

UNIVERSIDADE FEDERAL DO PARANÁ  
DEPARTAMENTO DE ENGENHARIA ELÉTRICA  
CURSO DE ENGENHARIA ELÉTRICA

**PEDRO ADOLFO DE SOUZA JUNIOR**

**SISTEMA DE CONTROLE DE ACESSO  
GERENCIADO VIA GPRS**

JULHO - 2011

UNIVERSIDADE FEDERAL DO PARANÁ  
DEPARTAMENTO DE ENGENHARIA ELÉTRICA  
CURSO DE ENGENHARIA ELÉTRICA

**PEDRO ADOLFO DE SOUZA JUNIOR**

**SISTEMA DE CONTROLE DE ACESSO  
GERENCIADO VIA GPRS**

Trabalho de Conclusão de Curso de Engenharia Elétrica, apresentado para obtenção de grau no Curso de Engenharia Elétrica da Universidade Federal do Paraná, sob orientação do Prof. Ademar Luiz Pastro.

JULHO - 2011

UNIVERSIDADE FEDERAL DO PARANÁ  
DEPARTAMENTO DE ENGENHARIA ELÉTRICA  
CURSO DE ENGENHARIA ELÉTRICA

JULHO - 2011



## LISTA DE FIGURAS

Figura 1 - Diagrama de Blocos.....	10
Figura 2 - Sistema RFID.....	12
Figura 3 - Protocolo Wiegand.....	15
Figura 4 - Paridade no protocolo Wiegand.....	16
Figura 5 - Arquitetura de um Sistema GPRS.....	18
Figura 6 - Estrutura e Camadas de Protocolos do GPRS.....	21
Figura 7 - Estrutura multi-frame.....	23
Figura 8 - Backbone GPRS.....	24
Figura 9 - Sistema Operacional em Tempo Real Embarcado.....	26
Figura 10 - Arquitetura Geral de um RTOS.....	29
Figura 11 - Sistema Operacional Baseado em Kernel Monolítico.....	30
Figura 12 - Sistema Operacional Baseado em Micro-Kernel.....	30
Figura 13 - Sistema Operacional Baseado em Exokernel.....	31
Figura 14 - Serviços do Kernel de um RTOS.....	32
Figura 15 - Leitor de Proximidade HID/Indala.....	35
Figura 16 - Modem GPRS Urmet Daruma.....	38
Figura 17 - M52259DEMOKIT - Freescale.....	39
Figura 18 - Diagrama de Blocos MCF5225x.....	40
Figura 19 - Conector J4 da placa M52259DEMOCOM.....	42
Figura 20 - Esquemático do Hardware Externo.....	43
Figura 21 - LM35 - Tensão x Temperatura.....	44
Figura 22 - Organização do MQX.....	46
Figura 23 - Estrutura da Aplicação.....	47
Figura 24 - Fluxograma da INIT task.....	47
Figura 25 - Fluxograma da RFID task.....	49
Figura 26 - Fluxograma da GPRS task.....	50
Figura 27 - Fluxograma LED task.....	51
Figura 28 - Fluxograma do Software de Gerenciamento.....	52
Figura 29 - Formulário de Cadastro de Usuário.....	53
Figura 30 - Foto do Hardware Externo Finalizado.....	56
Figura 31 - Foto com todos os periféricos conectados.....	58
Figura 32 - Debug na inicialização pela serial.....	58
Figura 33 - Execução de leitura de um cartão RFID.....	59

## LISTA DE TABELAS

Tabela 1 - Classificação RFID.....	14
Tabela 2 - Classificação RFID.....	15
Tabela 3 - Classe de Multislot dos Terminais.....	20
Tabela 4 - Funções dos pinos utilizados no projeto .....	43
Tabela 5 - Mensagens recebidas pelo Modem GPRS .....	50
Tabela 6 - Mensagens enviadas pelo Modem GPRS.....	50

## LISTA DE SIGLAS

AC -	Corrente Alternada
APN -	<i>Access Point Name</i>
BGP -	<i>Border Gateway Protocol</i>
BSS -	<i>Base Station System</i>
BSSGP -	<i>Base Station System GPRS Protocol</i>
BTS -	<i>Base Transceiver Station</i>
LC -	Circuito Capacitivo e Indutivo
ADSL -	<i>Asymmetric Digital Subscriber Line</i>
CHAP -	<i>Challenge Handshake Authentication Protocol</i>
C# -	C-Sharp – Linguagem de Programação
DL -	<i>Downlink</i>
EAS -	<i>Electronic Article Surveillance</i>
EIR -	<i>Equipment Identify Register</i>
IP -	<i>Internet Protocol</i>
IPsec -	<i>IP Security Protocol</i>
GGSN -	<i>Gateway GPRS Support Node</i>
GMSC -	<i>Gateway Mobile Services Switching Center</i>
GPRS -	<i>General Packet Radio Service</i>
GSM -	<i>Global System for Mobile Communications</i>
GTP -	<i>GPRS Tunneling Protocol</i>
HLR -	<i>Home Location Register</i>
LLC -	Logical Link Control
MAC -	<i>Media Access Control</i>
MS -	<i>Mobile Station</i>
MSC -	<i>Mobile Service Switching Center</i>
MT -	<i>Mobile Terminal</i>
NAT -	<i>Network Address Translation</i>
NS -	<i>Network Services</i>
OSI -	<i>Open Systems Interconnection</i>
OSPF -	<i>Open Shortest Path First</i>
PAP -	Password Authentication Protocol

PCU -	<i>Packet Control Unit</i>
PDP -	<i>Packet Data Protocol</i>
PDU -	<i>Protocol Data Unit</i>
PVC -	<i>Permanent Virtual Circuit</i>
PTP -	<i>Point-to-Point</i>
QoS -	<i>Quality of Service</i>
RADIUS -	<i>Remote Authentication Dial In User Service</i>
RFID -	<i>Radio Frequency Identification</i>
RIP -	Routing Information Protocol
RLC -	<i>Radio Link Control</i>
RTOS -	<i>Real Time Operating System</i>
SNDCP -	<i>Subnetwork Dependent Convergence Protocol</i>
SGSN -	<i>Serving GPRS Support Node</i>
SMS -	<i>Short Message Service</i>
TAG -	Modo como são chamado os dispositivos RFID
TCP -	<i>Transmission Control Protocol</i>
TDMA -	<i>Time Division Multiple Access</i>
TE -	<i>Terminal Equipment</i>
UL -	<i>Uplink</i>
UDP -	<i>User Datagram Protocol</i>
VLR -	<i>Visitor Location Register</i>



## SUMÁRIO

1.	INTRODUÇÃO.....	8
2.	PROPOSTA.....	9
3.	FUNDAMENTOS TEÓRICOS .....	11
3.1.	RFID.....	11
3.1.1.	História.....	11
3.1.2.	Funcionamento .....	11
3.1.3.	Aplicação .....	13
3.1.4.	Classificação.....	14
3.1.5.	Protocolo Wiegand.....	15
3.2.	GPRS .....	16
3.2.1.	Visão Geral de um sistema GPRS .....	17
3.2.2.	Arquitetura de um Sistema GPRS.....	18
3.2.3.	Protocolos de Transmissão GPRS.....	21
3.2.4.	Implementação da rede GPRS .....	23
3.3.	RTOS .....	25
3.3.1.	Por que utilizar um RTOS em aplicações em Tempo Real .....	26
3.3.3.	Conceitos equivocados sobre RTOS .....	27
3.3.4.	Características de um RTOS .....	27
3.3.5.	Arquitetura de um RTOS.....	29
3.3.6.	Kernel.....	29
3.4.1.	Principais características .....	32
4.	COMPONENTES PRINCIPAIS DO SISTEMA DE CONTROLE DE ACESSO 34	
4.1.	Leitor RFID Indala/HID 5365 (Wiegand) .....	34
4.1.1.	Características .....	34
4.1.2.	Tecnologia FlexSecur .....	35
4.2.	Modem GPRS Daruma .....	37
4.2.1.	Principais Características.....	38
4.3.	M52259DEMOKIT .....	39
4.3.1.	Familia MCF5225x .....	40
4.3.2.	Características - M52259DEMOMCU .....	40
4.3.3.	Características - M52259DEMOCOM .....	41
5.	ETAPAS DO DESENVOLVIMENTO .....	41
5.1.	Hardware Externo .....	42

5.1.1.	Esquemático .....	43
5.2.	Software Embarcado com RTOS MQX.....	45
5.2.1.	MQX-RTOS.....	45
5.2.2.	Estrutura da Aplicação para o Sistema de Controle de Acesso .	47
5.2.3.	INIT <i>task</i> .....	47
5.2.4.	RFID <i>task</i> .....	48
5.2.5.	GPRS <i>task</i> .....	49
5.2.6.	LED <i>task</i> .....	51
5.3.	Software de Gerenciamento em C#.....	51
6.	MODO DE FUNCIONAMENTO .....	52
6.1.	Cadastro de usuários .....	53
6.2.	Inicialização Geral do Sistema .....	54
6.3.	Requisições do Software de Gerenciamento .....	54
6.4.	Polling .....	55
7.	RESULTADOS .....	55
7.1.	Comunicação Serial .....	55
7.2.	Hardware Externo .....	56
7.2.1.	Construção.....	56
7.2.2.	Interface para o leitor RFID .....	57
7.2.3.	Demais circuitos .....	57
7.3.	Debug .....	58
7.4.	Gerenciamento.....	59
8.	CONCLUSÃO .....	60
9.	REFERÊNCIAS .....	62

## 1. INTRODUÇÃO

A falta de segurança hoje em dia é uma constante em nosso cotidiano. A falta dela, pelo descaso das autoridades públicas leva as grandes e pequenas empresas a investirem um alto valor de seu orçamento em soluções de segurança que visem diminuir ou pelo menos coibir a ação de marginais contra o seu patrimônio.

O presente trabalho foi desenvolvido com o objetivo de oferecer uma alternativa de solução para um problema em que Operadoras de Telefonia Fixa e Móvel e Operadoras de TV à cabo estão tendo com segurança que este trabalho foi desenvolvido. O problema citado trata-se de furtos de baterias, usadas para alimentação dos equipamentos instalados em armários espalhados por diversos pontos das cidades, com o objetivo de fornecer o acesso aos serviços oferecidos por estas operadoras a seus clientes. Os furtos ocorrem muitas vezes devido a facilidade do acesso aos locais onde se encontram as baterias, e a falta de controle deste acesso, pois em grande maioria dos casos os furtos são realizados por funcionários da própria Operadora ou funcionários de empresas terceirizadas contratadas pelas Operadoras para prestação de serviços de Manutenção.

O fornecimento do Controle de Acesso a esses armários para as Operadoras é então a solução que este trabalho apresentará, tendo como um objetivo final a apresentação de um protótipo para ratificar a viabilidade da solução.

O trabalho envolve tecnologias como RFID, GPRS além de um software embarcado fazendo uso de um Sistema Operacional em Tempo Real e uma aplicação desenvolvida para gerenciar o controle de acesso. Será então apresentado ao longo do trabalho um pouco sobre cada uma dessas tecnologias utilizadas para compor a solução final, além de um detalhamento completo sobre o desenvolvimento do Hardware, Software Embarcado e Software de gerenciamento.

## 2. PROPOSTA

Com o problema apresentado e uma possível solução sendo o Controle de Acesso, este capítulo apresenta uma proposta visando atender as necessidades das Operadoras em gerenciar o acesso aos seus armários espalhados pelas cidades.

Cada um desses armários, chamados a partir de agora de Pontos Remotos, terão um módulo instalado o qual será responsável por liberar ou não o acesso para o funcionário. Existirá uma central de controle, com um servidor onde será instalado o software de gerenciamento com sua base de dados.

Os requisitos básicos do Projeto são:

- Comunicação entre a central e os Pontos Remotos sem fio
- Identificação de cada acesso aos Pontos Remotos
- Alarme caso ocorra acesso ao Ponto Remoto sem autorização
- Software de Gerenciamento capaz de gerenciar vários pontos remotos ao mesmo tempo
- Histórico dos acessos realizados

Foi definido como sendo uma facilidade a mais no Projeto, porém se não atendida não implica em um bloqueio o Controle de Temperatura em cada um dos Pontos Remotos.

Para atender aos requisitos acima propostos foram utilizadas as seguintes tecnologias:

- RFID – "Radio-Frequency IDentification"
- GPRS – "General Radio Packet Service"
- RTOS embarcado
- Linguagem de Programação C#

Essas tecnologias foram escolhidas por atenderem os requisitos e por serem de uso freqüentes em soluções similares a esta apresentada neste trabalho.

Com os requisitos e as tecnologias para atendê-los definidos, o próximo passo foi definir o hardware que fará parte do módulo que será instalado em cada um dos Pontos Remotos, principalmente o processador que atenderá ao requisito suportar um RTOS embarcado.

Por haver interesse no desenvolvimento com processadores de 32bits foi escolhido o M52259DEMOKIT da Freescale, que terá suas características expostas

no capítulo quatro, item 4.3 o qual apresenta os componentes principais envolvidos no Projeto.

Para a identificação do acesso dos funcionários via RFID a leitora utilizada foi a de marca Indala que suporta a leitura de dados a 125kHz a qual terá suas características expostas no capítulo quatro, item 4.1.

A comunicação sem fio entre o software de gerenciamento e os Pontos Remotos se dará através de GPRS. Para fazer uso desta tecnologia é preciso ter um dispositivo que a suporte. No caso foi utilizado um modem GPRS da marca Daruma que faz uso de um módulo GPRS da Siemens MC35i. A comunicação entre o M52259DEMOKIT e o modem é feita através do protocolo RS-232. GPRS será tratado no capítulo três, item 3.2.

Com o hardware que fará parte do módulo a ser instalado nos Pontos Remotos definido, já é possível desenhar um diagrama de blocos para a proposta, apresentado na Figura 1.

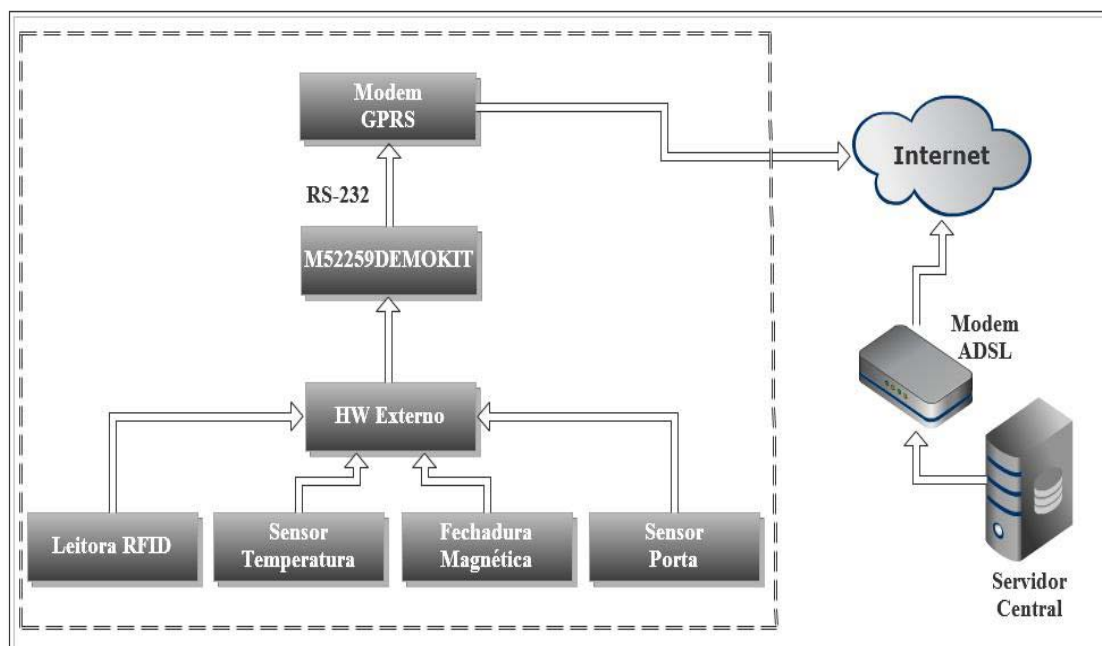


Figura 1 - Diagrama de Blocos

O modo de funcionamento da solução será apresentado no Capítulo 6 após todas as tecnologias que foram utilizadas terem seus aspectos principais expostos no Capítulo 3, a descrição dos componentes principais do sistema no Capítulo 4 e as etapas do desenvolvimento no Capítulo 5.

### 3. FUNDAMENTOS TEÓRICOS

Para o bom desenvolvimento do Projeto foi necessária uma ampla pesquisa sobre as tecnologias utilizadas na solução, pois quanto o maior o domínio sobre as mesmas menor será a dificuldade no desenvolvimento.

#### 3.1. RFID

##### 3.1.1. História

A primeira patente registrada para uma *tag* RFID, como são chamados os cartões, etiquetas e outros dispositivos que fazem uso desta tecnologia, foi feita em 23 de janeiro de 1973 por Mario W. Cardullo. No mesmo ano, Charles Walton, um empreendedor da Califórnia, registrou uma patente para um *transponder* passivo usado para destravar uma porta sem chave. Um cartão com um *transponder* embutido mandava um sinal para um leitor próximo a porta. Quando o leitor detectava um número de identificação válido armazenado dentro do tag RFID, o leitor liberava a porta. Walton licenciou a tecnologia para Schlage, um fabricante de cadeados, e outras empresas. [Ref. 3]

Após isso muito se desenvolveu sobre RFID. Empresas interessadas na possibilidade de controlar suas mercadorias de forma segura e rápida investiram muito no desenvolvimento para aprimoramento das *tags* RFID e suas respectivas leitoras.

##### 3.1.2. Funcionamento

Na Figura 2 é mostrado um sistema RFID. O leitor consiste em um transceptor de radiofrequência que opera em uma das frequências padronizadas para este tipo de aplicação (125 kHz, 13,56 MHz, 862 a 932 MHz ou 2,4 GHz).

