Universidade Federal do Ceará
Departamento de Engenharia de Teleinformática
Curso de Graduação em Engenharia de Teleinformática

Ricardo Gurgel de Sousa

Sistema de Identificação por Rádio-Frequência (RFID) Baseado em Plataforma ARM.

Fortaleza – Ceará
Dezembro 2011

Autor:
Ricardo Gurgel de Sousa

Orientador:
Prof. Dr. Helano de Sousa Castro

Sistema de Identificação por Rádio-Frequência (RFID) Baseado em Plataforma ARM.

Monografia de Conclusão de Curso apresentada à Coordenação do Curso de Graduação em Engenharia de Teleinformática da Universidade Federal do Ceará como parte dos requisitos para obtenção do grau de **Engenheiro de Teleinformática**.

Fortaleza – Ceará
Dezembro 2011

Ricardo Gurgel de Sousa

Sistema de Identificação por Rádio-Frequência (RFID) Baseado em Plataforma ARM.

Esta Monografia foi julgada adequada para a obtenção do diploma de Engenheiro do Curso de Graduação em Engenharia de Teleinformática da Universidade Federal do Ceará.

Ricardo Gurgel de Sousa

Banca Examinadora:

Prof. Dr. Helano de Sousa Castro
Orientador

Prof. Dr. George André Pereira Thé

Prof. Msc. Alexandre da Penha Coelho

Fortaleza, 13 de dezembro de 2011

Resumo

Identificação por Rádio-Frequência (do inglês, RFID) é uma tecnologia que usa ondas de rádio com o propósito de identificar e rastrear objetos ou pessoas. Essa tecnologia tem sido bastante utilizada atualmente, por tornar os processos de rastreamento muito mais dinâmicos, possibilitando a automatização destes e permitindo o atendimento a requisitos de tempo real, caso necessário. Um sistema de RFID consiste de uma etiqueta (tag) acoplada ao objeto a ser rastreado e uma leitora de etiqueta RFID. A etiqueta envia continuamente um sinal com uma codificação única que é captado pela leitora, que decodifica o dado e o usa para executar alguma ação sobre o objeto.

Uma possível aplicação seria a identificação de pessoas que portem etiquetas, permitindo executar ações de controle de acesso às mesmas. Com a adoção de protocolos robustos, como o EPC, e a maior gama de possibilidades de autenticação que uma tag oferece, a identificação por RFID é uma ótima opção para aumentar a confiabilidade desses sistemas. Tendo tais vantagens como motivação, este projeto propõe a concepção e implementação de um sistema de controle de acesso baseado em tecnologia RFID.

Neste projeto, será utilizada uma plataforma ARM, um microprocessador amplamente utilizado na área de sistemas embarcados. Tais microprocessadores são conhecidos por serem potentes e versáteis. Além disso, muitas organizações promovem o suporte de sistemas operacionais ao ARM, como o Linux. Combinando todas as vantagens fornecidas pela arquitetura com o arcabouço proveniente do Linux, temos um sistema computacional completo, compacto e de baixo custo, capaz de suportar as necessidades de processamento desta aplicação.

Palavras-chaves: RFID, ARM, Embedded Systems, Linux, LTIB.

Abstract

Radio Frequency Identification (RFID) is a technology which, through the use of radio frequency fields, identifies and/or tracks objects or persons. This technology has been very widespread - largely in logistic industry - by turning tracking and inventory processes much more dynamics, allowing the automatization of this processes and enabling the fulfillment of real-time constraints, if necessary. Basically, RFID system consists of a tag owned by the target and a tag reader. The tag sends a signal encrypted to the reader, which decrypts the data and uses it to performs some action over the target.

A common application for RFID is the identification of persons that carry tags, allowing to execute actions of access control. With the support of a robust protocol, as EPC, and the many possibilities of authentication that a tags offers, the usage of RFID in control access systems is a great option in order to magnify the reliability of this systems. Having these advantages as motivation, this project proposes the conception and implementation of a RFID-based access control system.

In this project, will be used a ARM-based platform, a microprocessor largely used on the embedded systems area. This microprocessors are know by have a powerful and versatile architecture. Besides, many organizations grants the support of operational systems to ARM, as Linux. Then, combining all the advantages provided by the ARM with the Linux environment, we have a complete computational system, compact and low cost, capable support all the requirements of this application.

Keywords: RFID, ARM, Embedded Systems, Linux, LTIB.

Dedico este trabalho primeiramente a Deus, pois sem Ele nada é possível. Dedico também a Francisco e Sileneide, meus amados pais.

"Tudo o que temos de decidir é o que fazer com o tempo que nos é dado."
J.R.R Tolkien

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

RFID *Radio Frequency Identification*

EPC *Electronic Product Code*

RF Rádio-Frequência

LF *Low Frequency*

HF *High Frequency*

UHF *Ultra High Frequency*

SHF *Super High Frequency*

RN16 *Random Number - 16 bits*

ACK *Acknowledgement*

CRC16 *Cyclic Redundancy Check - 16 bits*

ARM *Acorn RISC Machine*

RISC *Reduced Instruction Set Computing*

MMU *Memory Management Unit*

SO Sistema Operacional

DHCP *Dynamic Host Configuration Protocol*

TFTP *Trivial File Transfer Protocol*

BSP *Board Support Package*

LTIB *Linux Target Image Builder*

RPM *RPM Package Manager*

QSB *Quick Start Board*

PCI Placa de Circuito Impresso

DRC *Design Rule Check*

GPIO *General Purpose Input/Output*

Capítulo 1

Introdução

1.1 Objetivos

1.1.1 Geral

Desenvolver um sistema computacional de baixo custo capaz de suportar as principais aplicações da tecnologia RFID.

1.1.2 Específicos

i. Levantamento de requisitos e componentes necessários para a solução: sistema embarcado(ARM), leitora RFID e antenas;

ii. Confecção de placas de circuito impresso necessárias para integração entre os componentes;

iii. Desenvolvimento de um ambiente computacional capaz de gerenciar um sistema de controle de acesso;

iv. Testar as funcionalidades do sistema e analisar os resultados obtidos;

1.2 Organização do Trabalho

A seguir, é apresentada a estrutura e um breve resumo dos capítulos e apêndices que compõem este trabalho.

No capítulo 2, é apresentado um breve embasamento teórico acerca do RFID, principal assunto deste trabalho. Aqui discorre-se sobre os principais conceitos da

tecnologia, seus componentes e protocolos de comunicação utilizados, além de um sucinto histórico sobre o assunto.

Ainda na introdução teórica, o capítulo 3 descreve a arquitetura de microcontroladores ARM e suas características principais. A seguir, temos o capítulo 4, que conceitua o sistema operacional linux e seu uso na área de sistemas embarcados. Ainda neste capítulo, é descrita a ferramenta de desenvolvimento de imagens linux para plataformas customizadas, chamada LTIB.

No capítulo 5, a metodologia que este trabalho seguiu é explanada e se apresentam todos os componentes, tanto de hardware quanto de software, que definem o projeto. No capítulo de resultados, são mostrados os dados adquiridos acerca de testes de funcionalidade e de performance do sistema de controle de acesso.

