmissão.....	100
5.1.3 Interface com o instrumento microprocessado.....	100
5.1.4 Dimensões.....	101
5.2 INSTRUMENTO MICROPROCESSADO.....	101
5.2.1 Principais características.....	101
5.2.2 Dimensões.....	102
5.3 PROTOCOLO DE COMUNICAÇÃO.....	103
5.4 CONSUMO DE ENERGIA.....	103
6. CONCLUSÃO.....	105

7. REFERÊNCIAS BIBLIOGRÁFICAS	107

APÊNDICE	117

Lista de figuras

Figura 1.1: Diagrama de blocos de um instrumento microprocessado.....	5
Figura 1.2: Ilustração de um sistema computadorizado para o manejo de culturas.	7
Figura 1.3: Diagrama de um sistema de telemetria e telecontrole para auxiliar o manejo de culturas.	8
Figura 1.4: Diagrama de uma rede de computadores aplicada ao controle e supervisão de processos de produção.....	9
Figura 2.1: Diagrama de blocos com o objetivo ressaltado....	11
Figura 3.1: Diagrama de blocos do padrão 4-20 mA.	14
Figura 3.2: Diagrama de blocos de um sistema de proteção em cascata.	31
Figura 4.1: Diagrama de blocos de uma estação de dados.....	41
Figura 4.2: Configuração possível para uma estação de dados operando no campo: A) antena, transmissor/receptor de rádio; B) instrumento microprocessado; C) bateria; D) sensor.....	42
Figura 4.3: Diagrama de blocos mostrando como o sistema é dividido.	43

Figura 4.4: Diagrama de blocos da estação de dados, mostrando as interfaces entre o transceptor e o instrumento microprocessado.....	45
Figura 4.5: Diagrama de blocos da estação de dados, mostrando as principais partes do sistema.....	46
Figura 4.6: Diagrama de blocos do transceptor de rádio e interface.	47
Figura 4.7: Forma de conexão entre transmissor, receptor, isolador de 1/4 de onda e antena.....	48
Figura 4.8: Ilustração da antena discone vista lateralmente. ..	49
Figura 4.9: Ilustração do transformador de impedâncias (“balun”).	51
Figura 4.10: Ilustração da antena Yagi com 5 elementos.....	51
Figura 4.11: Diagrama de blocos do rádio transmissor TXM-418-A.	52
Figura 4.12: Diagrama de blocos do rádio receptor SILRX-418-A.	54
Figura 4.13: Diagrama esquemático do circuito com os módulos transmissor e receptor de rádio.	55
Figura 4.14: Diagrama esquemático do discriminador de tensão para o chaveamento da alimentação do transmissor/receptor de rádio.	56
Figura 4.15: Diagrama esquemático do adaptador de níveis lógicos e comunicação RS-485.	57

Figura 4.16: Diagrama esquemático do circuito regulador de tensão de 5V, com mínima tensão de entrada limitada apenas pela tensão coletor/emissor de saturação de Q3. ...	58
Figura 4.17: Diagrama esquemático do circuito de proteção contra surtos.....	59
Figura 4.18: Diagrama em blocos do instrumento microprocessado.....	61
Figura 4.19: Diagrama esquemático do microcontrolador e memórias de dados e de programa.....	62
Figura 4.20: Diagrama de blocos do microcontrolador 80C515.	63
Figura 4.21: Diagrama de blocos do relógio calendário.	67
Figura 4.22: Diagrama esquemático do relógio calendário.	68
Figura 4.23: Diagrama esquemático de uma interface digital balanceada multiponto baseada no padrão de comunicação RS485.....	69
Figura 4.24: Diagrama de blocos da interface junto ao instrumento microcontrolado.....	71
Figura 4.25: Diagrama esquemático da fonte de tensão ajustável controlada pelo instrumento microprocessado.....	72
Figura 4.26: Diagrama esquemático da fonte da estação de dados.....	74
Figura 4.27: As sete camadas do modelo ISO/OSI.....	76
Figura 4.28: Modelo OSI simplificado.	78
Figura 4.29: Conexão de dispositivos por meio de barramento.	79

Figura 4.30: Esquema de transmissão NRZ, por duração de pulso.	80
Figura 4.31: O efeito do "jitter" no código utilizado. A) Neste caso, o "jitter" pode levar a duração dos sinais para um tempo inválido. B) Neste caso, o "jitter" pode causar erros na recepção.	81
Figura 4.32: Fluxograma da rotina para gerar o código por largura de pulso.	83
Figura 4.33: Fluxograma da rotina de decodificação da linha.	85
Figura 4.34: Modelo simplificado de um sistema de comunicação.	87
Figura 4.35: Esquema de alocação de intervalos de tempo a cada estação de dados.	88
Figura 4.36: Diagrama da transmissão do preâmbulo de sincronização e dos lotes de palavras-código.	90
Figura 4.37: Estrutura das palavras código de endereço e dados.	91
Figura 4.38: Fluxograma parcial da rotina principal da estação de dados	94
Figura 4.39: Fluxograma da rotina de sincronização.	95

LISTA DE ABREVIATURAS E SIGLAS

A/D	- Analógico/Digital
AM	- Amplitude Modulation
ASK	- Amplitude Shift Keying

balun	- Balanced/Unbalanced
CMOS	- Complementary Metal-Oxide Semiconductor
CPU	- Central Processing Unit
EPROM	- Electrically Erasable Read Only Memory
FM	- Frequency Modulation
FSK	- Frequency Shift Keying
IBM-PC	- IBM Personal Computer
PM	- Phase Modulation
PSK	Phase Shift Keying
PWM	Pulse Width Modulation
RAM	Random Access Memory
RF	Radio Frequency
RFI	Radio Frequency Interference
ROM	Read Only Memory
filtro SAW	Surface acoustic Wave
UHF	Ultra High Frequency
VDR	Voltage Dependent Resistor
VHF	Very High Voltage

LISTA DE SÍMBOLOS

λ	- Comprimento de onda

1. Introdução

A produção agropecuária tem importância fundamental para a economia nacional. As condições climáticas e a grande extensão de terras cultiváveis tornam a agropecuária uma atividade econômica estratégica para o crescimento econômico e melhoria da qualidade de vida.

Os recursos que a ciência e a tecnologia podem oferecer devem ser empregados para a obtenção de maior produtividade da terra e da mão de obra, melhor qualidade e maior diversidade de produtos, buscando a competitividade econômica e maior racionalidade na utilização dos recursos ambientais disponíveis, para que o processo produtivo seja mantido no longo prazo.

1.1 Agropecuária e os fatores ambientais

A produção agropecuária é decisivamente influenciada em todas as suas fases por fatores ambientais, como as condições do solo, biológicas e atmosféricas.

As fases de plantio, germinação, crescimento,

floração frutificação, colheita, bem como o armazenamento e conservação da produção podem ter um manejo mais eficiente com o auxílio de informações obtidas a partir dos dados referentes aos fatores ambientais.

1.2 Agropecuária e telemetria

O acesso a dados agrometeorológicos em tempo-real ou quase-tempo-real, para auxiliar o manejo da produção agropecuária é uma ferramenta importante, como demonstra o número de trabalhos desenvolvidos nesta área.

Os dados em tempo real podem ser utilizados para auxiliar os agricultores em decisões relacionadas com as condições ambientais como a estimação da necessidade de água das culturas e a temporização da aplicação de pesticidas.

Com a otimização destas operações, pode-se economizar, ou utilizar-se de forma efetiva, a energia, os insumos, a mão de obra, a água, o solo.

Em processos dispersos, como normalmente é a produção agropecuária, a obtenção manual de dados referentes ao processo produtivo, pode não ser efetiva ou eficiente.

Leituras manuais de sensores localizados próximos ou junto à produção agropecuária, envolvem o deslocamento de técnicos realizando tarefas repetitivas, estando sujeitas a falhas humanas e consumindo mão de obra.

O emprego de telemetria automatizada para a obtenção destes dados pode ser uma solução vantajosa.

O projeto de sistemas para monitorar, processar e controlar diferentes parâmetros físicos em locais remotos, é um desafio relativamente novo e em demanda, como afirmado por AHMED & AL-AMOUD (1993).

1.3 Telemetria sem fios

Telemetria sem fios é uma solução obrigatória ou conveniente em aplicações que apresentam como requisito mobilidade, flexibilidade e/ou rapidez na instalação, ou ainda, em aplicações de caráter temporário.

Além disto, a substituição dos cabos pode eliminar desvantagens decisivas associadas aos mesmos como muita mão de obra para instalar, manutenção cara e demorada, proteção contra raios de difícil solução, sujeição à ação de roedores, depredações, roubos e acidentes como queimadas.

Nos processos produtivos agropecuários, a movimentação de máquinas, animais e trabalhadores é também um fator que pode tornar a utilização de cabos inconveniente ou inviável.

1.4 Instrumentação microprocessada

Atualmente, os sistemas de telemetria utilizam comunicação baseada na tecnologia de redes de computadores. São empregados instrumentos microprocessados, com conseqüente distribuição de inteligência no processo sob medida.

A figura 1.1 mostra um diagrama de blocos de um instrumento microprocessado.

A utilização de microcontroladores em instrumentos de campo tem sido fortemente impulsionada pela indústria de semicondutores de silício.

Quanto melhor os microcontroladores apresentam-se com relação ao desempenho, consumo de energia e custo, mais tornam-se capazes e convenientes para atender os requisitos da instrumentação de campo.

Instrumentos microprocessados apresentam desempenho superior aos não microprocessados devido a fatores como:

- 1) Comunicação com computadores;
- 2) Correção de não linearidades dos sensores;
- 3) Conversão do sinal do sensor para unidades de engenharia;
- 4) Mudança de faixa automática;
- 5) Compensação de influências ambientais como temperatura;
- 6) Autodiagnóstico, cujo resultado pode ser comunicado a um

dispositivo supervisor.

Figura 1.1: Diagrama de blocos de um instrumento microprocessado.

1.5 Telemetria computadorizada

Os fatores ambientais são variáveis no tempo e no espaço, e para um monitoramento adequado, são necessárias leituras periódicas de sensores espalhados pela área de interesse.

Sistemas computadorizados de telemetria podem varrer os sensores e/ou atuadores instalados próximos ou junto ao processo produtivo agropecuário de forma automática, com economia de mão de obra qualificada.

1.6 Telemetria computadorizada aplicada à agropecuária

Os elementos-chave de um sistema de telemetria microprocessado para auxiliar no manejo de culturas são:

- 1) Sensores confiáveis capazes de monitorar quantidades relativas ao solo, planta e clima.
- 2) Estações de dados operando na área de produção agropecuária, lendo múltiplos sensores e armazenando temporariamente e/ou transmitindo os dados a um computador em uma área de controle da produção (sala de controle).
- 3) Modelos e sistemas especialistas que auxiliem o agricultor a tomar decisões relativas ao manejo.

A figura 1.2 mostra um diagrama de um sistema computadorizado para o manejo de culturas.

1.7 Estações de dados

Neste trabalho, *estação de dados* é a designação empregada para instrumentos microprocessados com capacidade

de comunicar-se com computadores ou outras estações de dados, e ler sensores, controlar atuadores, processando, armazenando e transmitindo os dados referentes aos mesmos.

Figura 1.2: Ilustração de um sistema computadorizado para o manejo de culturas.

A figura 1.3 mostra um diagrama de um sistema de telemetria e telecontrole para auxiliar o manejo de culturas, com um diagrama de blocos indicando uma estação de dados.

Figura 1.3: Diagrama de um sistema de telemetria e telecontrole para auxiliar o manejo de culturas.

1.8 Redes industriais de instrumentos

O controle de processos industriais empregando redes de computadores é uma técnica com emprego crescente. A figura 1.4 mostra um diagrama de uma rede de computadores aplicada ao controle e supervisão de processos de produção.

Um número significativo de fornecedores, usuários, grupos e comitês de padronização e departamentos governamentais de diversos países estão envolvidos na criação de um padrão para redes que interligam equipamentos operando no chão de fábrica a equipamentos de controle/automação de nível mais elevado operando em uma área de controle.

A busca desse padrão é importante, na medida que um padrão bem estabelecido e aceito universalmente traz para

usuários e fornecedores redução de custos de fabricação, instalação e manutenção. A redução nos custos de fabricação dos dispositivos, devido à padronização, se deve à utilização de um mesmo conjunto de "chips" pelos fornecedores. A redução dos custos de instalação se deve à redução do trabalho de especialistas, que não terão de resolver caso a caso os problemas dos usuários, especialmente a interconexão de sistemas com protocolos diferentes, e a redução no custo da manutenção se deve à redução no número de componentes de reposição a serem estocados e ao treinamento dos técnicos.

Figura 1.4: Diagrama de uma rede de computadores aplicada ao controle e supervisão de processos de produção.

1.9 Agropecuária e as redes industriais de instrumentos

A tecnologia de redes locais de instrumentos, semelhante àquela empregada na indústria, poderá ser utilizada em sistemas de telemetria aplicados à agropecuária, com possíveis adaptações devido às diferenças dos processos e dos ambientes de operação.

Um aspecto relevante da utilização da tecnologia de redes industriais na agropecuária é a possibilidade de contar-se com equipamentos desenvolvidos para um mercado em que preço, qualidade, disponibilidade, confiabilidade, suporte de serviço e aplicação são requisitos importantes.

2. Proposição

O objetivo do trabalho é desenvolver um sistema de telemetria aplicado à agropecuária. O sistema deve permitir a transmissão de dados sem fios entre estações de dados operando no ambiente agropecuário e equipamentos retransmissores ou de controle e supervisão. A figura 2.1 mostra um diagrama de blocos em que o objetivo do trabalho é ressaltado.

Figura 2.1: Diagrama de blocos com o objetivo ressaltado.

São impostas severas restrições ao consumo de energia elétrica, para permitir que o equipamento seja alimentado por baterias, sem que as mesmas recebam manutenção com muita frequência, ou ainda, permitir o emprego

de células fotovoltaicas de baixa potência.

As estações de dados devem se comunicar sem fios com uma estação repetidora, operando sem restrições severas de consumo de energia.

3. Revisão da literatura

Os sistemas de telemetria evoluíram passando por tecnologias como transmissão pneumática, transmissão elétrica analógica, até atingir-se atualmente a utilização de tecnologia de redes de computadores.

3.1 Padrão 4-20ma

Um padrão de comunicação analógico, amplamente utilizado pela indústria de processos para transmitir dados entre uma área de produção e uma área de controle da produção é o padrão de corrente 4-20mA. A figura 3.1 mostra um diagrama de blocos de um sistema de telemetria empregando o padrão 4-20 mA.

Uma vantagem importante deste padrão, além da simplicidade, é a intercambiabilidade entre equipamentos de diferentes fornecedores.

A principal desvantagem do padrão 4-20mA é a transmissão em apenas uma direção. Ainda apresenta suscetibilidade a ruídos e resolução relativamente limitada. A

necessidade de um cabo para cada variável a ser transmitida torna elevado o custo de instalação e manutenção.

Figura 3.1: Diagrama de blocos do padrão 4-20 mA.

LINDNER (1990) afirma que a resolução oferecida pelo padrão 4-20mA não é suficiente para instrumentos com tecnologia digital.

3.2 Comunicação digital

Para explorar as potencialidades dos instrumentos microcontrolados, é necessária comunicação bidirecional imune a interferências, e na grande maioria dos sistemas encontrados na literatura, é utilizada comunicação digital.

A comunicação digital permite que um instrumento realize medidas de vários parâmetros, e transmita todos os dados através de um único canal de comunicação. Somando-se esta capacidade ao compartilhamento do mesmo meio de comunicação por vários instrumentos, pode-se reduzir o custo

da instalação do sistema, especialmente no caso de sistemas que utilizem cabos como meio físico de comunicação.

3.2.1 Migração da comunicação analógica para a digital

As indústrias usuárias de sistemas de telemetria e telecontrole estão, desde meados da década de 80, migrando da comunicação analógica para a comunicação digital.

A transição iniciou-se em 1983, com o surgimento de instrumentos digitais que transmitem dados de acordo com o padrão 4-20mA, ao qual é superposto um sinal digital FSK na faixa de frequência de voz, conforme relata KAZAHAYA (1987).

3.2.2 Comunicação digital e informações secundárias

A comunicação digital, além de permitir a transmissão dos dados dos sensores com imunidade a ruídos superior à comunicação analógica, também permite o acesso a dados referentes a outras informações relacionadas ao instrumento.

O instrumento pode transmitir a um supervisor dados a respeito de resultados de auto-diagnósticos, trazendo informações sobre seu estado operacional e condições de operação. Estas informações secundárias podem ser utilizadas na manutenção do sistema.

3.3 A padronização das redes industriais de instrumentos

No campo ou chão de uma fábrica são utilizados dispositivos como sensores, atuadores, controladores de campo, mostradores de campo e analisadores de processos.

A padronização dos protocolos de comunicação é essencial para a intercambiabilidade de instrumentos de diferentes fornecedores.

ZEBERMANN et al. (1990) e GATER (1988), afirmam que protocolos para redes industriais como "MAP", "PROWAY", não atendem os requisitos da comunicação de campo, tornando necessário o desenvolvimento de outras tecnologias. Comitês de padronização foram estabelecidos para criar um padrão, designado "fieldbus", nome formado pela junção das palavras "field" (campo, ou chão de fábrica), que segundo LINDNER (1990), é o nível mais baixo de uma indústria de processos, e "bus" (barramento): uma topologia de redes de computadores caracterizada por sua simplicidade.

Exemplos de trabalhos com "fieldbus", aplicados à indústria, realizados no Brasil são PELLE et al. (1992), MARTIN (1992) e NAZAR (1993).

Os usuários tem muito interesse na portabilidade das técnicas de controle de processos, de acordo com CARO (1989), e SÁ (1988).

Esta portabilidade proporciona a liberdade de escolher os sensores e configuração de sistemas que possam suprir melhor suas necessidades, independentemente do fornecedor.

Um padrão internacional de caráter aberto, que permita o mesmo grau de intercambiabilidade do padrão 4-20mA, vem sendo defendido em publicações técnicas dirigidas aos usuários de sistemas de telemetria/telecontrole.

Entre os argumentos utilizados, além da liberdade de escolha dos usuários, tem-se: a) elevação do custo dos equipamentos se os fornecedores implementarem várias versões de um mesmo produto; b) elevação do custo de manutenção se os usuários implementarem mais de uma versão de rede em suas fábricas.

3.3.1 A padronização de redes industriais para o chão de fábrica

Segundo LINDNER (1990), WOOD (1988), ZEBERMANN et al. (1990), desde meados de 1985 surgiram diversos grupos e comitês nacionais e internacionais buscando a padronização da comunicação entre dispositivos de campo. Estes grupos e comitês envolvem fornecedores, usuários e departamentos governamentais norte-americanos, ingleses, franceses, alemães e japoneses.

O número de entidades interessadas na

padronização das redes industriais de chão de fábrica é um indicativo da importância desta tecnologia, e pesquisas de mercado, como relata SCHUUR (1991), mostraram em 1986 à Philips Components (Eindhoven), que circuitos integrados para interfaces de redes industriais de chão de fábrica poderiam tornar-se um nicho explorável no mercado de componentes, desde que um padrão mundial, com somente algumas versões fosse estabelecido.

3.3.2 Dificuldades da padronização

O número de organizações e empresas envolvidas com a padronização tornou-se muito grande. LINDNER (1990), e MACZKA (1991), relatam que isto causou o surgimento de choques de interesses que trouxeram dificuldades à obtenção de soluções de consenso.

Além disto, segundo CROWDER (1991), tentou-se obter um padrão muito abrangente, que atenderia aos requisitos de áreas de aplicação diferentes como automação e controle de processos, gerando muitas controvérsias.

Estes problemas tornaram a padronização um problema intratável, fazendo com que cada empresa buscasse favorecer seus próprios produtos, ou formar grupos com empresas de interesses comuns, conforme afirma KIM (1992).

3.3.3 O que esperar da padronização

A padronização das redes industriais de chão de fábrica é um interesse tanto de usuários como de fornecedores, e poderá ocorrer pelo consenso, por ações normativas de departamentos nacionais, ou pela competição entre fornecedores. Um padrão com poucas variações para atender aos requisitos das diferentes aplicações é a solução ideal.

Segundo CARO (1989), o objetivo final será alcançado quando o usuário puder se concentrar na solução dos problemas do processo ao invés dos problemas de instrumentação.

3.4 Meios físicos e mecanismos de transmissão

Os meios físicos utilizados normalmente em telemetria são linhas de transmissão como cabos, fibras ópticas, linhas telefônicas ou de transmissão de energia, ondas eletromagnéticas propagando-se na atmosfera ou vibrações mecânicas propagando-se pela água.

A atmosfera é a solução mais empregada quando a utilização de cabos é impossível ou inconveniente, em aplicações que requerem mobilidade ou flexibilidade e rapidez na instalação, ou ainda aplicações de caráter temporário.

Além disto, uma solução sem fios pode eliminar desvantagens associadas aos cabos como grande emprego de

mão de obra para instalação, manutenção cara e demorada, proteção contra raios de difícil solução, sujeição a depredações, roubos, acidentes como queimadas, ação de roedores.

Em aplicações agropecuárias, a estas desvantagens somam-se a movimentação de máquinas e trabalhadores "dentro" do processo no qual são efetuadas as medidas.

Os sistemas encontrados na literatura que utilizam a atmosfera como meio de transmissão, são baseados em laser, radiação na faixa do infravermelho e rádio.