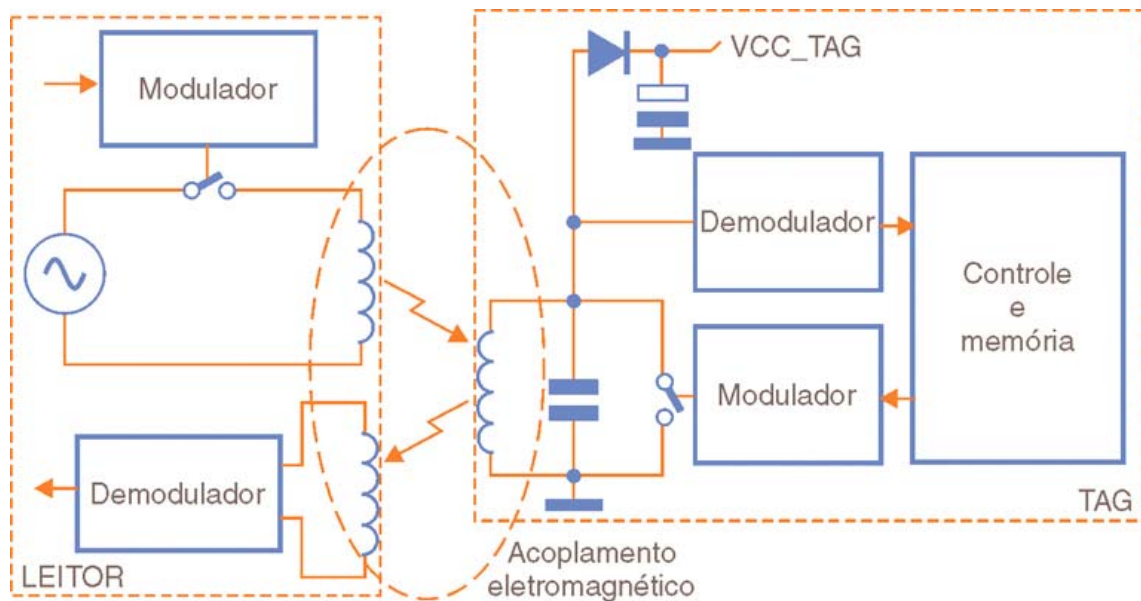


Figura 2 - Sistema RFID [Ref. 5]

Do outro lado temos um *transponder*, que irá responder aos comandos recebidos do leitor. Existem três tecnologias para implementação dos *transponders*: passivos (não possuem qualquer fonte de alimentação integrada), semi-passivos (possuem uma fonte de energia integrada apenas para alimentar a etapa de recepção) e ativos (possuem uma fonte de energia integrada e um circuito de transmissão ativo, o que garante um alcance muito maior).

No momento em que a tag é aproximada do leitor, a radiação eletromagnética emitida pela antena do leitor permeia a tag, induzindo uma tensão AC no circuito LC da mesma. Essa tensão é retificada e armazenada em um capacitor. Quando o circuito de controle da tag detecta que a tensão (VCC\_TAG na Figura 2) ultrapassou um determinado valor, as demais etapas da tag entram em operação e ela decodifica o sinal proveniente do leitor, devolvendo em seguida outro sinal codificado.

Esta transmissão da tag para o leitor é feita normalmente estabelecendo um curto-circuito na bobina, o que provoca uma alteração no campo eletromagnético que pode ser percebida pelo leitor. O leitor recebe este sinal e o envia a um circuito de processamento que irá verificar os dados recebidos. Todo este processo é realizado em uma fração de segundo.

### 3.1.3. Aplicação

Os sistemas RFID podem ser agrupados em quatro categorias de aplicação:

- Sistemas EAS (Electronic Article Surveillance)
- Sistemas Portáteis de Captura de Dados
- Sistemas em Rede
- Sistemas de Posicionamento

Os sistemas EAS são tipicamente sistemas de um bit, usados para identificação da presença ou falta de um item. O largo uso dessa tecnologia está nos bloqueios das lojas onde cada item possui uma *tag* e grandes antenas de leitura são colocadas em cada saída das lojas para detectar a saída desautorizada de um item.

Os sistemas portáteis são caracterizados pelo uso de terminais portáteis de coleta de dados, onde um sistema RFID está integrado do leitor com a antena. São utilizados em aplicações onde um alto grau de itens com *tags* pode ser exibido. Os terminais do tipo *hand-held* capturam os dados dos itens e então são transmitidos a um sistema de processamento central.

Sistemas em rede são aplicações caracterizadas pelo posicionamento fixo dos transceptores (leitores) e conectados por uma rede a um sistema de gerenciamento central. Os transceptores são fixados numa posição e os itens com os *tags* movem-se por esteiras, ou com pessoas, dependendo da aplicação.

Os sistemas de posicionamento usam *tags* para facilitar a locação automática e suporte de navegação para dirigir veículos. Os transceptores são localizados a bordo dos veículos e conectados por um sistema de transmissão a um sistema de gerenciamento central.

Onde haja a necessidade de coleta de dados um sistema RFID pode ser utilizado. O potencial de aplicação de sistemas RFID é enorme, tanto no setor da indústria, comércio e serviço. As principais áreas de aplicação dos sistemas RFID que atualmente podem ser identificadas são:

- Transporte e logística
- Fabricação e processamento
- Segurança

Existem outras áreas de atuação de RFID porém estas não são de uso comum no Brasil, porém em alguns países já bastante comum. São elas:

- Marcação de animal



- Acompanhamento postal
- Bagagem de aviões
- Controle de acesso a veículo
- Postos de pedágio
- Coleta de dados de medições de consumo de energia

O constante desenvolvimento de novos produtos RFID, a regulamentação e a redução de custos têm provocado o surgimento de novas aplicações em áreas até então ainda não exploradas.

### 3.1.4. Classificação

As Tabelas 1 e 2 a seguir trazem a classificação dos sistemas RFID por frequência, suas aplicações e particularidades.

Tabela 1 - Classificação RFID [Ref. 5]

<b>Tipo</b>	<b>LF</b>	<b>HF</b>
<b>Faixa de Frequência</b>	125 ou 134,2 kHz	13,56 MHz
<b>Alcance para leitura</b>	< 0,5m	≥ 1m
<b>Particularidades</b>	Precisa de antenas grandes, o que resulta em altos custos de produção. Sofre pequena degradação do sinal na presença de líquido e metais.	Melhor aproveitamento em alcance que as LF. É a melhor escolha para sistemas que não exijam um longo alcance e não seja necessária a leitura de um grande número de etiquetas ao mesmo tempo.
<b>Fontes de Energia</b>	Acoplamento magnético (campo próximo)	Acoplamento magnético (campo próximo)
<b>Aplicações Típicas</b>	Controle de Acesso Identificação de animais Imobilização de veículos	Controle de Acesso Controle de pagamento Identificação de objetos Controle de bagagens
<b>Leitura Múltipla</b>	Lenta	Média
<b>Leitura em ambientes metálicos ou com líquidos</b>	Melhor	Média
<b>Tamanho da etiqueta</b>	Grande	Médio

Tabela 2 - Classificação RFID [Ref. 5]

Tipo	UHF	Microondas
Faixa de Frequência	860 ou 930 MHz	2,45 ou 5,8 GHz
Alcance para leitura	Entre 4 e 5 m	≥ 1m
Particularidades	Muito mais baratas que as LH e HF. Tem os chips mais avançados e permitem a leitura de múltiplas etiquetas ao mesmo tempo.	Características parecidas com a UHF, com diferença de ser muito mais rápida na transmissão de dados. É muito afetada na presença de líquidos e metais.
Fontes de Energia	Acoplamento eletromagnético (campo distante)	Acoplamento eletromagnético (campo distante)
Aplicações Típicas	Identificação de caixas de equipamentos	Coleta de dados em tempo real. A frequência 5,8GHz vem sendo abandonada pelos sistemas RFID
Leitura Múltipla	Rápida	Rápida
Leitura em ambientes metálicos ou com líquidos	Ruim	Pior
Tamanho da etiqueta	Pequeno	Pequeno

### 3.1.5. Protocolo Wiegand

Para a transmissão dos dados lidos pelo transceptor para o sistema de gerenciamento dos dados o leitor faz uso de um protocolo de transmissão dos dados. O protocolo utilizado em grande maioria dos sistemas, inclusive no apresentado neste trabalho é o protocolo Wiegand 26 bits.

Este protocolo faz uso de duas linhas de dados D0 e D1 as quais indicam bits 0 e 1 respectivamente quando em nível baixo. A Figura 3 traz uma ilustração de como seria a identificação de um bit 0 ou bit 1, porém nesta ilustração estes ocorrem quando estas linhas estão em nível alto.

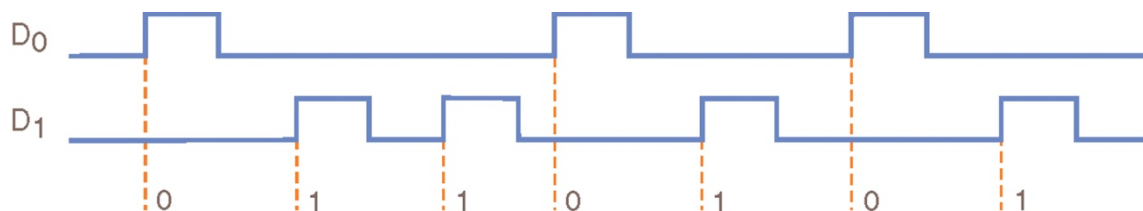


Figura 3 - Protocolo Wiegand [Ref. 5]

Na transmissão dos 26 bits que compõe o protocolo Wiegand dois bits são usados para paridade, a fim de checar a integridade do código transmitido. A Figura 4 mostra como se dá a transmissão dos bits de paridade.

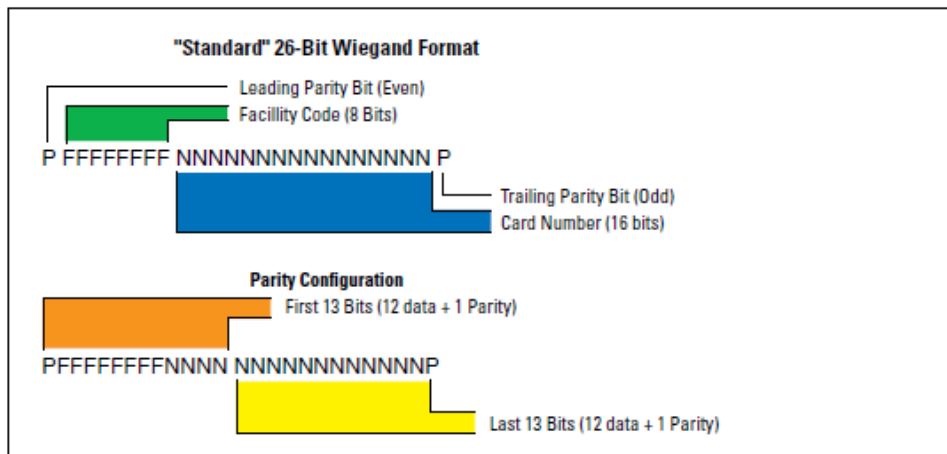


Figura 4 - Paridade no protocolo Wiegand

Para a conversão dos bits recebidos da linha D0 e D1 em códigos válidos é necessário seguir os passos abaixo:

- 1 Identificar 26 interrupções de borda de descida (D0/D1)
- 2 Iniciar o armazenamento pela direita (LSB)
- 3 Checar a paridade de acordo com a Figura 4
- 4 Remover os bits de paridade
- 5 Separar os bits de acordo com a Figura 4 a fim de formar o *facility code* e o *card number*.
- 6 Converter em hexadecimal os dois valores (*facility/card number*)

O leitor RFID que foi utilizado no projeto possui uma particularidade a mais, que será mencionada no Capítulo 5, onde as etapas de desenvolvimento serão relatadas. Esta particularidade não influi na tecnologia RFID, é só mais uma facilidade de segurança implementada pelo fabricante deste leitor.

### 3.2. GPRS

O Sistema Global de Comunicações Móveis (GSM) é um sistema de telefonia celular que essencialmente provê voz e serviços de dados por comutação de circuitos. Um sistema baseado no TDMA, GSM provê uma taxa de transferência de dados de aproximadamente 14.4kbps por *timeslot* TDMA.

Com o avanço da Internet e serviços baseados em WAP, um serviço de dados que ofereça comutação por pacotes parece ser mais apropriado, especialmente se o acesso aos dados é sobre uma interface de rádio, onde o custo

sobre o tempo de conexão é um importante parâmetro. O Serviço de Rádio de Pacote Geral (GPRS) foi desenvolvido para fornecer o serviço de comutação por pacotes sobre as interfaces de radio. Sendo um serviço de pacotes de dados, GPRS otimiza o uso da rede e recursos de radio. Baseado no mesmo espectro dos serviços de radio com mais ou menos a mesma arquitetura que o GSM, o GPRS permite uma fácil migração para uma operadora de GSM para GPRS.

### 3.2.1. Visão Geral de um sistema GPRS

O GPRS é uma evolução do GSM e, além disso, também é um sistema TDMA – com cada frequência dividida em quadros TDM e cada quadro TDMA contendo oito *timeslots*. A frequência e os *timeslots* são alocados dinamicamente pela rede para a estação móvel (MS) para transferência de dados sentido *uplink* (UL) e (DL) *downlink*. Recursos para UL e DL são alocados independentemente. Dependendo dos recursos alocados para a MS e a capacidade da MS, as taxas variam de 9 a 150kbps por usuário.

A operadora pode dinamicamente alocar recursos de radio entre a rede GSM que usa comutação por circuitos e a rede GPRS que faz uso de comutação por pacotes, dependendo do tipo de serviço.

GPRS suporta os protocolos comuns usados na transferência de dados como IP e X.25. Os protocolos IP e X.25, Unidades de Protocolo de Dados (PDUs), são transferidos de forma transparente pela rede GPRS entre um usuário a um nó destino IP ou X.25. Isto é conseguido através de túneis e encapsulamento PDU na rede GPRS. Isto permite adicionar novas redes ao GPRS, sem alterar a interface de radio GPRS.

GPRS suporta vários perfis de Qualidade de Serviço (QoS) para transferência de dados. Ele é projetado para reserva rápida para iniciar a transmissão de pacotes, tipicamente 0.5 a 1 segundo. A tarifação normalmente depende da quantidade de dados transferidos num determinado tempo em que a estação móvel (MS) fica conectada a rede. Os procedimentos de autenticação são idênticos ao GSM.

### 3.2.2. Arquitetura de um Sistema GPRS

A parte do sistema GPRS que realiza a comutação dos pacotes é chamada de SGSN (Serving GPRS Support Node). Ele é o centro da rede e fornece o roteamento para as demais partes, fazendo a interface entre a rede de comutação por pacotes (rede de dados) e a rede de comutação por circuitos (HLRs, EIRs e outros). Os principais componentes da arquitetura lógica de um sistema GPRS são mostrados abaixo na Figura 5.

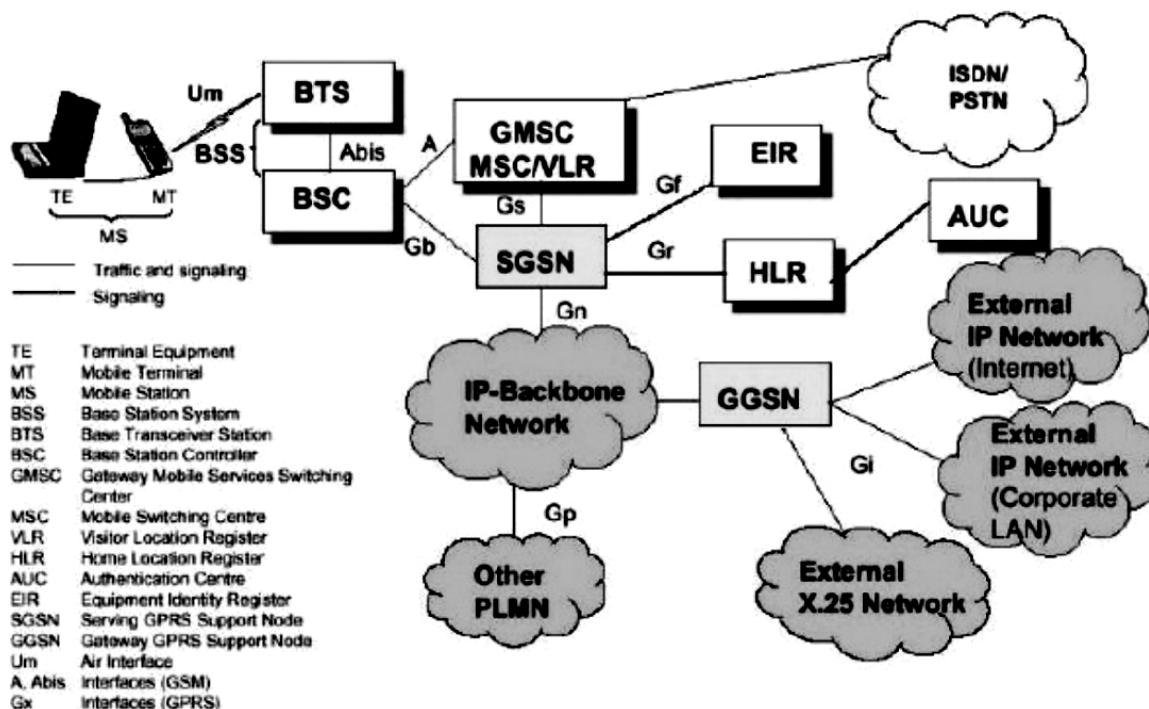


Figura 5 - Arquitetura de um Sistema GPRS [Ref. 8]

**TE (Terminal Equipment):** É o terminal onde o usuário final trabalha caracterizado pelo computador, o sistema recebe endereçamento IP para conectividade em uma rede local ou Internet.

**MT (Mobile Terminal):** É como um modem que fornece a conexão do TE na rede GPRS utilizando uma ligação com SGSN. É estabelecido um túnel entre o TE e o SGSN.

**MS (Mobile Station):** A junção de um TE e um MT formam um MS. Os componentes necessários para o MT e TE são agrupados no conceito de GPRS em um único equipamento. Dependendo da característica da rede GPRS, os MSs podem operar em três modalidades diferentes:

- **Classe A:** operação permite que um MS tenha uma conexão comutada por circuito ao mesmo tempo em que é envolvida em transferência do pacote.
- **Classe B:** esta operação permite que um MS faça conexão comutada por pacote e comutada por circuito, mas não ao mesmo tempo. Entretanto, o MS pode ser envolvido em transferência de pacote e receber uma chamada para o tráfego de comutação por circuito. O MS pode então suspender transferência do pacote enquanto durar a conexão por circuito e mais tarde recomeçar transferência do pacote.
- **Classe C:** esta modalidade de operação permite que um MS esteja ligado somente a um serviço. Um MS que suporta somente GPRS e o tráfego não comutado por circuito, trabalhará sempre na modalidade da classe C de operação.