Finalmente, em conclusão e trabalhos futuros discute-se se as expectativas apresentadas na metodologia foram cumpridas através da análise dos resultados, bem como as dificuldades encontradas durante a implementação, e propõe-se próximos passo no aprimoramento deste projeto.

Capítulo 2

Conceitos de RFID

2.1 Histórico

Como boa parte da tecnologia do século passado, o *Radio Frequency Identification* (RFID) - sigla inglesa para Identificação por Rádio Frequência - é uma herança da Segunda Guerra Mundial, onde sistemas de radares eram usados para identificar a presença de aviões que se aproximasse, mesmo antes de entrar em campo visual, permitindo que os militares se preparassem.

O sistema de radar funcionava a base de ondas eletromagnéticas que eram projetadas para o horizonte. Um receptor captava as ondas que eram refletidas pelos aviões e avisavam da presença dos mesmos. Conta-se que os aviões alemães executavam manobras ao se aproximar de suas bases, causando variações no sinal refletido e recebido pela base, o que era entendido como uma identificação de avião amigo pela base. Este é considerado o primeiro sistema de RFID passivo da história.

Assim, neste âmbito de guerra tecnológica, em 1937 Sir Robert Alexander Watson-Watt liderou um projeto que visava o desenvolvimento de um dispositivo de rádio frequência que pudesse ser implantado nos aviões britânicos e identificá-los para as bases. Seus esforços culminaram no IFF(*Identification Friend-or-Foe*)(HESSEL, 2009), que era um sistema composto por duas partes: um interrogador - o próprio radar em terra - e um transponder, que ficava a bordo da aeronave.

Desta forma, quando a aeronave se aproximasse de uma base aliada, o interrogador gerava um sinal de rádio frequência, que era captado pelo transponder,

o qual modulava e reenviava para a base, que, por sua vez, identificava ou não o avião como aliado. Apesar dos avanços tecnológicos, a dupla transponder-radar é usada até hoje na aviação civil ou militar.

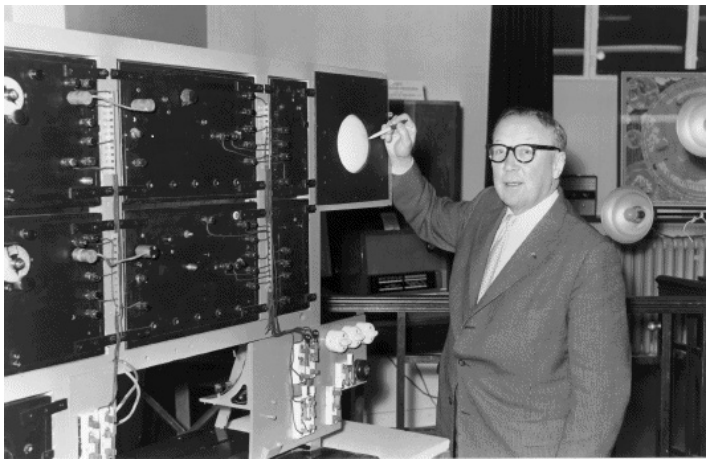


Figura 2.1: Sir Alexander Watson-Watt e seu invento.

Nas décadas posteriores ao seu invento, o RFID era de uso quase exclusivo militar, sendo ainda empregado em identificação de aeronaves. Porém, no início da década de 70, outras patentes começaram a surgir tendo como assunto o RFID, mas agora empregado na identificação de outros objetos, como itens de inventário ou em lojas de departamento.

Foi assim que o RFID ganhou um de seus mais importantes nicho de aplicação, o de prevenção de furtos em lojas de varejo. Nestes sistemas, as tags, equivalentes aos transponders, são acopladas aos objetos e quando na presença do campo de rádio frequência, emitem um sinal de identificação, que indica o extravio de um objeto. Até hoje nos deparamos como tais sistemas em grandes lojas - onde os leitores são posicionados próximos as saídas e as tags são pequenos broches, retirados pelo operador de caixa.

Com o crescente interesse das grandes indústrias e do Departamento de Defesa dos Estados Unidos pelo RFID, seu crescimento após a década de 70 foi rápido, não tardando para o barateamento do custo de implementação e o surgimento de aplicações das mais diversas na área de cadeia de suprimentos, tais como, inventário de pallets e containers, rastreamento de mercadorias e rebanhos, controle de acesso, entre outros.

Porém, por se tratar de uma tecnologia que usa como meio físico de propagação o ar - através de ondas eletromagnéticas - os sistemas RFID são passíveis de interferência em outros equipamentos, como telefones celulares e equipamentos de rádio-frequência em geral. Outro problema é a diversidade de frequências de operação dos sistemas, fato ocasionado pelo desenvolvimento do RFID em vários países ao mesmo tempo. Pensando nestas dificuldades e também com o intuito de formar um padrão universal, em 2003 foi formado o grupo EPCGlobal.

Figura 2.2: As várias aplicações da tecnologia RFID.

Este grupo tem como função a administração de padrões que definem os componentes e a arquitetura de sistemas RFID participantes da Rede EPCGlobal. O objetivo desta rede é reunir leitores e tags em um sistema de informação - através do *Electronic Product Code* (EPC) - capaz de tornar um produto rastreável ao longo de toda uma cadeia de suprimentos, de fabricantes a distribuidores, por fim chegando aos vendedores(GLOVER; BHATT, 2006).

Tal integração de informações tende a crescer cada vez mais, impulsionada pela tendência de barateamento do custo de implantação de sistemas RFID. As tags

passarão a ser implantadas em cada vez mais itens e dispositivos pessoais, como smartphones, possuirão leitoras RFID, assim como têm câmeras hoje.

Esta evolução culminará com o que é chamado por especialistas como a era da "Internet das Coisas", onde um objeto qualquer poderá ser identificado por sua assinatura única através de uma tag RFID, fazendo com que, por exemplo, possamos ler uma tag de um equipamento que não saibamos usar e, através de uma busca rápida na internet, obtenhamos seu manual e um vídeo explicativo do mesmo.

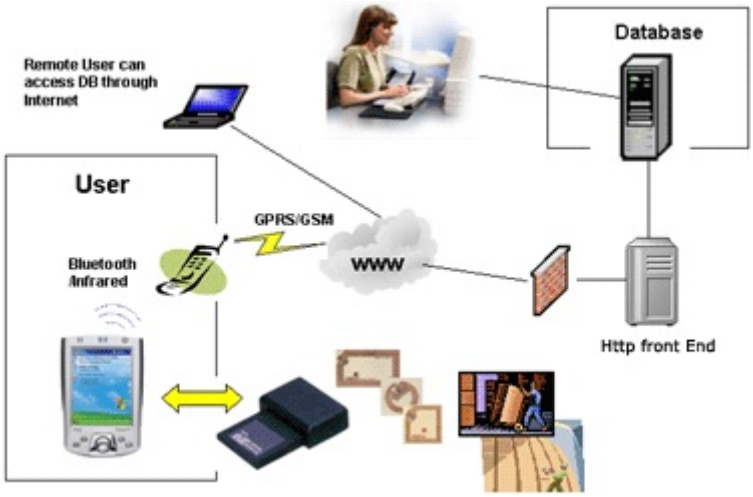


Figura 2.3: A internet das coisas.