3.4.1 Laser e infravermelho

Equipamentos que se comunicam através da propagação do laser e do infravermelho através da atmosfera tem como principal vantagem a não necessidade de licença para operar.

Uma das principais desvantagens do laser e do infravermelho como mecanismos de transmissão é sua elevada suscetibilidade às condições atmosféricas.

Ainda, necessitam de linha de visada totalmente desobstruída, o alinhamento entre receptor e transmissor é um fator crítico, há uma alta sensibilidade à luz solar direta, e o alcance é relativamente limitado.

O alcance limitado obriga, em muitas aplicações, a utilização de várias estações repetidoras. A adoção desta

solução leva à inoperância de todo o sistema em caso de falha de uma das estações.

Para aumentar o alcance ou confiabilidade do enlace, pode-se aumentar a potência óptica do feixe, mas este procedimento traz o aumento no custo, riscos à segurança e aumento no consumo de energia.

A propagação do laser ainda enfrenta sérios problemas com atenuação e perturbações do feixe quando o mesmo passa através de atmosfera turbulenta, contaminada e não homogênea, como em casos de chuva e neblina, mudanças no índice de refração do ar com a temperatura e turbulência causada por correntes de convecção, como afirmam ALLISON & LAND (1989).

3.4.2 Rádio

Os enlaces de rádio em telemetria são utilizados em grande parte dos trabalhos de telemetria sem fios encontrados na literatura.

De acordo com NEVE (1990), o surgimento do rádio digital como uma tecnologia madura e a sua utilização nos setores de consumo, comercial e industrial, acentuou o interesse de sua utilização em telemetria.

FLATMAN & WEISER (1992), afirmam que, devido ao alcance e à facilidade de uso, o rádio é o meio mais

apropriado para a maioria das redes de computadores sem fio em sistemas tanto dentro como fora de construções.

Uma das desvantagens do rádio é que os equipamentos devem obedecer normas estritas de utilização, homologação e/ou licenciamento, que são diferentes em cada país.

3.4.2.1 Faixa de freqüências

A faixa de freqüência a ser utilizada, é fundamental para o projeto de um enlace de rádio. Além de atender aos requisitos técnicos, deve obedecer as leis e normas de utilização para que os equipamentos recebam homologação e/ou licença de operação.

A freqüência a ser utilizada deve ser alta o suficiente para permitir o emprego de antenas pequenas com boa eficiência, e baixa o suficiente para não ser limitada de forma severa pela linha de visada.

Ainda não deve interferir ou sofrer interferências de serviços já existentes.

3.4.2.2 Modulação

A escolha do tipo de modulação é fundamental para o projeto de sistemas de comunicação baseados em rádio. Pode-se escolher entre modulação em amplitude (AM), modulação em freqüência (FM) e modulação em fase (PM), cada

uma proporcionando um nível de eficiência, ao custo de complexidade dos circuitos empregados.

A modulação FSK não é tão eficiente como PSK no que se refere à utilização de potência e largura de faixa, mas encontra ampla aplicação em transmissão de dados de baixa velocidade, de acordo com SHANMUGAM (1979).

Suas principais vantagens são as simplificações de "hardware", que resultam principalmente da utilização de demodulação não coerente e da relativa facilidade de se gerar os sinais.

Tais simplificações levam a uma diminuição no número e/ou complexidade dos componentes, o que contribui para diminuir os custos e/ou o volume do equipamento.

3.5 Legislação sobre radiotransmissão

O espectro de rádio frequência é um recurso natural limitado e de alto valor, e sua utilização é regulamentada por órgãos internacionais e governamentais.

Os dispositivos que radiam nesta faixa devem obter homologação e/ou licença para o seu funcionamento.

NEVE (1990), afirma que devido às restrições à largura de faixa, a utilização de rádio em telemetria e telecontrole deve utilizar um protocolo otimizado para as aplicações comuns de varredura de instrumentos para medidas,

alarmes e controles.

Os canais alocados para este serviço devem apresentar uma estrutura celular, para permitir a reutilização de frequências. Mas não se deve permitir a utilização de equipamentos móveis.

3.5.1 Legislação sobre radiotransmissão no brasil

Em BRASIL (1994), tem-se a Norma de Atribuição de Faixas de Frequências no Brasil, atualizada até 31/12/93, com a Tabela de Atribuição de Faixas de Frequências, fazendo citações das normas para cada faixa.

De acordo com BRASIL (1991) em torno de 415MHz foram destinados nove canais de 25kHz de largura de banda para comunicação de dados de baixa velocidade.

As correspondências Públicas, Privadas e Oficiais podem compartilhar o uso desta frequências, desde que uma estação nova não cause interferência em outra já existente.

Ainda podem ser empregados os 28 canais ida e volta alocados em torno de 928MHz e 940MHz, que segundo BRASIL (1990), são destinados a distribuição automática de dados via rádio visando o controle e supervisão de processos.

3.5.2 Legislação em outros países

Na Grã-Bretanha, conforme relata HICKMAN

(1993), uma faixa em torno de 418 MHz foi destinada à utilização por enlaces de rádio de baixa potência para telemetria de uso geral e uma faixa em torno de 458.MHz foi destinada à utilização por enlaces de rádio de baixa potência para telemetria de uso industrial e comercial. A Alemanha destinou uma faixa em torno de 433MHz para uso por enlaces de rádio de baixa potência para telemetria.

Estas regulamentações causaram o surgimento no mercado de equipamentos de baixo custo, com baixo consumo de energia e pequeno volume.

3.5.3 A legislação brasileira e a parte prática deste trabalho

Existe uma similaridade de destinações e uma relativa proximidade entre as frequências, entre as normas brasileiras e de outros países.

A utilização dos equipamentos disponíveis comercialmente atendendo às normas de outros países para o desenvolvimento dos protótipos deste trabalho foi considerado uma solução razoável, visto que serão implementados apenas algumas unidades.

Os boletins técnicos da RADIOMETRIX (1992a e b) informam que a empresa fornece sob encomenda módulos transmissores e receptores em frequências especificadas pelo usuário.

Desta forma, a implementação em escala comercial de sistemas como o descrito neste trabalho poderá ser realizada obedecendo a legislação brasileira com equipamentos de rádio comunicação fabricados no Brasil ou sob encomenda no exterior.

3.6 Proteção contra surtos de tensão

A utilização de equipamentos baseados em microprocessadores traz como principal desvantagem a perda da confiabilidade, como afirmam CLARK (1991), e STRINGFELLOW (1994).

Isto ocorre devido à sensibilidade dos dispositivos de silício, que são facilmente perturbados ou destruídos por surtos de tensão.

A função da proteção contra surtos de tensão é limitar as tensões em linhas de alimentação e de sinais, e desviar os surtos de corrente dos dispositivos sensíveis.

Segundo STRINGFELLOW (1994), as causas externas dos transientes de tensão são raios e chaveamentos de linhas de transmissão pelas concessionárias.

Uma falha em um componente, mesmo de baixo custo, significa descontinuidade na operação do sistema, deslocamentos de equipamentos de e/ou para manutenção e o emprego de mão de obra especializada, com um custo total que

pode ser muitas vezes maior que o custo do componente defeituoso.

A proteção de equipamentos de campo contra surtos de tensão não deve ser vista como algo secundário, deve ser levada em consideração em todas as etapas do projeto e em todas as partes do sistema.

3.6.1 Equipamentos de campo e os surtos de tensão

GATES et al. (1992), relatam que até 1992, não havia normas aprovadas nos EUA para testar controladores digitais de ambiente para produção animal com respeito à transientes de tensão, e ainda afirmam que o problema da proteção contra transientes não recebeu atenção adequada pelos fornecedores de controladores para aplicações pecuárias.

3.6.2 Severidade dos surtos

Um raio atingindo diretamente equipamentos eletrônicos tem efeitos destruidores. Um sistema de proteção contra raios procura controlar o ponto de atração do raio e oferecer um caminho de baixa impedância para a corrente do raio.

Um evento muito mais freqüente, é um raio atingir locais próximos ao equipamento, quando então são induzidos valores elevados de corrente por indução ou por elevação do

potencial de terra.

De acordo com CLARK (1989), os níveis de tensão produzidos em um cabo devido aos raios são relacionados diretamente à distância entre o raio e o cabo, à intensidade do raio e ao comprimento do cabo.

Segundo KOGA & MOTOMITSU (1985), os surtos de tensão causados por raios em cabos são aproximadamente proporcionais ao tamanho do cabo até 1 km, com pouco incremento com o comprimento do cabos após 2 km.

MASTER et al. (1985), apud CLARK (1991), relatam um caso em que um raio à 2,5 km de distância induziu em um fio suspenso de 460m uma tensão de 150.000V, correspondente a uma tensão de 326 V/m

BENNISON et al. (1973) afirmam que 99,8% dos surtos devidos à raios podem ser simulados em cabos de pares utilizando-se uma onda com 1000V de pico, tempo de subida de 10 μ S e tempo de decaimento de 1mS.

3.6.3 Aterramento

O aterramento é parte fundamental do circuito de proteção. Deve oferecer um caminho preferencial para desviar os surtos de correntes, induzidos em cabos, evitando correntes elevadas através do equipamento a ser protegido.

Muitos dos trabalhos publicados em que há um

detalhamento do sistema não se referem ao aterramento. AHMED & AL-AMOUD (1993) citam o aterramento como importante para eliminar ruídos durante a comunicação.

As correntes devidas aos surtos tem valor elevado, com altas taxas de crescimento. Segundo STRINGFELLOW (1994), apenas cravar hastes metálicas para atingir solo húmido ou bom condutor resulta em um aterramento com indutância elevada.

Uma forma usual de aterramento é uma combinação de hastes metálicas cravadas e condutores enterrados horizontalmente, ou ainda chapas metálicas enterradas.

3.6.4 Centelhadores a gás

Os centelhadores a gás suportam surtos de milhares de amperes, mas não disparam com surtos de tensão abaixo de 400 a 800V, como é relatado por CLARK (1989).

Devido ao tempo de atraso até a formação do arco voltaico entre os eletrodos do centelhador, da ordem de alguns microsegundos, o equipamento fica desprotegido durante o início do transiente, de acordo com GENERAL ELETRIC, (1978),

3.6.5 Varistores

Os varistores de óxido metálico, segundo CLARK (1989), suportam surtos de corrente relativamente elevados em relação aos supressores de transientes de silício, mas não são ceifadores de surtos de tensão tão eficientes quanto os supressores de transientes de silício.

3.6.6 Supressores de transientes de silício

Os supressores de silício são ceifadores quase ideais e ainda apresentam um tempo de resposta da ordem de sub-nanosegundos, porém possuem capacidade de dissipação de energia relativamente limitada, conforme relatam CLARK (1989) e GENERAL ELETRIC (1978).

3.6.7 Proteção com múltiplos estágios

A solução usual para a proteção de dispositivos de baixa tensão e alta sensibilidade a surtos, de acordo com CLARK (1989), é utilizar centelhadores, VDRs e supressores de surtos de silício em cascata, compondo um circuito de proteção com múltiplos estágios, como mostra o diagrama da figura 3.2.

Figura 3.2: Diagrama de blocos de um sistema de proteção em cascata.

3.7 Proteção contra temperatura e umidade

As condições de temperatura e umidade do ambiente agropecuário, devido à insolação e precipitação, entre outros, são severas para equipamentos com dispositivos semicondutores.

Além de contar com um abrigo coberto, sempre que possível, o encapsulamento dos circuitos do sistema deve prover proteções que permitam que o sistema apresente nível de confiabilidade suficiente.

As dimensões do equipamento é um fator crítico. Quanto menor, mais fácil é a proteção contra condições ambientais adversas.

3.7.1 Proteção contra umidade

A prática mais comum para proteger equipamentos que operam em ambiente agropecuário contra umidade, segundo

TANNER (1990), é encapsular as placas de circuito impresso do sistema em caixas seladas, com conectores de cabeça circular para todos os sensores e interfaces de comunicação.

3.7.2 Proteção contra temperatura

Os ciclos térmicos afetam negativamente a vida útil dos componentes eletrônicos. A expansão diferencial nas partes de um componente causa tensões mecânicas que podem levar à quebra ou à separação de junções coladas, como afirma SANGWINE (1987).

Para proteger o equipamento contra os ciclos térmicos, é necessário utilizar uma camada de material isolante em torno do equipamento.

Para proporcionar uma "inércia" térmica, diminuindo-se a amplitude dos ciclos térmicos, pode-se colocar no interior do compartimento isolado termicamente um corpo com capacidade calórica relativamente elevada.

3.8 Velocidade de transmissão

A velocidade de transmissão é fundamental para o projeto do sistema objeto deste trabalho. Maiores velocidades de transmissão requerem mais energia, e aumentam o consumo dos equipamentos. Assim deve-se escolher a velocidade de transmissão criteriosamente.

A velocidade necessária para o sistema pode ser determinada pelo número de bits gerado por unidade de tempo. O número de bits gerado depende da quantidade de sensores e ou atuadores a serem varridos, pela resolução das quantidades associadas aos sensores e ou atuadores. A esse número devem ser adicionados os bits necessários para a transmissão de cabeçalhos de mensagens, comandos e endereços das estações.

As velocidades de transmissão dos sistemas descritos na literatura vão de algumas centenas de baud até 9600 baud.

FEUER (1986), afirma que a velocidades de 300 a 2400 baud são suficientes para praticamente todas as aplicações em produção agropecuária.

3.9 Trabalhos desenvolvidos na área

LIMA et al. (1983), descrevem um sistema de telemetria em que estações remotas microprocessadas comunicam-se com uma estação microprocessada local. O meio de comunicação é rádio, modulação em amplitude (ASK), "half-duplex". O controle de acesso ao meio é do tipo "polled", com um mestre e até quatro escravos, com multiplexação por divisão no tempo. Os receptores das estações remotas devem permanecer energizados aguardando o recebimento de convites-consultas. O endereço das estações remotas é digital, sendo

transmitido na forma de um trem de pulsos durante o "polling", ao passo que os dados das estações remotas são transmitidos analogicamente por modulação por largura de pulso (PWM). A faixa de frequência utilizada pelo sistema está em torno de 27MHz, e o sistema é indicado para "curtas distâncias".

CROMER & McLENDON (1984), descrevem um sistema de telemetria empregando comunicação digital, composto por uma estação de campo e uma estação de base, sendo utilizado para medir água no solo. O meio de comunicação empregado é rádio e o receptor da estação de campo permanece energizado aguardando convites para transmitir. Utilizou-se modem FSK, na faixa de voz. O controle de erros é feito com um bit de paridade para cada 8 bits, e em caso de detecção de um erro, é solicitada nova transmissão. A taxa de amostragem é de 1 leitura a cada 15 minutos. O sistema é alimentado por baterias chumbo-ácido. São citados problemas com RFI.

HOWEL et al. (1984), descrevem uma rede computadorizada de coleta de dados de estações climatológicas para auxiliar em pesquisas de irrigação de culturas agrícolas. A rede é constituída por duas estações climatológicas e um computador em um centro de processamento interligados via linha telefônica. São medidos radiação solar, temperatura do ar e do solo, velocidade e direção do vento, precipitação. A taxa

de amostragem é de 1 leitura por minuto, das quais são calculadas as médias horárias e máximas e mínimas diárias para transmissão. A velocidade de transmissão é de 300 baud.

CULLUM & LEWIS (1985), descrevem uma estação de coleta de dados, que armazena em fita cassete, dados referentes à temperatura do ar e do solo, radiação solar, umidade relativa e velocidade do vento. A estação possui a possibilidade de empregar telecomunicação para transferência de dados, sendo alimentada por bateria e células fotovoltaicas. A taxa de amostragem utilizada é de uma leitura por hora.

HAYHOE & BALCHIN (1987), descrevem um sistema de telemetria digital aplicado à agricultura, medindo a temperatura do solo e detectores de congelamento. O meio de transmissão utilizado é rádio. O sistema é composto por uma estação remota e uma estação de controle. A estação remota é alimentada por baterias. Utilizou-se rádios "walkie-talkie", operando na faixa de UHF, adaptados a modems, com um alcance de até 40 km com antenas Yagi. A velocidade de transmissão é de 300 baud, com taxa de amostragem de 1 medida por hora, sendo que os dados são transmitidos uma vez ao dia.

ALLISON & LAND (1989) descrevem um sistema de telemetria microprocessada, empregando comunicação analógica. O sistema é composto por uma estação de base e

várias estações remotas, sendo desenvolvido visando aplicações em agropecuária. O meio de comunicação empregado é laser propagando-se na atmosfera. Apenas a estação de base possui uma fonte de laser, dirigindo-o a cada estação remota, que por sua vez, ao detectar o feixe de laser, aciona um "chopper" mecânico que modula o laser a ser refletido. O alcance atinge até 305m.

BANDYOPADHYAY et al. (1989) descrevem um sistema de telemetria empregando comunicação digital, para monitoração de nível de poluição em rio. O sistema é composto por uma estação flutuante e uma estação repetidora na margem do rio. As quantidades medidas na água são: condutividade elétrica, temperatura, potencial hidrogeniônico, e oxigênio dissolvido. O enlace entre a estação flutuante utiliza como meio de comunicação o rádio, na faixa de VHF. Não há controle de acesso ao meio, a estação flutuante transmite os dados dos sensores periodicamente, gravando simultaneamente os dados em fita cassete. A estação flutuante é alimentada por baterias chumbo-ácido. Utilizou-se transmissão assíncrona, com um "start-bit" e um "stop-bit" para cada um dos bits transmitidos.

COCKERHAM & ORTEGA (1989), descrevem um sistema de telemetria digital para operar em uma estação experimental agropecuária, para medidas de temperatura e velocidade do ar, potencial matricial do solo, estado de bomba

de água, fluxo de água em condutos. O meio de comunicação utilizado é radiação na faixa do infravermelho propagando-se na atmosfera. O sistema é composto por 11 estações de campo microprocessadas, dois repetidores e uma estação base, em um laço encadeado, em que cada estação passa adiante as mensagens recebidas. O controle de acesso ao meio é do tipo "polled", com um mestre e vários escravos. A alimentação das estações remotas é fornecida por baterias em conjunto com células fotovoltaicas. Foi implementado um esquema para diagnóstico do sistema, em que as estações de campo enviam mensagens automaticamente em períodos de aproximadamente 30 minutos. A velocidade de transmissão não é citada, mas é utilizada a interface serial RS-232 de um computador pessoal IBM-PC compatível.

FRANZOTTI (1990) desenvolveu um sistema de telemetria via rádio, para coletar dados que permitem avaliar a qualidade da água em reservatórios. São monitorados o oxigênio dissolvido, o potencial hidrogeniônico, a condutividade elétrica e a temperatura. São previstas coletas a intervalos programáveis em 20, 40 ou 60 minutos. A estação remota é montada sobre uma plataforma flutuante, alimentada por baterias e painéis de células fotovoltaicas.

LATIMER & REDDELL (1990), descrevem um sistema de irrigação automático que inclui um sistema de

telemetria baseada em comunicação digital. Três estações remotas de medida e controle comunicam-se com um computador em uma sala de controle. Utiliza-se dois meios de comunicação, rádio e infravermelho. O enlace de infravermelho é do tipo encadeado, com alcance de até 800 metros por lance. O enlace de rádio empregado possui alcance de até 400 m, e a frequência é 166 MHz. As estações remotas são alimentadas por baterias.

GRISMER (1992), descreve um sistema de telemetria digital para monitoração de salinidade, temperatura e umidade de solos em culturas agrícolas irrigadas. Cabos são utilizados como meio de comunicação, com um comprimento de 2 km, fios separados para energia e dados, e velocidade de transmissão de 1200 bauds. O sistema é composto por um barramento serial com 15 estações escravas remotas microcontroladas e uma estação mestre em uma sala de controle. O controle de acesso ao meio é do tipo "polled". O controle de erros é realizado com "check sum". Há um controlador de alimentação das estações remotas, na ausência de dados no barramento, as estações remotas são desligadas. A estação mestre é conectada à linha telefônica através de modem, comunicando-se com um centro de processamento de dados distante.

AHMED & AL-AMOUD (1993), descrevem um

sistema de telemetria e telecomando utilizando comunicação digital. O meio de comunicação utilizado é radiação na faixa do infravermelho propagando-se na atmosfera. O sistema mede potencial matricial do solo e controla uma bomba de água em uma cultura agrícola irrigada, em região de clima árido. O sistema é constituído por três estações escravas remotas e uma estação mestre de base. Alimentação por baterias chumbo-ácido. O controle de acesso ao meio é por "polling". A velocidade de transmissão dentro da faixa de 300 a 2400 baud. A taxa de amostragem é de uma leitura a cada 29 minutos. São citadas dificuldades com a propagação de infravermelho em condições atmosféricas desfavoráveis.

TORRE (1995) desenvolveu um sistema de telemetria, baseado em arquitetura serial multiponto. A transmissão de dados é realizada por enlace de rádio e linha telefônica discada, à velocidade de 9600 bauds. O sistema tem capacidade para ler até 240 sensores, sendo empregado para coletar dados climatológicos, temperatura do solo e potencial mátrico, a uma taxa de amostragem de 1 leitura a cada 10 minutos. São citados problemas com surtos gerados por raios.

4. Desenvolvimento

Buscando atingir os objetivos propostos, foram desenvolvidos protótipos de equipamento composto por um instrumento microprocessado e uma interface de comunicação empregando rádio. A esse equipamento, foi dada a designação *estação de dados*.