**BSS** (*Base Station*): Uma BSS consiste de uma BSC (Base Station Controller) e uma BTS (Base Transceiver Station). O BTS é um equipamento de rádio que envia e recebe informações sobre a comunicação das BSCs com os MSs. Um grupo de BTSs é controlado por um BSC. O BTS deve conter softwares específicos para GPRS. A BSC gerencia as chamadas comutadas por circuitos e comutadas por pacote e deve ser equipada com software e hardware para o sistema GPRS. O BTS separa as chamadas comutadas por circuito MS originadas da transmissão de dados do pacote, antes que o BSC envie chamadas para MSC/VLR, e dados da comutação por pacote para o SGSN.

**MSC** (*Mobile Service Switching Center*): é responsável pelo tráfego de pacotes que tem os MSs como origem ou destino e por funções como autenticação, roteamento, gerenciamento de mobilidade, controle de acesso e contabilidade de uso da rede de rádio.

**GMSC** (*Gateway Mobile Services Switching Center*): Faz o roteamento das chamadas entre a rede GSM e a PSTN. Não há mudança neste produto para o GPRS.

**GGSN** (*Gateway GPRS Support Node*): gerencia o roteamento entre a rede GPRS e outras redes de dados (Internet, X.25, etc). Também é responsável por controlar a alocação de endereços IP por parte dos MSs e por traduzir os formatos

de pacotes de endereços externos para o formato de endereçamento GPRS e vice-versa.

**HLR** (*Home Location Register*): é a base de dados que armazena a informação da subscrição para cada pessoa que possui uma conta na operadora de GSM/GPRS. O HLR armazena a informação tanto para uma comunicação por circuito como por pacote. A informação encontrada no HLR inclui, por exemplo, serviços suplementares, parâmetros de autenticação, Access Point Name (APN) como o *Internet Service Provider*, endereço IP. Além disso, o HLR inclui a informação sobre a posição (localização Geografica) do MS. Para o GPRS, a informação do cliente é trocada entre HLR e SGSN. Nota-se que as informações de autenticação para GPRS são recuperadas diretamente do HLR para SGSN. São armazenadas nos HLRs informações de clientes quando se destina a outras operadoras, *roaming*.

**VLR** (*Visitor Location Register*): é uma base de dados que contém a informação sobre todos os MSs que ficam situados atualmente na região da MSC ou da distribuição de SGSN respectivamente. O SGSN contém a funcionalidade de VLR para uma comunicação por pacotes.

**SGSN** (*Serving GPRS Support Node*): um componente primário na rede GSM para o uso do GPRS. O SGSN envia e recebe pacotes destinados para a MS que estejam dentro da área de serviço SGSN. O SGSN fornece roteamento de pacotes e serve a todos os usuários GPRS que estão fisicamente dentro da área de serviço geográfica de SGSN. Um usuário GPRS pode ser servido por todo o SGSN na rede, tudo depende da localização do mesmo.

As especificações do GSM definem 29 classes para os terminais conforme o número de *timeslots* utilizados na recepção ou transmissão. Assim um terminal classe 8 ou 4 + 1 é aquele que pode receber dados em 4 *timeslots* e enviar em 1. A classe varia com o modelo do terminal sendo os mais comuns os que suportam até as 12 primeiras classes apresentadas na Tabela 3.

Tabela 3 - Classe de Multislot dos Terminais

Classe de Multislot	Número máximo de time-slots		
	Rx	Tx	Soma
1	1	1	2

2	2	1	3
3	2	2	3
4	3	1	4
5	2	2	4
6	3	2	4
7	3	3	4
8	4	1	5
9	3	2	5
10	4	2	5
11	4	3	5
12	4	4	5

### 3.2.3. Protocolos de Transmissão GPRS

A interface chamada *Um* é o link lógico na comunicação de um MS (*Mobile Station*) e uma BSS (*Base Station System*). A estrutura de protocolos responsável pela transmissão de dados do usuário, é construída na forma de camadas, semelhante a estrutura de camadas OSI. A primeira camada é implementada no BTS (*Base Transceiver Station*). O PCU (*Packet Control Unit*), como é um novo componente, é tratado em outra camada do protocolo. A Figura 6 mostra a estrutura de camadas de protocolo para o GPRS.

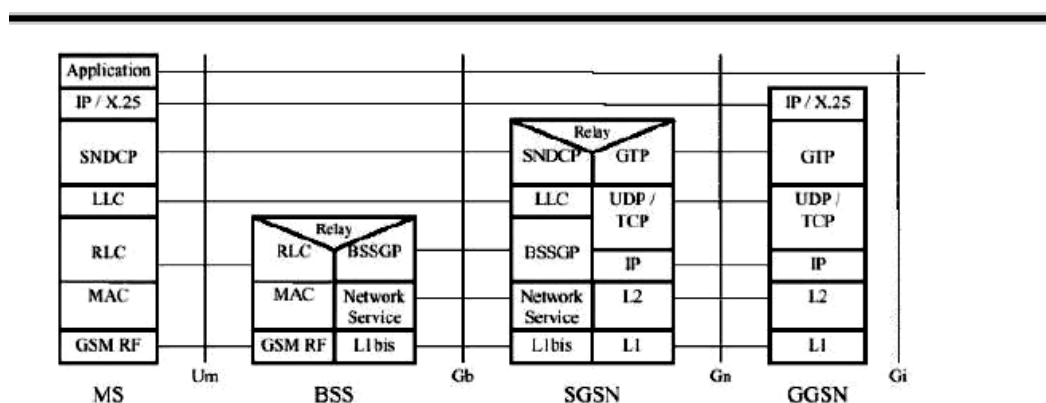


Figura 6 - Estrutura e Camadas de Protocolos do GPRS

**SNDCP** (*Subnetwork Dependent Convergence Protocol*) permite às camadas superiores acessarem as facilidades de transmissão ao nível de rede.



Localiza-se abaixo da camada de rede e acima da LLC (Logical Link Control). Possui as seguintes subfunções:

- Multiplexação dos pacotes em uma ou várias camadas;
- Compressão de protocolos de informação e controle bem como dados do usuário;
- Segmentação e montagem.

A camada **LLC** (*Logical Link Protocol*) estabelece conexões lógicas cifradas independentemente dos protocolos da interface de rádio entre os MS e a SGNS. O LLC permite que o usuário permaneça com a mesma conexão quando se move entre as células de uma mesma SGNS. A LLC suporta:

- Processos de transferência de PDUs (Packets Data Unit) LLC em ambos os modos *Acknowledged* e *unacknowledged*.
- Processo de detecção de correção de PDUs LLC perdidos ou corrompidos.
- Processo de controle de fluxo e cifragem de PDUs LLC.

As camadas **RLC** (*Radio Link Control*) / **MAC** (*Media Access Control*) realizam a conexão entre o MS e o BSS através da interface aérea.

A camada **GSM RF** é a conexão física (via radio frequência) entre a MS e a BSS.

O **BSSGP** (*Base Station System GPRS Protocol*) transporta informações de rotas e parâmetros de QoS (*Quality of Service*) entre a BSS e o SGSN. O NS (*Network Services*) transporta o PDU do BSSGP.

**L1** e **L2** são as camadas de enlace e meio físico.

O **GTP** (*GPRS Tunneling Protocol*) é responsável pelo tunelamento dos dados do usuário entre o SGSN e os elementos do Backbone GPRS. O GTP encapsula todos os dados PTP (Point-to-Point), PDP (Packet Data Protocol) e PDU (Protocol Data Units). Fornece mecanismos de controle de fluxo entre os GGSNs caso seja solicitado.

**TCP** carrega os PDUs no backbone da rede GPRS para os protocolos que necessitam de transporte confiável, como o x.25.

**UDP** utilizado para protocolos que não necessitam de confiabilidade na transmissão.

**IP** fornece o roteamento dos pacotes e sinalização.

Na Figura 7 é mostrada a estrutura de multi-frame, encapsulada nas diversas camadas.

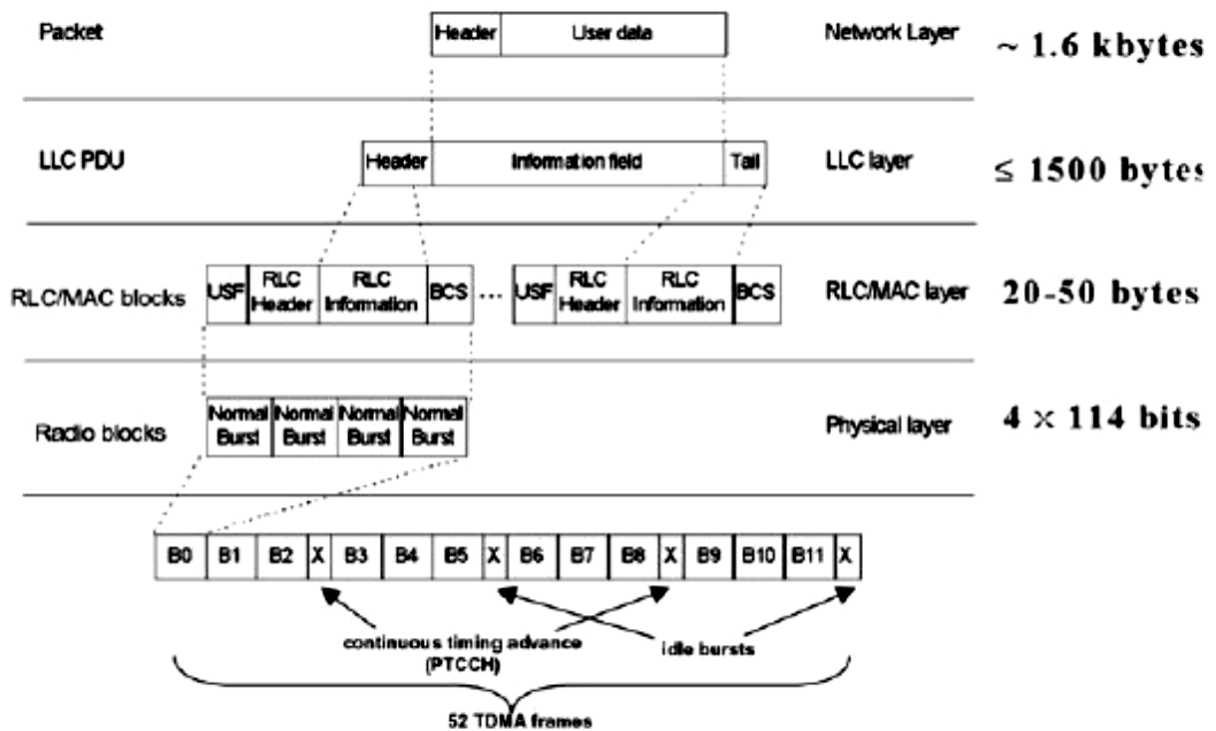


Figura 7 - Estrutura multi-frame

### 3.2.4. Implementação da rede GPRS

O centro de uma rede GPRS está concentrado na GSNs que são compostos pelos seguintes componentes:

- Gerenciamento da rede: incluindo operação e manutenção da bilhetagem.
- Serviços de rede: o que prove os serviços de Internet. Este conjunto de componentes é denominado de “GPRS Internal Backbone” formado pela conexão de roteadores entre os GSNs. No entanto é possível incluir roteadores separados do backbone GPRS, desde que possua um equipamento DCE entre os GSNs.

A conectividade IP do GPRS fornece:

- Comunicação entre as diferentes partes de um sistema GPRS incluindo: Mobile Station, SGSN, GGSN, administração de hosts e prove acesso a usuários na rede;

- Comunicação com a Internet

Existem dois diferentes níveis de comunicação IP: Comunicação com a rede GPRS para sinalização, controle, etc. A Figura 8 demonstra um tipo de implementação de GPRS *Internal Backbone*.

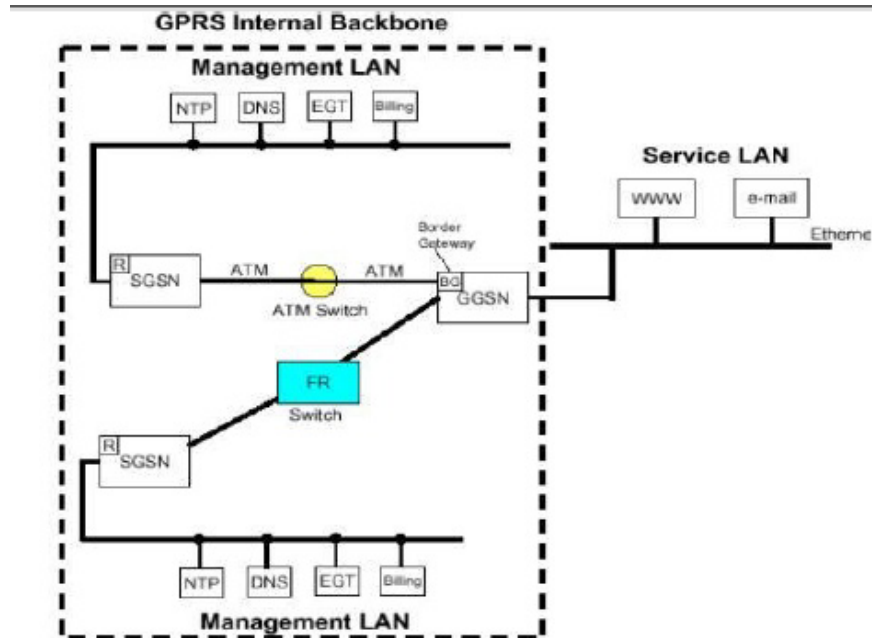


Figura 8 - Backbone GPRS

O sistema GPRS fornece a conectividade IP entre MSs e IHS utilizando um padrão do sistema. A transferência é baseada nos protocolos de Internet, a transmissão é realizada fim-a-fim utilizando os meios aéreos. O MS é responsável por fornecer a conexão do usuário da rede através do modem. O endereço IP utilizado na comunicação do sistema pode ser público, privado, dinâmico ou estático. Os endereços dos MS públicos ou privados são definidos de acordo com a topologia de rede adotada e a forma de acesso a Internet. Os endereços dinâmicos são geralmente adotados para visitantes do SGSN quando o mesmo está em *roaming* recebe um endereço temporário, e o mesmo permanece com este endereço somente enquanto estiver na rede visitada, sendo necessário desta forma um range menor de endereços. O endereço IP estático é definido no HLR geralmente, é utilizado para acesso a redes seguras que utilizam o IP como parte da verificação de acesso.

Os roteadores realizam o roteamento de todos os MSs conectados ao sistema GPRS enviando os pacotes para o *Backbone*. Para a segurança desta transação são envolvidos aplicações e serviços como:

- autenticação em RADIUS (utilizando PAP (Password Authentication Protocol) ou CHAP (Challenge Handshake Authentication Protocol)); uso de IPSec ou PVC para tunelamento dos dados;
- uso de NAT (Network Address Translation) em caso de uso de endereços privados.

Para comunicação entre os roteadores são suportados vários protocolos de roteamento como RIP (Routing Information Protocol), OSPF (Open Shortest Path First) e BGP (Border Gateway Protocol).

### **3.3. RTOS**

Podemos dividir Sistema Operacional em Tempo Real em dois componentes, como sua própria nomenclatura o faz, Sistema Operacional e Tempo Real.

Tempo Real indica uma resposta rápida ou reação a um evento no instante em que acontece. A resposta retrata o tratamento lógico do resultado produzido. O instante da evolução dos acontecimentos “retrata o prazo para a produção do resultado”.

Sistema Operacional (OS) é um programa do sistema que fornece uma interface entre o hardware e as aplicações. OS é geralmente equipado com recursos como: multitarefa, sincronização, tratamento de eventos e interrupções, acesso aos pinos de entrada e saída do processador, comunicação entre as tarefas, temporizadores e gerenciamento de memória para cumprir seu papel principal da gestão de recursos de hardware para atender às demandas de programas de aplicação.

RTOS é conseqüentemente um sistema operacional que suporta aplicações em tempo real e sistemas embarcados, fornecendo logicamente resultados corretos dentro do prazo exigido. Essas características definem seu comportamento no determinístico e natureza limitada de recursos.

### 3.3.1. Por que utilizar um RTOS em aplicações em Tempo Real

RTOS não é um componente necessário em toda a aplicação em tempo real em sistemas embarcados. Um sistema embarcado simples não requer RTOS. Mas quando a complexidade das aplicações se expande para além de tarefas simples, benefícios de ter um RTOS superam largamente os custos associados.

Sistemas embarcados estão cada vez mais complexos. E, como cada vez mais recursos são colocados para eles, aplicações em execução em plataformas de sistemas embarcados serão cada vez mais complexas a maneira como eles se esforçam para satisfazer as exigências de resposta do sistema. Um RTOS será eficaz para permitir as aplicações em tempo real serem projetadas e ampliadas com mais facilidade, de acordo com o desempenho exigido.

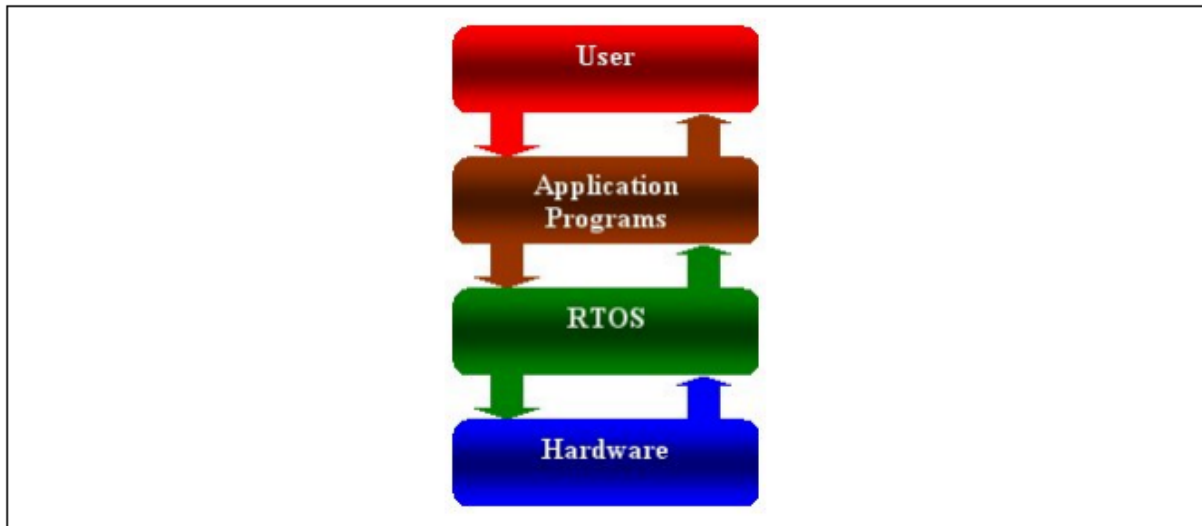


Figura 9 - Sistema Operacional em Tempo Real Embarcado

### 3.3.2. Classificação de RTOS

RTOS podem ser classificados em três tipos, *Hard Real Time RTOS*, *Firm Real Time RTOS* e *Soft Real Time RTOS* conforme descrito abaixo:

- *Hard Real Time RTOS*: grau de tolerância para prazos não cumpridos é extremamente pequeno ou zero. O não atendimento dentro do prazo tem resultados catastróficos para o sistema.
- *Firm Real Time RTOS*: perder um prazo pode resultar em uma redução de qualidade inaceitável.