Apesar de parecer bastante vantajoso, ainda haverá muito a se discutir até chegarmos a esta era. Um dos pontos discutíveis está relacionado com a invasão de privacidade. De fato, como as tags se tornarão cada vez menores e com uma vida útil maior, muitos usuários podem sentir que sua privacidade está sendo violada quando tags estão presentes em produtos e os usuários não forem avisados. De fato, se uma pessoa possui um objeto que contém uma tag, tal indivíduo agora também é passível de ser rastreado. Porém, estes são limites que a ética e a legislação ainda haverão de resolver.

2.2 Componentes

Um sistema RFID é formado basicamente por dois componentes: leitoras e tags.

2.2.1 Leitoras

Leitoras(ou readers) RFID são os dispositivos responsáveis por estabelecer a interface entre um sistema de computação e as tags. Sua estrutura básica é composta por um microcontrolador, que fornece a interface digital e administra os protocolos de comunicação, e de um circuito de Rádio-Frequência (RF), necessário para gerar o campo magnético que interage com as tags.

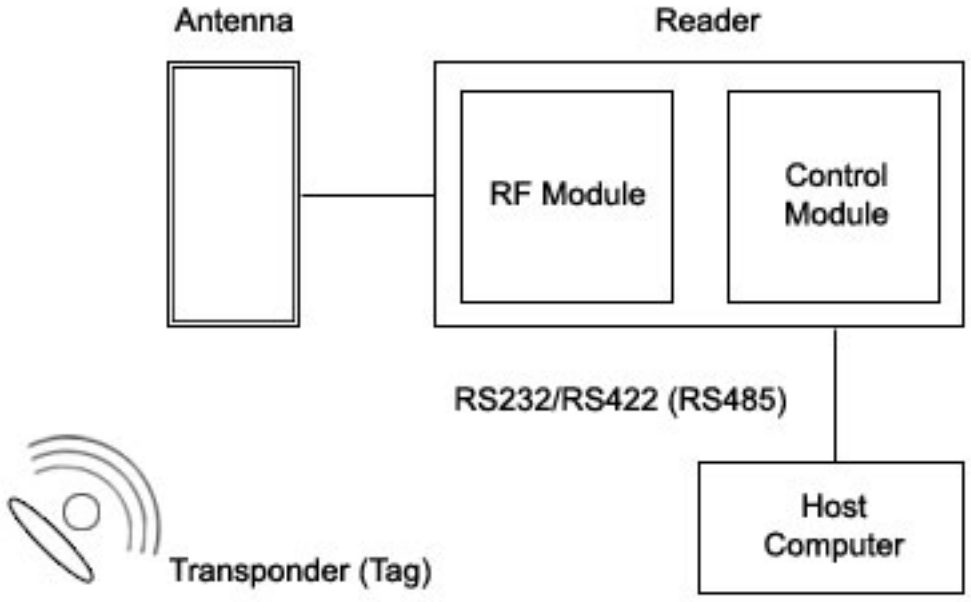


Figura 2.4: Diagrama de blocos de uma leitora.

Quanto ao alcance, as leitoras podem ser classificadas como de proximidade ou de longo alcance. As leitoras de proximidade tem o seu campo de ação limitado a poucos centímetros da antena. Ou seja, é empregada em aplicações onde apenas uma tag é apresentada por vez, como em controle de acesso.

Já as leitoras de longo alcance são dotadas de antenas que abrangem um campo relativamente grande, a depender da aplicação(galpões, lojas, fazendas). Tais leitoras são capazes de manipular várias tags ao mesmo tempo e listá-las de acordo com seus números de identificação, procedimento conhecido como inventário.

Leitoras RFID também podem ser classificadas quanto a frequência em que operam. Para garantir que a comunicação entre tags e leitoras não esteja em uma banda já alocada para outro serviço, causando interferências, foram estabelecidas classes de frequências, que especificam a faixa de operação permitida. A seguir, é

mostrada uma tabela com as frequências mais utilizadas pela tecnologia RFID e suas principais aplicações.

Frequência	Nome	Aplicações
135 kHz	*Low Frequency* (LF)	Controle de Acesso, Smart Cards
13.553 13.567 MHz	*High Frequency* (HF)	Smart Cards
860 - 960 MHz	*Ultra High Frequency* (UHF)	Smart Labels
2.446 - 2.454 GHz	*Super High Frequency* (SHF)	Identificação de Veículos

Tabela 2.1: As principais faixas de frequência do RFID.

Outro componente que influencia bastante no funcionamento da leitora é a antena utilizada. A depender da aplicação, um tipo diferente de layout de antena é empregado. O layouts mais comuns de antenas são: Portais, Túneis e antenas portáteis, como apresentado na figura 2.5. Antenas em portal são geralmente empregadas em aplicações típicas de logística, onde as tags são identificadas quando passam por um portão onde a leitora é instalada. É ideal para controle de fluxo de mercadorias e prevenção de furtos.

Antenas em túnel são empregadas em casos onde os objetos possam ser dispostos em uma linha de montagem, ou esteira rolante. Sua estrutura fechada possibilita o uso de técnicas de blindagem contra ondas de rádio, o que torna o layout em túnel bem menos susceptível a interferências externas em suas leituras.

(a) (b) (c)

Figura 2.5: Portal(a), túnel(b) e dispositivo portátil(c).

As antenas portáteis são integrantes de dispositivos RFID handheld, que são pequenos sistemas embarcados usados para fazer leituras in loco em situações nas quais seja inconveniente a locomoção do objeto até o alcance de uma antena. Neste

tipo de aplicação, o uso de RFID assemelha-se ao código de barras, por isso é comum que tais dispositivos também possuam esta funcionalidade(GLOVER; BHATT, 2006).

2.2.2 Tags

Enquanto as leitoras RFID são os componentes que ficam - geralmente - fixos em um local, as tags(ou etiquetas) são os componentes que se encontram acoplados nos objetos identificados. Sua função é basicamente agregar informação ao objeto vinculado e fornecer estes dados quando interrogada por uma leitora.

Figura 2.6: Alguns tipos de tags.

No que se refere a custo, as tags representam hoje o maior gargalo para a implantação em massa do RFID, pois o custo de oportunidade das tags deve competir com as etiquetas de código de barra, que são basicamente papel e tinta. Porém, grandes avanços no processo de fabricação já estão sendo alcançados, sendo possível o preço de R$ 0,60 centavos por unidade, para quantidades de centenas ou alguns milhares(HESSEL, 2009).

Uma tag é composta por uma máquina de estado e de um circuito de RF, responsável - como nas leitoras - pela modulação e decodificação dos dados. Além de uma máquina de estado, as tags também possuem uma memória não-volátil, que armazena seu identificador único. Uma tag pode ser classificada de acordo com dois parâmetros: Quanto ao tipo de armazenamento e quanto a sua fonte de alimentação. De acordo como tipo de memória presente nas tags, elas podem ser classificadas como(HESSEL, 2009):

Somente Leitura : Seu identificador é gravado durante a fabricação, sem possibilidades de modificações posteriores.