4.1 Estação de dados

As estações de dados desenvolvidas neste trabalho são constituídas basicamente por um instrumento microprocessado, fonte de alimentação e recursos de telecomunicações, encapsulados para operar sob as condições ambientais agropecuárias. A figura 4.1 mostra um diagrama de blocos de uma estação de dados.

A estação de dados deve operar junto ou próximo às atividades agropecuárias de interesse, comunicando-se sem fios com uma estação de controle/supervisão. A figura 4.2 mostra uma configuração possível para uma estação de dados, junto ao ambiente agropecuário.

Figura 4.1: Diagrama de blocos de uma estação de dados.

A antena deve permanecer a uma altura em que a vegetação, construções, movimentação de pessoas, animais e máquinas não causem interferência excessiva, tendo seu mastro aterrado eletricamente.

Os encapsulamentos do instrumento microprocessado e do transceptor devem oferecer proteção contra as condições ambientais adversas. No caso de utilização de bateria de chumbo-ácido, os encapsulamentos do instrumento microprocessado e da bateria devem ser separados, para evitar a ação de vapores ácidos sobre as partes metálicas dos circuitos.

4.1.1 Configuração da estação de dados

A estação de dados é dividida em duas partes separadas por cabos: uma delas com o transceptor de rádio e a antena, e outra com o instrumento microprocessado, sensores e bateria. Na figura 4.3 tem-se um diagrama de blocos mostrando como o sistema é dividido.

Figura 4.2: Configuração possível para uma estação de dados operando no campo: A) antena, transmissor/receptor de rádio; B) instrumento microprocessado; C) bateria; D) sensor.

Esta solução permite diminuir o comprimento do cabo de transmissão de RF, e aumenta-se o comprimento do par trançado, onde as frequências de trabalho são bem menores, e a incorporação dos componentes para proteção do sistema contra

surtos é mais fácil.

Torna-se possível, considerando-se as pequenas dimensões do rádio transmissor e do rádio receptor, instalar o transceptor junto à antena.

Figura 4.3:Diagrama de blocos mostrando como o sistema é dividido.

4.1.1.1 Interface entre o instrumento microcontrolado e o transceptor.

Foram construídas interfaces entre o instrumento microcontrolado e o transceptor, com as funções de permitir utilização de alguns metros de cabo de conexão para transportar

dados e energia, compatibilizar níveis lógicos e proteger contra surtos de tensão induzidos no cabo de conexão por raios.

Transmissão de dados entre o instrumento microprocessado e o transceptor, por meio de fio único ou par não trançado não são soluções satisfatórias. Estas soluções podem ser empregadas para pequenas distâncias, em sistemas relativamente livres de ruídos, como por exemplo comunicação entre placas de circuito impresso vizinhas, como informa TEXAS INSTRUMENTS (1985).

Pares de fios trançados são uma solução muito empregada, apresentando as seguintes vantagens:

- 1) Cancelamento do ruído devido ao acoplamento mútuo entre linhas adjacentes;

- 2) Ambos os fios são igualmente afetados por ruído acoplado magneticamente e eletrostaticamente, resultando em uma tensão em modo comum, que pode ser rejeitada por receptores com entradas balanceadas;

- 3) Diferenças de tensão entre os locais dos excitadores/receptores aparecem como sinais no modo comum, sendo rejeitados;

- 4) Impedância uniforme ao longo da linha torna fácil a colocação de terminações.

Cada condutor de conexão acrescentado ao cabo de interface deve necessariamente receber proteção. Para diminuir

o número de circuitos de proteção, buscou-se diminuir o número de condutores do cabo.

Empregando-se o padrão RS-485, a transmissão dos dados entre o instrumento microcontrolado e os módulos de rádio é feita por meio de um único par trançado. A figura 4.4 mostra um diagrama de blocos da estação de dados com as interfaces entre o transceptor e o instrumento microprocessado.

Figura 4.4: Diagrama de blocos da estação de dados, mostrando as interfaces entre o transceptor e o instrumento microprocessado.

Para alimentar os módulos transmissor e receptor com apenas um condutor além do “terra”, utilizou-se um discriminador de tensão. Esta solução também permite diminuir o número de circuitos de proteção necessários.

A figura 4.5 mostra um diagrama de blocos do sistema, detalhando as principais partes. Na parte superior da

figura, tem-se o transceptor de rádio e na parte inferior o instrumento microprocessado.

Figura 4.5: Diagrama de blocos da estação de dados, mostrando as principais partes do sistema.

4.1.2 Transceptor de rádio

O transceptor de rádio é composto pela antena, isolador de $\frac{1}{4}$ de onda, rádio transmissor, rádio receptor, discriminador de tensão de alimentação, interface RS-485, adaptador de níveis lógicos, regulador de tensão e proteção contra surtos de tensão. A figura 4.6 mostra um diagrama de blocos do transceptor de rádio e interface

Figura 4.6: Diagrama de blocos do transceptor de rádio e interface.

4.1.2.1 Antena e isolador de $\frac{1}{4}$ de onda

Para que o transmissor e o receptor utilizem a mesma antena, a entrada do receptor e a saída do transmissor foram isoladas através de uma linha de transmissão de comprimento de $\frac{1}{4}$ de onda, e de diodos que operam como chaves controladas pela tensão de alimentação do transmissor. Quando o transmissor é ativado, ambos os diodos operam como uma chave fechada, e o sinal do transmissor é enviado à antena. Quando o transmissor é desativado, ambos os diodos operam como uma chave aberta, e o sinal da antena é enviado ao receptor. O diagrama da figura 4.7 mostra como são conectados transmissor, receptor, isolador de $\frac{1}{4}$ de onda e antena.

Figura 4.7: Forma de conexão entre transmissor, receptor, isolador de $\frac{1}{4}$ de onda e antena.

Foram construídos dois tipos de antena: discone e Yagi.

A antena discone é composta por um disco e um

cone. O condutor central do cabo coaxial que alimenta a antena é conectado ao disco, e a malha ao cone. A figura 4.8 mostra uma ilustração da antena discone.

O diagrama de radiação da antena discone é omnidirecional no plano horizontal, sendo essencialmente semelhante ao de um dipolo vertical. Uma diferença significativa é que o diagrama sofre alterações com a frequência, de forma que as frequências mais altas são radiadas com maior intensidade em direção ao solo. Isto tem efeitos positivos, ao evitar a propagação de harmônicos gerados pelo transmissor de RF.

Figura 4.8: Ilustração da antena discone vista lateralmente.

Outras características desta antena é sua construção e alimentação simples, o que a torna recomendada para sistemas que requerem simplicidade de instalação e projeto mecânico, de acordo com KANDOIAN (1946).

NAIL (1953) relata a construção de antena discone à prova de água, em que a junção entre o disco e o cone é vedada com material isolante.

Os cálculos da antena discone utilizada neste trabalho foram realizados por MORAIS et al. (1994).

A antena Yagi emprega dipolo dobrado de comprimento $\lambda/2$, dimensionada para apresentar uma impedância de 200 ohms. O casamento com a impedância de 50 ohms do cabo coaxial foi obtida por meio de um transformador de impedâncias de $\frac{1}{2}$ onda “balun”, construído com cabo coaxial. A figura 4.9 mostra uma ilustração do “balun” empregado.

Construída com 5 elementos (refletor, dipolo e três diretores), esta antena proporciona um ganho aproximado de 8 dB, o que permite um aumento no alcance e/ou na confiabilidade. A figura 4.10 mostra uma ilustração da antena Yagi construída.

Figura 4.9: Ilustração do transformador de impedâncias ("balun").

Figura 4.10: Ilustração da antena Yagi com 5 elementos.

4.1.2.2 rádio transmissor e rádio receptor

Empregou-se um conjunto de rádio transmissor e rádio receptor projetados para compor enlaces digitais de baixa potência, operando em FSK, fabricados por RADIOMETRIX Ltd (1992a e b). Entre as características destes dispositivos pode-se ressaltar a estabilidade, o baixo consumo de energia e as pequenas dimensões.

Estas características tornam estes módulos particularmente atrativos para este trabalho. O baixo consumo de energia permite a utilização efetiva de fontes de energia sem fios, como baterias e células fotovoltaicas, e as pequenas dimensões facilitam o encapsulamento para a proteção contra as condições ambientais adversas.

A figura 4.11 mostra um diagrama de blocos do módulo transmissor, e a figura 4.12 mostra um diagrama de blocos do módulo receptor.

Figura 4.11: Diagrama de blocos do rádio transmissor TXM-418-A.

Segundo RADIOMETRIX Ltd (1992b) o rádio transmissor apresenta as seguintes características:

1) Frequência central da portadora 418MHz, estabilizada por ressonador SAW (circuito ressonante baseado em ondas acústicas de superfície).

2) 10Kbps de largura de banda de dados;

3) Faixa de tensões de alimentação de 6 a 12 V;

4) Corrente de alimentação de 10 mA;

5) Potência radiada (ERP) de -8 dBm

6) Dimensões de (30 x 10 x 6)mm.

Segundo RADIOMETRIX Ltd (1992a) o rádio receptor apresenta as seguintes características:

1) Frequência central da portadora 418MHz, estabilizada por filtro SAW;

2) 5 Kbps de largura de banda de dados;

3) Faixa de tensões de alimentação de 4.5 a 9V;

4) Corrente de alimentação de 14 mA;

5) Sensibilidade para relação sinal/ruído de 20 dB:

0,5 μ V;

6) Superheterodino com dupla conversão;

7) Dimensões de (48 x 20 x 6)mm.

A figura 4.13 mostra o diagrama esquemático do circuito com os módulos transmissor e receptor de rádio. O receptor e o transmissor foram montados sobre lados opostos de uma placa de circuito impresso, sendo que o receptor recebeu uma blindagem de cobre.

Figura 4.12: Diagrama de blocos do rádio receptor SILRX-418-A.

4.1.2.3 Discriminador de tensão de alimentação

O circuito discriminador de tensão tem a função de alimentar seletivamente os módulos receptor e transmissor, não permitindo que ambos sejam alimentados ao mesmo tempo. Este circuito é baseado em duas referências de tensão e comparadores. Para tensões abaixo de 7V, é ativado o receptor, e para tensões acima de 8V, é ativado o transmissor. O intervalo de 1V entre as “janelas” de tensão assegura que ambos não serão ativados ao mesmo tempo.

Figura 4.13: Diagrama esquemático do circuito com os módulos transmissor e receptor de rádio.

A figura 4.14 mostra o diagrama esquemático do discriminador de tensão. As tensões de referência são obtidas a partir de D11, R25, R27 e R26, que são comparadas com amostras da tensão de entrada obtidas através de R23 e R24. A saída dos comparadores, de configuração coletor aberto, ativarão os transistores Q7 ou Q8 seletivamente.

Figura 4.14: Diagrama esquemático do discriminador de tensão para o chaveamento da alimentação do transmissor/receptor de rádio.

4.1.2.4 Interface RS-485 e adaptador de níveis lógicos

A interface de conexão entre o coletor de dados e os módulos de rádio é baseada no padrão RS-485, sendo composta por dois circuitos: um junto ao instrumento microprocessado, e outro junto ao trancceptor de rádio. O diagrama esquemático do circuito que faz a adaptação de níveis lógicos e a comunicação baseada no padrão RS-485 é mostrado na figura 4.15.

A compatibilização dos níveis lógicos entre os sinais TTL do circuito excitador/receptor RS-485 e os sinais do transmissor e do receptor de rádio é feita pelos transistores Q6

e Q9. A direção dos dados é determinada pela tensão de alimentação do receptor junto aos pinos 2 e 3 do integrado U1.

Figura 4.15: Diagrama esquemático do adaptador de níveis lógicos e comunicação RS-485.

4.1.2.5 Regulador de tensão

O "chip" excitador/receptor RS-485 deve ser alimentado à partir de uma tensão de 5V a 6V. Para obter estes valores de tensão partindo de uma faixa de tensões de 6V a 9V, não é possível a utilização de reguladores integrados como os da família LM78XX, pois estes dispositivos necessitam de uma tensão de alimentação em torno de 2,3V superior à tensão regulada, de acordo com NATIONAL SEMICONDUCTOR (1982). O diagrama esquemático do circuito regulador de tensão utilizado é mostrado na figura 4.16.

Trata-se de um circuito convencional, onde o amplificador diferencial formado por Q4 e Q5 recebe na entrada não inversora uma referência de tensão gerado por R25 e D11, e na entrada inversora uma fração da tensão de saída, obtida pelos resistores R10 e R11.

Figura 4.16: Diagrama esquemático do circuito regulador de tensão de 5V, com mínima tensão de entrada limitada apenas pela tensão coletor/emissor de saturação de Q3.

4.1.2.6 Proteção contra surtos de tensão

O cabo entre o coletor e os módulos transmissor/receptor deve conduzir a alimentação e os dados do receptor e os dados.

Cada condutor do cabo deve necessariamente estar protegido contra surtos de tensão, pois os surtos induzidos em cabos devidos à raios podem chegar a algumas centenas de volts por metro de cabo.

Os circuitos de proteção, para que sejam eficazes, devem apresentar estágios em cascata, englobando centelhadores, varistores e supressores de silício, e assim apresentam dimensões relativamente grandes. Se o número de condutores a proteger for elevado o volume dos circuitos de proteção trará dificuldades no encapsulamento para proteção contra as condições ambientais adversas.

Para diminuir o número de circuitos de proteção, além de adotar-se apenas um par trançado para a transmissão de dados entre o coletor e os módulos de rádio, adotou-se apenas um condutor para alimentar tanto o módulo transmissor como o módulo receptor.

Figura 4.17: Diagrama esquemático do circuito de proteção contra surtos.

O primeiro estágio do circuito de proteção é

baseado em um centelhador, que é ativado por tensões a partir da faixa de 400-800V. O segundo estágio é baseado em varistores, com tensão de limiar de 12V, e o terceiro e último estágio é em um ceifador tendo por base um diodo zener.

Esta combinação de dispositivos de proteção busca reunir as melhores características de cada um deles, visando um circuito de proteção efetivo. A figura 4.17 mostra o diagrama esquemático do circuito de proteção contra surtos.

4.1.3 Instrumento microprocessado

O instrumento microprocessado é composto por um microcontrolador, memória de programa ("ROM"), memória de dados ("RAM"), relógio calendário e interface RS-485. A figura 4.18 mostra um diagrama de blocos do instrumento microprocessado.

O instrumento microprocessado pode efetuar leituras de sensores, controlar atuadores, armazenar e comunicar dados. A figura 4.19 mostra o diagrama esquemático do instrumento microprocessado, com o microcontrolador, memórias e interface RS-485.

Figura 4.18: Diagrama em blocos do instrumento microprocessado.

Figura 4.19: Diagrama esquemático do microcontrolador e memórias de dados e de programa.

4.1.3.1 Microcontrolador

Os microcontroladores são uma opção atrativa para proporcionar inteligência a um instrumento devido aos vários dispositivos de "hardware" incorporados em um único "chip", eliminando a necessidade de projeto e depuração de "hardware" convencional, associando pequeno volume, baixo consumo, e maior confiabilidade à flexibilidade de uso.

Figura 4.20: Diagrama de blocos do microcontrolador 80C515.

Neste trabalho, utilizou-se o microcontrolador SAB80C535, membro da família Intel (Santa Clara, Ca.) MCS8051, produzido por SIEMENS (1989). A figura 4.20 mostra um diagrama de blocos do microcontrolador SAB80C515, que difere do SAB80C535 apenas pela presença de memória ROM ("read only memory"), ausente no SAB80C535, que busca instruções exclusivamente em memória externa.

A família de microcontroladores de oito bits 8051 tem boa aceitação pelo mercado, sendo produzida por vários fabricantes de dispositivos semicondutores como ADVANCED MICRO DEVICES (1989), INTEL CORPORATION (1987), DALLAS SEMICONDUCTORS (1990), PHILIPS COMPONENTS (1985), SIEMENS, ATMEL.

Atualmente são oferecidos componentes da família 8051 com total compatibilidade de "software", tanto com mais recursos de "hardware", como com menos recursos que o "chip" original, visando abranger maior número de aplicações.

Devido ao sucesso comercial da família, um grande número de ferramentas para o desenvolvimento de sistemas baseados na mesma se encontra disponível no mercado, como montadores, simuladores, emuladores, compiladores de linguagens de alto nível.

Os fatores acima são importantes, pois indicam que os componentes provavelmente não terão sua produção

interrompida, ou produzidos em pequena escala. Isto tem um impacto positivo no custo dos dispositivos.

Entre as características do microcontrolador utilizado, tem-se:

- 1) 4 "ports" de entrada/saída de oito bits (dois "ports" são utilizados como barramentos de dados/endereços);
- 2) Três temporizadores /contadores de eventos de 16 bits;
- 3) UART;
- 4) 12 fontes de interrupção, com 4 níveis de prioridade;
- 5) Conversor AD com resolução de 8 bits, ou 10 bits com uma segunda conversão, com 8 entradas multiplexadas;
- 6) "Watchdog timer" de 16 bits;
- 7) Até 64K bytes de memória de dados.

Cada "port" de entrada/saída é constituído de oito pinos, que são utilizados para ler e controlar os dispositivos periféricos.

Os contadores/temporizadores podem operar como contadores de eventos externos, ou como temporizadores acionados pelo oscilador interno.

A interface de comunicação serial síncrona e assíncrona permite fácil interconexão com computadores.

A existência de um "watchdog timer" é uma

característica relevante deste membro da família de microcontroladores MCS8051. Quando a CPU é levada por interferências a uma seqüência não funcional, por tempo indefinido, o "overflow" do contador do "watchdog" causa um "reset" no microcontrolador, que por sua vez reinicializa todo o instrumento.

4.1.3.2 Memória de dados e programa

A memória de dados utilizada é do tipo estática CMOS, de baixo consumo quando em "standby", com capacidade de 32 Kbytes. A memória de programa é do tipo EPROM CMOS, com capacidade de até 64 Kbytes.

4.1.3.3 Relógio calendário

O relógio calendário utilizado é um integrado compatível com o barramento I²C, o PCF8583, PHILIPS COMPONENTS (1989). Um diagrama de blocos do relógio calendário é mostrado na figura 4.21.

O consumo do relógio calendário quando apenas o oscilador e os contadores estão operando, é tipicamente 10 μ A (valor máximo: 50 μ A).

O relógio calendário possui uma saída de sinal de alarme. O alarme é programável em horas, minutos e segundos. O diagrama esquemático do relógio calendário é mostrado na figura 4.22.

Figura 4.21: Diagrama de blocos do relógio calendário.

O sinal de alarme do relógio calendário é um dos sinais que pode ativar a fonte de alimentação do instrumento microprocessado. O acionamento da fonte pelo sinal de alarme permite que todo o instrumento microprocessado seja desligado, após o microcontrolador programar o alarme do relógio, sendo empregado para permitir o desligamento da estação de dados entre os "slots" de tempo de acesso.

O "chip" do relógio calendário ainda incorpora 240 bytes de memória RAM, que é utilizada para armazenar o "status" do instrumento. Assim, quando o instrumento microprocessado é "despertado" pelo relógio, ele vem buscar nesta memória as informações sobre as atividades a executar em

seguida.

Figura 4.22: Diagrama esquemático do relógio calendário.

4.1.3.4 Interface de comunicação RS-485

A interface de comunicação entre o instrumento microprocessado e os módulos de rádio é baseada no padrão RS485.

Este padrão permite que até 32 interfaces compartilhem um mesmo barramento, utilizando como meio físico uma linha de transmissão de dados balanceada. Normalmente utiliza-se como linha de transmissão um par trançado, terminado pela sua impedância característica. A figura 4.23 mostra um esquema do padrão de comunicação RS-

485.

Em cada ponto, tem-se usualmente um receptor e um excitador compondo um transceptor. O sinal no receptor deve apresentar uma tensão diferencial de pelo menos 200 mV.

Figura 4.23: Diagrama esquemático de uma interface digital balanceada multiponto baseada no padrão de comunicação RS485.

A interface entre o microcontrolador e o "chip" transceptor é feita através dos "ports" de entrada/saída disponíveis no microcontrolador. Um pino é utilizado como entrada de dados, outro como saída de dados e um terceiro pino como controle da direção dos dados.

4.1.3.5 Conversor A/D

O conversor A/D, incorporado ao “chip” do microcontrolador, é precedido por um multiplexador com oito entradas e um circuito amostrador/retentor. Um diagrama de blocos do conversor A/D é mostrado pela figura 4.20.

O conversor A/D utiliza o método de aproximações sucessivas, com 8 bits de resolução e tempo de conversão de 15 μ S (com cristal de 12MHz). Com uma segunda conversão, alterando-se as tensões de referência, pode-se obter uma resolução de 10 bits.

4.1.3.6 Interface entre o instrumento microprocessado e o transceptor

A interface entre o instrumento microprocessado e o transceptor realiza a comunicação via padrão RS-485 e a proteção contra surtos de tensão. Um diagrama de blocos da interface junto ao coletor de dados é mostrado na figura 4.24.

A interface junto ao instrumento microprocessado tem ainda a função de alimentar o transceptor com os valores adequados ao funcionamento do discriminador de tensão de tensão de alimentação.

A figura 4.25 mostra o diagrama esquemático da fonte de tensão ajustável, que é controlada pelo instrumento microprocessado, deve permitir enviar dois níveis de tensão aos

módulos de rádio. O sinal denominado ALIM. TX/RX habilita ou inibe a alimentação, enquanto que o sinal SELEÇÃO TX/RX permite seleccionar o valor da tensão entre dois valores, um em torno de 6V, para alimentar o rádio receptor, e o outro em torno de 9V, para alimentar o transmissor.