- *Soft Real Time RTOS*: prazos podem não ser cumpridos, mas o sistema pode se recuperar. Redução na qualidade do sistema é aceitável.

### 3.3.3. Conceitos equivocados sobre RTOS

- Um RTOS deve ser rápido

A resposta de um RTOS depende de seu comportamento determinístico e não de sua velocidade de processamento. A capacidade de um RTOS para a resposta a eventos dentro de um prazo não implica que este é rápido.

- RTOS apresenta grande quantidade de *overhead* na CPU

Um RTOS normalmente requer apenas entre 1% a 4% do uso da CPU.

- Todos os RTOS são iguais

RTOS são geralmente projetados para 3 tipos de sistemas em tempo real mencionados no item 3.3.2. Além disso, eles podem ainda ser classificados de acordo com os tipos de dispositivos de hardware (por exemplo, 8 bits, 16 bits, 32 bits) suportados.

### 3.3.4. Características de um RTOS

O projeto de um RTOS é essencialmente um equilíbrio entre a prestação de um recurso razoavelmente rico de facilidades para o desenvolvimento da aplicação e, não sacrificar previsibilidade e o tempo. Um RTOS básico será equipado com as seguintes facilidades:

#### 3.3.4.1. Multitarefa e Preempção

Um RTOS deve ser multi-tarefa e preemptivo para executar múltiplas tarefas em aplicações em tempo real. O gerenciador deve ser capaz de antecipar qualquer tarefa no sistema e alocar os recursos para a tarefa que precisa dele mais ainda no pico carga.

#### 3.3.4.2. Prioridade da Tarefa

Preempção define a capacidade de identificar a tarefa que precisa de um recurso a mais e atribuir-lhe o controle para obter o recurso. Em RTOS, tal capacidade é conseguida através da atribuição de tarefas individuais com o nível de prioridade adequada. Assim, é importante para RTOS ser equipado com esse recurso.

#### 3.3.4.3. Mecanismo Confiável de Comunicação Inter-Tarefa

Para várias tarefas se comunicarem em tempo hábil e para garantir a integridade dos dados entre si, uma comunicação e sincronização confiável entre as tarefas são obrigatórios.

#### 3.3.4.4. Herança de prioridade

Para permitir que aplicações com requisitos rigorosos de prioridade a ser implementada, RTOS deve ter um número suficiente de níveis de prioridade ao usar agendamento prioridade.

#### 3.3.4.5. Latências predefinidas

Um RTOS precisa ter definido com precisão o tempo de chamadas de seu sistema. As métricas de comportamento são:

- Tarefa de latência de comutação: O tempo necessário para salvar o contexto de uma tarefa atualmente em execução e alternar para outra tarefa é desejável ser curto.
- Latência de interrupção: O tempo decorrido entre a execução da última instrução da tarefa interrompida e as primeiras instruções na função de tratamento de interrupção.
- Interrupção de latência expedição: O tempo desde a última instrução no manipulador de interrupção até a próxima tarefa programada para ser executada.

#### 3.3.4.6. Controle de gerenciamento de memória

Para garantir uma resposta previsível a uma interrupção, um RTOS deve fornecer uma maneira para a tarefa bloquear o seu código e dados na memória real.

### 3.3.5. Arquitetura de um RTOS

A arquitetura de um RTOS depende da complexidade de sua implantação. RTOSs bons podem se adaptar a atender diferentes conjuntos de exigências para diferentes aplicações. Para aplicações simples, um RTOS geralmente compreende apenas um kernel. Para sistemas embarcados mais complexos, um RTOS pode ser uma combinação de vários módulos, incluindo o kernel, pilhas de protocolos de rede e outros componentes, como mostrado na Figura 10.

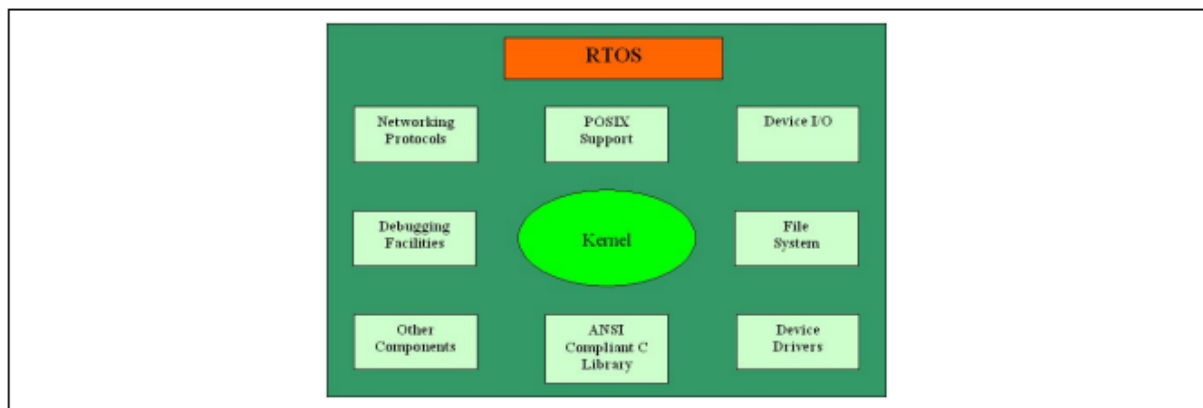


Figura 10 - Arquitetura Geral de um RTOS

### 3.3.6. Kernel

Um sistema operacional geralmente consiste de duas partes: o espaço do kernel (modo kernel) e espaço do usuário (modo usuário). kernel é o componente menor e central de um sistema operacional. Seus serviços incluem gerenciamento de memória e dispositivos e também para fornece uma interface para aplicações de software usar os recursos. Serviços adicionais, tais como proteção e gestão de programas e multitarefa podem ser incluídos dependendo da arquitetura do sistema operacional. Existem três grandes categorias de modelos de kernel disponíveis, a saber:

#### 3.3.6.1. Kernel monolítico

Responsável por todos os serviços básicos do sistema (ou seja, processo e gerenciamento de memória, manipulação de interrupção e comunicação I/O, sistema de arquivo, etc) no espaço do kernel. Como tal, kernels monolíticos fornecem abstrações ricas e poderosas do hardware. Quantidade de mudanças de contexto e



mensagens envolvidas é muito reduzida, o que faz com que ele rode mais rápido do que um microkernel. Exemplos clássicos podem ser citados como o Linux e Windows.

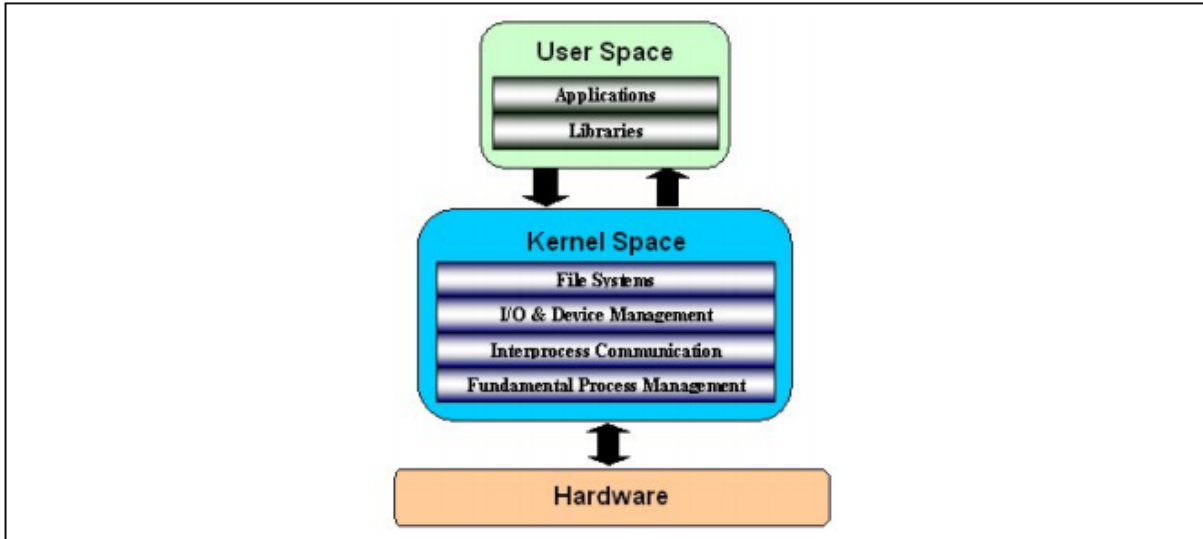


Figura 11 - Sistema Operacional Baseado em Kernel Monolítico

### 3.3.6.2. Micro-Kernel

Executa apenas o processo de comunicação básico (mensagens) e controle de I/O. Os outros serviços do sistema (sistema de arquivos, de rede, etc) reside no espaço do usuário na forma de *daemons*/servidores. Assim, micro-kernel fornece um conjunto simples e menor de abstrações hardware. É mais estável do que o kernel monolítico como não é afetado mesmo se os servidores falharem (exemplo sistema de arquivos). Exemplos de OS são AmigaOS e QNX.

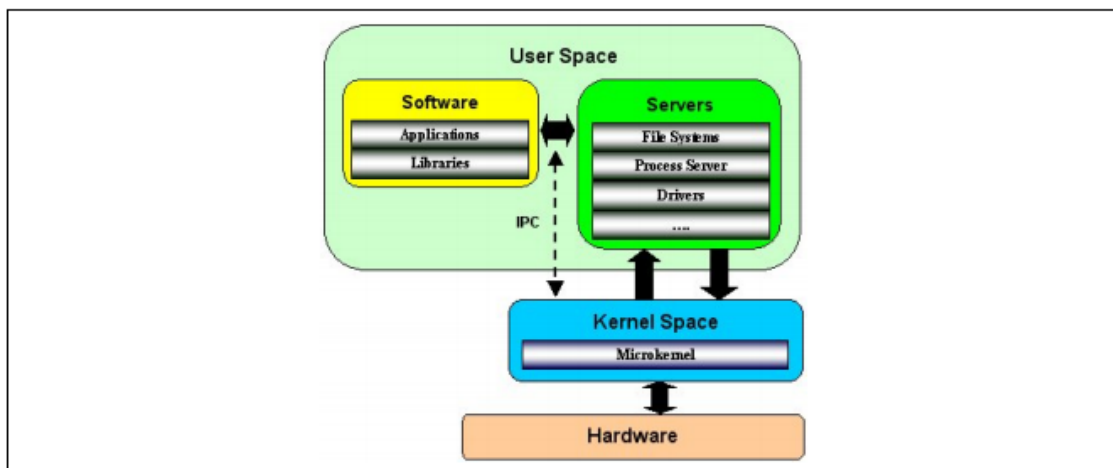


Figura 12 - Sistema Operacional Baseado em Micro-Kernel

### 3.3.6.3. Exokernel

O conceito é ortogonal ao de micro-kernel e kernel monolíticos dando um controle eficiente para a aplicação sobre o hardware. Executa apenas os serviços protegendo os recursos (ou seja, acompanhamento da propriedade, guardando o uso, revogando acesso aos recursos, etc), proporcionando baixo nível de interface para sistemas operacionais biblioteca (libOSes) e deixando a gestão para a aplicação.

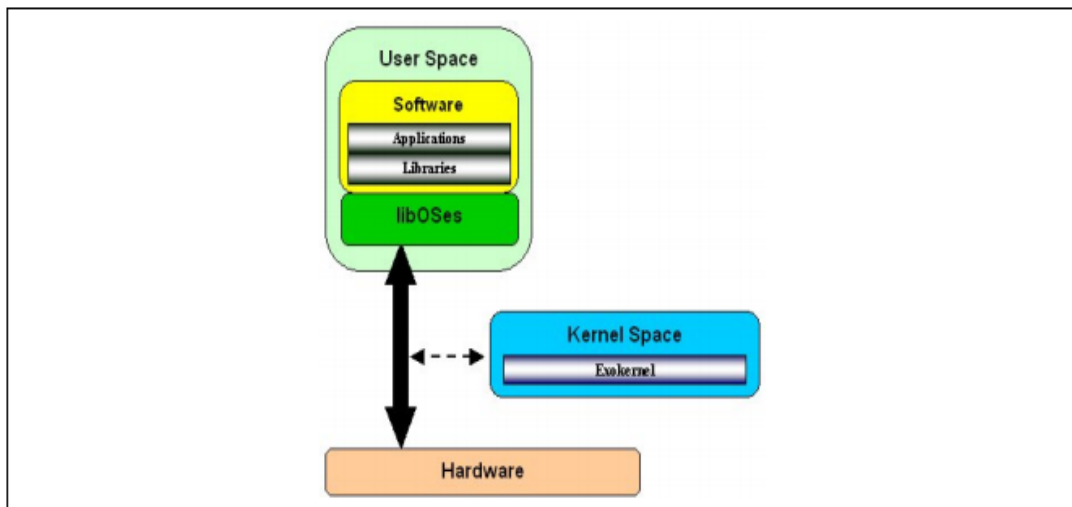


Figura 13 - Sistema Operacional Baseado em Exokernel

Um RTOS geralmente evita implementação do kernel como um grande programa monolítico. O kernel é desenvolvido, ao invés disso, como um micro-kernel, com adição de funcionalidades configuráveis. Esta implementação traz benefícios, resultando em aumento de opções de configuração do sistema, como cada aplicativo incorporado requer um conjunto específico de serviços do sistema no que diz respeito às suas características.

O kernel de um RTOS fornece uma camada de abstração entre o software de aplicativos e hardware. Essa camada de abstração é composta por seis tipos principais de serviços comuns pelo kernel para o software de aplicação. A Figura 14 mostra os seis serviços comuns de um kernel RTOS.

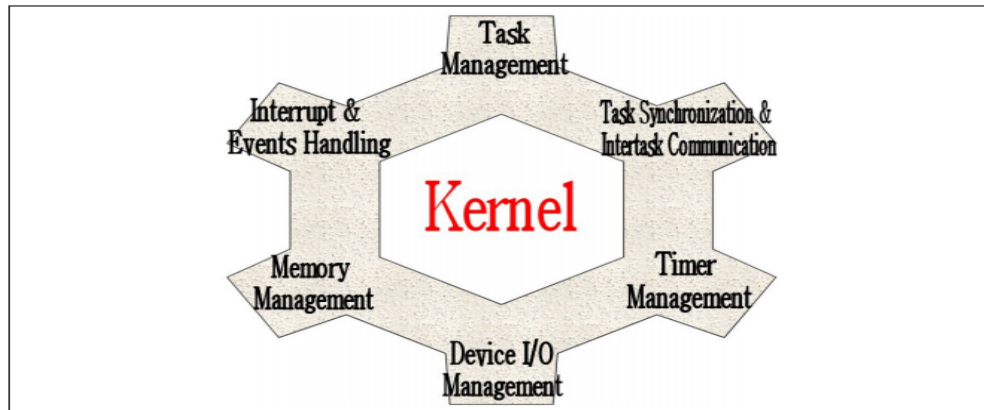


Figura 14 - Serviços do Kernel de um RTOS

### 3.4. Linguagem C#

A linguagem de programação C# foi desenvolvida em 2000. Seu arquiteto chefe é o dinamarquês Anders Hejlsberg. Anders Hejlsberg também é conhecido por ter projetado o Turbo Pascal, Delphi e mais tarde, uma ferramenta visual para construir aplicações cliente-servidor.

C# deve suas raízes para C++, Delphi e Java e é baseado na sintaxe da linguagem C++, a qual é uma extensão do C. No entanto, em vez de excluir muitas características do C como na maioria das linguagens orientadas a objeto, os autores do C# resolveram manter as características e flexibilidade de C, porém construindo-as com mais segurança.

Um fato importante na história do C# aconteceu em 1996 quando a Microsoft adquiriu a Colusa Software. A Colusa tinha lançado um produto, em 1995, chamado OmniVM. OmniVM foi um ambiente de máquina virtual que oferecia duas vantagens distintas sobre as primeiras versões do Java. Primeiro, evitando a interpretação e usando uma arquitetura virtual RISC, o que fornecia execução de código quase nativa. Em segundo lugar, implementou um robusto isolamento da aplicação através de um gerenciador de memória virtual. Isso o fez um ambiente muito seguro para execução de códigos para aplicações móveis por exemplo. OmniVM formaria a base para o CLR (*Common Language Runtime*) da Microsoft.

#### 3.4.1. Principais características

O fundamento principal de qualquer linguagem orientada a objetos é seu suporte para definir e trabalhar com classes. Classes determinam novos tipos,

permitindo que você estenda a linguagem e consiga manipular melhor o problema que você está tentando solucionar. C# contém palavras-chave para declaração de novas classes e de seus métodos e propriedades e para implementação de encapsulamento, polimorfismo, os três pilares da programação orientada a objetos.

No C#, tudo relacionado a uma declaração de classe localiza-se na própria declaração. Definições de classe C# não necessitam de arquivos de cabeçalho ou arquivos separados (IDL, Interface Definition Language). Além disso, o C# permite documentação direta (inline) que simplifica a criação online e impresa de documentação para um aplicativo.

C# também permite interfaces, um meio de fazer um contrato com uma classe para processos que a interface determina. No C#, uma classe pode herdar de apenas um objeto pai (parent), mas uma classe pode implementar múltiplas interfaces. Quando uma classe implementa uma interface, na prática promete proporcionar funcionalidade que as interfaces especificam. C# proporciona também um suporte a structs (estruturas), um conceito que tem mudado seu significado de forma relevante a partir do C++. No C#, uma estrutura é um limitado tipo supérfluo que quando instanciado, faz menos pedidos ao sistema operacional e à memória, do que uma classe convencional faz. Um struct pode herdar de uma classe ou ser herdada por uma classe, mas um struct pode implementar uma interface.