Uma Escrita/Várias Leituras Pode ser gravada uma única vez depois de sua fabricação. Após isso, não é permitida a alteração de seus dados.

Escrita/Leitura : Permite o livre manuseio de seus dados ao longo de sua vida útil. Por ser mais vulnerável a falsificações, geralmente possuem a opção de serem bloqueadas, através de uma senha, ou mesmo desativadas permanentemente. É comum a presença de uma área de memória para uso geral nestas tags.

A vantagem dos dois primeiros tipos de tag é o preço, pois o fato de serem fabricadas em grande escala, sem grandes diferenças entre os lotes, faz com que o custo caia bastante, o que é necessário para aplicações de grande rotatividade de tags, como smartcards e crachás eletrônicos.

Já o terceiro tipo de tag é muito utilizado em aplicações onde necessita-se de uma capacidade maior de customização das tags. Por exemplo, uma tag de escrita/leitura, além de seu código identificador, poderia carregar alguma informação crítica em relação ao objeto identificado, como o prazo de validade, caso se trate de um produto perecível.

Quanto a fonte de alimentação, uma tag pode ser classificada como passiva, semi-ativa ou ativa. Tags passivas compõem o tipo mais difundido destas, pois são bem mais baratas e de fácil manuseio, sendo apresentadas no mercado em diversos tamanhos e formas: desde adesivos(smartlabels) a chaveiros.

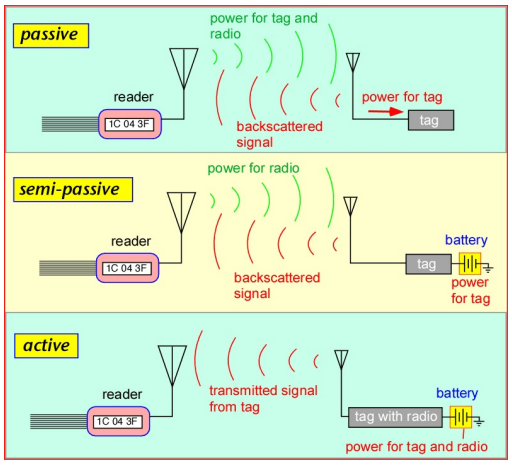


Figura 2.7: Classificação das tags quanto a alimentação.

A principal característica das tags passivas é a ausência de bateria ou qualquer fonte interna de alimentação, utilizando apenas a energia induzida pela portadora

do sinal de RF emitido pela leitora. Assim, este tipo de tag tem seu alcance e capacidade de resposta limitados pela potência que a leitora é capaz de fornecer. Por não necessitarem de trocas de baterias, possuem uma vida útil bastante longa.

Similares as passivas, as tags semi-passivas só estabelecem uma comunicação quando na presença de uma leitora. Porém, estas possuem uma bateria interna, responsável por alimentar sua máquina de estados. Tal fato permite que a tag possa responder a distâncias maiores, além de exigir menor potência da leitora para estabelecer uma comunicação.

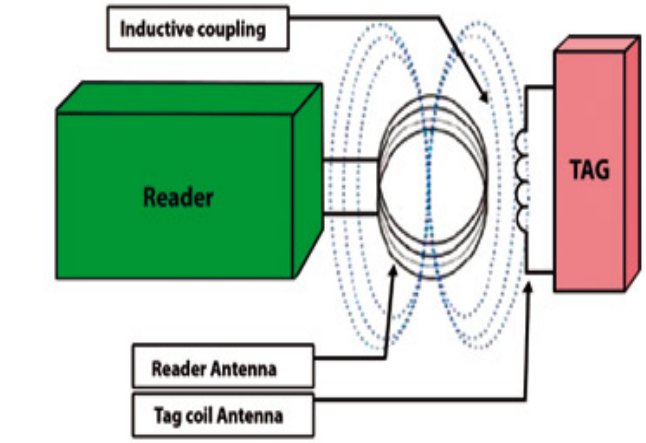
Ambos os tipos têm em comum o fato de não possuirem um circuito de transmissão, pois, apesar de terem uma antena, não são capazes de gerar seu próprio campo magnético. Assim, para estabelecer uma comunicação com a leitora, as tags passivas e semi-passivas utilizam duas técnicas de comunicação: acoplamento indutivo ou modulação backscatter.

No acoplamento indutivo, as leitoras fornecem corrente para suas antenas, que tem uma forma espiral. Esta corrente produzirá um campo magnético, que por sua vez induzirá uma corrente no circuito de recepção da tag, como mostrado na figura 2.8(a). Com essa carga produzida, a tag pode então comunicar-se com a leitora, provocando modificações no campo magnético que serão percebidas como alterações de voltagem nos terminais da antena da leitora. No acoplamento indutivo, tag e leitora devem estar próximas e alinhadas, o que é típico de sistemas que trabalham em frequências das faixas LF e HF.

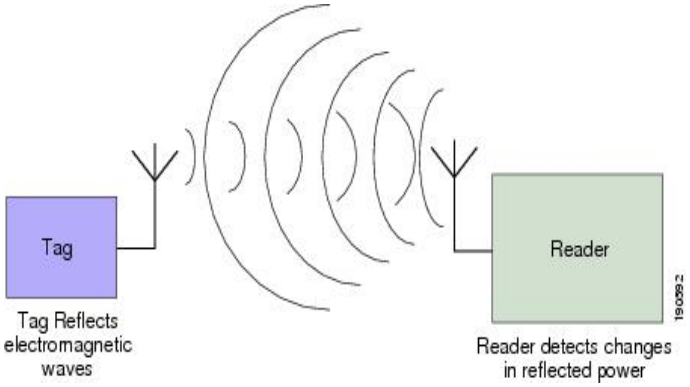
Em (GLOVER; BHATT, 2006), a modulação backscatter é definida como uma técnica onde a tag, similar ao acoplamento indutivo, utiliza-se do campo magnético fornecido, refletindo ou não uma parte das ondas de rádio recebidas, que são captadas de volta pela leitora.

Uma analogia interessante seria a de duas pessoas se comunicando a distância com um espelho e uma lanterna, onde o indivíduo com o espelho pode alternar entre refletir ou não a luz da lanterna, gerando assim um código que pode ser captado pelo primeiro. Esta técnica é utilizada por sistemas que trabalham na faixa de UHF, onde tag e leitora não precisam estar exatamente alinhadas nem próximas.

Diferente dos dois primeiros tipos de tags, as do tipo ativa possuem um transmissor próprio, que as possibilita um alcance ainda maior do que as semi-passivas. Tais tags têm como característica principal a capacidade de tomar a



(a)



(b)

Figura 2.8: Acoplamento indutivo(a) e modulação backscatter(b).

iniciativa em uma comunicação com leitoras vizinhas.

Em exemplares mais sofisticados, as tags são dotadas de sensores responsáveis por monitorar alguma característica do objeto - como temperatura, por exemplo. Tal configuração permite que a tag possa gerar um alarme e dispará-lo prontamente, ou mesmo manter um registro das leituras de seus sensores em sua memória não-volátil.