Figura 4.24: Diagrama de blocos da interface junto ao instrumento microcontrolado.

4.1.3.7 Proteção contra surtos de tensão

O circuito de proteção contra surtos de tensão empregado junto ao instrumento microprocessado é semelhante ao empregado no transceptor, cujo diagrama esquemático é mostrado na figura 3.17.

4.1.3.8 Fonte de alimentação

Para alimentar os protótipos desenvolvidos,

adotou-se bateria de chumbo-ácido. O consumo de energia é um fator crítico em sistemas alimentados por baterias, recebendo atenção no projeto de todo o sistema. Uma das principais providências tomadas para diminuir o consumo foi a utilização de um protocolo de comunicação adequado.

Figura 4.25: Diagrama esquemático da fonte de tensão ajustável controlada pelo instrumento microprocessado.

O protocolo utilizado busca diminuir o consumo através da adoção do método de multiplexação por divisão de tempo para controlar o acesso ao meio, permitindo que todo o instrumento, ou pelo menos o receptor de rádio permaneça desligado durante a maior parte do tempo.

O único dispositivo que permanece ligado, alimentado à partir de uma bateria dedicada, é o relógio calendário, que consome apenas alguns micro ampères.

A fonte de alimentação incorpora entradas de controle para que o relógio calendário, previamente ajustado pelo instrumento microprocessado, acione a mesma.

Durante a operação normal, a fonte do coletor de dados é ligada, pouco antes de chegar a janela de tempo da estação de dados, pelo sinal de alarme do relógio calendário.

Após o recebimento dos comandos e sua execução, o coletor de dados reprograma o relógio calendário para a próxima janela, e desliga a fonte.

A figura 4.26 mostra o diagrama esquemático da fonte da estação de dados.

Três sinais controlam o funcionamento da fonte: manual, o sinal de alarme do relógio calendário, e um sinal gerado pelo instrumento microprocessado. O controle manual será utilizado para a inicialização do sistema, o controle pelo sinal de alarme será utilizado para ativar a estação de dados

4.2 Protocolo de comunicação

As limitações de consumo de energia impostas como objetivo a ser atingido por este trabalho não permitem o emprego imediato de um dos padrões propostos para utilização com cabos, pois isto exigiria que os receptores das estações de dados permanecessem ligados, ou seja, consumindo energia.

Isto dificulta ou até inviabiliza a utilização de baterias como fonte de energia para as estações de dados remotas, pois as manutenções freqüentes necessárias para a recarga das baterias podem resultar em um dispêndio de mão de obra que, a princípio, o sistema busca minimizar.

A principal característica do protocolo de comunicação, que busca a diminuição do consumo de energia, é uma otimização do método de acesso ao meio.

O método de acesso ao meio utiliza multiplexação por divisão do tempo, com alocação fixa, sendo sincronizado pelo mestre. Utilizou-se uma adaptação do código POCSAG ("Post Office Code Standardization Advisory Group) utilizado em sistemas de telemensagens ("radio pagers"), como proposto em NEVE (1990).

Este método de acesso ao meio permite que as estações de dados permaneçam desligadas, ou pelo menos o receptor de dados, durante a maior parte do tempo, com conseqüente economia de energia.

O protocolo utilizado neste trabalho baseia-se no Modelo de Referência para Interconexão de Sistemas Abertos (OSI), proposto como padrão internacional pela ISO (International Standard Organization).

Figura 4.27: As sete camadas do modelo ISO/OSI

O modelo OSI é tomado por base de praticamente todos os padrões propostos para redes industriais de chão de fábrica.

4.2.1 Modelo OSI

O modelo OSI divide o processo de comunicação em sete camadas, atribuindo a cada camada funções, e definindo as interfaces entre as camadas. A figura 4.27 mostra um diagrama de blocos das camadas deste modelo, indicando as principais funções de cada uma.

O principal benefício desta divisão é a possibilidade de mudanças de implementação dentro de cada camada, desde que as interfaces acima e abaixo sejam mantidas.

4.2.2 Modelo OSI simplificado

Os protocolos propostos como padrão para redes industriais empregam apenas 3 camadas: camada física, camada de enlace de dados e camada de aplicação. As outras 4 camadas são eliminadas.

Figura 4.28: Modelo OSI simplificado.

Segundo DAMSKER (1988), isto é possível porque: a camada 3, camada de rede, é necessária somente se várias redes estão conectadas por meio de comportas ou pontes; a camada 4, de transporte, não é necessária porque as mensagens são muito curtas; a camada 5, de sessão, não é necessária em redes onde somente são utilizadas mensagens não conectadas; e a camada 6, de apresentação, é desnecessária porque a camada superior, de aplicação, garante a adoção de formato, sintaxe e código preestabelecido. A figura 4.28 mostra o modelo OSI simplificado.

4.2.3 Topologia da rede

A topologia normalmente empregada em redes industriais de chão de fábrica, normalmente é do tipo barramento.

Em um barramento, as mensagens são trocadas diretamente pelas estações, sem intermediários, não necessitando de roteamento ou armazenamento intermediário de mensagens.

Isto torna o barramento uma topologia de rede relativamente simples. Estas simplificações podem tornar a comunicação mais rápida, o que é desejável em aplicações em tempo real.

Figura 4.29: Conexão de dispositivos por meio de barramento.

Neste trabalho, devido ao método de acesso ser sincronizado pelo mestre, e que os escravos não comunicam-se entre si, a topologia da rede não deve ser necessariamente do tipo barramento. Apenas deve-se assegurar o enlace entre o mestre e cada escravo. A figura 4.29 mostra a conexão de

dispositivos, em que a topologia da rede é um barramento.

4.2.4 Camada física

A função da camada física é transmitir bits através de um canal de comunicação. Ela define qual é a representação física de um bit. Como descrito no item 3.1.2, utilizou-se o rádio modulado em FSK como meio de comunicação.

Utilizou-se um esquema de transmissão semelhante ao utilizado em um sistema de rádio-alarme produzido comercialmente pela Motorola, descrito por BERLIN (1983).

Figura 4.30: Esquema de transmissão NRZ, por duração de pulso.

Neste esquema, os dados são codificados utilizando duração de pulso. O nível lógico não é significativo, apenas o tempo entre as transições define se o bit recebido é '1' ou '0'. A figura 4.30 mostra o esquema de transmissão empregado. Utilizou-se para a modulação e demodulação em duração de pulso os contadores/temporizadores e interrupções

disponíveis no microcontrolador. O código das rotinas utilizadas para a transmissão e recepção encontram-se no apêndice.

Figura 4.31: O efeito do "jitter" no código utilizado. A) Neste caso, o "jitter" pode levar a duração dos sinais para um tempo inválido. B) Neste caso, o "jitter" pode causar erros na recepção.

4.2.4.1 Geração do código

Define-se um tempo t_0 associado ao bit '0' e um

tempo t_1 associado ao bit '1'. Ambas as frequências definidas pelos inversos de t_0 e t_1 devem ser inferiores à largura de banda de dados do sistema.

Para gerar o código de transmissão de uma palavra, pode-se seguir o seguinte algoritmo:

1- Mantém-se a linha sem transições durante um tempo t_0+t_1 , indicando um início da transmissão de uma palavra.

2- Inicia-se o quadro através de uma transição.

3- Espera-se um tempo t_0 ou t_1 , de acordo com o bit a ser transmitido, e executa-se uma transição no canal de comunicação.

4- Repete-se o passo anterior até que todos os bits tenham sido transmitidos.

A figura 4.32 mostra um fluxograma da geração do código utilizando largura de pulso.

4.2.4.2 Recepção do código

Na recepção, mede-se o período entre transições para decidir se o bit recebido é '0' ou '1', conhecendo-se o tempo de duração do pulso dos bits. Deve-se utilizar uma margem de tolerância que depende do "jitter" t_j admitido no sistema.

1- Se não ocorrem transições por um tempo superior a t_1+t_j , então não há dados na linha, e a recepção de

uma palavra será iniciada pela primeira transição.

Figura 4.32: Fluxograma da rotina para gerar o código por largura de pulso.

2- Se o tempo entre duas transições consecutivas for inferior a $(t_0 - t_j)$ ou superior a $(t_1 + t_j)$, o bit é inválido, e o quadro é descartado, aguardando-se o início de outro quadro.

3- Se o tempo de um bit válido é menor que $(t_0 + t_1)/2$, então o bit recebido é '0', caso contrário o bit recebido é '1'.

4- Quando forem recebidos 32 bits válidos, recebeu-se uma palavra, que será verificada pelo controle de erros.

A figura 4.33 mostra um fluxograma da rotina de recepção do código.

Com este esquema, elimina-se a necessidade de "start bit" e "stop bit", utilizados em transmissão assíncrona, bem como dos quadros de sincronização utilizados em transmissão síncrona.

A largura do pulso correspondente a cada bit pode ser utilizada para detectar erros de transmissão a cada bit recebido. Para que não ocorram erros devidos ao "jitter", a diferença entre a duração do bit '0' e '1' deve ser maior que o "jitter" admitido no sistema.

Na figura 4.31A, tem-se os valores mínimo e máximo para a duração dos pulsos, e valores fora desta faixa são considerados como erros de transmissão. A figura 4.31B, mostra como o "jitter" pode causar erros na recepção, ao tornar

o tempo de um sinal correspondente a um bit "0" mais longo e vice-versa.

Figura 4.33: Fluxograma da rotina de decodificação da linha.

Além disto, como o "jitter" não é cumulativo, a soma dos tempos de todas as larguras dos bits ao final de um quadro também pode ser utilizada para detectar erros em quadros.

Estas duas formas de controle de erro, duração do bit e duração da palavra diminuem a possibilidade da aceitação de mensagens geradas pelo recebimento de dados aleatórios.

Adotou-se a taxa de 1200 bits por segundo, que, como indicado pela revisão da literatura, é suficiente para a grande parte das aplicações em agropecuária.

4.2.5 Camada de enlace de dados

A função da camada de enlace de dados é controlar o acesso ao meio de comunicação, o endereço físico e tornar o canal de comunicação em uma linha aparentemente livre de erros de transmissão para a camada imediatamente superior.

A unidade de transferência de dados na camada de enlace de dados corresponde a uma palavra, composto de uma seqüência de 32 bits.

4.2.5.1 Controle de erros

As condições de propagação de RF são dinâmicas, e para obter-se um grau de confiabilidade aceitável quando o meio físico de comunicação apresenta imperfeições, torna-se necessário um controle de erros. A figura 4.34 mostra um

modelo simplificado de um sistema de comunicação.

Figura 4.34: Modelo simplificado de um sistema de comunicação.

O controle de erros deve permitir a operação segura do sistema em face ao "fading", recepção de dados aleatórios e interferências de outros sistemas operando alguns quilômetros distante.

Além dos erros que podem ser detectados por meio do esquema de transmissão empregado, por tempo de bit inválido ou tempo de quadro inválido, há o acréscimo de bits de redundância para detecção e correção de erros.

Utilizou-se o código BCH (Bose-Chaudhuri-Hocquenghem), que segundo WIGGERT (1988), são a classe mais poderosa

conhecida de códigos de bloco para correção de erros aleatórios.

O código BCH utilizado possui vetores de tamanho 31 e dimensão do espaço vetorial 21, portanto, com 10 bits de redundância. O vetor gerador, dado por STENBIT (1964), é dado por:

$$x = (3551) \text{ (octal)}$$

A capacidade de correção de erros do código BCH utilizado é de 5 bits, conforme afirma WIGGERT (1988),.

4.2.5.2 Método de acesso ao meio

O acesso ao meio é baseado em multiplexação por divisão no tempo síncrona com alocação estática (STDM), com um mestre e vários escravos, garantindo a todas as estações de dados acesso regular ao meio. Cada estação de dados transmite ou recebe somente quando seu intervalo de tempo chega.

Figura 4.35: Esquema de alocação de intervalos de tempo a cada estação de dados.

Uma vantagem de um barramento com um único mestre é a sincronização de todas as comunicações, com a temporização dos dados controlada precisamente. A figura 4.35 mostra o esquema de alocação de intervalos de tempo a cada canal de comunicação.

Segundo CARO (1989), a maior parte das aplicações requer a configuração de uma base de dados para varredura de sensores e atuadores. Para sistemas que requerem notificação rápida de eventos críticos, BERLIN (1983) sugere um protocolo híbrido, em que uma parte do tempo seja alocada para avisos das estações de dados.

A cada estação de dados remota (escravo) é atribuído uma janela de tempo fixa em relação a uma palavra de sincronismo.

Após a transmissão pelo mestre de um preâmbulo para sincronização, são transmitidos lotes de palavras de código de acordo com o número de mensagens a serem enviadas. A figura 4.36 mostra um diagrama da transmissão do preâmbulo de sincronização e dos lotes de palavras-código.

Estas palavras-código são empregadas no código POCSAG, como descreve DRUDE (1990).

Cada lote consiste de dezessete palavras de código de 32 bits de comprimento. A primeira palavra de código de um lote possui um padrão fixo e é utilizada para a obtenção de

sincronismo de palavra. Os dezesseis códigos seguintes são arranjados em oito quadros (numerados de 0 a 7) de duas palavras código cada. O número do quadro faz parte do endereço do usuário. Cada instrumento é alocado a um quadro.

Figura 4.36: Diagrama da transmissão do preâmbulo de sincronização e dos lotes de palavras-código.

São utilizados dois tipos de palavras código, uma para os endereços das estações de dados e outra para os dados. A palavra código de endereço é iniciada por um bit '0', que é seguido por 18 bits de endereço, 2 bits de função, 10 bits de CRC e um bit de paridade.

A palavra código de dados é iniciada por um bit '1', que é seguido de 20 bits de dados, 10 bits de CRC e um bit de paridade. A figura 4.37 mostra a estrutura das palavras

código.

Palavra Código Endereço	0	18 Bits de Endereço	2 Bits de Função	10 Bits CRC	P
Palavra Código Dados	1	20 Bits de Mensagem		10 Bits CRC	P

Figura 4.37: Estrutura das palavras código de endereço e dados.

4.2.6 Camada de aplicação

A camada de aplicação cuida de oferecer ao instrumento microprocessado os serviços de transmissão e recepção de dados com a estação de supervisão e controle.

4.2.7 Rotina principal

A rotina principal executada nas estações de dados remotas se encarrega de decodificar e executar os comandos recebidos.

Os comandos implementados são: a transmissão de dados de sensores acoplados às entradas analógicas do instrumento microprocessado; acerto do relógio calendário e transmissão de autodiagnóstico.

Se o equipamento de controle/supervisão falhar, a estação de dados deve levar o sistema a uma condição segura. A estação de dados pode detectar falha no mestre ou no enlace, verificando a ausência das palavras de sincronismo.

Para que os atuadores falhem com segurança, devem operar como multivibradores monoestáveis redispáráveis, com tempo de disparo superior ao período entre as mensagens de comandos; obedecer a comandos com tempo de início e fim determinados ou ainda as mensagens de comando devem ser repetitivas ou atualizadas ciclicamente, durante a execução do comando.

A figura 4.38 mostra um fluxograma parcial da rotina principal da estação de dados, que continua na figura 4.39.

A função de sincronização com a estação de controle/supervisão é atribuída à rotina principal, e não à camada de acesso ao meio. O sincronismo determina os períodos em que a estação de dados permanece desativada em uma aplicação de simples leitura de sensores. Entretanto, quando a execução de um comando requer a ativação em períodos específicos, percebe-se a importância da ativação e desativação ser controlada pela rotina principal.

Uma estrutura de dados armazena o status do sincronismo na memória não volátil disponível no “chip” do relógio-calendário.

Se o estado atual indica perda de sincronismo, a estação de dados busca pela palavra de sincronismo, durante no máximo o período de multiplexação completo. Em caso de

falha, a estação é desativada indefinidamente.

Quando a palavra de sincronismo é recebida, as variáveis de sincronismo são atualizadas, indicando que há sincronismo, e o alarme do relógio calendário é ajustado para a próxima ativação.

Se ocorrer falha na recepção da palavra de sincronismo devido a condições desfavoráveis temporárias no enlace, incrementa-se uma variável da estrutura de dados de sincronismo, que conta as falhas na recepção da palavra de sincronismo. Quando esta variável é igual ou maior que três, considera-se o sincronismo perdido, e inicia-se uma nova busca contínua pela palavra de sincronismo. O fluxograma da rotina de sincronização é mostrado na figura 4.39. Após o reestabelecimento do sincronismo, a variável que conta as perdas de sincronismo recebe o valor zero.

Figura 4.38: Fluxograma parcial da rotina principal da estação de dados

Figura 4.39: Fluxograma da rotina de sincronização.

4.3 Teste dos protótipos

O teste dos protótipos foi realizado em laboratório e em ambiente agropecuário.

4.3.1 Procedimento de teste

Montados os protótipos, o funcionamento dos mesmos foi testado em laboratório, por meio de troca de mensagens ininterruptamente durante 48 horas. Simples haste metálica foi utilizada como antena, os protótipos foram alimentados por fonte de laboratório, e não estavam encapsulados.

Após encapsulados, os protótipos foram testados em ambiente agropecuário. Durante o teste, as condições atmosféricas permaneceram boas, sem chuvas.

Dois tipos de antenas foram testados: antena do tipo yagi de 5 elementos (ganho aproximado de 8 dB) e antena do tipo discone, omnidirecional.

A um dos protótipos foi acrescentado um mostrador alfanumérico de cristal líquido.

As rotinas desenvolvidas para o teste colocaram uma das estações ecoando todas as palavras código recebidas. A outra estação recebeu a incumbência de gerar palavras, transmiti-las, esperar o “eco” e comparar com a palavra transmitida.

Após um levantamento inicial para estimar o alcance do enlace, as estações foram montadas a distâncias conhecidas, e as rotinas de teste executadas durante uma hora.

O total de palavras transmitidas, bem como o

percentual de erros foram monitorados pelo mostrador alfanumérico.

5. Resultado e discussão

Neste capítulo é apresentado e discutido o resultado do trabalho. O resultado foi dividido nos seguintes tópicos: Dispositivo de comunicação via rádio, Instrumento microprocessado, Protocolo de comunicação e Consumo de energia.

5.1 Dispositivo de comunicação via rádio

A interface de comunicação via rádio é composta de transmissor, receptor, interface entre antena, receptor e transmissor, antena, fonte de alimentação, discriminador de tensão de alimentação, proteção contra surtos de tensão e interface com o instrumento microprocessado. O resultado deste conjunto é relatado e discutido a seguir.

5.1.1 Alcance

Os protótipos construídos, operando nas condições de teste descritas no capítulo 4, item 4.3, empregando a antena Yagi e discone descritas no capítulo 4, item 4.1.2.1 na estação

remota e estação repetidora respectivamente, estabeleceram um enlace com alcance de 2,4 km, com uma taxa de erros de 1,5% em de 3600 transmissões/retransmissões, totalizando 7200 comunicações. Deve ser ressaltado que todos os erros foram detectados pela codificação de linha empregando modulação por duração de pulso.

A codificação de linha, empregando modulação por duração de pulso é realizada pelo microcontrolador, com o auxílio de seus contadores e interrupções. Uma característica deste método é que ele permite dois tipos de verificação de erros: se cada bit ou o conjunto de bits que forma a palavra não apresentaram duração que possa ser considerada excessiva ou insuficiente.

Nas mesmas condições de teste, para uma distância entre antenas de 2 km, a taxa de erro obtida nas condições de teste empregadas foi igual a zero.

Com antenas Yagi em ambas as estações repetidora, e distância entre antenas igual a 2,4 km, a taxa de erros obtida também foi nula.

O alcance obtido é suficiente para transportar os dados da área de produção para uma área onde esteja disponível a rede de energia elétrica. Um exemplo de aplicação é o sistema de irrigação por pivô central. Estações de dados lendo sensores dentro da área irrigada comunicam-se com o equipamento de

controle/supervisão por meio de uma estação repetidora junto ao pivô, que opera alimentada pela rede ac, sem restrições severas de consumo, completando o enlace até o local da estação de supervisão/controle.

5.1.2 Velocidade de transmissão

A velocidade de transmissão do sistema é variável, pois os símbolos zero e um representam tempos entre transições diferentes. Quando todos os símbolos são iguais a um, a velocidade de transmissão é igual a 1200 bits por segundo. Se todos os bits são iguais a zero, a velocidade de transmissão atinge um máximo de 2400 bits por segundo. Se a probabilidade de ocorrer símbolos zero for igual à probabilidade de ocorrer símbolos um, a velocidade de transmissão média vale 1800 bytes por segundo.

A velocidade de transmissão obtida é compatível com o volume de dados relativamente baixo gerado por variáveis ambientais.

5.1.3 Interface com o instrumento microprocessado

A interface entre os módulos que compõem a estação de dados permite minimizar o comprimento do cabo que alimenta a antena. O discriminador de tensão de alimentação permite alimentar e chavear entre transmissor e receptor com

apenas dois condutores. O discriminador de tensão ainda determina a direção dos dados através da interface RS-485. Assim são apenas quatro condutores entre os módulos, dois para alimentação/controle e dois para comunicação. Este pequeno número de condutores facilita a proteção do equipamento contra os surtos decorrentes de descargas atmosféricas próximas.