C# proporciona suporte total de delegates (delegados): para invocação de métodos indiretamente. Em outras palavras, como no C++ você pode achar funcionalidade similar (como em ponteiros para funções de membros), mas *delegates* são referência de tipos seguros que encapsulam métodos com assinaturas e tipos de retorno específicos. *Delegates* têm sido estendidas significativamente, primeiro no C# 2.0 e novamente no C# 3.0, primeiramente com anonymous delegates e agora com as Expressões Lambda (Lambda expressions), colocando a base para o LINQ.

C# proporciona características orientadas a componentes, como propriedades, eventos e construtores declarativos (como atributos). Programação orientada a componente é sustentado pelo armazenamento de metadado com o código para a classe. O metadado descreve a classe, incluindo seus métodos e propriedades, bem como sua necessidade de segurança e outros atributos, assim

como será que pode ser serializado; o código contém a lógica suficiente para executar suas funções.

Dessa forma, uma classe compilada é uma unidade independente. Então um ambiente de armazenamento de dados que sabe como ler um metadado e um código de uma classe, não precisa de nenhuma outra informação para fazer utilização disso. Isso é possível, usando o C# e o Common Language Runtime (CLR) para adicionar um metadado personalizado a uma classe pela criação de atributos personalizados. Dessa mesma forma, é possível ler um metadado de classe utilizando os tipos do CLR que suportam reflexão. Quando você compila seu código, cria um *assembly*. Um *assembly* é uma coleção de arquivos que aparecem ao programador como uma única DLL, biblioteca de link dinâmico (Dynamic Link Library) ou executável (EXE). Em .NET, um *assembly* é a unidade básica de reutilização, versionamento, segurança e implantação. O CLR proporciona um número de classes para manipulação de *assemblies*.

Ainda algumas características importantes sobre C#, são:

- Acesso direto à Memória utilizando ponteiros do estilo C++
- Palavras-chave para incluir operações como inseguras
- Avisa o coletor de lixo CLR para não coletar objetos referenciados por ponteiros até que eles sejam liberados

#### **4. COMPONENTES PRINCIPAIS DO SISTEMA DE CONTROLE DE ACESSO**

Este capítulo é destinado à descrição dos principais componentes envolvidos no desenvolvimento deste projeto, que são o Leitor RFID, o modem GPRS e o M52259DEMOKIT.

##### **4.1. Leitor RFID Indala/HID 5365 (Wiegand)**

O leitor RFID Indala/HID, pode ser montado em ambientes internos ou externos, desde que o isolamento após a montagem seja feito de forma correta.

##### **4.1.1. Características**

- Aceita tensão de alimentação entre 5 e 16 Volts.

- Encontra-se disponível com interface Wiegand ou Interface Clock-and-Data.
- Faz uso da tecnologia FlexSecur desenvolvida pela HID.
- Permite um aprimoramento de fita magnética a leitora de proximidade sem necessidade de refazer o cabeamento ou instalar novas extensões para os cabos.
- Oferece alta confiabilidade, características consistentes de alcance de leitura e baixo consumo de energia.
- Montagem direta em metal sem alteração no desempenho do alcance da leitura.
- Proporciona LED multicolorido, compatibilidade com todos os sistemas padrão de controle de acesso e controle interno ou no computador central do LED e Buzzer.



Figura 15 - Leitor de Proximidade HID/Indala

#### 4.1.2. Tecnologia FlexSecur

A popularidade de controle de acesso por proximidade usando cartões de 26 bits de formato aberto, combinado com leitores de proximidade intercambiáveis tem resultado em uma situação em que diferentes clientes podem receber cartões idênticos, e os cartões de uma instalação podem obter acesso a outra instalação.

Os leitores e cartões da HID Indala oferecem uma tecnologia de segurança exclusiva chamado FlexSecur® onde cartões e leitores são programadas como um conjunto único para cada cliente. Portanto, os cartões pertencentes a um cliente não

podem ser lidos por leitores de outro cliente. FlexSecur® proporciona uma melhoria considerável no sistema de controle de acesso de segurança, sem nenhum custo adicional, e funciona com qualquer sistema de controle de acesso.

A maioria dos leitores de cartão de proximidade irá transmitir os dados do cartão de proximidade para o painel de controle de acesso sem verificar se o cartão é autorizado para essa instalação. A maioria dos cartões de proximidade não pode ser configurado para operar em apenas uma instalação específica. Os leitores são totalmente intercambiáveis e os cartões serão lidos em qualquer leitor da mesma marca. Distribuidores locais têm geralmente leitores intercambiáveis e cartões de 26-bit formato aberto em estoque.

Enquanto isso é conveniente para o agente de segurança, isto também pode representar um risco de segurança significativo para o cliente. Isso significa que é possível para os clientes receberem cartões que são idênticas aos dos outros. Alguns clientes têm acidentalmente descoberto que os seus cartões podem acessar a entrada para as instalações de outro cliente, e eles estão preocupados que alguém de outra instalação poderia ganhar entrada para as suas instalações.

Além disso, a maioria dos cartões de controle de acesso de proximidade contêm dados não criptografados. Isto significa que se uma pessoa com algum conhecimento técnico obtém um cartão de um site particular, mesmo se o cartão é anulado no banco de dados do sistema de controle de acesso, o código de instalação e o número de identificação poderiam ser determinados usando um leitor de cartão comprado de qualquer distribuidora. Cartões adicionais no mesmo intervalo de números poderiam ser comprados, permitindo a entrada não autorizada.

FlexSecur® multi-camada de segurança impede a entrada acidental ou tentativas fraudulentas através da criação de formatos únicos para o clientes individualmente, que incluem uma senha única para ser especificada, e criptografia única dos dados do cartão de modo que não pode ser interpretado por intrusos em potencial. Estas medidas de segurança são programadas tanto nos cartões como nos leitores, para que o leitor só envie os dados para o sistema de controle de acesso somente quando os cartões corresponderem a aquele cliente.

O tecnologia FlexSecur® consiste nos seguintes componentes:

- Número Formato Indala - Um número de referência único é atribuído a cada formato. Novos formatos são atribuídos aos próximos números disponíveis em

seqüência. O número de formato não tem relação direta com o formato dos dados do cartão. O número de formato é cadastrado no banco de dados Indala, e links para um "arquivo de projeto" com os seguintes componentes usados em cartões de programa e os leitores:

- Senha - Este valor de 10 dígitos (30 bits) numérico é armazenado em ambos os cartões e leitoras. Senhas são opcionais, mas altamente recomendadas.
- Chave de Criptografia - Uma seqüência de bits gerada aleatoriamente atribuídos a cada número novo formato, ele é usado durante o processo de codificação do cartão para criptografar os dados do cartão e é usado pelo leitor para "descriptografar" os dados antes de enviar a saída para o Sistema de Controle de Acesso. A chave é armazenada no leitor mas não no cartão.
- Leitor de cartão e Formato de Dados - Este formato descreve como os dados de controle de acesso estão codificados nos cartões, incluindo o número de bits, comprimentos de campo, tipos e posições, dados do pedido e esquema de paridade. Ele também diz ao leitor quantos bits são codificados no cartão e quantos bits para transmitir para o sistema de controle de acesso.
- Parâmetros do Leitor - O arquivo de projeto de formato também inclui a configuração do leitor especial que o cliente exige, como buzzer on/off, LEDs on/off, simples ou duplo controle de LED, Wiegand largura de pulso, parâmetros etc. Os parâmetros do leitor também podem ser modificados usando através de "Cartões de Opção".

#### 4.2. Modem GPRS Daruma

O modem GSM/GPRS da Urmet Daruma é um equipamento para comunicação de dados, utilizando tecnologia celular GSM/GPRS. Recomendado para uso de comunicação entre máquinas (M2M) suporta ambientes industriais e afins, pode ser conectado a um PC, Notebook e outros equipamentos por comunicação serial V24/RS232 ou USB.

Para uma maior flexibilidade e disponibilidade na comunicação, o MIN200 suporta simultaneamente dois SIM cards, controlados pela aplicação do terminal conectado (Computador), permitindo a seleção do melhor sinal entre as respectivas operadoras.





Figura 16 - Modem GPRS Urmet Daruma

#### 4.2.1. Principais Características

- Módulo GSM embarcado: Cinterion MC55i
- Transmissão e recepção de dados
- Transmissão e recepção de SMS
- SMS tipo MT/MO/CB/PDU
- Transmissão de dados assíncrona não transparente  
GSM 2400 bit/s 4800 bit/s 9600 bit/s
- CSD até 14.4 kbps
- Conectividade GPRS
- MULTISLOT GPRS classe 10
- Estação móvel GPRS classe B
- GPRS max. 85.6 kbps (downlink)
- Esquemas de codificação CS1, CS2, CS3, CS4
- Interface de dados: V.24 RS-232 ou USB 2.0
- Controle de comandos AT (padrão ETSI 07.05,07.07)
- Potência de Saída:
  - Classe 4 (2W) para EGSM900 e EGSM850
  - Classe 1 (1W) para EGSM1800 e EGSM1900

- Alimentação: de 10V à 30V
- Entrada de Corrente:
  - em repouso : 50mA 12VDC
  - em operação : 250mA 12VDC
- Temperatura de Operação: -10°C a +50°C
- Temperatura de Armazenagem: de -40°C a +85°C
- Protocolos da Pilha TCP/IP: TCP, UDP, HTTP, FTP, SMTP, POP3  
(Acesso por comandos AT)

#### 4.3. M52259DEMOKIT

O KIT de demonstração M52259DEMOKIT para microcontrolador Freescale MCF5225x inclui as placas M52259DEMOMCU e M52259DEMOCOM.

O M52259DEMOMCU tem dois conectores para permitir que recursos adicionais possam ser adicionados de forma rápida e eficiente. Além do microcontrolador MCUMCF52259, a placa possui um sistema integrado, Open Source, USBDM, e uma porta USB com conector mini-AB. A IDE Freescale CodeWarrior também está incluída no KIT para facilitar o desenvolvimento de aplicações e de depuração.

O M52259DEMOCOM é uma placa de expansão conectada diretamente ao M52259DEMOMCU. Ela fornece as interfaces Ethernet 10/100, RS-232, e Hi-Speed CAN. Conectores montados na placa também fornecem a expansão adicional de novas funcionalidades.

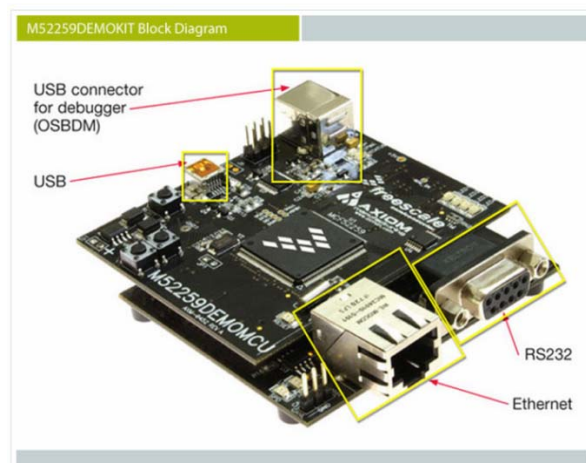


Figura 17 - M52259DEMOKIT - Freescale

#### 4.3.1. Família MCF5225x

A família MCF5225x consiste de microcontroladores com periféricos como USB, Ethernet, CAN e funções de criptografia. A família MCF5225x é baseada em um Coldfire 32-bits com frequência de até 80MHz, 512kB de memória flash e 64kB de SRAM. A interface de barramento externo fornece flexibilidade para adicionar mais memória ou mais periféricos. Essas e outras características fazem da família MCF5225x dispositivos ideais para a indústria e aplicações médicas que requerem uma grande quantidade de periféricos e um alto nível de desempenho.

A Freescale fornece ferramentas de desenvolvimento e software que ajudam os desenvolvedores de projetos a desenvolver com maior rapidez e facilidade.

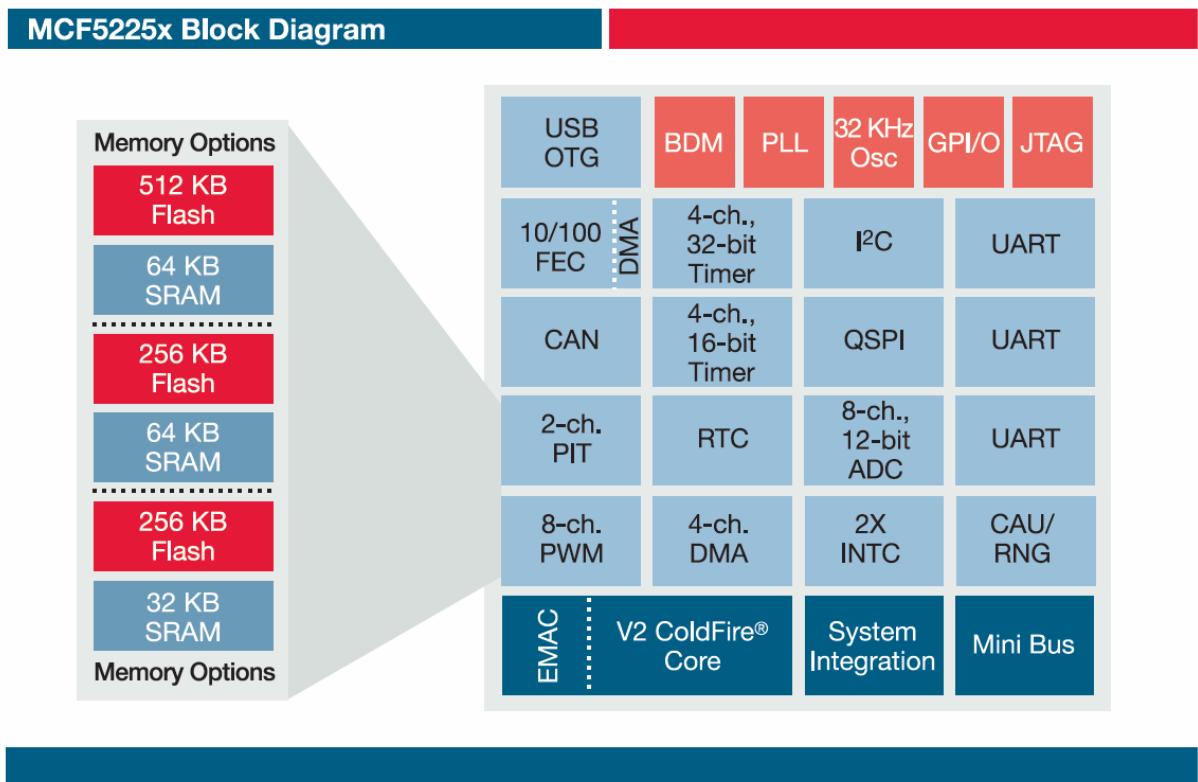


Figura 18 - Diagrama de Blocos MCF5225x

#### 4.3.2. Características - M52259DEMOMCU

- MCF52259 MCU
- Fast Ethernet Controller (FEC)
- USB Physical Layer Interface (PHY)
- Mini-FlexBus External Bus Interface

- FlexCAN 2.0B Module
- Integrated, Open-Source, USB BDM

#### 4.3.3. Características - M52259DEMOCOM

- 10/100 Ethernet
  - KZS8041 Ethernet PHY
  - Configurado para operação MII
  - Conector RJ45
  - Isolamento Magnético
  - LEDs para indicação da velocidade e estado do Link
- RS-232 PHY
  - 1-Canal com controle de fluxo
  - Operação a 2 fios
  - Conector DB-9
- HS CAN PHY
  - Taxa de transferência de 1 Mb
  - Configurado no modo de operação *slave*
  - Conector de 3 pinos
- Sinais de IO
  - 40-pos
  - Não montado

## 5. ETAPAS DO DESENVOLVIMENTO

O desenvolvimento do projeto consistiu em quatro tarefas. Uma delas já foi apresentada, a pesquisa sobre as tecnologias utilizadas, e as outras três serão apresentadas neste capítulo, que basicamente compõe o desenvolvimento da solução com Hardware (Externo + M52259DEMOKIT), Software Embarcado e Software de Gerenciamento.

## 5.1. Hardware Externo

O hardware externo é chamado dessa maneira pelo fato deste ser um hardware que foi desenvolvido especificamente para este projeto e é conectado ao M52259DEMOKIT, ou seja, ele não faz parte do KIT da Freescale.

A necessidade do hardware externo se deu devido ao fato de que é necessário uma interface entre os periféricos usados neste projeto e o M52259DEMOKIT. É necessária a adaptação dos níveis de tensão dos dispositivos para trabalharem no mesmo nível de tensão do MCF52259, conectores específicos e circuitos externos necessários para acionamento de relês.

O M52259DEMOKIT fornece um conector de 40 pinos (J4) na M52259DEMOCOM, que teve que ser montado, com o objetivo de fornecer o acesso a pinos de I/O (IRQs, SPI, etc.) não usados já pelos periféricos montados no KIT. É nesse conector (J4) da placa M52259DEMOCOM que o hardware externo é conectado.

A Figura 19 mostra uma parte do esquemático da placa M52259DEMOCOM com o conector J4, indicando os pinos que o mesmo é conectado ao MCF52259 na placa M52259DEMOMCU.