2.3 Protocolos de Comunicação

Por se tratar do ar o meio físico utilizado pela tecnologia RFID, muitas vezes a comunicação entre tags e leitoras é alvo de interferências vindas tanto de fontes externas quanto de eventuais colisões provocadas por duas ou mais tags que respondam a requisição da leitora ao mesmo tempo. Com o intuito de mitigar leituras espúrias - seja por colisões ou ruídos externos - protocolos de comunicação

específicos são empregados.

Estes protocolos têm como característica principal a divisão da comunicação em turnos, dividindo assim as tags em grupos menores, fazendo com que a probabilidade de duas tags responderem ao mesmo tempo diminua. A seguir, serão discutidos os dois principais protocolos utilizados: o Slotted Aloha e o EPC Class 1 Generation 2.

2.3.1 Slotted Aloha

O protocolo Slotted Aloha - derivado do Aloha puro, um protocolo utilizado na AlohaNet - é típico em aplicações de múltiplo acesso ao meio, como é o caso do RFID. Ele destaca-se por sua simplicidade, o que torna o overhead da comunicação bem menor, distribuindo o controle da comunicação entre os elementos passivos e ativos - no caso tags e leitoras, respectivamente.

Basicamente, o Slotted Aloha funciona da seguinte forma: A leitora envia um sinal de requisição através de seu campo, fazendo com que todas as tags que o escutaram respondam enviando o seu identificador. Porém, quando a requisição é feita, a leitora informa às tags quantas unidades de tempo estão disponíveis para as mesmas, chamadas slots. Assim, cada tag escolhe aleatoriamente um slot, que será o período no qual a tag reenviará o seu identificador. Após o último slot, a contagem é reiniciada. A este ciclo, damos o nome de turno, ou round(figura 2.9).

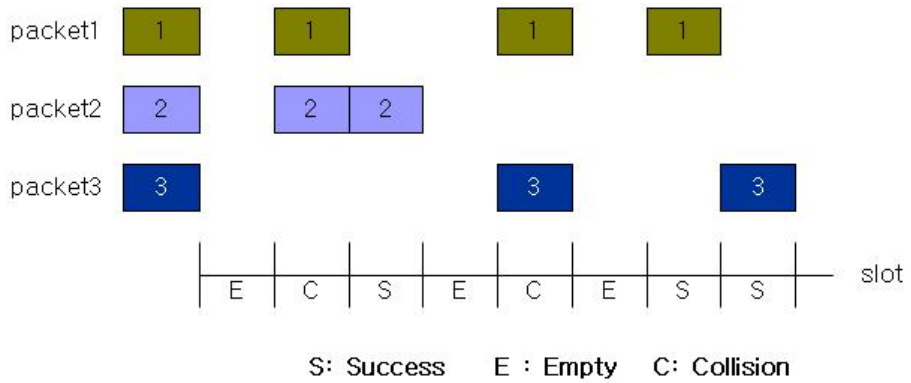


Figura 2.9: Transmissão no protocolo Slotted Aloha.

Após enviar a requisição, a leitora aguarda a resposta das tags, esperando até que se possa fazer uma leitura válida. Quando ocorre uma colisão, por ser o elemento passivo, a tag não toma conhecimento do que aconteceu e continua a

operar. Apenas a leitora, conhecendo os padrões de tensão resultantes de uma leitura com interferência, detecta a mesma e a descarta.

Caso ocorra uma transmissão sem colisão, a leitora envia um comando direcionado a tag com o identificador lido, chamado select. Ao receber este comando, a tag entra em um estado de dormência e não responde mais em seu slot de tempo. O algoritmo continua até que não haja mais resposta de nenhuma tag.

Como foi dito, a principal qualidade deste protocolo é sua simplicidade, pois requer poucas sinalizações entre as leitoras e as tags antes que se inicie a comunicação de fato. Porém, este protocolo não é indicado para situações de alta população de tags, visto que, caso a leitora forneça um número de slot inferior ao de tags, muitas colisões aconteceram, forçando a leitora a enviar outra requisição com mais slots até que a ocorrência de colisões alcance níveis tolerável. A depender do número de tags presentes, este processo pode acarretar atrasos consideráveis.

2.3.2 EPCGlobal Class1 Gen2

Como vimos em 2.1, a EPCGlobal é uma instituição sem fins lucrativos que visa criar uma padronização mundial para a tecnologia RFID. Desta forma, em 2004, o padrão EPCGlobal Class 1 Generation 2, ou simplesmente EPC Gen2, foi criado. Ele define os requisitos físicos e lógicos para um sistema operando na faixa de 860-960MHz implementado com tags passivas, prezando pela segurança dos dados(EPCGLOBAL, 2011).

Semelhante ao Slotted Aloha, no protocolo EPC Gen2 a leitora é responsável por iniciar toda e qualquer comunicação. Além disso, a população de tags também é dividida em grupos menores, com a intenção de evitar colisões. Porém, no protocolo EPC Gen2, existem mais divisões, são elas:

Sessões : São quatro(de S0 a S3) e uma tag só pode participar de uma por vez;

Tipos : Todas as tags pertencem a um dos tipos(A ou B), sendo possível a troca de um tipo para o outro, caso a leitora requisite;

Slots : Número de espaços de tempo definidos que a leitora disponibiliza para as tags.

Em um sistema EPC Gen2, as leitoras possuem três funções básicas:

Seleção : Processo no qual, especificando a sessão e o tipo de tag, a leitora isola um subconjunto da população de tags;

Inventário : Listagem de todas as tags que respondam a requisição de identificação em uma determinada sessão, através da obtenção de seu EPC;

Acesso : Acessar a área de memória de uma tag, modificar permissões de escrita/leitura(Lock) e desativar tag(Kill)

Na etapa de seleção, a leitora deve especificar em seu comando de requisição - chamado Query - qual sessão e tipo de tag deve responder, dando início a um inventário, onde todas as tags selecionadas responderão em seus slot de tempo. Assim, no fim de um inventário, a leitora possui todos os identificadores referentes a população selecionada, sendo possível o acesso individual a cada tag. Neste estágio, a leitora pode escrever ou ler na área de memória da tag, a depender da presença de uma senha de acesso.

Segundo (EPCGLOBAL, 2011), para obedecer os padrões de comunicação do protocolo, uma tag deve se comunicar utilizando modulação backscatter e obedecer os comandos das leitoras quando participando de um inventário. Além disso, a tag deve implementar os seguintes estados: Ready, Arbitrate, Reply, Acknowledged, Open, Secured e Kill.

Assim que a tag está pronta para operação - ou seja, adquiriu energia suficiente para funcionar - esta deve entrar no estado Ready, onde a tag espera por uma requisição de uma leitora para participar de um inventário. É o estado inicial da comunicação.

Caso receba um comando Query, a tag entra no estado Arbitrate. Neste estado, a tag deve gerar um número aleatório dentro de um limite fornecido pela leitora, carregá-lo em um contador - slot counter - e decrementá-lo toda vez que a leitora enviar o comando QueryRep.