5.1.4 Dimensões

Como as estações de dados devem operar em ambientes hostis, uma característica interessante é que as dimensões do transceptor resultaram tais que permitiram seu encapsulamento em caixa de PVC vedada por meio de anel de borracha, de dimensões internas 154x110x70mm.

5.2 Instrumento microprocessado

O “software” do instrumento microprocessado foi desenvolvido parte em linguagem de montagem e parte em linguagem C. O emprego de linguagem de alto nível permitiu uma boa modularização das rotinas, facilitando o desenvolvimento e a depuração.

5.2.1 Principais características

O instrumento microprocessado montado apresenta as seguintes características:

- Capacidade de memória de programas (ROM): 64Kbytes;
- Capacidade de memória de dados (RAM): 32Kbytes;
- Oito entradas analógicas, com resolução de oito bits, podendo atingir 10 bits, com uma segunda conversão;
- Relógio-calendário com alarme de tempo real, com área de 240 bytes de memória RAM não volátil.

Dos 64 Kbytes disponíveis para programas, aproximadamente 16 Kbytes foram empregados para as rotinas básicas de comunicação, restando 48 Kbytes para as rotinas de aquisição/controle.

Dos 32 Kbytes disponíveis para dados, 1 Kbyte foi empregado pelas rotinas de comunicação, restando 31 Kbytes para armazenar dados.

5.2.2 Dimensões

O instrumento microprocessado, a fonte de alimentação e o circuito de proteção contra surtos de tensão foram encapsulados em uma caixa de PVC com vedação por meio de anel de borracha, normalmente empregada em instalações elétricas de dimensões internas 190x145x90 mm.

5.3 Protocolo de comunicação

O protocolo de comunicação desenvolvido, empregando alocação por divisão no tempo estática, permite até 59 estações por enlace, com taxa de amostragem de uma amostra por hora.

A escolha do protocolo é de fundamental importância para o consumo de energia do sistema. A alocação dos períodos de tempo a cada estação é sincronizada pela estação de controle/supervisão, e as estações remotas podem permanecer desativadas durante a maior parte do tempo.

Foi implementada detecção correção de erros empregando o código BCH (Bose-Chaudhuri-Hocquenghem). Suficiente para detectar e corrigir até 5 bits em posição aleatória dentro da palavra. Por final, os 31 bits é acrescentado um bit de paridade.

A proteção do equipamento contra erros de comunicação procura tornar o sistema imune à recepção de dados aleatórios durante falhas no enlace.

5.4 Consumo de energia

O consumo de energia é uma característica muito importante de equipamentos que operam alimentados por baterias e/ou células fotovoltaicas. Consumo elevado pode

significar baterias e células fotovoltaicas de alta capacidade, manutenções para recarga de baterias freqüentes, com repercussões negativas no custo e nas dimensões do equipamento.

A estação de dados, composta pelo instrumento microprocessado e interface de comunicação via rádio pode ser alimentada por uma fonte de tensão de valor entre 12 a 15 volts.

O instrumento microprocessado, com o tranceptor desligado, consome 68 mA.

O consumo máximo da estação, que ocorre quando o receptor de rádio é ativado, é de 175 mA. Considerando que a taxa de amostragem dos dados seja uma amostra por hora, e que o tempo de coleta/transmissão seja de 15 segundos para coleta e 15 segundos para transmissão, a estação de dados consumirá 0,58 ampères/dia. Este valor para o consumo permite a utilização efetiva de baterias, com número relativamente pequeno de manutenções para recarga.

6. Conclusão

Os resultados apresentados pelos protótipos, especificamente o consumo de energia, o alcance, o volume ocupado pelos circuitos, a velocidade de transmissão, a taxa de amostragem e o número de estações admitidas por canal indicam que o trabalho atingiu os objetivos propostos de forma satisfatória.

A escolha criteriosa de cada componente do sistema, bem como da configuração em que foram arrançados proporciona ao sistema desenvolvido as seguintes características:

1. O baixo consumo médio de energia permite o emprego de fontes de energia sem fio.
2. O alcance, relativamente baixo, é suficiente para transportar os dados da área onde os fios são inconvenientes para uma área onde a rede de distribuição de energia esteja disponível.
3. A velocidade de transmissão é compatível com grande parte das aplicações de telemetria e telecontrole em agropecuária.

4. A taxa de amostragem adotada é suficiente para o monitoramento de variáveis ambientais.

5. O número de estações permitida por canal permite uma utilização racional do espectro de radiofrequência.

6. O pequeno volume dos circuitos torna fácil sua proteção contra ambientes hostis.

7. A configuração adotada, permitindo que o transceptor permaneça próximo à antena, minimiza as perdas do sinal de RF, evitando perdas na sensibilidade e na potência radiada.

Sugestões para trabalhos futuros

O alcance do transceptor pode ser aumentado, construindo-se antenas com maior ganho, o que não causa aumentos no consumo de energia. Também a inclusão de um pré-amplificador precedendo o receptor, com pequeno aumento no consumo, ou ainda com um amplificador de RF após o transmissor, com aumento no consumo de energia.

O protocolo pode ser modificado, alocando “slots” de tempo para a transmissão de avisos pelas estações remotas.

7. Referências bibliográficas

AHMED, A.S.; AL-AMOUD, A.I. Infrared telemetry for data acquisition and telecontrol in automatic irrigation scheduling. Computers and Electronics in Agriculture, v.8, p.73-85, 1993.

ADVANCED MICRO DEVICES, 80515/80535 8-bit single-chip microcontroller, Sunnyvale, 1989, 35p.

ALLISON, J.M.; LAND, J.E. Laser-based communication system. Transactions of the ASAE, v.32, n.1, p.298-303, 1989.

BAKER, C. Surge protection for instrumentation. In: TEMPCOM '78, London, 1978, p.1-7.

BANDYOPADHYAY, S.; BHATTACHARYA, U.K.; NIOGI, S.; SOM, P.; PATRANABIS, D. Instrumentation for continuous in situ monitoring of water quality. IEEE Transactions on Instrumentation and measurement, v.38, n.3, p.820-3, 1989.

BENISON, E.; GHAZI, A.J.; FERLAND, P. Lightning surges in open wire, coaxial and paired cables. IEEE Transactions on Communications, v.21, n.10, p.1136-43, 1973.

BERLIN, M.A. 960 megahertz radio as a telemetry transmission medium. Advances in Instrumentation, v.38, pt.1, p.311-15, 1983.

BRASIL. Presidência da República. Secretaria Nacional de Comunicações. Portaria número 7, de 11 de janeiro de 1991. Diário Oficial da União, Brasília, 14 de janeiro de 1991, seção 1, p.929.

BRASIL. Presidência da República. Secretaria Nacional de Comunicações. Portaria número 105, de 09 de novembro de 1990. Diário Oficial da União, Brasília, 12 de novembro de 1990, seção 1, p.21551.

BRASIL. Presidência da República. Ministério das Comunicações. Portaria número 45, de 11 de fevereiro de 1994. Diário Oficial da União, Brasília, 16 de fevereiro de 1994, seção 1, p.2223.

CARO, R.H. Field bus applications. Advances in Instrumentation, v44, pt.3, p.989-94, 1989.

CLARK, O.M. Transient voltage threats and protection of communication interfaces. ASAE paper n.89-3036, 1989.

CLARK, O.M. Specifying and installing surge protectors for industrial computers at incoming power lines and data i/o ports. In: INDUSTRIAL COMPUTING CONFERENCE, Anaheim, California, 1991. Proceedings. Research Triangle Park, ISA, 1991. v.1, p.571-86.

COCKERHAM, S.T.; ORTEGA, T.R. Remote data acquisition for agricultural experiment station management. Applied Agricultural Research, v.4, n.2, p.144-7, 1989.

CROMER, W.A.; McLENDON, B.D. Microprocessor-based radio telemetry system for determination of soil moisture status. ASAE Paper n.84-1593, 1984, 19p.

CROWDER, R.S. Common fieldbus functionality for manufacturing & process applications. Advances in Instrumentation, v.46, pt.2, p.1847-55, 1991.

CULLUM, R.F.; LEWIS, C.E. Automated environmental data collection for research in remote locations. Agroborealis, v.17, n.2, p.19-23, 1985.

DALLAS SEMICONDUCTOR (Dallas, Texas), Product data

book. Dallas, 1990-1991, 936p.

DAMSKER, D.J. Assessment of industrial data network standards. IEEE transactions on Energy Conversion, v.3, n.2, p199-204, 1988.

DRUDE, S. A single chip decoder for the POCSAG radiopaging code. IEEE Transactions on Consumer Electronics, v.36, n.4, p.897-902, Nov. 1990.

FEUER, L. Infrared telemetry for agricultural applications. In: AGRIMATION, 2., CHICAGO, 1986. Proceedings... Chicago, 1986, p.207-13.

FLATMAN, A.V.; WEISER, P.G. Evolution off wireless networks. ICL Technical Journal, v.8, n.2, p.242-71, 1992.

FRANZOTTI, C. L. Sistema automático de medidas e aquisição de dados da qualidade da água em reservatórios. São Carlos, 1990. Tese (mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo.

GATER, C. Field bus. Measurement and Control, v.21, p.37-41, 1988.

GATES,, R.S.; TURNER, L.W.; OVERHULTS, D.G. Transient

overvoltage testing of environmental controllers.
Transactions of the ASAE, v.35, n.2, p.727-33, 1992.

GENERAL ELECTRIC, Transient voltage suppression manual.
2 ed. New York, 1978.

GRISMER, M.E. Field sensor networks and automated
monitoring of soil waters sensor. Soil Science, v.154, n.6,
p.482-9, 1992.

HAYHOE, H.N.; BALCHIN, D. Automatic retrieval of data
agrometeorological using a microcomputer and radio telemetry.
Computers and Electronics in Agriculture, v.2, p.129-34,
1987.

HICKMAN, I. Low power radio links. ElectronicsWorld &
Wireless World, n.1683, p.140-4, Feb. 1993.

HOWELL, T.A.; MEEK, D.W.; PHENE, C.J.; DAVIS, K.R.;
McCORMICK, R.L. Automated weather data collection for
research on irrigation scheduling. Transactions of ASAE,
v.27, n.2, p.386-391, 1984.

INTEL CORPORATION (Santa Clara, Ca), Embedded
controller handbook. Santa Clara, 1987. paginação
irregular.

KANDOIAN, A.G. Three new antenna types and their applications. Proceedings of the IRE, n.34, p.70w-75W, Feb. 1946.

KAZAHAYA, M. Field bus: New standard? Control Engineering, v.34, n.11, p.21-4, 1987.

KIM, I. Four new drivers for the fieldbus. Chemical Engineering, v.99, n.10, p.48, 1992.

KOGA, H.; MOTOMITSU, T. Lightning-induce surges in paired telephone subscriber cable in Japan. IEEE Transactions on Electromagnetic compatibility, v.27, n.3, p.152-61, 1985.

LATIMER, E.A.; REDDELL, D.L. Components for an advance rate feedback irrigation system. Transactions of the ASAE, v.33, n.4, p.1162-70, 1990.

LIMA, R.N.; FREIRE, R.C.S.; DEEP, G.S. Sistema de telemetria bidirecional. In: CONGRESSO SA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 3., Campinas, 1983. Anais. Campinas, UNICAMP, 1983. v.1, p.155-69.

LINDNER, K. Fieldbus: A milestone in field instrumentation technology. Measurement and Control, v.23, n.9, p.272-7,

Nov. 1990.

MACZA, W.J. Digital field communications set to ride SP50 bus. InTech, v.38, n.6, p.26-8, 1991.

MARTINS, J O fieldbus, ou barramento de campo, no contexto das redes industriais. EM. p.56-61, agosto 1992.

MORAIS, E.M.; OLIVEIRA, H.J.Q.; SARTORI, J.C. Cálculos de três antenas com frequência central de 418 MHz. São Carlos, EESC-Dep. Eng. Elétrica, 1994. (Relatório de Antenas - SEL170).

NAIL, J.J. Designing discone antennas. Electronics, p.167-169, Aug. 1953.

NATIONAL SEMICONDUCTOR CORPORATION (Santa Clara, Ca), Voltage regulator handbook. Santa Clara, 1982. paginação irregular.

NAZAR, R.F. Uso de rede de campo para aplicação em instrumentação e controle em tempo real. São Carlos, 1993.257p. Tese (mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo.

NEVE, B.D. Progress on radio fieldbus. Measurement & Control, v.23, n.2, p.14-9, 1990.

PELLE, D.; WILLRICH, R.; FRITSKE, U. JR.; SILVA, E.S. Um modelo de implementação do barramento de campo PROFIBUS. In: CONAI - CONGRESSO NACIONAL DE AUTOMAÇÃO INDUSTRIAL, 5., São Paulo, 1992. Anais. São Paulo, 1992. p. 253-62.

PHILIPS COMPONENTS (Eindhoven), Integrated circuits: microprocessors, microcontrollers and peripherals. Eindhoven, 1985, 1334p.

PHILIPS COMPONENTS (Eindhoven), Integrated circuits: I2C-bus compatible ICs. Eindhoven, 1989, 1384p.

RADIOMETRIX Ltd (Southall, Middlesex), SILRX-418-A data sheet: UHF radio telemetry receive module. Southall, Middlesex, 1992a.

RADIOMETRIX Ltd (Southall, Middlesex), TXM-418-A data sheet: Low power UHF radio telemetry transmit module. Southall, Middlesex, 1992b.

SÁ, D. The evolution of the intelligent measurement. Measurement & Control, v.21, n.6, p.142-4, 1988.

SANGWINE, S.J. Electronic components and technology. London, Chapman & Hall, 1987. 181p.

- SANTOS, G.D.; SALLES, H.C.A.S. Rádio bicanal digital: Evolução tecnológica para atendimento rural. Revista Telebrás, v.16, n.56, p.113-8, 1992.
- SHANMUGAM, K.S. Digital and analog communications systems. New York, John Wiley, 1979.
- SIEMENS AG (München) Siemens microcomputer components SAB80C515/80C535: User's manual 10.89. München, 1989.
- STENBIT, J.P. Tables of generators for Bose-Chaudhuri Codes. IEEE Transactions on Information Theory, v.10, n.4, p.390-1.
- STRINGFELLOW, M.F. Lightning and surge protection. Broadcasting Engineering, v.38, n.1, p.16, n.2, p.16, n.3, p.16, 1994.
- TANNER, B.D. Automated weather stations. Remote Sensing Review, v.5, n.1, p. 73-98, 1990.
- TEXAS INSTRUMENTS, Linear and interface circuits applications, 1985, v.2.
- TORRE, A.N. Estudo e implementação de um sistema de monitoramento remoto de variáveis edafo-ambientais. São Carlos, 1995.146p. Tese (doutorado) - Instituto de Física

de São Carlos, Universidade de São Paulo.

WIGGERT, D. Codes for error control and synchronization.
Norwood, MA, Artech, 1988. 203p.

WOOD, G.G. Current fieldbus activities. Computer
Communications, v.11, n.3, p.118-23, 1988.

ZEBERMANN, C.; KASTER, L.; RAKE, H. PROFIBUS open
on-site communication. In: TRIENNIAL WORLD CONGRESS
OF THE INTERNATIONAL FEDERATION ON AUTOMATIC
CONTROL, 11, Tallin, Aug. 13-17, 1990, Proceedings...
Elmsford: Pergamon, 1991. p.143-146.

MASTER, M. et al. Voltages induced on overhead line by the
lightning stepped leader. IEEE Transactions on EMC. v.28,
n.3, p.168-71, 1986.

Apêndice

Listagem das rotinas desenvolvidas para as estações de dados, em linguagem de montagem e linguagem C.

Índice

Acerta_Relogio	146
AcertaAlarmeData	147
ArmazenaStatus	158
BcdBin	135
BinBcd	134
BuscaPalavrasEnderecoComando	144
BuscaPalavraSinc	153
BuscaStatus	156
CalculaCRC	159
ClearIntAlarm	152
ComparaHora	166
ComparaStatus	157
CSTARTUP	120
Data_Segundos	166
delay	135
DesabilitaAlarmeRelogio	152
DesativaAteProximoComando	155
DesativaAteProximoSincronismo	154
DesativaTempoIndeterm	155
DesligaTxRx	169

dputc	129
dputchar	162
dputinst.....	129
EsperaPalavra	145
fb.h	137
inidisp	128
IniSer	164
interr	121
IntOverTimer2	121
IntRcxTD	121
LeAlarmeDatado.....	150
LeFlagIntRelogio	151
LeRelogio.....	149
Paridade.....	133
PreparaRec	168
ptrx	136
putchar	136
putchar_i2cbus.....	128
putframe	130
RcxI2C_ack	126
RcxI2C_no_ack	126
Rotina de teste da estacao mestre	140
rotina de teste do enlace de radio, estacao escrava.....	160
rotina de teste do enlace de radio, estacao mestre	161

rotina de teste estacao escrava	141
Segundos_Data	167
Sincroniza	154
Start_I2C_Bus	164
Stop_I2C_Bus	164
T2Timer	166
T2TimerReload	161
TrxI2C	126


```

;-----
; Arquivo : CSTARTUP.S03
; rotina de inicializacao
;-----
        NAME      CSTARTUP

$DEFEM.INC          ; modelo de memoria medium ou large
        PUBLIC  exit
        EXTERN  IntOverTimer2, IntRcxTD
        EXTERN  _R
        EXTERN  main

;-----
; Alocando memoria para segmento de pilha virtual
;-----
        RSEG      XSTACK
        DS        1024
xstack_end:
;-----
; Alocando memoria para a pilha interna
;-----
        RSEG      ISTACK
stack_begin:
        DS        45

        RSEG      CSTART
startup:
        SJMP      init_C
        ORG       startup+2bh ; overflow timer 2
        LJMP      IntOverTimer2
        ORG       startup+53h ; IEX3  P1.0
        LJMP      IntRcxTD
        ORG       startup+5bh ; IEX4  P1.1
        LJMP      IntRcxTD

init_C:
;-----
; Desligando transmissor/receptor
;-----
        CLR       P1.7
;-----
;-----
; Ativando o banco de registradores selecionado no momento

```

```

; da "linkagem"
;-----
MOV      A,#_R
MOV      C,ACC.3
MOV      PSW.3,C
MOV      C,ACC.4
MOV      PSW.4,C

MOV      SP,#stack_begin
;-----
; Inicializando variaveis globais e estaticas
; (usar sempre com o modelo large)
;-----

EXTERN ?SEG_INIT_L17
LCALL    ?SEG_INIT_L17 ; Initialize segments

;-----
; Inicializando hardware
;-----
MOV      R6,#HIGH(xstack_end)
MOV      R7,#LOW(xstack_end)
MOV      DPTR,#0          ; Sem parametros
LCALL    main             ; main()
exit:
SJMP     $
END      startup

;-----
; Rotinas de tratamento de interrupcao do
; timer 2 e dos pinos de recepcao
;-----
MODULE interr

PUBLIC   IntOverTimer2, IntRcxTD
PUBLIC   EmRecepcao
EXTERN   BufferQuadro

$io515.inc

;-----
; definindo a posicao das variaveis de recepcao
;-----

; flag para indicar que a recepcao de um quadro esta em curso

```

```

EmRecepcao EQU 8

; valor para a recarga do timer 2
RecT2H EQU 9
RecT2L EQU 0AH

; buffer para o valor do timer 2
LatchT2H EQU 0BH
LatchT2L EQU 0CH

; valor minimo de T2 para bit vahlido
VMinT2H EQU 0DH
VMinT2L EQU 0EH

; valor de limiar para T2
VLimH EQU 0FH
VLimL EQU 10H

; ponteiro de bits do buffer de recepcao
PBB EQU 11H

; buffer de recepcao
BR0 EQU 12H
BR1 EQU 13H
BR2 EQU 14H
BR3 EQU 15H

; flag de quadro recebido
FQR EQU 16H

RSEG CODE

;-----
; Rotina de tratamento da interrupcao por overflow de T2
;-----
;
IntOverTimer2:
    CLR EAL
    ; inibindo a contagem do timer 2
    ANL T2CON, #0FCH ; 24

    ; indicando que a recepcao de um quadro nao esta em
curso
    MOV EmRecepcao, #0

    ; inicializando o ponteiro de bits do buffer de recepcao
    MOV PBB, #0

    ; inicializando o buffer de recepcao

```

```

MOV    BR0, #0
MOV    BR1, #0
MOV    BR2, #0
MOV    BR3, #0

```

```

; limpando a flag de overflow do timer 2
CLR    TF2

```

```

; retorna-se com T2 parado
SETB   EAL
RETI

```

IntRcxTD:

```

; numero de microsegundos antes da interrupcao parar o
contador

```

```

; 2 + 2 = 4

```

```

; inibindo a contagem do timer 2
ANLT2CON, #0FCH    ; 24

```

```

; numero de microsegundos antes do timer voltar a contar
; 1 + 2 + 2 + 2 + 2 + 1 = 10

```

```

;