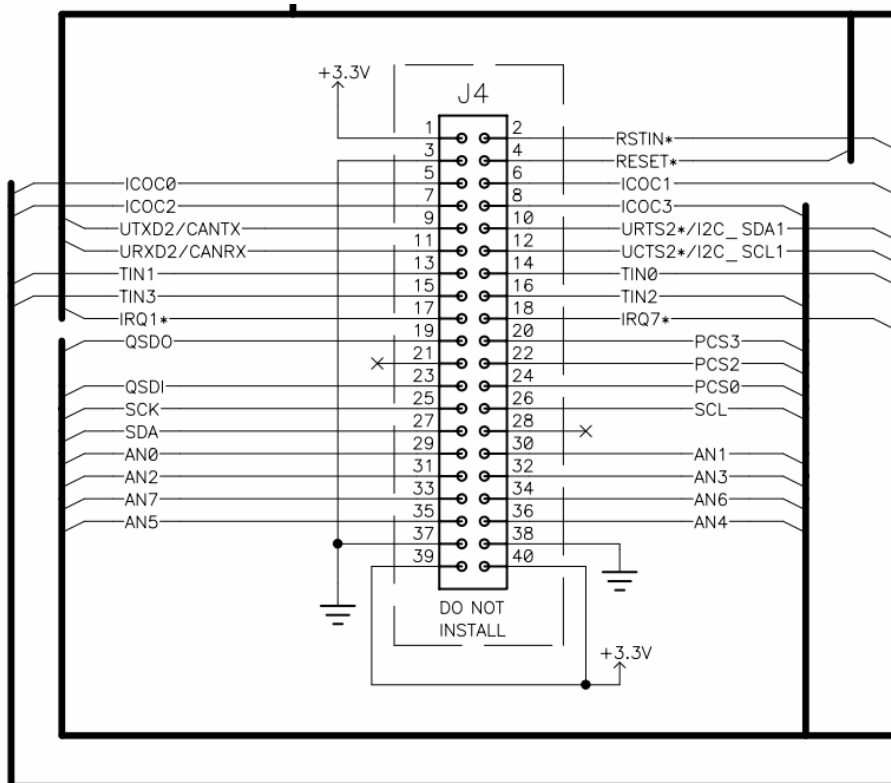


Figura 19 - Conector J4 da placa M52259DEMOCOM

A Tabela 4 mostra os pinos que foram utilizados neste projeto e a função para qual cada um deles foi designado.

Tabela 4 - Funções dos pinos utilizados no projeto

Pino	Nome	Função
17	IRQ0	Pino configurado como interrupção externa de borda de descida e conectado da linha de Dados D1 do leitor RFID
18	IRQ7	Pino configurado como interrupção externa de borda de descida e conectado da linha de Dados D0 do leitor RFID
29	AN0	Pino configurado como A/D e conectado ao circuito do sensor de temperatura
30	AN1	Pino configurado como saída e conectado ao buzzer do leitor RFID
31	AN2	Pino configurado como saída e conectado ao LED do leitor RFID
32	AN3	Pino configurado como saída e conectado ao circuito de acionamento do RELE 1
34	AN6	Pino configurado como entrada e conectado ao sensor da porta
33	AN7	Pino configurado como saída e conectado ao circuito de acionamento do RELE 2

### 5.1.1. Esquemático

A Figura 20 mostra o esquemático do hardware externo que foi desenhado utilizando a ferramenta Eagle.

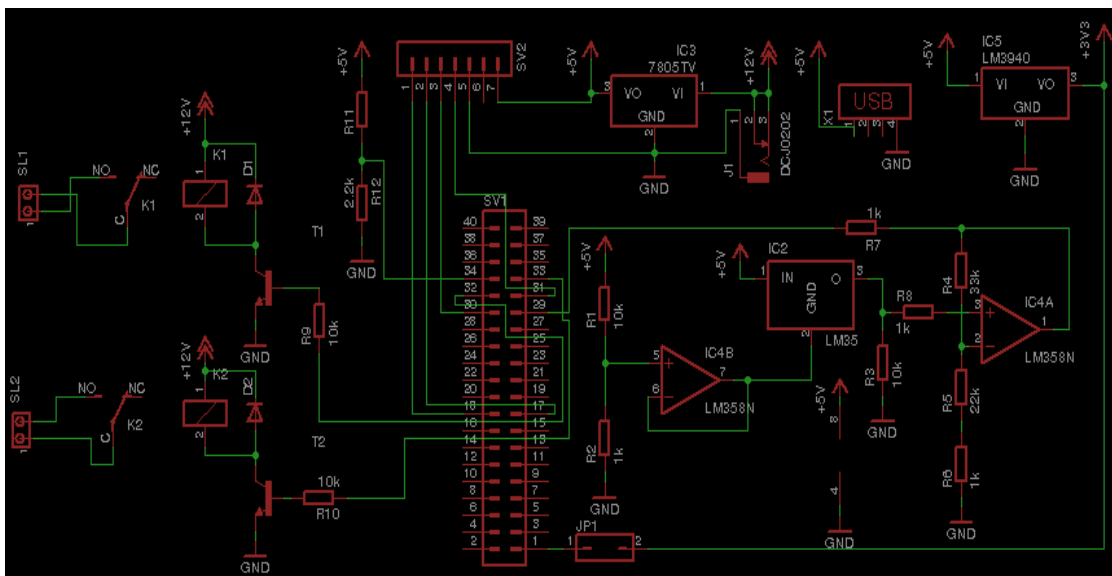


Figura 20 - Esquemático do Hardware Externo

Observando o esquemático podemos identificar o conector SV1 de 40 pinos que vai ser conectado ao conector J4 da placa M52259DEMOCOM.

No circuito do hardware externo foi incluído dois reguladores de tensão, um para a tensão de 5V (LM7805) para alimentação do leitor RFID e do M52259DEMOKIT, através do conector USB X1, e outro de 3.3V (LM3940) com o intuito de alimentar o KIT pelo pino 1 do conector SV1. A entrada ou a alimentação do regulador LM7805 é proveniente de uma fonte externa de 12V DC (1A) que é conectada ao J1. Já o LM3940 recebe a alimentação do LM7805.

No esquemático da Figura 20, é possível notar a presença de um circuito integrado amplificador operacional que foi utilizado para dar um ganho de tensão para as diferenças de tensão fornecidas pelo sensor de temperatura LM35, uma vez que uma mínima diferença de tensão pode indicar uma grande variação de temperatura. E para que esta pequena diferença entre níveis de tensão seja percebida pelo conversor A/D do MCF52259 é necessário o amplificador que foi projetado para ter um ganho de mais ou menos 2,5. A Figura 21 mostra um gráfico de tensão por temperatura do LM35.

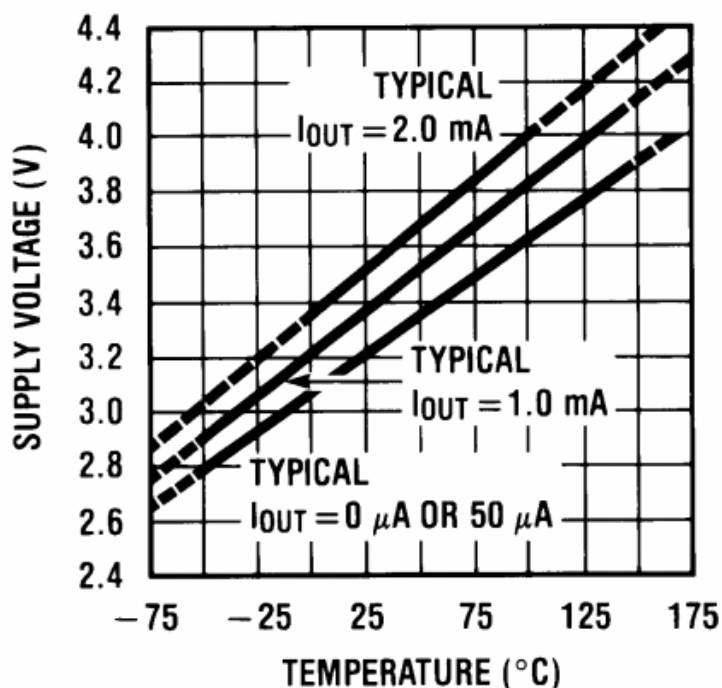


Figura 21 - LM35 - Tensão x Temperatura

O Leitor RFID irá conectado ao SV2 que tem seus pinos ligados diretamente ao conector SV1. Aqui é importante fazer uma observação importante que se refere a adaptação do nível de tensão fornecido pelo Leitor RFID para o microcontrolador

MCF52259. O Leitor RFID é alimentado com 5V, portanto os níveis de tensão gerados por ele devem ser reduzidos para 3.3V antes de chegar ao MCF52259. Isto não foi feito diretamente no hardware externo, então teve que ser feito no cabo do Leitor RFID, inserindo resistores de 2.2k $\Omega$  em série nas linhas de dados, LED e Buzzer.

O sensor de estado da porta (aberta ou fechada) é representado no esquemático pelo resistor R11, que vai conectado a um resistor de *pull-down* e conectado ao pino de entrada AN6.

Os conectores SL1 e SL2 são ligados aos circuitos com reles que tem o objetivo de acionar uma sirene de alarme e acionar uma fechadura magnética para abertura da porta.

O desenho da placa, dupla face, pode ser encontrado nos anexos deste trabalho.

## 5.2. Software Embarcado com RTOS MQX

No software embarcado foi utilizado o RTOS MQX da Freescale que possui as características de um RTOS monolítico, que foi explicado no Capítulo 3.

O desenvolvimento com RTOS ganha em eficiência à medida que o desenvolvedor vai conhecendo suas principais características, funcionalidades de suas bibliotecas e entendendo a maneira como o RTOS acessa o hardware para então iniciar a construção de *drivers* para acesso a dispositivos conectados aos seus pinos de I/O.

Neste capítulo é feita uma descrição superficial sobre o MQX e como a aplicação desenvolvida trabalha com o Sistema Operacional.

### 5.2.1. MQX-RTOS

O Sistema Operacional em Tempo Real MQX da MQX Embarcado, foi desenvolvido para sistemas embarcados em tempo real com um único processador, multi-processador e sistemas distribuídos. Para alavancar o sucesso do sistema operacional MQX, a Freescale Semiconductor adotou esta plataforma de software para as suas famílias de microprocessadores ColdFire® e PowerPC™.



Comparado com as distribuições originais do MQX, a distribuição Freescale MQX foi feita mais simples de configurar e usar. Uma versão única agora contém o sistema operacional MQX além de todos os outros componentes de software com suporte para um determinado microprocessador.

MQX é uma biblioteca de funções que os programas usam em tempo de execução para se tornarem aplicações multi-tarefas em tempo real. As principais características são o seu tamanho escalonável, orientado a componentes, arquitetura e facilidade de uso. O MQX suporta aplicações multi-processador e pode ser utilizado com flexibilidade em sistemas embarcados de rede, comunicações de dados, e gerenciamento de arquivos.

#### 5.2.1.1. Organização do MQX

O MQX consiste de componentes do kernel (não opcional) e componentes opcionais. No caso dos componentes do kernel, apenas as funções que o MQX ou a aplicação chama estão incluídas na imagem. Para atender seus requisitos, uma aplicação configura componentes do kernel, adicionando componentes opcionais. O diagrama a seguir mostra os componentes do kernel no centro com componentes opcionais ao redor.

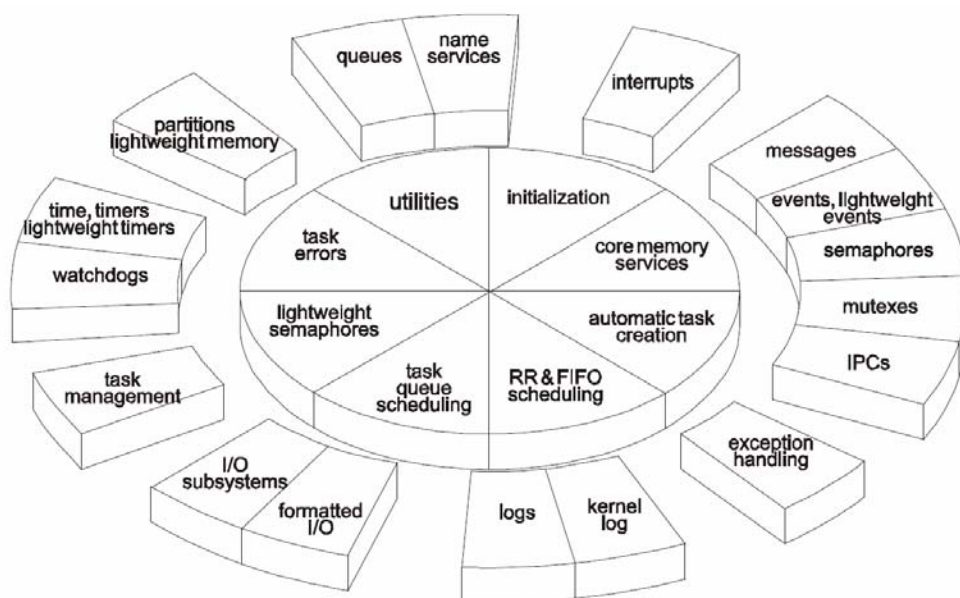


Figura 22 - Organização do MQX

### 5.2.2. Estrutura da Aplicação para o Sistema de Controle de Acesso

A aplicação desenvolvida neste projeto foi dividida em quatro *tasks*, fazendo uso da possibilidade de se trabalhar com multi-tasks no MQX. A Figura 23 mostra como ficou estruturada a aplicação, já destacando as duas *tasks* principais com suas responsabilidades.

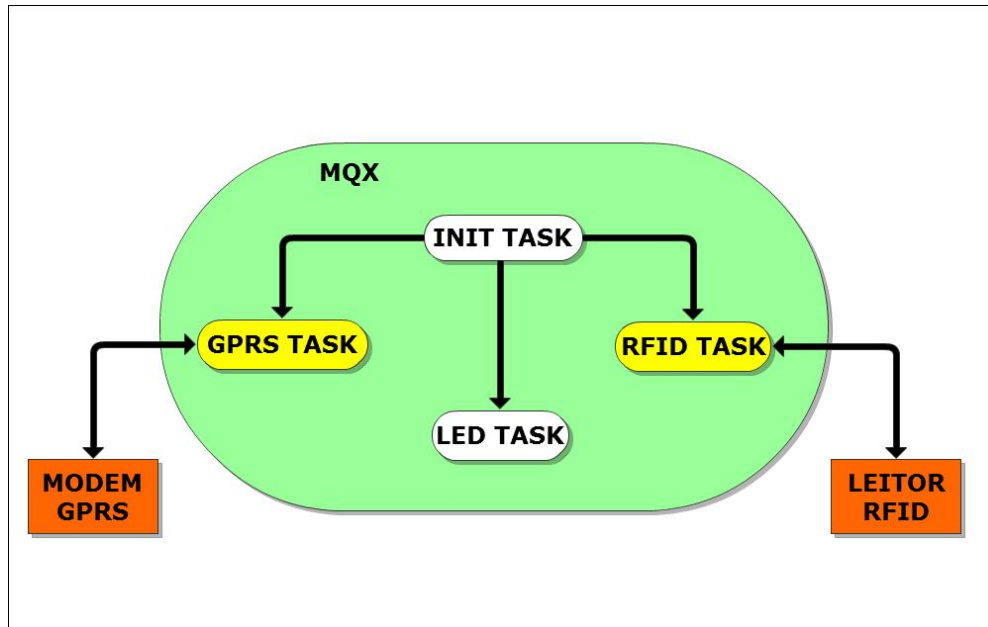


Figura 23 - Estrutura da Aplicação

### 5.2.3. INIT *task*

A INIT *task* tem a função de inicialização do sistema e, portanto é a primeira *task* a entrar em funcionamento após a inicialização do MQX. Esta *task* que inicia cada uma das outras *tasks* também chama uma função para inicialização de um servidor telnet que foi usado no projeto somente com a função de debug.

O fluxograma da Figura 24 mostra as funções da INIT *task*.

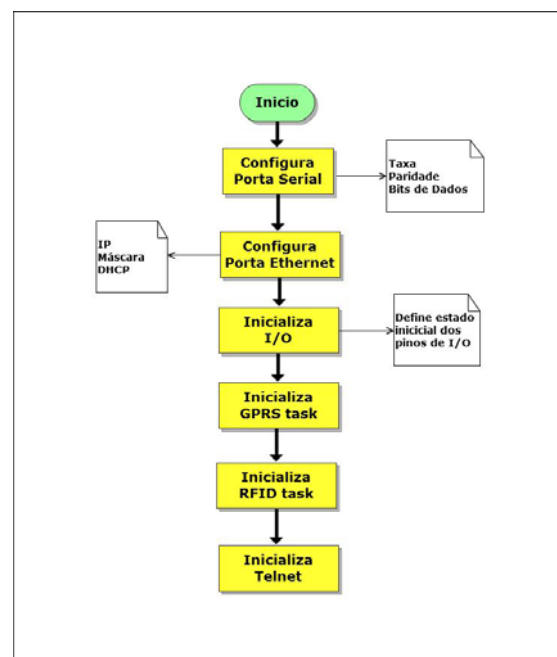


Figura 24 - Fluxograma da INIT *task*

#### 5.2.4. RFID *task*

A RFID *task* tem a função de gerenciar o leitor RFID. Esta *task* que receberá os 26 bits enviados pelo leitor RFID (Wiegand-26) pelas linhas de dados D0 e D1 a cada aproximação de um cartão ao leitor. Além de receber os bits ela os decodifica e monta o código do cartão para a consulta a base de dados local para então decidir se libera ou não o acesso para aquele número de cartão.

Para receber esses bits a RFID *task*, faz uso de um driver que foi escrito para facilitar as operações envolvendo o leitor RFID. Esse driver contém as funções de abertura, fechamento, leitura e escrita. Abaixo é explicado o porquê de cada uma dessas funções.

Quando trabalhamos com um sistema operacional como o MQX, não acessamos os pinos de I/O diretamente para obter a informação desejada, mas sim requisitamos ao MQX esta informação. O driver então nada mais é do que funções que quando chamadas requisitam ou enviam ao Sistema Operacional, MQX no caso, uma informação que deve ser enviada a um determinado pino no caso de uma operação de escrita ou retirada no caso de uma operação de leitura. Essa abstração de hardware é uma das facilidades mais importantes no desenvolvimento com um Sistema Operacional embarcado.

Portanto temos no driver uma função de abertura, que é feita através de um file descriptor. E é através desse file descriptor que iremos, caso não tenha nenhum erro na abertura, ter acesso aos pinos para escrita e leitura.

Na aplicação desenvolvida para este projeto, o driver tem uma função para tratamento das interrupções IRQ1 e IRQ7, a qual armazena os bits das linhas D1 e D0 respectivamente em uma fila. Quando a aplicação requisita uma operação de leitura ao driver, ela passa um ponteiro para o driver o qual será usado como fila para armazenar os bits. O driver retorna o número de bits armazenados na fila para que a aplicação possa ter acesso aos mesmos. Ou seja, a RFID *task* executa uma vez a operação de abertura do driver e entra em um loop executando a cada iteração uma operação de leitura, para checar se o driver tem algum bit a enviar. Se a resposta do driver é diferente de 0, significa que um cartão foi aproximado do leitor e a RFID TASK deve então executar a montagem e verificação do código lido na base de dados local.

A Figura 25 apresenta o fluxograma da RFID *task* com suas principais funções.