Quando o contador atingir zero, a tag deve entrar no estado Reply, onde a mesma gera e envia um *Random Number - 16 bits* (RN16) pra a leitora. Caso receba um*Acknowledgement* (ACK), entra no estado Acknowledged e envia para a leitora seu EPC e o respectivo *Cyclic Redundancy Check - 16 bits* (CRC16). Neste estado, caso receba uma requisição da leitora, a tag pode ir para o estado de Secured, se não tiver senha de acesso, ou Open, caso haja uma senha.

Neste ponto, a tag pode receber comandos de acesso a memória, bloqueio de escrita/leitura ou até ser desativada, passando para o estado Kill. Uma tag que se encontre neste estado não responde mais a qualquer comando de leitoras. Na figura 2.10, é mostrado um fluxograma do protocolo EPC Gen2.

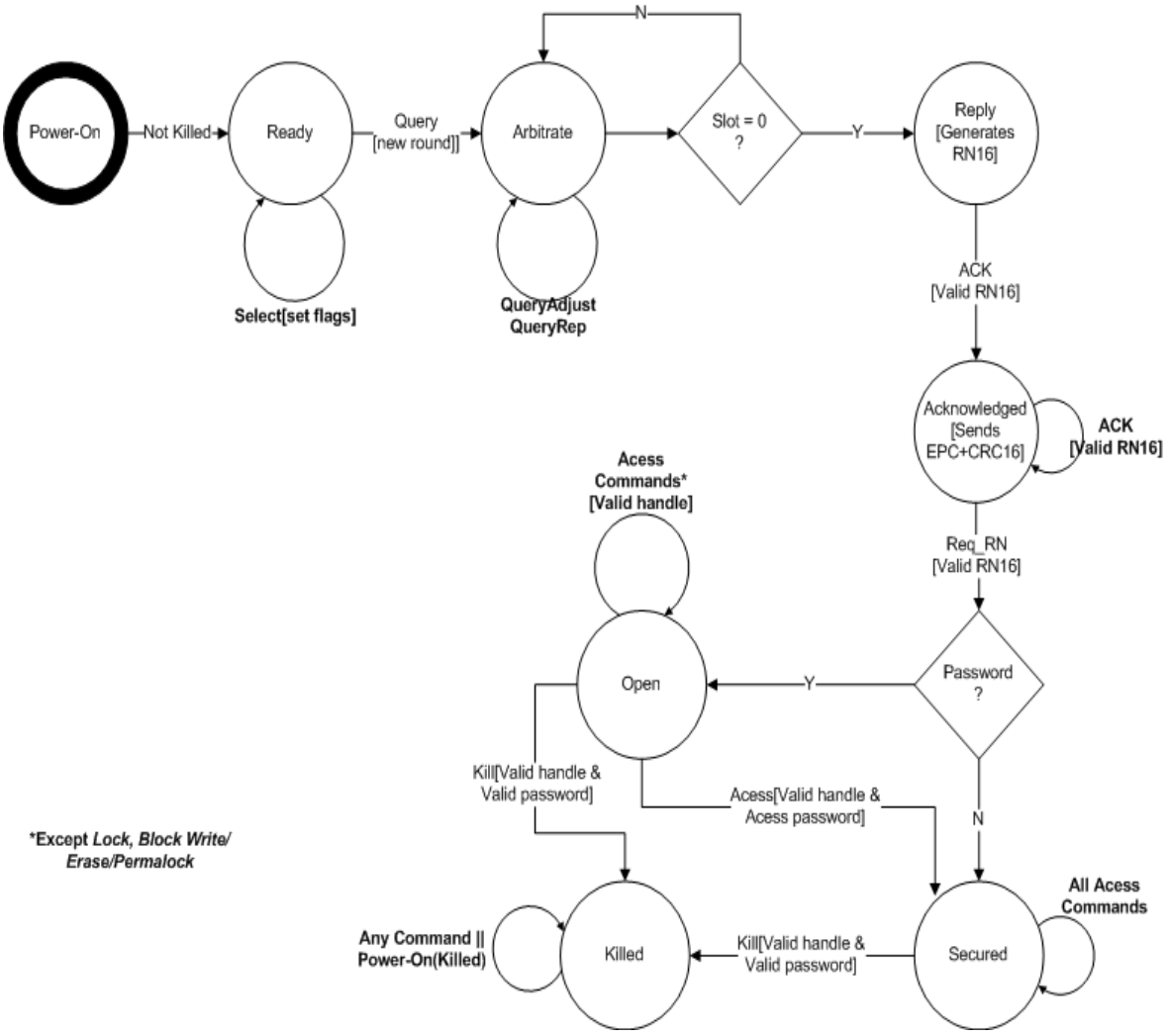


Figura 2.10: Máquina de estados de uma tag EPC Gen2.

Como podemos ver, todas as comunicações entre tag e leitora são acompanhadas de um número de verificação CRC16. Este número, gerado a partir da combinação lógica entre os dados e uma máscara, chamada polinômio, é obrigatório no padrão EPC Gen2 e tem como finalidade adicionar robustez ao protocolo. Desta forma, quando uma leitora recebe, por exemplo, o EPC de uma tag, o CRC16 é calculado novamente e comparado com o fornecido, permitindo-se a detecção de falhas na comunicação, ocasionadas por colisões ou interferências externas.

Como mostrado neste capítulo, um dos principais empecilhos para a implantação em massa do RFID é o custo de implantação da tecnologia. Desta forma, para que os sistemas RFID tenham uma boa performance e um baixo custo, é necessário, dentre outras coisas, o uso de uma arquitetura de microcontrolador otimizada. No próximo capítulo, apresentaremos a arquitetura ARM, uma boa solução para este problema.

Capítulo 3

Arquitetura ARM

3.0.3 Introdução

Um dos microprocessadores mais populares do mundo, o *Acorn RISC Machine* (ARM) é uma arquitetura de 32 bits *Reduced Instruction Set Computing* (RISC) criada pela Acorn Computadores, e hoje está presente nas mais diversas aplicações, desde smartphones e tablets até em TVs digitais e câmeras fotográficas.

Boa parte de seu sucesso se deve ao fato de possuir um modelo de programação bastante simples e versátil, o que causou a rápida adoção do ARM por várias comunidades desenvolvedoras de software. Atualmente vários sistemas operacionais suportam a arquitetura ARM, como o Android, Linux, Windows CE, iOS, entre outros.

O termo ARM refere-se apenas ao *core* do processador, sendo este desenvolvido atualmente pela empresa ARM Holdings. A partir deste *core*, várias outras empresas licenciadas criam sua própria solução de microcontrolador, adicionando periféricos e características específicas para aplicações. Hoje, a Freescale é a principal produtora de processadores ARM voltados para uso automotivo, o qual requer um nível elevado de exigências contra falhas e condições de operação.

A família IMX 53 - mostrada na figura 3.1 - é projetada para experiências de alta performance em aplicações automotivas e de multimídia. Possui suporte a processamento de vídeo 2D e 3D, múltiplas opções de conectividade e um alto nível de integração do sistema. Seu kit de desenvolvimento, o IMX53 QSB é uma solução de baixo custo e versátil. Possui interface de rede Ethernet, saída de vídeo, portas USB, entre outras.