```

```

; total : 10 - 4 = 6 microsegundos

```

```

; inibindo interrupcoes

```

```

CLR    EAL

```

```

; armazenando o valor do timer 2 no buffer

```

```

MOV    LatchT2H, TH2

```

```

MOV    LatchT2L, TL2

```

```

; recarregando o timer 2 para sincronizacao

```

```

MOV    TH2, RecT2H

```

```

MOV    TL2, RecT2L

```

```

; habilitando contagem de T2

```

```

ORLT2CON, #1    ; 24

```

```

CLR    TF2

```

```

PUSH   PSW

```

```

PUSH   ACC

```

IF_011: ; se e' o inicio de um quadro

```

MOV    A, EmRecepcao

```

```

JNZ    EI_011

```

```

MOV    EmRecepcao, #1

```

```

MOV    PBB, #0

```

```

MOV    BR0, #0

```

```

MOV    BR1, #0

```

```

MOV    BR2, #0

```

```

MOV    BR3, #0

```

```

POP    ACC

```

```

POP    PSW

```



```

        CLR    IEX3
        CLR    IEX4
        SETB   EAL
        RETI
EI_011:
        ; armazenando registradores afetados na pilha interna
        PUSH   B
        PUSH   0
        PUSH   1
        PUSH   DPH
        PUSH   DPL
IF_001: ; se o tempo de bit e' valido
        MOV    A, LatchT2L
        CLR    C
        SUBB   A, VMinT2L
        MOV    A, LatchT2H
        SUBB   A, VMinT2H
        JC     EL_001
TH_001: ; neste ponto, um bit foi recebido, e o valor de T2
        ; indica se e' '0' ou '1'
IF_002: ; se o bit corresponde a um '0' ou a um '1'
        MOV    A, VLimL
        CLR    C
        SUBB   A, LatchT2L
        MOV    A, VLimH
        SUBB   A, LatchT2H
        JC     EL_002
TH_002: ; e um '1'
        MOV    R1, #0
        SJMP   EI_002
EL_002: ; e' um '0'
        MOV    R1, #1
EI_002:
        MOV    B, #8
        MOV    A, PBB
        DIV    AB
IF_003: ; SE O RESULTADO INTEIRO E' MAIOR QUE 4
        MOV    R0, A
        MOV    A, #4 ; NUMERO DE BYTES DO
BUFFER RECEPCAO
        CLR    C
        SUBB   A, R0
        JNC    EI_003
TH_003: ; o PBB apontava para fora do buffer
        MOV    R0, #0
EI_003: ; o ponteiro PBB e' valido
        MOV    A, R0
        ADD    A, #BR0
        MOV    R0, A ; R0 APONTA PARA UM BYTE DO

```

```

BUFF RECEP
    ; EM B, TEM-SE O NRO DE DESLOCAMENTOS
    ; EM R1, O BIT RECEBIDO
    INC    B
    MOV    A, R1
DO_002:
    RR     A
           DJNZ    B, DO_002
ED_002:
    ORL    A, @R0
    MOV    @R0, A
    INC    PBB
IF_004: ; se o PBB e' maior ou igual o maximo (31)
           MOV    A, PBB
           CLR    C
           SUBB   A, #32
           JC     EI_004
TH_004: ; PBB com o valor maximo: dado recebido
           ; colocando o buffer de recepcao no buffer de dados
    MOV    R1, #4           ; numero de bytes
    MOV    DPTR, #BufferQuadro
    MOV    R0, #BR0         ; buffer de recepcao
DO_001:
    MOV    A, @R0
    MOVB   @DPTR, A
    INC    R0
    INC    DPTR
           DJNZ    R1, DO_001
ED_001:
    MOV    FQR, #1         ; flag de quadro recebido

           ; indicando que a recepcao de um quadro nao esta em
curso
    MOV    EmRecepcao, #0
           ; inicializando o ponteiro de bits do buffer de recepcao
    MOV    PBB, #0
           ; inicializando o buffer de recepcao
    MOV    BR0, #0
    MOV    BR1, #0
    MOV    BR2, #0
    MOV    BR3, #0
EI_004: ; o quadro nao esta' completo
           SJMP    EI_001
EL_001: ; neste caso, o tempo do bit nao e' valido
           ; indicando que a recepcao de um quadro pode estar em
curso
    MOV    EmRecepcao, #1
           ; inicializando o ponteiro de bits do buffer de recepcao
    MOV    PBB, #0

```

```

; inicializando o buffer de recepcao
MOV    BR0, #0
MOV    BR1, #0
MOV    BR2, #0
MOV    BR3, #0
; colocando nos registradores afetados o valor original
; e retornando
EI_001:
POP    DPL
POP    DPH
POP    1
POP    0
POP    B
POP    ACC
POP    PSW
CLR    IEX3
CLR    IEX4
SETB   EAL
RETI
END

```

```

;-----
; arquivo: i2c.c
; rotinas de comunicacao barramento i2c
;
; int TrxI2C(unsigned char val);
; unsigned char RcxI2C_no_ack(void);
; unsigned char RcxI2C_ack(void);
;-----
MODULE I2C
SDA EQU P1.3
SCL EQU P1.4
PUBLIC TrxI2C
PUBLIC RcxI2C_ack
PUBLIC RcxI2C_no_ack
RSEG CODE

; rotina de transmissao barramento I2C
TrxI2C:
MOV    R0, #8
MOV    A, R3
FOR_001:
RLC    A
MOV    SDA, C
LCALL  TEMPOI2C
SETB   SCL
LCALL  TEMPOI2C

```

```

        CLR    SCL
        LCALL  TEMPOI2C
        DJNZ   R0, FOR_001
EFOR_001:
        SETB   SDA
        LCALL  TEMPOI2C
        SETB   SCL
        LCALL  TEMPOI2C
        MOV    C, SDA
        CLR    SCL
        LCALL  TEMPOI2C
        MOV    A, #0
        RLC    A
        MOV    R3, A
        MOV    R2, #0
        RET

```

; rotina de recepcao com ack barramento I2C

```

RcxI2C_ack:
        MOV    R0, #8
FOR_002:
        SETB   SCL
        LCALL  TEMPOI2C
        MOV    C, SDA
        RLC    A
        CLR    SCL
        LCALL  TEMPOI2C
        DJNZ   R0, FOR_002
EFOR_002:
        CLR    SDA
        LCALL  TEMPOI2C
        SETB   SCL
        LCALL  TEMPOI2C
        CLR    SCL
        LCALL  TEMPOI2C
        SETB   SDA
        LCALL  TEMPOI2C
        MOV    R3, A
        RET

```

; rotina de recepcao sem ack barramento I2C

```

RcxI2C_no_ack:
        MOV    R0, #8
FOR_003:
        SETB   SCL
        LCALL  TEMPOI2C
        MOV    C, SDA
        RLC    A
        CLR    SCL

```



```

        LCALL  TEMPOI2C
        DJNZ   R0, FOR_003
EFOR_003:
        SETB   SDA
        LCALL  TEMPOI2C
        SETB   SCL
        LCALL  TEMPOI2C
        CLR    SCL
        LCALL  TEMPOI2C
        MOV    R3, A
        RET

```

```

TEMPOI2C:
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        RET
        END

```

```

;-----
; Arquivo : puti2c.s03
; char putchar_i2cbus (int val);
;-----
MODULE  putchar_i2cbus
$fb.inc

```

```

        PUBLIC  putchar_i2cbus
        RSEG    CODE
putchar_i2cbus:
        MOV     SBUF, R3
        RET
        END

```

```

;-----
; Arquivo : INIDISP.S03
; rotina inicializacao display
;          void inidisp(void);
;-----
MODULE  inidisp
PUBLIC  inidisp

```

```

PUBLIC dputinst
$io515.inc
EN_DIS EQU P5.2
RD_WR_DIS EQU P5.1
RS_DIS EQU P5.0
D_DISPLAY EQU P4
RSEG CODE
inidisp:
MOV r3, #38H
LCALL dputinst
MOV r3, #38H
LCALL dputinst
MOV r3, #06H
LCALL dputinst
MOV r3, #0EH
LCALL dputinst
MOV r3, #01H
LCALL dputinst
MOV r3, #0CH ; cursor off
LCALL dputinst
RET

dputinst:
clr RS_DIS
mov D_DISPLAY, r3
clr RD_WR_DIS
setb EN_DIS
clr EN_DIS
MOV R2, #200
djnz r2, $ ; DANDO UM TEMPO P/ O CONTROLADOR
DO DISP.
RET
END

```

```

;-----
; Arquivo : DPUTCHAR.S03
; rotina basica de saida para display LCS
;          int dputchar (int val);
;-----
MODULE dputc
PUBLIC dputc
$io515.s03
EN_DIS EQU P5.2
RD_WR_DIS EQU P5.1
RS_DIS EQU P5.0
D_DISPLAY EQU P4
RSEG CODE

```

```

dputc:
    setb    RS_DIS
    mov     D_DISPLAY, r3
    clr     RD_WR_DIS
    setb    EN_DIS
    clr     EN_DIS
    mov     D_DISPLAY, 0ffh
    mov     r2, #20
    djnz    r2, $ ; DANDO UM TEMPO P/ O CONTROLADOR
DO DISP.
    RET
    END

```

```

;-----
; Arquivo : PUTFRAME.S03
; Envia um quadro via serial
; void putframe(unsigned long *d)
;-----

```

```

NAME      putframe(16)
RSEG      CODE(0)
RSEG      UDATA(0)
PUBLIC    putframe(18,96)
EXTERN    ?ENTER_L17
EXTERN    ?LEAVE_L17
EXTERN    ?LD_REL4_L11
EXTERN    ?CL8051_2_20_L17
RSEG      CODE

```

\$io515.inc

```

;-----
;      definindo a posicao das variaveis de transmissao
;      (compartilhadas pela recepcao)
;-----

```

```

ContBit0TrxH    EQU 9
ContBit0TrxL    EQU 0AH
ContBit1TrxH    EQU 0BH
ContBit1TrxL    EQU 0CH

```

putframe:

```

    MOV     A,#4
    MOV     R0,#0
    MOV     R1,#4
    LCALL    ?ENTER_L17
    LCALL    ?LD_REL4_L11
    DW 1024
    MOV     DPTR, #bufreg ; APONTA PARA BUFFER
    MOV     A, 20H
    MOVB    @DPTR, A
    INC     DPTR

```

```

MOV    A, 21H
MOVX   @DPTR, A
INC    DPTR
MOV    A, 22H
MOVX   @DPTR, A
INC    DPTR
MOV    A, 23H
MOVX   @DPTR, A
MOV    20H, R2
MOV    21H, R3
MOV    22H, R4
MOV    23H, R5
MOV    C, 20H.7      ; 12
LCALL  putbit        ; 24
MOV    C, 20H.6      ; 12
LCALL  putbit
MOV    C, 20H.5      ; 12
LCALL  putbit
MOV    C, 20H.4      ; 12
LCALL  putbit
MOV    C, 20H.3      ; 12
LCALL  putbit
MOV    C, 20H.2      ; 12
LCALL  putbit
MOV    C, 20H.1      ; 12
LCALL  putbit
MOV    C, 20H.0      ; 12
LCALL  putbit
MOV    C, 21H.7      ; 12
LCALL  putbit
MOV    C, 21H.6      ; 12
LCALL  putbit
MOV    C, 21H.5      ; 12
LCALL  putbit
MOV    C, 21H.4      ; 12
LCALL  putbit
MOV    C, 21H.3      ; 12
LCALL  putbit
MOV    C, 21H.2      ; 12
LCALL  putbit
MOV    C, 21H.1      ; 12
LCALL  putbit
MOV    C, 21H.0      ; 12
LCALL  putbit
MOV    C, 22H.7      ; 12
LCALL  putbit
MOV    C, 22H.6      ; 12
LCALL  putbit
MOV    C, 22H.5      ; 12

```



```

LCALL putbit
MOV C, 22H.4 ; 12
LCALL putbit
MOV C, 22H.3 ; 12
LCALL putbit
MOV C, 22H.2 ; 12
LCALL putbit
MOV C, 22H.1 ; 12
LCALL putbit
MOV C, 22H.0 ; 12
LCALL putbit
MOV C, 23H.7 ; 12
LCALL putbit
MOV C, 23H.6 ; 12
LCALL putbit
MOV C, 23H.5 ; 12
LCALL putbit
MOV C, 23H.4 ; 12
LCALL putbit
MOV C, 23H.3 ; 12
LCALL putbit
MOV C, 23H.2 ; 12
LCALL putbit
MOV C, 23H.1 ; 12
LCALL putbit
MOV C, 23H.0 ; 12
LCALL putbit
nop
nop
nop
CPL TXD ; FINALIZANDO ULTIMO BIT
MOV R4, #2
DO_003: ; DA UM TEMPO DE 2 BITS '1', INDICANDO FIM
DE TRANSMISSAO
ANL T2CON, #0FCH ; 24 PARA DE CONTAR
MOV TH2, ContBit1TrxH ; 24
MOV TL2, ContBit1TrxL ; 24
ORL T2CON, #1 ; 24 VOLTA A CONTAR
JNB TF2, $
ANL T2CON, #0FCH ; 24 PARA DE CONTAR
CLR TF2 ; 12
DJNZ R4, DO_003
ED_003:
ANL T2CON, #0FCH ; 24 PARA DE CONTAR
CLR TF2 ; 12
; retornando o buffer de transmissao e
; area bit-enderecavel
MOV DPTR, #bufreg ; APONTA PARA BUFFER
MOVBX A, @DPTR

```

```

MOV    20H, A
INC    DPTR
MOVX   A, @DPTR
MOV    21H, A
INC    DPTR
MOVX   A, @DPTR
MOV    22H, A
INC    DPTR
MOVX   A, @DPTR
MOV    23H, A
?0000:
    LJMP    ?LEAVE_L17
putbit: ; o bit vem no carry
    CPL    TXD                ; 12
    ANL    T2CON, #0FCH        ; 24 PARA DE CONTAR
    CLR    TF2                ; 12
    JC     trans1              ; 24
trans0:
    MOV    TH2, ContBit0TrxH    ; 24
    MOV    TL2, ContBit0TrxL    ; 24
    ORL    T2CON, #1            ; 24 VOLTA A CONTAR
    JNB    TF2, $
    RET
trans1:
    MOV    TH2, ContBit1TrxH    ; 24
    MOV    TL2, ContBit1TrxL    ; 24
    ORL    T2CON, #1            ; 24 VOLTA A CONTAR
    JNB    TF2, $
    RET
RSEG    UDATA
bufreg:
    DS    4
    END

```

```

;-----
; arquivo: paridade.s03
; calcula a paridade de uma palavra de 32 bits
;-----

```

```

NAME    paridade(16)
RSEG    CODE(0)
PUBLIC  Paridade(22,32)
EXTERN  ?ENTER_L17
EXTERN  ?LEAVE_L17
EXTERN  ?LD_REL4_L11
EXTERN  ?CL8051_2_20_L17
RSEG    CODE
Paridade:

```

```

MOV     A,#4
MOV     R0,#0
MOV     R1,#4
LCALL   ?ENTER_L17
LCALL   ?LD_REL4_L11
DW 1024
MOV     B, #0
MOV     A, R2
MOV     C, P
MOV     B.0, C
MOV     A, R3
MOV     C, P
MOV     B.1, C
MOV     A, R4
MOV     C, P
MOV     B.2, C
MOV     A, R5
MOV     C, P
MOV     B.3, C
MOV     A, B
MOV     B, #0
MOV     C, P
MOV     B.0, C
MOV     R2, #0
MOV     R3, B
?0000:
LJMP     ?LEAVE_L17
END

```

```

;-----
;  arquivo: binbcd.s03
;  unsigned char BinBcd(unsigned char val);
;-----
MODULE BinBcd
PUBLIC BinBcd
RSEG CODE
BinBcd:
MOV     A, R3
MOV     B, #10
DIV     AB
MOV     R3, B
MOV     B, #10
DIV     AB
MOV     A, B
SWAP    A
ORL     A, R3
MOV     R3, A

```

```

    RET
    END

;-----
; arquivo: bcdbin.s03
; unsigned char BcdBin(unsigned char val);
;-----
    MODULE BcdBin
    PUBLIC BcdBin
    RSEG CODE
BcdBin:
    MOV     A, R3
    ANL     A, #0F0H
    SWAP    A
    MOV     B, #10
    MUL     AB
    MOV     B, A
    MOV     A, R3
    ANL     A, #0FH
    ADD     A, B
    MOV     R3, A
    RET
    END

;-----
; arquivo:          DELAY.S03
; void delay (int val);
;-----
    MODULE delay
    PUBLIC delay
    RSEG CODE
delay:
    mov     a, r2      ; 1
    orl     a, r3      ; 1
    jnz     continua  ; 2
    ret
continua:
    lcall   delay974    ; 1000 - 11 - 2 - 13 = 974
loop:
    clr     c          ; 1
    mov     a, r3      ; 1
    subb    a, #1       ; 2
    mov     r3, a       ; 1
    mov     a, r2      ; 1
    subb    a, #0       ; 2
    mov     r2, a       ; 1
    mov     a, r2      ; 1
    orl     a, r3      ; 1
    jnz     continua_2 ; 2 total 13
    ret

```



```

continua_2:
    lcall delay983    ; 1000 - 2 - 13 - 2 = 983
    ljmp  loop        ; 2
delay983:
    mov  r4, #3       ; 2
    nop                      ; 1
    djnz r4, $         ; 2 * 3 = 6
delay974:
    mov  r4, #242     ; 2
ld1:
    nop                      ; 1
    nop                      ; 1
    djnz r4, ld1       ; 2    4 * 242 = 968
    nop                      ; 1
    nop                      ; 1
    ret                       ; 2
    END

```

```

;-----
; Arquivo : PUTCHAR.S03
; Envia um caracter via serial
; int putchar (int val);
;-----
    MODULE putchar
    PUBLIC  putchar
    RSEG   CODE
putchar:
    JNB    TI,putchar
    CLR    TI
    MOV    SBUF, R3
    RET
    END

```

```

;-----
; arquivo ptrx.s03
; prepara a recepcao
;-----
    MODULE ptrx
    PUBLIC  ptrx
    RSEG   CODE
ptrx:
    mov    r3, #30
loop:
    cpl    TXD      ;12
    mov    r2, #180 ;24

```

```

    djnz    r2, $    ;24
    djnz    r3, loop
    ret
    END

```

```

/*****
*****/

```

```

/*    arquivo fb.h    */

```

```

#define AddrPCF8583 0xA0 /* com a0 low */
#define WR 0
#define RD 1
#define PWRTRRX_bit 0x97 /* P1_7_bit */
#define PWRCTRL 0x96 /* P1_6_bit */
#define N6_9V_bit 0x95 /* P1_5_bit */
#define EmRecepcao 8
#define RecT2H 9
#define RecT2L 0xA
#define LatchT2H 0xB
#define LatchT2L 0xC
#define VMinT2H 0x0D
#define VMinT2L 0x0E
#define VLimH 0x0F
#define VLimL 0x10
#define PBB 0x11
#define BR0 0x12
#define BR1 0x13
#define BR2 0x14
#define BR3 0x15
#define FQR 0x16
#define NroPulBit0 ((unsigned int)((1200000.0/12)/1200.0))
#define NroPulBit1 ((unsigned int)((1200000.0/12)/600.0))
#define NroMinPul ((unsigned int)(NroPulBit0 * 0.9))
#define NroMaxPul ((unsigned int)(NroPulBit1 * 1.1))
#define TMRecarga 15
#define TMPPararT2 5
#define VRecargaT2 (0x100001 - NroMaxPul + TMRecarga)
#define ValorMinT2 (0x100001 - NroMaxPul + NroMinPul +
TMPPararT2)
#define NroPulLim ((unsigned
int)((NroPulBit0+NroPulBit1)/2))
#define VBit0 ((unsigned int)(0x100001 - NroMaxPul +
NroPulBit0 + TMPPararT2))
#define VBit1 ((unsigned int)(0x100001 - NroMaxPul +
NroPulBit1 + TMPPararT2))
#define VLim ((unsigned int)(VBit1 + VBit0)/2)
#define ContBit0TrxH 9

```

```

#define ContBit0TrxL 0x0A
#define ContBit1TrxH 0xB
#define ContBit1TrxL 0xC
#define ContBit0Trx (0x100001 - NroPulBit0 + 17)
#define ContBit1Trx (0x100001 - NroPulBit1 + 17)
#define MAXNUMPALCOM 16
#define PERIODOSLOT 8 /* segundos */
#define NUMEROSLOTS 4
#define INSTRUM01 0X1231
#define INSTRUM02 0X1232
#define INSTRUM03 0X1233
#define INSTRUM04 0X1234
#define INSTRUM05 0X1235
#define INSTRUM06 0X1236
#define INSTRUM07 0X1237
#define MEUSLOT 2
#define PALAVRACODSINC 0xAAAA
#define PERIODOSETUP 3
#define MAXNUMPERDASINC 3
#define TEMPOESPERASINC 4
#define TEMPOESPERAENDERECO 4
#define TEMPOESPERAPAL 5
#define MSEGSAINC 3
#define SINCRONISMO 1
#define COMANDO 0
#define NS4A ((double)126230400.0)
#define NSA 315360001
#define NSAB 316224001
#define NSD 864001
#define ANOBASE 1990
/* Mapa da memoria do relógio:
  00 - 0F : contadores do relógio e registradores do alarme
  10H- 11H: (int) ano do relógio
  12H- 13H: (int) ano do alarme
  14H      : (char) Status.Sincronismo
  15H      : (char) Status.ContPerdaSinc
  16H      : (char) Status.ProximoSinc
  17H      : (char) Status.ProximoSinc
  18H      : (char) Status.ProximoSinc
  19H      : (char) Status.ProximoSinc
  1AH 1BH : (unsigned) Status.CheckSum
*/
#define ANORELOGIO 0x10
#define ANOALARME 0x12
#define STATUS 0x14
struct status_t
{
    char Sincronismo_Comando;
    char Comando;