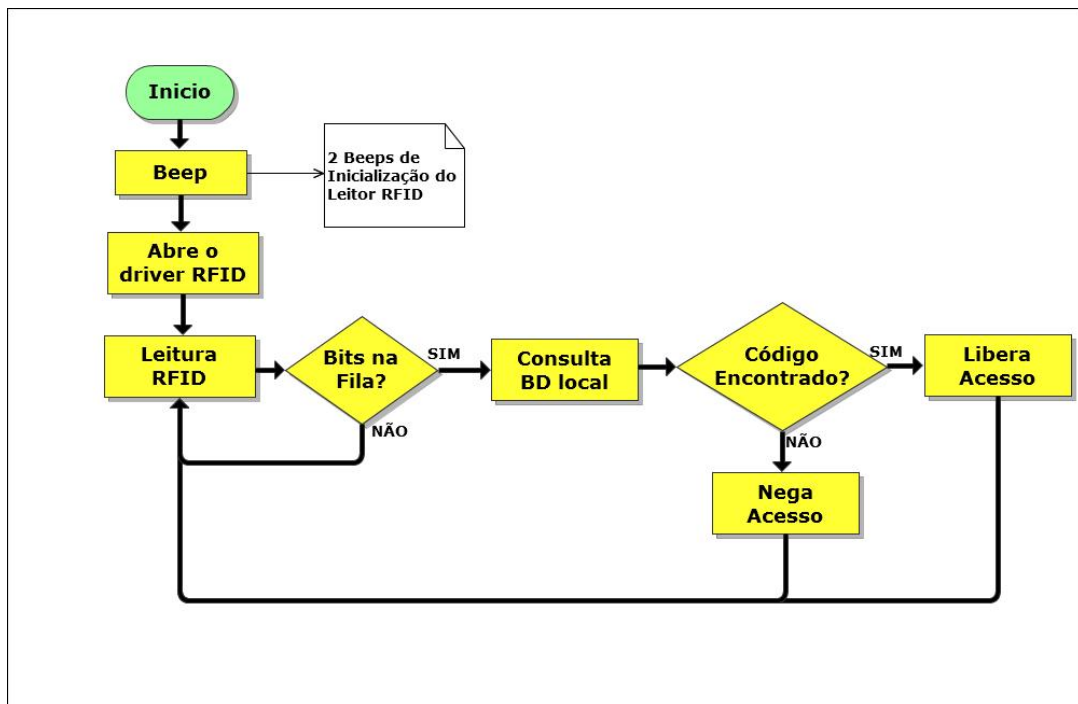


Figura 25 - Fluxograma da RFID *task*

#### 5.2.5. GPRS *task*

A GPRS *task* gerencia as operações envolvendo o modem GPRS. Como a comunicação com o modem é feito através da interface serial, que já foi configurada na INIT *task*, não foi necessário escrever um driver para acessar o modem. Foram utilizadas as próprias funções nativas do MQX que fazem as operações de leitura e escrita na interface serial.

As principais funções dessa *task* são a inicialização do modem, verificação de mensagens recebidas pelo modem, provenientes do software de gerenciamento e o envio de mensagens ao software de gerenciamento.

A função de inicialização do modem executa uma série de comandos AT no modem através da interface serial a fim de configurar parâmetros no modem como o modo de mensagens de erro, e principalmente o socket TCP, passando os parâmetros necessários para que o modem tenha acesso a Internet. Se ocorrer tudo sem erros na função de inicialização, esta ao final envia um SMS para um número de celular definido na aplicação com a informação de que o módulo GPRS iniciou sem problemas e contendo o IP que o mesmo recebeu da operadora.

Após a inicialização do modem GPRS, a *task* entra em um loop e a cada iteração executa uma operação de leitura do *socket* TCP do modem GPRS. Essa operação de leitura é feita através de um comando AT que retorna o conteúdo do *socket* TCP que deve ser lido então pela GPRS TASK.

As Tabelas 5 e 6 mostram as mensagens que podem ser recebidas (Tabela 5) e ou enviadas (Tabela 6) pelo modem GPRS para o Software de Gerenciamento pelo *socket* TCP.

Tabela 5 - Mensagens recebidas pelo Modem GPRS

Mensagem	Função
add<número do cartão>	Adicionar um novo código de cartão a base de dados local
rem<número do cartão>	Remove um código de cartão da base de dados local
Temp	Requisita a informação de temperatura
Status	Requisita a informação de sensor

Tabela 6 - Mensagens enviadas pelo Modem GPRS

Mensagem	Função
temp<temperatura>	Requisita a informação de temperatura
status<True/False>	Requisita a informação de sensor

A Figura 26 mostra o fluxograma da GPRS *task*.

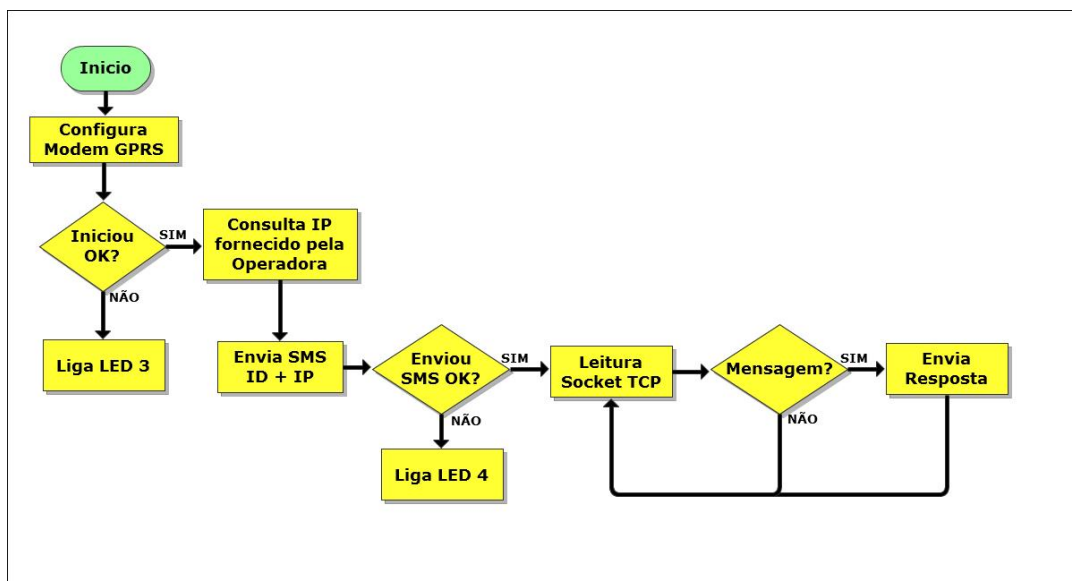


Figura 26 - Fluxograma da GPRS task

### 5.2.6. LED task

É a *task* mais simples do sistema, porém tem uma função fundamental de mostrar ao operador que o sistema está em funcionamento.

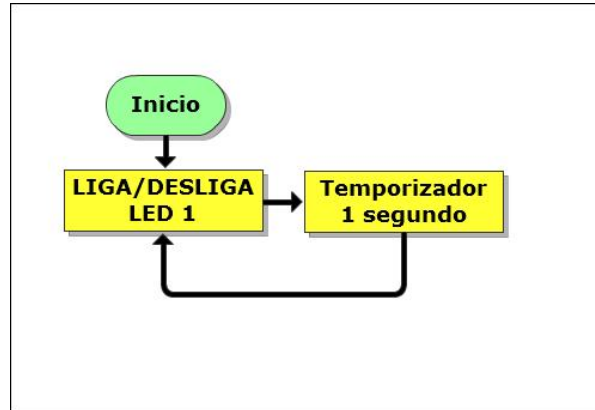


Figura 27 - Fluxograma LED task

Ela simplesmente tem a função de piscar o LED1 da placa M52259DEMOMCU durante todo o tempo em que o módulo fica ligado. É a *task* de menor prioridade, porém deve estar sempre rodando, o que indica ao operador caso o LED1 fique constantemente ligado ou desligado uma falha no sistema. Essa falha pode ter sido causada por algum tipo de travamento em alguma outra *task* e portanto o operador deve reiniciar o sistema.

### 5.3. Software de Gerenciamento em C#

O Software de Gerenciamento é o responsável pelo envio dos códigos de cartão para os Pontos Remotos e requisição de remoção de códigos das bases de dados locais de cada Ponto Remoto.

O fluxograma, apresentado na Figura 28 mostra as funções do Software de Gerenciamento.

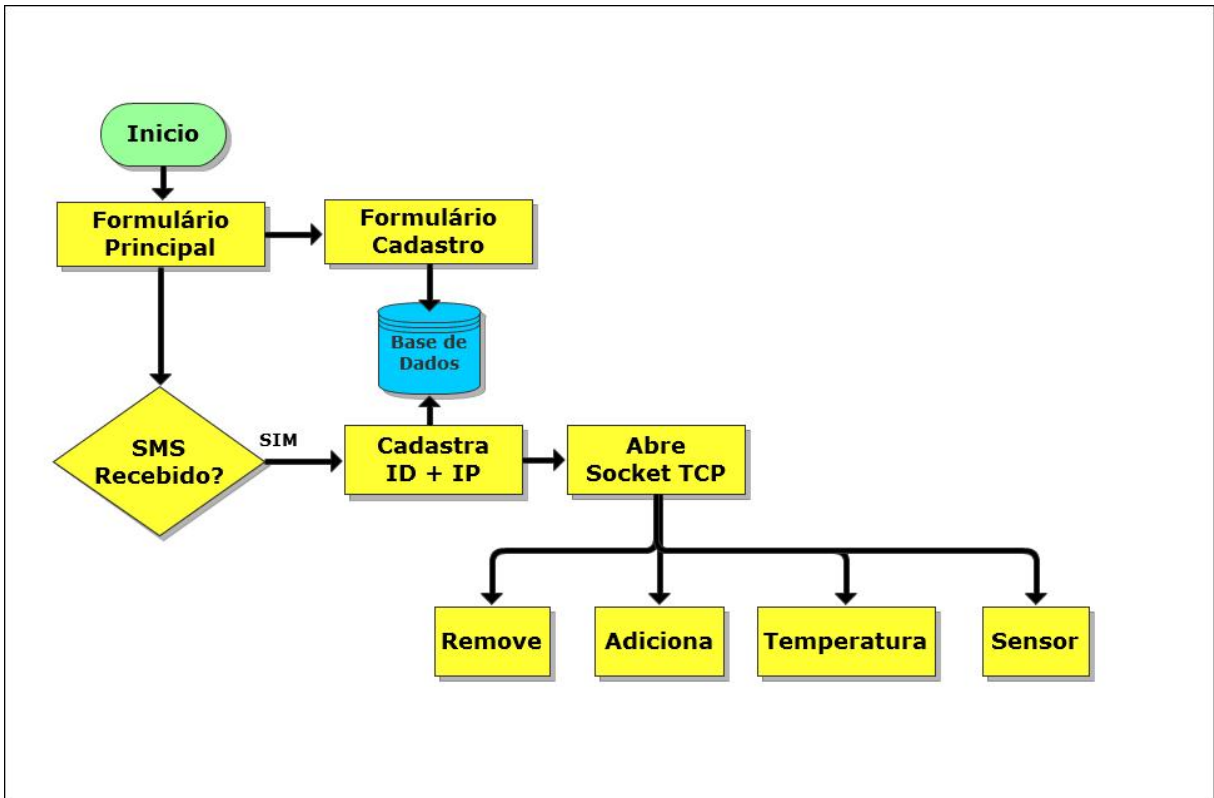


Figura 28 - Fluxograma do Software de Gerenciamento

## 6. MODO DE FUNCIONAMENTO

No diagrama de blocos da Figura 1, é apresentada a solução completa já com o Servidor Central com a base de dados. Observe que na Figura 1 temos os blocos que estão envolvidos por uma linha tracejada, os quais deverão repetir-se para cada Ponto Remoto onde a solução estiver instalada. Ainda na Figura 1 é possível notar que os periféricos usados na solução como Leitor RFID, sensores e fechadura magnética não são conectados ao M52259DEMOKIT diretamente, mas sim por meio de um hardware externo. Este hardware externo tem a função de fornecer as interfaces para os periféricos fazendo a adaptação dos devidos níveis de alimentação entre cada periférico e o micro-controlador. O hardware externo foi detalhadamente explicado no capítulo cinco que apresentou as etapas do desenvolvimento.

Este Capítulo tem o objetivo de esclarecer como cada um dos blocos presentes no diagrama da Figura 1 atua no contexto geral da solução proposta neste trabalho e como se darão as etapas de instalação da solução em um cliente.

## 6.1. Cadastro de usuários

Deve haver inicialmente um cadastro de todos os funcionários no servidor central visando obter uma base de dados e junto com este cadastro para cada funcionário será distribuído um crachá que é um cartão de RFID que deverá ter seu número cadastrado juntamente com os dados do funcionário. Ou seja, cada funcionário terá um código de RFID que será utilizado para habilitar ou não o acesso deste funcionário a determinados pontos remotos. Com a base de dados no Servidor Central definida e com os cartões de RFID distribuídos para os funcionários, é possível a instalação do Sistema de Controle de Acesso via GPRS em cada um dos pontos remotos os quais se deseja ter controle.

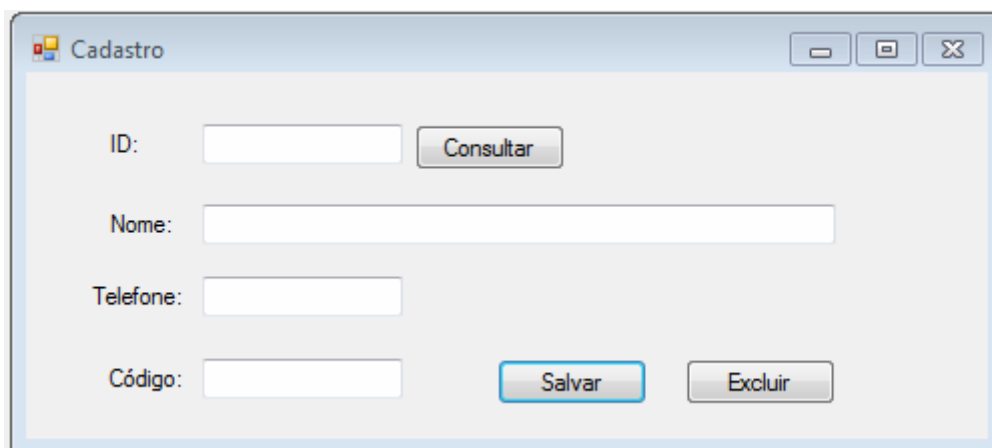


Figura 29 - Formulário de Cadastro de Usuário

Inicialmente pensou-se em ter uma base de dados local em cada ponto remoto, a qual armazenaria os códigos de RFID permitidos para aquele ponto remoto. Por uma limitação de tempo, a solução proposta agora é o envio dos códigos de RFID pelo software de gerenciamento toda vez que um módulo instalado em um Ponto Remoto é inicializado.



## 6.2. Inicialização Geral do Sistema

Na função de inicialização do software embarcado, mais precisamente da GPRS *task*, é realizada a configuração do modem GPRS para este receber um IP da operadora e em cima deste IP abrir um *socket* TCP em uma porta padrão, no projeto definida como 13000. Ainda na função de inicialização é feito o anúncio de cada ponto remoto que se dará através de um SMS que conterà o ID do ponto remoto e o IP que o mesmo obteve da operadora, como mostrado no fluxograma da Figura 26.

O ideal seria ter na central outro modem GPRS, o qual poderia receber esse SMS e já cadastrar automaticamente o ID e seu respectivo IP na base de dados do Servidor Central. Atualmente, por possuir somente um modem GPRS a solução apresentada é enviar um SMS para o celular de um operador e este terá que cadastrar o ID e seu respectivo IP no software de gerenciamento.

De posse do IP do ponto remoto o software de gerenciamento pode estabelecer a conexão TCP na porta padrão definida, 13000. Com a comunicação estabelecida, é possível enviar os códigos de RFID autorizados para aquele determinado ponto remoto. Essa seria então a inicialização do sistema de um modo geral.

Depois de inicializado o sistema de controle de acesso funciona localmente sem depender do software de gerenciamento no servidor central, pois o mesmo já tem agora os códigos de RFID que deve ou não autorizar o acesso. Mesmo assim a comunicação entre o software de gerenciamento e o ponto remoto fica estabelecida, com o objetivo de cadastro de novos códigos e ou remoção de códigos cadastrados, requisição de dados de acesso e estado dos sensores.

Desta forma o operador que estiver responsável pelo sistema no servidor central terá as informações de acesso e ou violação sobre todos os pontos remotos monitorados, sendo então esta uma solução de segurança para as operadoras sobre seus equipamentos instalados nesses pontos remotos.

## 6.3. Requisições do Software de Gerenciamento

Como já foi demonstrado no fluxograma da Figura 26, o modem GPRS recebe mensagens provenientes do Software de Gerenciamento. Essas mensagens, mostradas na Tabela 5, tem sempre uma resposta que deve ser enviada pelo Ponto

Remoto como mostra a Tabela 6. Assim então fica estabelecida a comunicação entre o Software de Gerenciamento e os Pontos Remotos.

#### 6.4. *Polling*

Para que o link estabelecido entre o Modem GPRS e o Servidor Central não caia, é necessário que antes de terminar o tempo de conexão sem tráfego de dados (configurado no Modem GPRS), é enviado uma mensagem para o Software de Gerenciamento somente para que o temporizador seja reiniciado. Essa tarefa de polling é feita também pela GPRS task.

## 7. RESULTADOS

Neste Capítulo serão apresentados os resultados obtidos após a implementação de cada uma das partes envolvidas neste complexo Sistema de Controle de Acesso via GPRS.

### 7.1. Comunicação Serial

Estabelecer a comunicação serial entre o modem GPRS Daruma e o M52259DEMOKIT, não foi uma tarefa trivial. Começando pelo fato dele recomendar que se trabalhe com as linhas de controle de fluxo CTS, RTS.

Os primeiros testes com o modem GPRS foram feitos com o PC, ou seja, utilizando um terminal no PC (minicom) o qual era possível enviar os comandos AT para o modem GPRS e receber as respostas.

Para a utilização do modem com o kit M52259DEMOKIT, foi necessária a confecção de um cabo serial DB9(macho)-DB9(macho), incluindo os pinos de CTS/RTS para estabelecer a comunicação com o modem. Ocorreu um sério problema na aplicação, mais especificamente na função que lia a resposta do modem e devido a esse problema o andamento do projeto atrasou. Basicamente, o problema era que o algoritmo de leitura não esperava tempo suficiente para que o modem mandasse todos os caracteres, fazendo com que a aplicação retornasse como resposta sempre somente o primeiro caractere recebido.