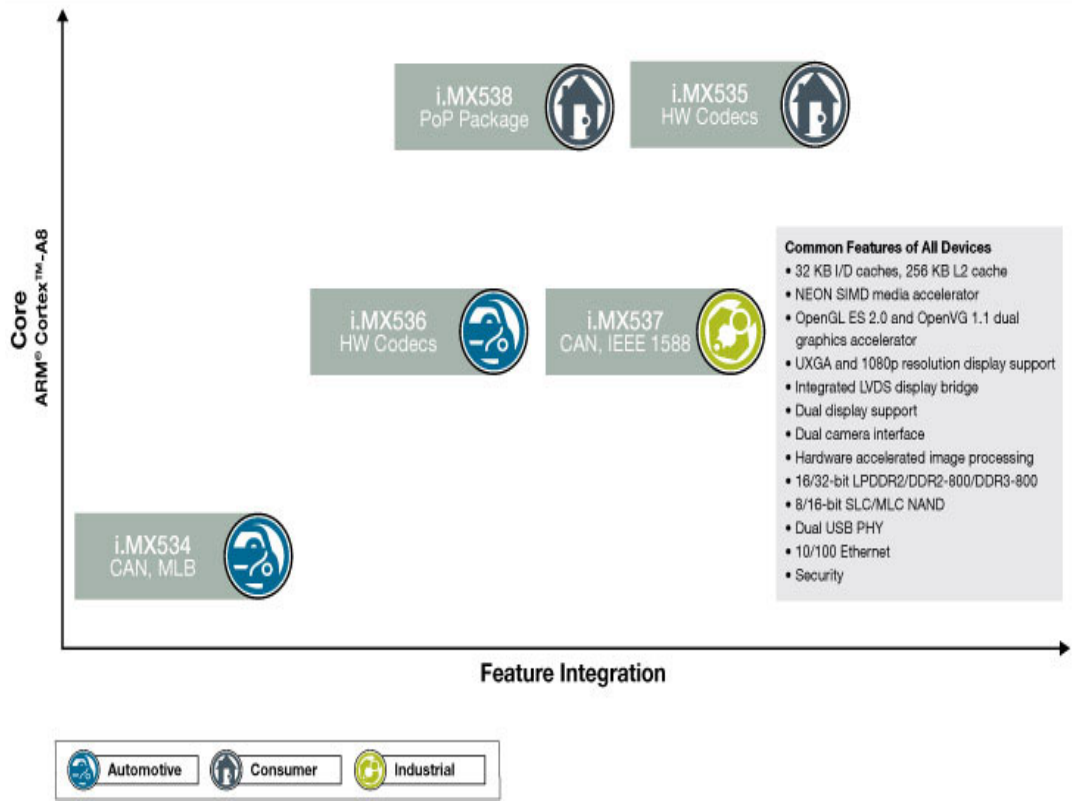


Figura 3.1: Família IMX53.

3.1 Modelo de programação

O modelo de programação que caracteriza um processador ARM é centrado em três principais conceitos: mapeamento de memória e dispositivos, modos de execução e tratamento de exceções.

Para gerenciar o acesso a memória e aos dispositivos de entrada/saída, o processador ARM utiliza um único espaço de endereçamento, com largura de 32bits. Nesta configuração, endereços específicos e contíguos são separados para cada dispositivo, podendo-se assim usufruir de todas as funcionalidades do dispositivo através de acessos a memória.

Geralmente, uma arquitetura baseada em ARM possui três níveis de memória, que são usadas para fins específicos. Da de maior desempenho para a de menor, temos: memória cache, memória RAM e memória de armazenamento secundário. A memória cache é uma unidade de armazenamento de alta velocidade que se encontra dentro do chip.

A memória cache tem como objetivo aumentar o desempenho do processador, disponibilizando dados e instruções que são usados com alta frequência pelo mesmo. Por ser mais cara de se implementar, é de pequena capacidade, necessitando de uma lógica especial para coordenar quais dados estarão presentes nela.

Para gerenciar o armazenamento, cores ARM contam com uma *Memory Management Unit* (MMU), que é responsável por fornecer um endereçamento virtual para as regiões de memória. Esta MMU é capaz de endereçar páginas de até 1Mb e implementa controle de permissão a regiões de memória. Esta última característica é de suma importância para o suporte a diversos modos de operação, tornando possível em hardware a autenticação do usuário para o uso de memória.

Os modos de execução implementados pelo processador são sete. Cada um deles possui seu uso particular e permissões diferentes para acessar os registros do ARM. O principal é o modo de execução Usuário. Nele, uma aplicação pode executar em segurança, sem que haja perigo de invadir regiões de memória que estejam protegidos. Para tanto, não conta com privilégios.

Todos os outros modos são dotados de privilégios para modificar as permissões de acesso a memória e são usados para atender situações especiais do processador, chamadas exceções. Estas podem ser interrupções, erros de escrita/leitura, instrução indefinida ou reset do processador.

3.2 Instruções

Por ser uma arquitetura RISC, o core ARM tem como característica principal prezar por uma otimização do processamento, oferecendo um conjunto de instruções pequeno, porém versátil, que sejam capazes de executar em um período de clock. Estas capacidades são alcançadas a partir da adoção de quatro fundamentos(SLOSS; SYMES, 2004):

Instruções Devem ser em número reduzido e capazes de serem executadas em um ciclo de clock. Para sintetizar instruções complexas, o compilador deve combinar uma série de instruções básicas.

Pipeline O processamento de instruções pode ser estratificado em unidades menores.

Registradores Fornece uma grande quantidade de registradores de uso genérico, que podem guardar tanto endereços quanto dados.

Arquitetura *load-store* Permite o armazenamento de dados em registros do processador, diminuindo o acesso a memória e aumentando a velocidade do ciclo de busca e execução.

Característica marcante da arquitetura, o pipeline é uma técnica que separa a execução de uma instrução em passos mais simples e independentes. No caso de um pipeline de três estágios, estes passos são: busca na memória(*fetch*); decodificação da instrução(*decode*) e execução(*execute*)(SLOSS; SYMES, 2004). Desta forma, após o processador executar um *fetch*, paralelo ao *decode* é feito um acesso a memória para instrução seguinte. Assim as instruções vão entrando em um ritmo encadeado, promovendo um paralelismo na execução.

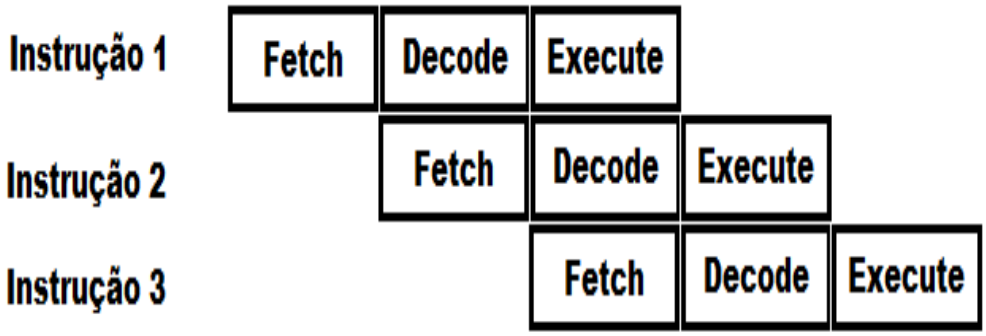


Figura 3.2: Pipeline de três estágios.