```



```

    char ContPerdaSinc;
    long ProximoSinc;
    long ProximaAtivacao;
    unsigned int CheckSum;
};
struct tempo {
    char tm_sec;
    char tm_min;
    char tm_hour;
    char tm_day;
    char tm_mon;
    int tm_year;
    char tm_wday;
};
void T2TimerReload(int BPS);
void T2Timer(void);
void putframe(unsigned long d);
void IniSer(unsigned char ValorTh1);
void PreparaRec(void);
void PreparaTrans(void);
void DesligaTxRx(void);
void inidisp(void);
int dprintf(const char *,...);
void T2Timer(void);
void delay (int val);
void ptrx(void);
unsigned long CalculaCRC(unsigned long d);
int Paridade(unsigned long d);
void Start_I2C_Bus(void);
void Stop_I2C_Bus(void);
int TrxI2C(unsigned char val);
unsigned char RcxI2C_no_ack(void);
unsigned char RcxI2C_ack(void);
unsigned char BcdBin(unsigned char val);
unsigned char BinBcd(unsigned char val);
void LeRelogio(struct tempo *rc);
void Acerta_Relogio(struct tempo *rc);
int AcertaAlarmeData(struct tempo *rc);
int ClearIntAlarm(void);
int DesabilitaAlarmeRelogio(void);
int ComparaHora(struct tempo *t1, struct tempo *t2);
long Data_Segundos(struct tempo *t);
void Segundos_Data(long Seg, struct tempo *t);
void LeAlarmeDatado(struct tempo *rc);
int LeFlagIntRelogio();
int BuscaStatus(struct status_t *st);
int ArmazenaStatus(struct status_t *stt);
unsigned long ProximoSinalSincronismo(unsigned long tp);
void DesativaTempoIndeterm(struct status_t *stt);

```



```

void DesativaAteProximoComando(struct status_t *stt, struct
tempo *HS);
void DesativaAteProximoSincronismo(struct status_t *stt);
void ExecutarComando(unsigned long *Com);
int AtivacaoManual(struct status_t *st);
char EsperaPalavra(int TempoEspera, unsigned long *Pal);
void RecuperaSinc(struct status_t *stt);
char BuscaPalavrasEnderecoComando(unsigned long *VetCom);
int BuscaPalavraSinc(long TempoEsperaSinc, struct tempo
*Tmp);

```

```

/*****
*****/
/* Rotina de teste da estacao mestre */
/* arquivo:rfbm.c */
#include <io515.h>
#include "fb.h"
unsigned long BufferQuadro;
struct tempo Hora;
long Inst[] = {0, 0X1231, 0X1232, 0X1233, 0X1234, 0X1235,
0X1236, 0X1237};

void main(void)
{
    long t, ta, ttps;
    unsigned long ul;
    int Instrum;

    PreparaTrans();
    inidisp();
    Le_Hora(&Hora);
    Acerta_Relogio(&Hora);
    putframe(PALAVRACODSINC);
    t = Data_Segundos(&Hora);
    ttps = t + PERIODOSLOT*NUMEROSLOTS;
    t += PERIODOSLOT;
    Instrum = 1;
    for(;;){
        LeRelogio(&Hora);
        ta = Data_Segundos(&Hora);
        if(ta == (ttps-1)){
            PreparaTrans();
            delay(100);
            dprintf("sinc");
            putframe(PALAVRACODSINC);
            ttps += PERIODOSLOT*NUMEROSLOTS;
            t += PERIODOSLOT;

```

```

        PreparaRec();
    }
    if(ta == t){
        dprintf(" Inst%d",Instrum);
        PreparaTrans();
        delay(50);
        putframe(Inst[Instrum]);
        delay(50);
        putframe(2);
        delay(50);
        putframe(Hora.tm_min);
        delay(50);
        putframe(Hora.tm_sec);
        PreparaRec();
        t += PERIODOSLOT;
        if(Instrum < NUMEROSLOTS-1){
            Instrum++;
        }
        else{
            Instrum = 1;
        }
    }
    if(input(FQR)){
        ul = BufferQuadro;
        output(FQR, 0);
        dprintf(" %ld", ul);
    }
}
}

```

```

/*****
*****/
/*  arquivo:rfbe6.c   rotina de teste estacao escrava */
#include "fb.h"
struct status_t Status;
unsigned long BufferQuadro;
long TempoSincronismo;
struct tempo HoraAtivacao;
struct tempo H;
unsigned long PalCom[MAXNUMPALCOM + 1];
#define MEUENDERECO 0X1232 /* slot 2 */

void main()
{
    unsigned long t;

    LeRelogio(&HoraAtivacao);

```

```

t = Data_Segundos(&HoraAtivacao);
if(BuscaStatus(&Status)) {
    if(Status.ProximaAtivacao == t){
        if( Status.ContPerdaSinc < MAXNUMPERDASINC){
            if(Status.Sincronismo_Comando != COMANDO){
                Status.Sincronismo_Comando = SINCRONISMO;
            }
            if(Status.Sincronismo_Comando == COMANDO){
                if(BuscaPalavrasEnderecoComando(&PalCom)){
                    ExecutarComando(&PalCom);
                    /* aqui, o sincronismo e' valido */
                    Status.ContPerdaSinc = 0;
                }
                Status.ProximaAtivacao = Status.ProximoSinc -
MSEGSINC;
                Status.Sincronismo_Comando = SINCRONISMO;
                Segundos_Data(Status.ProximaAtivacao, &H);
                ArmazenaStatus(&Status);
                AcertaAlarmeData(&H);
                clear_bit(PWRCTRL);
                for(;;);
            }
        }
        else{ /* Ativacao para sincronismo */
            if(!BuscaPalavraSinc(TEMPOESPERASINC, &H)){
                Status.ContPerdaSinc++;
                if(Status.ContPerdaSinc >= MAXNUMPERDASINC){
                    RecuperaSinc(&Status); /* inclui uma desativacao */
                }
            }
            else{
                Status.Sincronismo_Comando = COMANDO;
                Status.ProximaAtivacao = Status.ProximoSinc +
(MEUSLOT * PERIODOSLOT) -
MSEGSINC;
                Status.ProximoSinc = Status.ProximoSinc +
(NUMEROSLOTS *
PERIODOSLOT);
                Segundos_Data(Status.ProximaAtivacao, &H);
                ArmazenaStatus(&Status);
                AcertaAlarmeData(&H);
                clear_bit(PWRCTRL);
                for(;;);
            }
        }
        else{
            t = Data_Segundos(&H);
            Status.ContPerdaSinc = 0;
        }
        Status.Sincronismo_Comando = COMANDO;
        Status.ProximaAtivacao = t + (MEUSLOT *

```

```

PERIODOSLOT) - MSEGSLINC;
        Status.ProximoSinc = t + (NUMEROSLOTS *
PERIODOSLOT);
        Segundos_Data(Status.ProximaAtivacao, &H);
        ArmazenaStatus(&Status);
        AcertaAlarmeData(&H);
        clear_bit(PWRCTRL);
        for(;;);
    }
}
}
}
if(AtivacaoManual(&Status)){
}
else{
    RecuperaSinc(&Status);
}
}

```

```

void RecuperaSinc(struct status_t *stt)
{
    struct tempo H;
    long t;

    if(BuscaPalavraSinc(PERIODOSLOT * (NUMEROSLOTS+1),
&H)){
        t = Data_Segundos(&H);
        stt->Sincronismo_Comando = COMANDO;
        stt->ProximaAtivacao = t + (MEUSLOT * PERIODOSLOT) -
MSEGSLINC;
        stt->ProximoSinc = t + (NUMEROSLOTS * PERIODOSLOT);
        Segundos_Data(stt->ProximaAtivacao, &H);
        ArmazenaStatus(stt);
        AcertaAlarmeData(&H);
        clear_bit(PWRCTRL);
        for(;;);
    }
    else{
        clear_bit(PWRCTRL);
        for(;;);
    }
}

```

```

int BuscaPalavraSinc(long TempoEsperaSinc, struct tempo
*Tmp)
{
    long ti, ta;

```



```

unsigned long ul;

PreparaRec();
LeRelogio(Tmp);
ti = Data_Segundos(Tmp);
do{
    LeRelogio(Tmp);
    ta = Data_Segundos(Tmp);
    if(input(FQR)){
        ul = BufferQuadro;
        output(FQR, 0);
        if(ul == PALAVRACODSINC){
            /* desligando receptor */
            clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */
            return 1; /* a hora de recebimento da pal. esta em Tmp */
        }
    }
}while( ( (ta-ti) < TempoEsperaSinc) );
/* desligando receptor */
clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */
return 0;
}

```

```

char BuscaPalavrasEnderecoComando(unsigned long *VetCom)
{
    long ti, ta;
    int i = 1;
    struct tempo Tmp;

    PreparaRec();

    LeRelogio(&Tmp);
    ti = Data_Segundos(&Tmp);
    *VetCom = 0;
    do{
        LeRelogio(&Tmp);
        ta = Data_Segundos(&Tmp);
        if(input(FQR)){
            *VetCom = BufferQuadro;
            output(FQR, 0);
        }
    }while( ( (ta-ti) < TEMPOESPERAENDERECO+MSEGSINC)
    && (*VetCom != MEUENDERECO));
    if(*VetCom != MEUENDERECO){
        clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */
        return 0;
    }
}

```

```

    }
    if(!EsperaPalavra(TEMPOESPERAPAL, VetCom+1)){
        for(i = 1; i < MAXNUMPALCOM; i++){
            VetCom[i] = 0;
            clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */
            return 1;
        }
    }
    i = 2;
    while((EsperaPalavra(TEMPOESPERAPAL, VetCom+i)) && (i
< MAXNUMPALCOM)
        && (i < VetCom[0])){
        i++;
    }
    VetCom[MAXNUMPALCOM+1] = i;
    clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */
    return 1;
}

```

```

char EsperaPalavra(int TempoEspera, unsigned long *Pal)
{
    long ti, ta;
    struct tempo Tmp;

    LeRelogio(&Tmp);
    ti = Data_Segundos(&Tmp);
    do{
        LeRelogio(&Tmp);
        ta = Data_Segundos(&Tmp);
        if(input(FQR)){
            *Pal = BufferQuadro;
            output(FQR, 0);
            return 1;
        }
    }while( (ta-ti) < TempoEspera);
    return 0;
}

```

```

void ExecutarComando(unsigned long *Com)
{
    int i;

    PreparaTrans(); /* *** fase depuracao */
    delay(80);
    for(i=0;i<3;i++){

```

```

        delay(200);
        putframe(Com[i+1]);
    }
}

```

```

int AtivacaoManual(struct status_t *st)
{
    st;
    return 0;
}

```

```

/*****
*****/
/*  arquivo:relogio.c */
#include "fb.h"
typedef long time_t;
/*
 * Acerta_Rologio: transfere os dados da estrutura que contem
 * a hora e a data para os contadores do relógio calendario,
 * e desabilita o alarme.
 * Adicionalmente, armazena o ano nas posicoes de
memoria
 * ANORELOGIO e ANORELOGIO+1.
 *
 */
#define CONTROLSTATUSREG 0
#define ALARMCONTROL 8
void Acerta_Rologio(struct tempo *rc)
{
    unsigned uc;

    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    TrxI2C(CONTROLSTATUSREG); /* endereco byte */
    TrxI2C(0x80); /* dado posicao 00h - para contador */
    TrxI2C(0); /* dado posicao 01h - cent segundos */
    TrxI2C(BinBcd(rc->tm_sec)); /* dado posicao 02h - segundos
 */
    TrxI2C(BinBcd(rc->tm_min)); /* dado posicao 03h - minutos
 */
    TrxI2C(BinBcd(rc->tm_hour) & 0x3f); /* dado posicao 04h -
 horas */
    uc = (rc->tm_year - ((rc->tm_year/4) * 4)) << 6;
    TrxI2C(uc|(BinBcd(rc->tm_day) & 0x3f)); /* dado posicao
 05h - dia /ano */
    TrxI2C(BinBcd(rc->tm_mon) & 0x1f); /* dado posicao 06h -
mes */

```

```

TrxI2C(0); /* dado posicao 07h - timer */
TrxI2C(0); /* dado posicao 08h controle alarme*/
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
TrxI2C(CONTROLSTATUSREG); /* endereco byte */
TrxI2C(0x04); /* dado posicao 00h - habilita contador */
Start_I2C_Bus();
/* armazenando ano na memoria */
TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
TrxI2C(ANORELOGIO); /* endereco byte */
TrxI2C(rc->tm_year >> 8);
TrxI2C(rc->tm_year);
Stop_I2C_Bus();
}

/*
 * AcertaAlarmeData: Habilita e acerta o alarme do relógio
 * calendario
 * no modo alarme datado (segundos, minutos, horas, dias, e
 * mes,
 * sendo que o dia da semana e o ano sao ignorados).
 * O valor de retorno indica se as operacoes de escrita no
 * barramento I2C foram sucedidas de "acknowledge"
 */

int AcertaAlarmeData(struct tempo *rc)
{
    unsigned char uc;
    unsigned char csreg, aereg, aah, abh, ach, adh, aeh,
    a9h2, aah2, abh2, ach2, adh2, aeh2;

    /* lendo reg. controle/status */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    TrxI2C(CONTROLSTATUSREG);
    TrxI2C(0x04); /* dado posicao 00h - habilita alarme reg */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    TrxI2C(0x9);
    TrxI2C(0x00); /* dado posicao 09h - cent segundos */
    aah = BinBcd(rc->tm_sec);
    TrxI2C(aah); /* dado posicao 0ah - segundos */
    abh = BinBcd(rc->tm_min);
    TrxI2C(abh); /* dado posicao 0bh - minutos */
    ach = BinBcd(rc->tm_hour) & 0x3f;
    TrxI2C(ach); /* dado posicao 0ch - horas */
    uc = (rc->tm_year - ((rc->tm_year/4) * 4)) << 6;
    adh = uc|(BinBcd(rc->tm_day))&0x3f;

```



```

TrxI2C(adh); /* dado posicao 0dh - dia /ano */
aeh = BinBcd(rc->tm_mon) & 0x1f;
TrxI2C(aeh); /* dado posicao 0eh - mes */
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
TrxI2C(ANOALARME);
TrxI2C(rc->tm_year >> 8);
TrxI2C(rc->tm_year);
/* habilitando alarme data */
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
TrxI2C(ALARMCONTROL); /* endereco reg. controle alarme
*/
TrxI2C(0xB0); /* dado posicao 8h - habilita alarme data */
/* conferindo alarme */
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | WR);
TrxI2C(CONTROLSTATUSREG);
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | RD);
csreg = RcxI2C_no_ack();
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | WR);
TrxI2C(0x8); /* endereco alarme control */
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | RD);
acreg = RcxI2C_ack();
a9h2 = RcxI2C_ack();
aah2 = RcxI2C_ack();
abh2 = RcxI2C_ack();
ach2 = RcxI2C_ack();
adh2 = RcxI2C_ack();
aeh2 = RcxI2C_ack();
Stop_I2C_Bus();
return ((csreg == 0x4) && (acreg == 0xb0) &&
(a9h2 == 0) && (aah == aah2) && (abh == abh2) &&
(ach == ach2) && (adh == adh2) && (aeh == aeh2));
}

/*
 * LeRelógio: Le o relógio calendario, armazenando os dados
em
 * uma estrutura.
 * O ano e' armazenado nas duas posicoes de memoria
imediatamente
 * superiores `as posicoes do alarme. Caso o ano do relógio
seja
 * diferente dos dois bits menos significativos do ano

```

```

armazenado
*   na memoria, incrementa-se o ultimo.
*
*/

void LeRelogio(struct tempo *rc)
{
    unsigned char uc, ud, csr;

    /* mantendo ultima contagem nos registradores de captura */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(CONTROLSTATUSREG);
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | RD);
    csr = RcxI2C_no_ack() & 0xbf;
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(CONTROLSTATUSREG);
    TrxI2C(csr);
    /* lendo os contadores */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(0x02); /* endereco contador segundos */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | RD);
    uc = RcxI2C_ack();
    rc->tm_sec = BcdBin(uc);
    uc = RcxI2C_ack();
    rc->tm_min = BcdBin(uc);
    uc = RcxI2C_ack();
    rc->tm_hour = BcdBin(uc & 0x3f);
    uc = RcxI2C_ack();
    /* agora, em uc tem-se ano/dia */
    rc->tm_day = BcdBin(uc & 0x3f);
    ud = RcxI2C_no_ack();
    rc->tm_mon = BcdBin(ud & 0x1f);
    /* lendo ano na memoria */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(ANORELOGIO); /* endereco RAM */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | RD);
    rc->tm_year = 256 * RcxI2C_ack();
    rc->tm_year = rc->tm_year + RcxI2C_no_ack();
    ud = uc >> 6;
    if((rc->tm_year < ANOBASE) || (rc->tm_year > 2010)){
        Start_I2C_Bus();
        TrxI2C(AddrPCF8583 | WR);
    }
}

```

```

    TrxI2C(ANORELOGIO); /* endereco RAM */
    TrxI2C(ANOBASE >> 8);
    TrxI2C(ANOBASE);
}
uc = rc->tm_year & 0x3;
if( ud != uc ){
    rc->tm_year++;
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(ANORELOGIO); /* endereco RAM */
    TrxI2C(rc->tm_year >> 8);
    TrxI2C(rc->tm_year);
}
/* habilitando atualizacao dos registradores de captura */
Start_I2C_Bus();
TrxI2C(AddrPCF8583 | WR);
TrxI2C(CONTROLSTATUSREG);
TrxI2C(csr | 0x40);
Stop_I2C_Bus();
}

/*
 *   LeAlarmeDatado: Le os registradores do alarme,
armazenando-os
 *   em uma estrutura.
 *   Como o ano nao e' levado em consideracao pelo relógio
quando
 *   o alarme e' datado, o ano lido e' o ano do relógio.
 *
 */

void LeAlarmeDatado(struct tempo *rc)
{
    unsigned char uc, ud;

    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(ANOALARME); /* endereco RAM */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | RD);
    rc->tm_year = 256 * RcxI2C_ack();
    rc->tm_year += RcxI2C_ack();
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(0xa); /* endereco segundos alarme */
    uc = RcxI2C_ack();
    rc->tm_sec = BcdBin(uc);
    uc = RcxI2C_ack();
    rc->tm_min = BcdBin(uc);

```

```

uc = RcxI2C_ack();
rc->tm_hour = BcdBin(uc & 0x3f);
uc = RcxI2C_ack();
/* agora, em uc tem-se ano/dia */
rc->tm_day = BcdBin(uc & 0x3f);
ud = RcxI2C_no_ack();
rc->tm_mon = BcdBin(ud & 0x1f);
Stop_I2C_Bus();
}

/*
 * LeFlagIntRelogio: Verifica se o sinal de interrupcao do
 *                   relógio está ativo. Se ativo, retorna
 *                   1, senão, 0.
 */

int LeFlagIntRelogio()
{
    unsigned char uc, ud;

    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR);
    TrxI2C(CONTROLSTATUSREG); /* endereço reg.
    controle/status */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | RD);
    ud = RcxI2C_no_ack();
    /* se o "alarm enable" do "control/status" está habilitado */
    if(ud & 0x4){
        Start_I2C_Bus();
        TrxI2C(AddrPCF8583 | WR);
        TrxI2C(0x08); /* endereço reg. controle alarme */
        Start_I2C_Bus();
        TrxI2C(AddrPCF8583 | RD);
        uc = RcxI2C_no_ack();
        /* se a função de alarme está habilitada */
        if(uc & 0x30){
            /* se o sinal de alarme está habilitado */
            if(uc & 0x80){
                /* se a "flag" de interr. está em "high" */
                if(ud & 0x2){
                    Stop_I2C_Bus();
                    return 1;
                }
            }
        }
    }
}

```



```

    Stop_I2C_Bus();
    return 0;
}

/*
 * ClearIntAlarm: limpa o sinal de interrupcao do pino 7
 * do relógio calendario (ativo em "low")
 * O valor de retorno indica se as operacoes de escrita
 * no barramento I2C foram sucedidas de acknowledge
 */

int ClearIntAlarm(void)
{
    unsigned char uc, ue = 0;

    Start_I2C_Bus();
    ue = ue | TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    ue = ue | TrxI2C(0x0); /* endereco reg. controle/status */
    Start_I2C_Bus();
    ue = ue | TrxI2C(AddrPCF8583 | RD); /* endereco relógio */
    uc = RcxI2C_no_ack() & 0xFC;
    Start_I2C_Bus();
    ue = ue | TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    ue = ue | TrxI2C(0x0); /* endereco reg. controle/status */
    ue = ue | TrxI2C(uc);
    Stop_I2C_Bus();
    return !ue;
}

/*
 * DesabilitaAlarmeRelógio: Desabilita a funcao alarme do
 relógio
 * O valor de retorno indica se as operacoes de escrita no
 * barramento I2C foram sucedidas de "acknowledge".
 */

int DesabilitaAlarmeRelógio(void)
{
    unsigned char uc, ue = 0;

    Start_I2C_Bus();
    ue = ue | TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    ue = ue | TrxI2C(0x8); /* endereco reg. controle alarme */
    Start_I2C_Bus();
    ue = ue | TrxI2C(AddrPCF8583 | RD); /* endereco relógio */
    uc = RcxI2C_no_ack() & 0x30;
    Start_I2C_Bus();

```

```

    ue = ue | TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    ue = ue | TrxI2C(0x8); /* endereco reg. controle alarme */
    ue = ue | TrxI2C(uc);
    Stop_I2C_Bus();
    return !ue;
}