Após algum tempo debugando a aplicação, e com algumas conversas com colegas e o Professor Pastora respeito do problema foi encontrada a solução que foi esperar um tempo maior na função de leitura da serial a fim de receber todos os caracteres que o modem tenha para enviar.

## 7.2. Hardware Externo

O Hardware Externo apresentado no Capítulo 5, item 5.1, foi implementado com sucesso, desempenhando o papel esperado para ele no Sistema.

### 7.2.1. Construção

O resultado obtido após a confecção de uma placa dupla face em fibra de vidro, com o roteamento do esquemático, apresentado na Figura 20, feito pelo Software Eagle pode ser visualizado na Figura 30 que mostra a foto do HW Externo. É interessante observar que, trata-se de um protótipo e não de uma versão final. E por esse motivo, como em quase todo protótipo, houve correções de hardware após a montagem da placa.

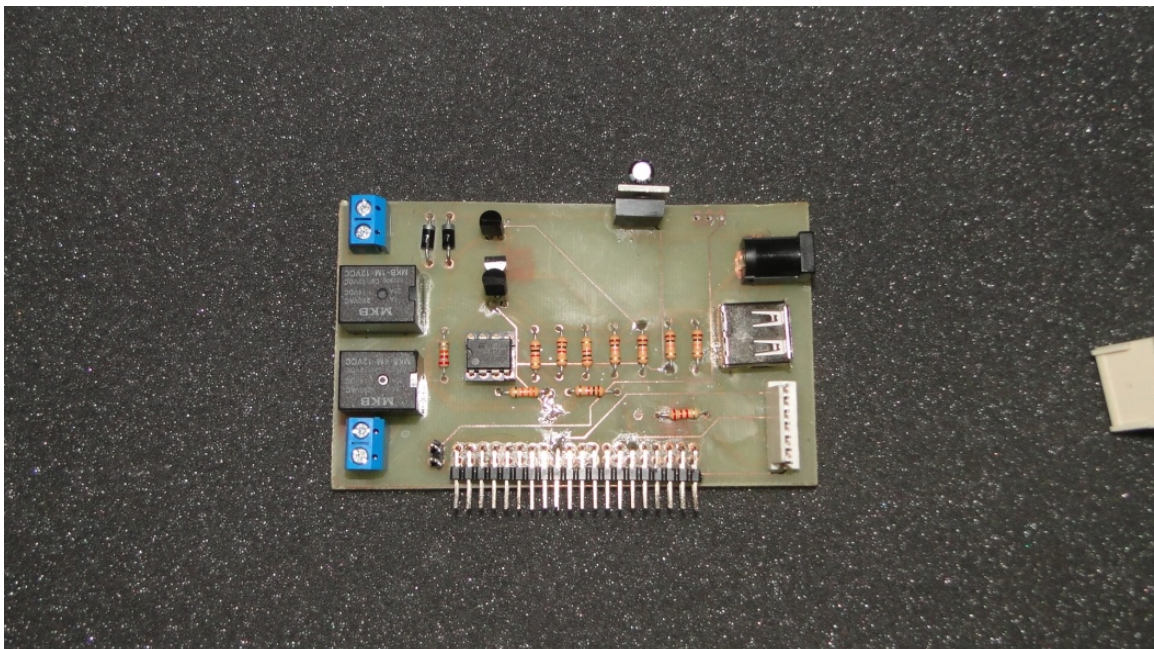


Figura 30 - Foto do Hardware Externo Finalizado

### 7.2.2. Interface para o leitor RFID

O leitor RFID é conectado ao hardware externo através do conector SV2 e este então tem seus pinos conectados aos respectivos pinos de IRQ e I/O do conector J4 da placa M52259DEMOCOM que por sua vez estão conectados ao processador MCF52259.

Aqui não houve problemas, e foi a primeira parte do hardware externo testada e que funcionou de acordo com o esperado. As linhas de dados D0/D1 provenientes do leitor RFID são conectadas aos pinos de IRQ7/IRQ1 e com esses pinos configurados corretamente na aplicação, esta consegui identificar as 26 interrupções ocorridas a cada aproximação do cartão ao leitor RFID. A partir desse ponto se deu o desenvolvimento com sucesso da parte de leitura e montagem do código do cartão e facility code que é o padrão utilizado nos sistemas de controle de acesso.

### 7.2.3. Demais circuitos

Os demais circuitos desenvolvidos no hardware externo também funcionaram sem problemas, com exceção do circuito que envolve o sensor de temperatura que não foi testado por falta de tempo. O desenvolvimento da aplicação ficou comprometido devido a perda de tempo considerável com os problemas na interface serial entre o modem GPRS e M52259DEMOKIT que era a base do projeto.

Fora isso, os circuitos de acionamentos de fechadura e sirene como os dois reles funcionaram e o sensor da porta também funcionou de acordo, sendo simulado com uma chave liga/desliga.

O circuito regulador de tensão LM3940 (3.3V) não foi encontrado nas lojas de componentes eletrônicos aqui de Curitiba e como é possível ver na foto da Figura 30 só temos o regulador LM7805. Portanto, a alimentação do M52259DEMOKIT tem que ser feita via cabo USB, uma vez que é a única maneira de alimentar o KIT, tanto na interface USB/BDM como na interface mini-USB.

A Figura 31 traz uma foto do Sistema com todos os componentes conectados.

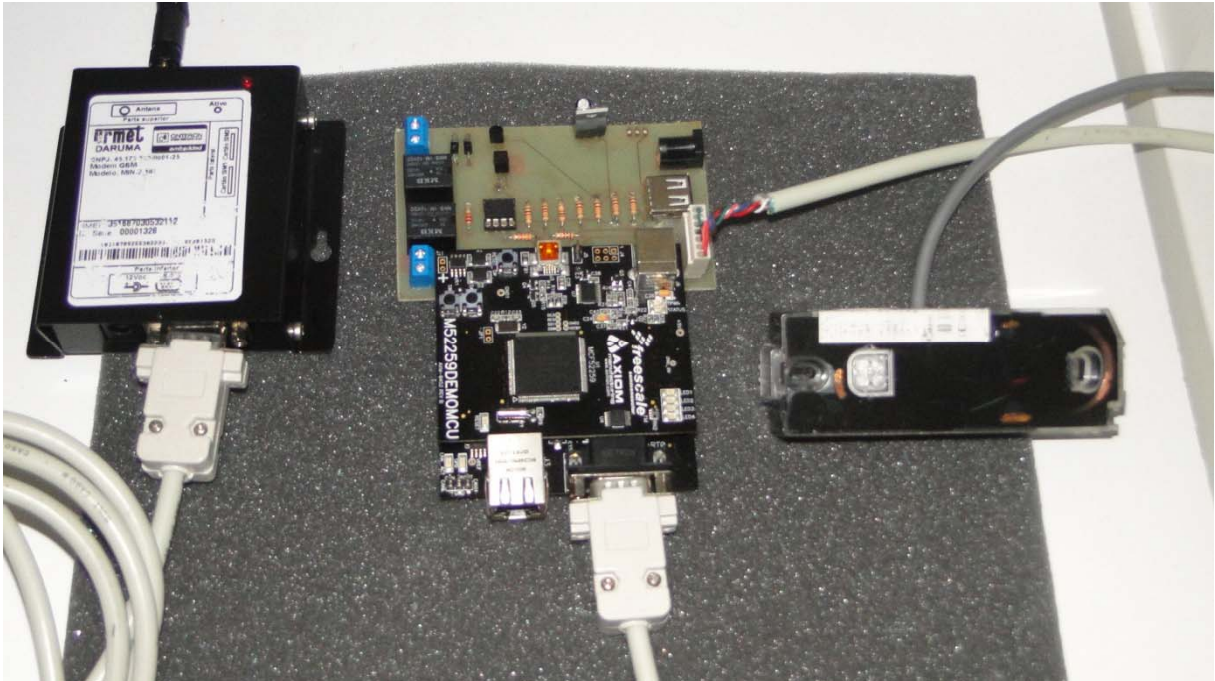


Figura 31 - Foto com todos os periféricos conectados

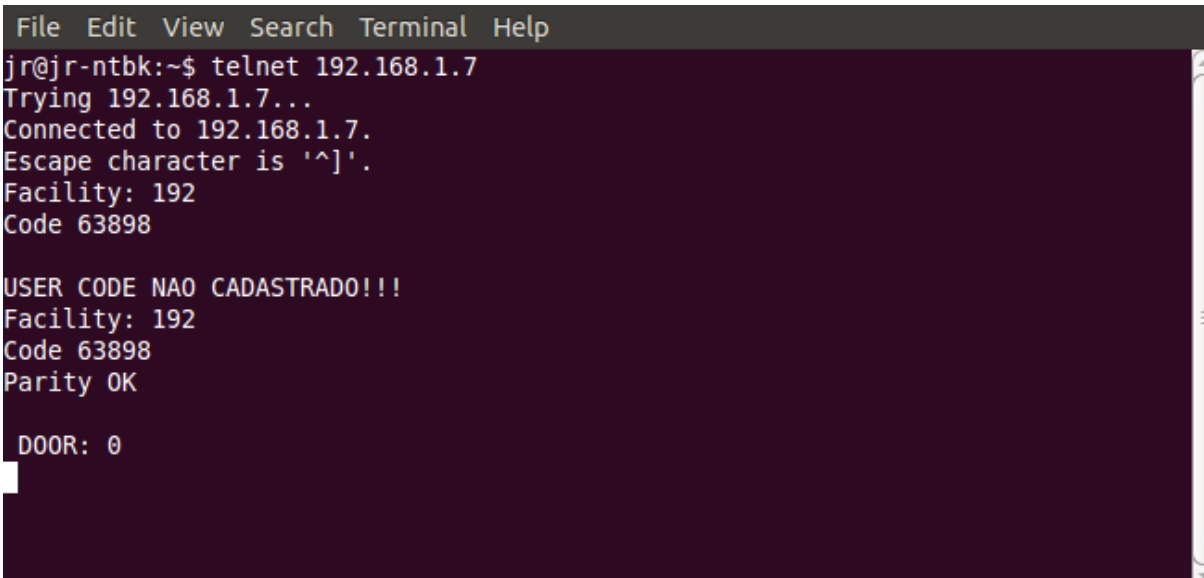
### 7.3. Debug

A Figura 32 apresenta uma tentativa de inicialização do M52259DEMOKIT, enviando três vezes o comando at. Como não obteve resposta, o LED4 fica permanentemente aceso indicando que houve falha na inicialização do modem GPRS. Na inicialização como ainda não tem o servidor telnet em execução, é necessário que o debug seja feito através da interface serial do PC, como é o caso da Figura 32.

```
File Edit View Search Terminal Help
Welcome to minicom 2.4
OPTIONS: I18n
Compiled on Jun 3 2010, 13:46:31.
Port /dev/ttyUSB0
Press CTRL-A Z for help on special keys
at
at
at
```

Figura 32 - Debug na inicialização pela serial

Já com o servidor *telnet* em execução, é possível executar uma operação de leitura de cartão e a aplicação imprime os resultados na interface de *telnet* o que facilita bastante a depuração. A Figura 33 demonstra o resultado de uma situação de aproximação de um cartão ao leitor RFID. Observe que na primeira leitura, a aplicação indica que o código do usuário não está cadastrado na base de dados local e, portanto o acesso é negado. Somente na segunda leitura o acesso é liberado e o código do cartão, *facility code* e a indicação de paridade são impressas. Isto foi implementado na aplicação para teste, uma vez que não tinha mais de um cartão para testes. O sistema inicia sem nenhum cartão cadastrado na base de dados local e a primeira vez que é executada a rotina de leitura de RFID é adicionado o código do cartão a base de dados.

A terminal window with a dark background and light text. The menu bar at the top includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main text shows a telnet session starting with 'jr@jr-ntbk:~\$ telnet 192.168.1.7'. It shows connection details, escape character, and two data reads. The first read shows 'Facility: 192' and 'Code 63898' followed by 'USER CODE NAO CADASTRADO!!!'. The second read shows 'Facility: 192', 'Code 63898', and 'Parity OK'. At the bottom, it shows 'DOOR: 0'.

```
File Edit View Search Terminal Help
jr@jr-ntbk:~$ telnet 192.168.1.7
Trying 192.168.1.7...
Connected to 192.168.1.7.
Escape character is '^]'.
Facility: 192
Code 63898

USER CODE NAO CADASTRADO!!!
Facility: 192
Code 63898
Parity OK

DOOR: 0
```

Figura 33 - Execução de leitura de um cartão RFID

Ainda com relação a Figura 33 é interessante observar que é impresso o estado do sensor da porta, no caso em 0, portanto fechado.

#### 7.4. Gerenciamento

Ainda não é possível apresentar nenhum resultado concreto sobre o Software de Gerenciamento. Devido aos atrasos, já comentados, o desenvolvimento do Software ficou bastante comprometido.

Os testes com a abertura do *socket* TCP no C# ocorreram sem problemas, portanto já existe essa classe implementada no código. A base de dados do servidor

também está implementada, porém o que falta implementar é justamente a interação de mensagens entre o Software de Gerenciamento e o modem GPRS, de acordo com as Tabelas 5 e 6.

## **8. CONCLUSÃO**

O trabalho contribuiu muito para enriquecer o conhecimento sobre o desenvolvimento de aplicações em Tempo Real, apresentando todas as dificuldades que um complexo projeto de desenvolvimento tem. A complexidade, inicialmente julgada, como não sendo um grande obstáculo, demonstrou-se ao longo da implementação como um grande desafio a ser vencido, pois a medida que o desenvolvimento caminhava, novas funcionalidades apareciam para serem implementadas.

As vantagens no desenvolvimento de uma aplicação, fazendo uso de um sistema operacional ficaram claras, o que faz com que seja considerado para Projetos futuros, mesmo de baixa complexidade, o uso de um RTOS mesmo que impactando no custo inicialmente. Este impacto no custo inicialmente pode ser compensado com um menor tempo de desenvolvimento e soluções de problemas futuros de forma mais rápida e eficaz.

A pesquisa sobre GPRS e RFID, duas tecnologias atualmente muito utilizadas, não só em sistemas de segurança, mas como em várias outras aplicações, fez com que idéias de novos projetos aparecessem podendo ser usado como base toda a implementação feita para esta Solução de Controle de Acesso.

O objetivo continua sendo, fazer deste Trabalho de Conclusão de Curso, um produto que possa ser oferecido a empresas de segurança com contrato com Operadoras de Telefonia móvel e fixa, com o objetivo de minimizar os problemas que as mesmas possuem com o acesso não autorizado aos seus armários espalhados por todo o país.

Por mais que, neste exato momento, o Projeto não atenda a todos os requisitos já mencionados ao longo do trabalho, por faltar parte da implementação do Software de Gerenciamento, pode ser considerado que o Projeto até aqui foi muito bem conduzido e por uma questão de tempo e seu alto nível de complexidade ainda não tem todas as funcionalidades propostas implementadas. Porém, teve grande parcela das funcionalidades implementadas e testadas, fazendo com que o

Projeto

seja

considerado

viável.



## 9. REFERÊNCIAS

1. SHILDT, H. **C Completo e Total**. 3.ed. São Paulo: Pearson Education do Brasil, 1996.
2. SHARP, J. **Microsoft Visual C# 2005 Passo a Passo**. Porto Alegre: Bookman, 2007.
3. **The History of RFID Technology**. Disponível em: <<http://www.rfidjournal.com/article/view/1338>>. Último acesso em 04/07/2011.
4. **RFID (Identificação por Radiofrequência)**. Disponível em: <[http://www.teleco.com.br/tutoriais/tutorialrfid/pagina\\_3.asp](http://www.teleco.com.br/tutoriais/tutorialrfid/pagina_3.asp)>. Último acesso em 04/07/2011.
5. **Leitor RFID**. Disponível em: <<http://www.sabereletronica.com.br/secoes/leitura/685>>. Último acesso em 04/07/2011.
6. **General Packet Radio Services**. Disponível em: <<http://www.scribd.com/doc/51277813/GPRS>>. Último acesso em: 04/07/2011.
7. **GPRS**. Disponível em: <[http://www.teleco.com.br/tutoriais/tutorialgprs/pagina\\_1.asp](http://www.teleco.com.br/tutoriais/tutorialgprs/pagina_1.asp)>. Último acesso em: 04/07/2011.
8. **Redes GSM e GPRS**. Disponível em: <<http://www.scribd.com/doc/9227795/Redes-GSM>>. Último acesso em: 04/07/2011.
9. **General RTOS Concepts**. Disponível em: <[http://documentation.renesas.com/eng/products/tool/apn/res05b0008\\_r8cap.pdf](http://documentation.renesas.com/eng/products/tool/apn/res05b0008_r8cap.pdf)>. Último acesso em: 04/07/2011.
10. **C# - Brief History**. Disponível em: <<http://www.cs.uwf.edu/~eelsheik/pl/csharp/home.html>>. Último acesso em: 04/07/2011.
11. **A Linguagem C#**. Disponível em: <[http://altabooks.tempsite.ws/capitulos\\_amostra/programando.pdf](http://altabooks.tempsite.ws/capitulos_amostra/programando.pdf)>. Último acesso em: 04/07/2011.
12. **Leitora MiniProx**. Disponível em: <[http://www.hidglobal.com/documents/miniprox\\_ds\\_pt.pdf](http://www.hidglobal.com/documents/miniprox_ds_pt.pdf)>. Último acesso em: 04/07/2011.
13. **HID Indala FlexSecur Technology**. Disponível em: <[http://www.hidglobal.com/documents/hidIndalaFlexsecur\\_wp\\_en.pdf](http://www.hidglobal.com/documents/hidIndalaFlexsecur_wp_en.pdf)>. Último acesso em: 04/07/2011.
14. **How to Develop I/O Drivers for MQX**. Disponível em: <[http://cache.freescale.com/files/32bit/doc/app\\_note/AN3902.pdf?fbsp=1&WT\\_TYPE=Application%20Notes&WT\\_VENDOR=FREESCALE&WT\\_FILE\\_FORMAT=pdf&WT\\_ASSET=Documentation](http://cache.freescale.com/files/32bit/doc/app_note/AN3902.pdf?fbsp=1&WT_TYPE=Application%20Notes&WT_VENDOR=FREESCALE&WT_FILE_FORMAT=pdf&WT_ASSET=Documentation)>. Último acesso em: 04/07/2011.