```

```

/*****
*****/

```

```

/* arquivo:sinc.c */

```

```

#include "fb.h"

```

```

extern unsigned long BufferQuadro;

```

```

/*
 * BuscaPalavraSinc: Busca receber a palavra código de
 * sincronismo durante Te segundos. Se a recepção for bem
 * sucedida, retorna 1, caso contrário, retorna 0
 * A hora de recepção da palavra de sincronismo
 * é armazenada em TmpSinc
 */

```

```

char BuscaPalavraSinc(long Te, struct tempo *TmpSinc)
{
    long ti, ta;
    unsigned long ul;

```

```

    LeRelógio(TmpSinc);

```

```

    return 1;

```

```

    ti = ta = Data_Segundos(TmpSinc);

```

```

    PreparaRec();

```

```

    do{

```

```

        LeRelógio(TmpSinc);

```

```

        ta = Data_Segundos(TmpSinc);

```

```

        if(input(FQR)){

```

```

            ul = BufferQuadro;

```

```

            output(FQR, 0);

```

```

            if(ul == PALAVRACODSINC){

```

```

                /* desligando receptor */

```

```

                clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */

```

```

                return 1; /* a hora de recebimento da pal. esta em Tmp */
            }

```

```

        }

```

```

    }while( (ta-ti) < Te);

```

```

    /* desligando receptor */

```

```

    clear_bit(PWRTXRX_bit); /* desabilit. alim. tx e rx */

```

```

    return 0;

```

```

}

```

```

/*
Sincroniza: Sincroniza este escravo com a estacao mestre.
*/
void Sincroniza(struct status_t *stt)
{
    struct tempo Hora;

    if(BuscaPalavraSinc(TEMPOESPERASINC, &Hora)){
        DesativaAteProximoComando(stt, &Hora);
    }
    else{
        stt->ContPerdaSinc++;
        if(stt->ContPerdaSinc > MAXNUMPERDASINC){
            /* buscando sincronismo durante todo um lote */
            if(BuscaPalavraSinc(PERIODOSLOT *
(NUMEROSLOTS+1), &Hora)){
                DesativaAteProximoComando(stt, &Hora);
            }
            else{
                DesativaTempoIndeterm(stt);
            }
        }
        else{
            DesativaAteProximoSincronismo(stt);
        }
    }
}

/*
*
*   DesativaAteProximoSincronismo: Determina o proximo
horario no qual
*   e' esperado um sinal de sincronismo, acerta-se o
alarme
*   para este horario, acerta-se e armazena-se o status,
*   e desativa-se a estacao.
*/
void DesativaAteProximoSincronismo(struct status_t *stt)
{
    struct tempo HS;

    stt->ProximaAtivacao = stt->ProximoSinc =
        ProximoSinalSincronismo(stt->ProximoSinc);
    stt->Sincronismo_Comando = SINCRONISMO;
    ArmazenaStatus(stt);
    Segundos_Data( stt->ProximoSinc - MSEGSLINC, &HS);
    AcertaAlarmaData(&HS);
    clear_bit(PWRCTRL);
}

```

```

    for(;;);
}

/*
 * DesativaAteProximoComando: Determina o proximo horario
no qual
 *      e' esperado um comando, acerta-se o alarme para este
 *      horario, acerta-se e armazena-se o status,
 *      e desativa-se a estacao.
 */
void DesativaAteProximoComando(struct status_t *stt, struct
tempo *HS)
{
    long tp;

    tp = Data_Segundos(HS);
    stt->Sincronismo_Comando = COMANDO;
    stt->ContPerdaSinc = 0;
    stt->ProximoSinc = tp + (NUMEROSLOTS * PERIODOSLOT);
    stt->ProximaAtivacao = tp + (MEUSLOT * PERIODOSLOT);
    Segundos_Data(stt->ProximaAtivacao, HS);
    ArmazenaStatus(stt);
    AcertaAlarmeData(HS);
    clear_bit(PWRCTRL);
    for(;;);
}

/*
 *
 * DesativaTempoIndeterm: Desabilita-se o sinal de alarme do
relógio,
 *      e desativa-se a estacao, que entao so pode ser
 *      ativada manualmente.
 */
void DesativaTempoIndeterm(struct status_t *stt)
{
    stt->Sincronismo_Comando = SINCRONISMO;
    ArmazenaStatus(stt);
    DesabilitaAlarmeRelógio();
    ClearIntAlarm();
    for(;;);
}

unsigned long ProximoSinalSincronismo(unsigned long tp)
{
    return tp + NUMEROSLOTS * PERIODOSLOT;
}

```



```

}

/*****
*****/
/*  arquivo:status.c  */

#include "fb.h"
/*
 *   BuscaStatus:   Le as variaveis que indicam para o
microcontrolador
 *   a cada ativavacao qual tarefa deve realizar. Estas
 *   variaveis sao armazenadas na memoria nao volatil
 *   disponivel no "chip" do relógio-calendario.
 *
 *   nome da variavel      funcao
 *   -----|-----
 *   Sincronismo_Comando   |   indicar se a ativacao e'
para
 *   |   receber o sincronismo ou para
 *   |   receber comandos
 *   ContPerdaSinc        |   Contar quantas vezes, devido
 *   |   a possiveis interferencias, a
 *   |   palavra-codigo de sincronismo
 *   |   nao foi recebida.
 *   ProximoSinc          |   Armazenar a hora de recepcao
 *   |   de palavra de sincronismo
 */
int BuscaStatus(struct status_t *st)
{
    unsigned int cs;
    unsigned char uc;

    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    TrxI2C(STATUS); /* endereco status na ram do "chip" relógio
calendario */
    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | RD); /* endereco relógio */
    cs = st->Sincronismo_Comando = RcxI2C_ack();
    st->Comando = RcxI2C_ack();
    cs += st->Comando;
    st->ContPerdaSinc = RcxI2C_ack();
    cs += st->ContPerdaSinc;
    uc = RcxI2C_ack();
    st->ProximoSinc = uc;

```

```

cs += uc;
st->ProximoSinc = st->ProximoSinc << 8;
uc = RcxI2C_ack();
st->ProximoSinc += uc;
cs += uc;
st->ProximoSinc = st->ProximoSinc << 8;
uc = RcxI2C_ack();
st->ProximoSinc += uc;
cs += uc;
st->ProximoSinc = st->ProximoSinc << 8;
uc = RcxI2C_ack();
st->ProximoSinc += uc;
cs += uc;
uc = RcxI2C_ack();
st->ProximaAtivacao = uc;
cs += uc;
st->ProximaAtivacao = st->ProximaAtivacao << 8;
uc = RcxI2C_ack();
st->ProximaAtivacao += uc;
cs += uc;
st->ProximaAtivacao = st->ProximaAtivacao << 8;
uc = RcxI2C_ack();
st->ProximaAtivacao += uc;
cs += uc;
st->ProximaAtivacao = st->ProximaAtivacao << 8;
uc = RcxI2C_ack();
st->ProximaAtivacao += uc;
cs += uc;
st->Checksum = RcxI2C_ack();
st->Checksum = st->Checksum << 8;
st->Checksum += RcxI2C_no_ack();
Stop_I2C_Bus();
if(st->Checksum == cs){
    return 1;
}
return 0;
}

```

```

int ComparaStatus(struct status_t *a, struct status_t *b)
{
    return(
        (a->Sincronismo_Comando == b->Sincronismo_Comando)
        &&
        (a->Comando == b->Comando) &&
        (a->ContPerdaSinc == b->ContPerdaSinc) &&
        (a->ProximoSinc == b->ProximoSinc) &&
        (a->ProximaAtivacao == b->ProximaAtivacao) &&
        (a->Checksum == b->Checksum) );
}

```

```

}

/*
 *   ArmazenaStatus:   Armazena na memoria nao volatil
disponivel
 *                   no "chip" do relógio-calendario as variaveis que
indicam
 *                   para o microcontrolador a cada ativavacao qual tarefa
 *                   deve realizar.
 */
int ArmazenaStatus(struct status_t *stt)
{
    unsigned int cs;
    unsigned char uc;
    struct status_t StCf;

    Start_I2C_Bus();
    TrxI2C(AddrPCF8583 | WR); /* endereco relógio */
    TrxI2C(STATUS); /* endereco status na ram do "chip" relógio
calendario */
    TrxI2C(stt->Sincronismo_Comando);
    cs = stt->Sincronismo_Comando;
    TrxI2C(stt->Comando);
    cs += stt->Comando;
    TrxI2C(stt->ContPerdaSinc);
    cs += stt->ContPerdaSinc;
    uc = stt->ProximoSinc >> 24;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximoSinc >> 16;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximoSinc >> 8;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximoSinc;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximaAtivacao >> 24;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximaAtivacao >> 16;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximaAtivacao >> 8;
    cs += uc;
    TrxI2C(uc);
    uc = stt->ProximaAtivacao;
    cs += uc;

```

```

TrxI2C(uc);
stt->Checksum = cs;
uc = cs >> 8;
TrxI2C(uc);
TrxI2C(cs);
Stop_I2C_Bus();
BuscaStatus(&StCf);
return(ComparaStatus(&StCf, stt) );
}

```

```

/*****
*****/

```

```

/* arquivo crc.c */
#define NROLINHAS 10
int Paridade(unsigned long d);
const unsigned long Bhct[NROLINHAS] = {
    0xD5E4A0001,
    0x6AF250001,
    0x357928001,
    0xCF5830001,
    0x67AC18001,
    0xE632A8001,
    0xA6FDF0001,
    0x537EF8001,
    0xFC5BD8001,
    0xABC948001
};

```

```

unsigned long CalculaCRC(unsigned long d)
{
    int i;
    unsigned long tmp;

    d = d & 0xfffff8001;
    for(i = 0; i < NROLINHAS; i++){
        tmp = Paridade(d & Bhct[i]);
        if(tmp){
            d = d | (tmp << (NROLINHAS - i));
        }
    }
    d = d | Paridade(d);
    return d;
}

```

```

/*****
*****/

```



```

*****/
/* arquivo ping.c
 * rotina de teste do enlace de radio, estacao escrava */

#include <io515.h>
#include "fb.h"

unsigned long BufferQuadro;
void main(void)
{
    unsigned long ul;
    int i;

    PreparaRec();
    for(;;){
        if(input(FQR)){
            ul = BufferQuadro;
            output(FQR, 0);
            PreparaTrans();
            for(i = 0 ; i < 500; i++){
                putframe(ul);
                PreparaRec();
            }
        }
    }
}

/*
 * T2TimerReload : coloca o timer 2 no modo timer reload
 */
void T2TimerReload(int BPS)
{
    long lg;
    unsigned char uc;

    lg = 1000000/BPS;
    lg = 0x100001 - lg;
    output(CRCL, lg);
    output(CRCH, lg/256);
    clear_bit(EXEN2_bit); /* des. T2EX */
    uc = input(T2CON);
    uc = (uc & 0x60) | 0x11;
    output(T2CON, uc);
    /* fclock/12 */
    /* T2 no modo auto-reload por overflow */
    /* timer function */
}

```

```

/*****
*****/
/* arquivo pingd.c
   rotina de teste do enlace de radio, estacao mestre */
#include <io515.h>
#include "fb.h"
unsigned long BufferQuadro;

void main(void)
{
    int i, j;
    unsigned long ul1 = 0;
    unsigned long ul2;

    inidisp();
    for(;;){
        PreparaTrans();
        dprintf("\nVou transmitir...");
        putframe(++ul1);
        PreparaRec();
        dprintf("\nVou esperar...");
        for(i = 0; i < 1000; i++){
            if(input(FQR)){
                ul2 = BufferQuadro;
                output(FQR, 0);
                dprintf("\nRecebi!");
                dprintf("\nQT: %lX", ul1);
                dprintf("\nQR: %lX", ul2);
                for(j = 0 ; j < 5000; j++){
                    break;
                }
            }
        }
    }
}

/*
 * T2TimerReload : coloca o timer 2 no modo timer reload
 */

void T2TimerReload(int BPS)
{
    long lg;
    unsigned char uc;

    lg = 10000001/BPS;
    lg = 0x100001 - lg;
    output(CRCL, lg);
    output(CRCH, lg/256);
    clear_bit(EXEN2_bit); /* des. T2EX */
}

```

```

uc = input(T2CON);
uc = (uc & 0x60) | 0x11;
output(T2CON, uc);
/* fclock/12 */
/* T2 no modo auto-reload por overflow */
/* timer function */
}

```

```

/*****
*****/
/* Arquivo : DPUTCHAR.C  saida para display */
#include <semente.h>
#include <io515.h>
/*
    x: colunas de 0 a 39
    y: linhas de 0 a 1
*/
char linhadisplay = 0; /* de 0 a 1 */
char colunadisplay = 0 ; /* de 0 a 15 */
char bf1[17];
char bf1[17];

void dputchar(char c)
{
    int k;

    if((colunadisplay<16)&&(c !='\n')){
        bf1[colunadisplay++] = c;
        bf1[colunadisplay] = 0;
        dputc(c);
    }
    else{
        if((colunadisplay == 16) ){
            if(linhadisplay == 0){
                gotoxy(0,1);
                dputc(c);
                linhadisplay = 1;
                colunadisplay = 1;
            }
            else{
                gotoxy(0,0);
                k = 0;
                while(bf1[k]){
                    dputc(bf1[k++]);
                }
                while(k<16){
                    dputc(' ');
                }
            }
        }
    }
}

```

```

        k++;
    }
    gotoxy(0,1);
    for(k = 0 ; k < 16 ; k++){
        dputc(' ');
    }
    gotoxy(0,1);
    dputc(c);
    bf1[0] = c;
    bf1[1] = 0;
    colunadisplay = 1;
    }
}
else{
    if(linhadisplay == 0){
        gotoxy(0,1);
        linhadisplay = 1;
        colunadisplay = 0;
    }
    else{
        gotoxy(0,0);
        k = 0;
        while(bf1[k]){
            dputc(bf1[k++]);
        }
        while(k<16){
            dputc(' ');
            k++;
        }
        gotoxy(0,1);
        for(k = 0 ; k < 16 ; k++){
            dputc(' ');
        }
        gotoxy(0,1);
        bf1[0] = 0;
        colunadisplay = 0;
        linhadisplay = 1;
    }
}
}
}
}

```

```

/*****
*****/

```

```

/* arquivo iniser.c InicSer : inicializa o port serial */
#include "io51.h"

```



```

void IniSer(unsigned char ValorTh1)
{
    unsigned char c;

    c = ((input(TMOD) & 0x0f)|0x20);
    output(TMOD,c);
    output(TH1, ValorTh1);
    c = (input(PCON) | 0x80);
    output(PCON,c);
    set_bit(TR1_bit);
    output(SCON, 0x50);
    set_bit(TI_bit);
    return;
}
/*****
*****/
/*
    arquivo:stti2c.c
*/
#include "io515.h"
#include "fb.h"
#define SCL P1_4_bit
#define SDA P1_3_bit

/* rotina de inicializacao barramento I2C */
void Start_I2C_Bus(void)
{
    set_bit(SDA);
    delay(1);
    set_bit(SCL);
    delay(1);
    clear_bit(SDA);
    delay(1);
    clear_bit(SCL);
    delay(1);
    set_bit(SDA);
    delay(1);
    return;
}

/* rotina de finalizacao barramento I2C */
void Stop_I2C_Bus(void)
{
    clear_bit(SCL);
    delay(1);
    clear_bit(SDA);
    delay(1);
    set_bit(SCL);

```

```

    delay(1);
    set_bit(SDA);
    delay(1);
    clear_bit(SCL);
    delay(1);
    return;
}

```

```

/*****
*****/
/* atoi : converte uma string ascii em um inteiro.
   espacos +-dddd */
#include <ctype.h>
int atoi(char *s)
{
    int i = 0;
    int Sinal= 1;
    int NumeroInteiro = 0;

    while(s[i] == ' '){
        i++;
    }
    if((s[i]=='+')){
        i++;
    }
    else{
        if((s[i]=='-')){
            Sinal = -1;
            i++;
        }
    }
    while(isdigit(s[i] & 0x7f)){
        NumeroInteiro = (NumeroInteiro * 10) + ((s[i]-0x30) &
0x7f);
        i++;
    }
    return(NumeroInteiro * Sinal);
}

```

```

/*****
*****/
/* T2Timer: coloca o timer 2 no modo timer, desabilitando a
contagem
*/
#include <io515.h>

```

```

void T2Timer(void)
{
    unsigned char uc;

    clear_bit(EXEN2_bit); /* des. T2EX */
    uc = input(T2CON);
    uc = (uc & 0x60);
    output(T2CON, uc);
}

/*****
*****/
/* arquivo: tempo rotinas auxiliares de temporizacao */
#include <math.h>
#include "fb.h"
int dmes[] = {31,28,31,30,31,30,31,31,30,31,30,31};

int ComparaHora(struct tm *t1, struct tm *t2)
{
    long seg1, seg2;

    seg1 = Data_Segundos(t1);
    seg2 = Data_Segundos(t2);
    if( seg1 > seg2){
        return -1;
    }
    if( seg1 < seg2){
        return 1;
    }
    return 0;
}

long Data_Segundos(struct tm *t)
{
    int i, difano, n3a, n4a;
    long tempo;

    tempo = t->tm_sec + t->tm_min*60 + t->tm_hour*3600 + (t-
>tm_day-1)*NSD;
    difano = t->tm_year - ANOBASE;
    n4a = (difano/4);
    tempo += n4a * NS4A;
    n3a = difano - (n4a*4);
    if(n3a){
        n3a--;
        tempo += NSAB;
        while(n3a){

```

```

        tempo += NSA;
        n3a--;
    }
    for(i = 0; i < t->tm_mon - 1; i++){
        tempo += (dmes[i]) * NSD;
    }
}
else{
    for(i = 0; i < t->tm_mon-1; i++){
        tempo += (dmes[i]) * NSD;
    }
    if(t->tm_mon > 2){
        tempo += NSD;
    }
}
return tempo;
}

void Segundos_Data(long Seg, struct tm *t)
{
    double n4a;
    int i, anb = 0;
    long nsr;

    modf(Seg/NS4A, &n4a);
    t->tm_year = ANOBASE + (n4a*4);
    nsr = Seg;
    for(i = n4a; i > 0; i--){
        nsr -= NS4A;
    }
    if( nsr > NSAB){
        t->tm_year++;
        nsr -= NSAB;
    }
    else{
        anb = 1;
    }
    while(nsr >= NSA){
        t->tm_year++;
        nsr -= NSA;
    }
    if (anb == 1){
        dmes[1] = 29;
    }
    t->tm_mon = 1;
    for(i = 0 ; i < 12; i++){
        if( nsr >= (dmes[i] * NSD) ){
            nsr -= (dmes[i] * NSD);
            t->tm_mon ++;
        }
    }
}

```



```

    }
    else{
        break;
    }
}
t->tm_day = nsr/NSD;
nsr -= t->tm_day*NSD;
t->tm_day++;
t->tm_hour = nsr/3600;
nsr = nsr-(t->tm_hour*3600);
t->tm_min = nsr/60;
t->tm_sec = nsr - (t->tm_min*60);
}

```

```

/*****
*****/
#include <io515.h>
#include "fb.h"
/* PreparaRec: inicializa a recepcao */
void PreparaRec(void)
{
    clear_bit(EAL_bit);
    set_bit(PWRTXRX_bit); /* habilit. alim. tx e rx */
    clear_bit(N6_9V_bit); /* rx */
    output(RecT2H, VRecargaT2 >> 8);
    output(RecT2L, VRecargaT2);
    output(EmRecepcao, 0);
    output(VMinT2H, ValorMinT2 >> 8);
    output(VMinT2L, ValorMinT2);
    output(VLimH, VLim >> 8);
    output(VLimL, VLim);
    output(PBB, 0);
    output(BR0, 0);
    output(BR1, 0);
    output(BR2, 0);
    output(BR3, 0);
    output(FQR, 0);
    T2Timer();
    clear_bit(I3FR_bit); /* PINO P1.0 - int3, int. borda descida
*/
    set_bit(EX3_bit);
    clear_bit(IEX3_bit);
    set_bit(EX4_bit);
    clear_bit(IEX4_bit);
    set_bit(ET2_bit); /* habilita interrupcao timer 2 */
    clear_bit(TF2_bit); /* flag de overflow de T2 */
    set_bit(EAL_bit);

```

```
}

/* PreparaTrans: inicializa a Transmissao */
void PreparaTrans(void)
{
    clear_bit(EAL_bit);
    set_bit(PWRTXRX_bit); /* hab. alim. tx e rx */
    set_bit(N6_9V_bit);
    clear_bit(EX3_bit);
    clear_bit(IEX3_bit);
    clear_bit(EX4_bit);
    clear_bit(IEX4_bit);
    clear_bit(ET2_bit); /* desabilita interrupcao timer 2 */
    clear_bit(TF2_bit); /* flag de overflow de T2 */
    output(ContBit0TrxH, ContBit0Trx >> 8);
    output(ContBit0TrxL, ContBit0Trx);
    output(ContBit1TrxH, ContBit1Trx >> 8);
    output(ContBit1TrxL, ContBit1Trx);
    T2Timer();
    delay(100);
    ptrx();
}

void DesligaTxRx(void)
{
    clear_bit(PWRTXRX_bit); /* desligando tx e rx */
    clear_bit(N6_9V_bit);
}
```