O conjunto de instruções do ARM pode ser dividido em 10 classes. São elas: Controle de fluxo(Branch), Processamento de Dados(por exemplo:AND, OR, ADD), Multiplicação, Aritméticas Especiais, Acesso a Registro de Status, Load and Store, Semáforo, Geração de Exceções e Instruções de Coprocessador.

Uma característica pertencente a todas as instruções é a capacidade de serem executadas condicionalmente. Ou seja, a depender do valor do campo *cond* presente em seu opcode, a instrução pode ser ignorada, caso a situação não seja condizente com a condição requerida.

Para que seja possível usufruir do alto desempenho do processador ARM e, ao mesmo tempo, contar com um ambiente de desenvolvimento ágil e completo,

é necessário o uso de um sistema operacional de alta performance, que é o caso do linux. No próximo capítulo, discutiremos sobre este sistema.

Capítulo 4

Linux para Sistemas Embarcados

O Sistema Operacional (SO) Linux é um derivado do Unix, um sistema multitarefa e multiusuário baseado na linguagem C. Em 1991, o então universitário Linus Torvalds lança o Linux em uma lista de discussões na internet e propõe seu SO como um substituto Open Source para o Minix, desenvolvido por Andrew Tannenbaum.

Junto com o kernel, Linus fornecia apenas utilitários básicos para o funcionamento do SO, ficando a cargo do usuário a tarefa de encontrar seus programas, compilá-los e configurá-los. Esta característica acabou fazendo com que o linux se popularizasse no meio acadêmico e entre os usuários mais técnicos.

Contando com o grande suporte que as comunidades na internet fornecem, ao longo do tempo surgiram várias empresas que desenvolviam sua própria versão do linux, com interface gráfica e aplicativos personaizados. A estas versões, damos o nome de distribuição. Entre distribuições de uso geral e específico, hoje contamos com dezenas delas, para as mais diversas plataformas de hardware, desde computadores pessoais a smartphones.

De fato, a principal característica que torna o linux uma solução bastante atraente para os sistemas embarcados é a sua alta capacidade de customização, o que era um problema para os antigos SOs de sistemas embarcados, extremamente dependentes do hardware, tornando o processo de migração para uma nova plataforma demorado e laborioso. A seguir, são listadas mais algumas vantagens:

- O linux é um SO maduro, de alta performance e estável;

- Suporta nativamente várias aplicações e protocolos de rede;
- Permite a escalabilidade, possibilitando seu uso em produtos comerciais;
- Pode ser implantado sem cobrança de royalties por propriedade intelectual;
- Amplo suporte da comunidade desenvolvedora e rápido lançamento de versões para novas arquiteturas e plataformas.

Um dos maiores indicadores do sucesso do linux no mundo dos sistemas embarcados é sem duvida a plataforma Android, da empresa Google. Presente hoje em 42% dos smartphones no mercado mundial[1], o sofisticado e versátil sistema Android é basicamente composto de uma máquina virtual, bibliotecas de dispositivos e um kernel linux modificado.

4.1 Estrutura de um Sistema Embarcado Linux

Nesta sessão, apresentaremos a estrutura geral de qualquer sistema embarcado operando Linux. Basicamente, um sistema linux pode ser dividido em três grandes blocos: Bootloader, Kernel e Aplicação(figura 4.1).

Bootloader

Segundo definido em (HOLLABAUGH, 2002), o bootloader é o firmware responsável por carregar o kernel e sua estrutura de suporte na memória do sistema, e iniciar a execução do mesmo. Para isso, no momento em que o sistema é energizado, o bootloader entra em ação e seleciona qual kernel deve ser inicializado e carrega um pequeno sistema de arquivos na memória RAM. Este contém apenas as informações necessárias para se montar o sistema de arquivos root, iniciando o processo de boot propriamente dito.

Além disso, outra importante função do bootloader é a de configurar a plataforma, programando os controladores de memória, inicializando os registradores do processador, habilitando dispositivos de hardware e caso preciso, implementando os serviços de suporte a rede local, como o *Dynamic Host Configuration Protocol* (DHCP). Todas estas configurações são passadas como parâmetros para o kernel, que as usa durante todo a sua operação.

[1]Fonte: http://www.tecmundo.com.br/12071-android-conquista-quase-50-do-mercado-de-smartphones.htm. Acessado em 11/10/2011

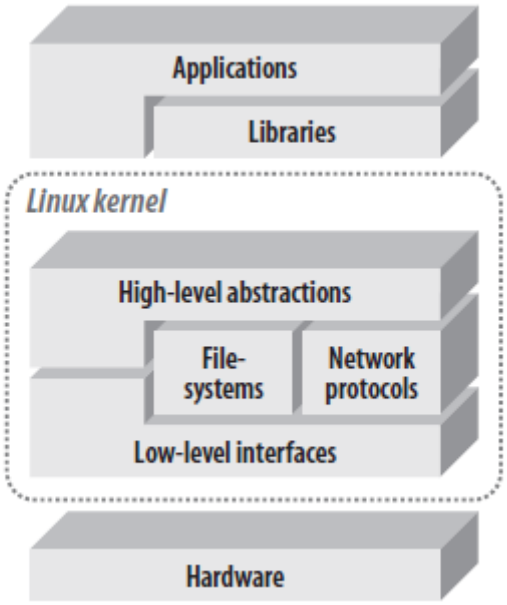


Figura 4.1: Arquitetura de um sistema embarcado linux.

Kernel Linux

O kernel é a parte fundamental de um SO, pois centraliza funções cruciais para a manutenção do sistema, como escalonamento de processos, alocação de memória RAM e acesso e gerenciamento de dispositivos de entrada/saída através de abstrações. Esta última é sem dúvida uma característica importante para a camada de aplicação, pois torna possível o acesso a uma porta serial, por exemplo, sem que seja necessário conhecimento específico sobre o hardware.

Um dos principais blocos integrantes do kernel é o sistema de arquivos. Este nada mais é do que a forma como o sistema operacional organiza hierarquicamente seus diretórios e arquivos a partir de um diretório-pai, chamado diretório *root*, dentro de um dispositivo de armazenamento, como um disco rígido. Nesta árvore de diretórios, estão contidos todos os dados do SO, desde arquivos de configuração e binários de aplicativos a filmes e músicas de um usuário qualquer.

Porém, este sistema de arquivos nem sempre está disponível em um dispositivo local, podendo ser carregado através de um serviço de rede, como o *Trivial File Transfer Protocol* (TFTP). Esta configuração é geralmente empregada durante o desenvolvimento de aplicações para a plataforma, permitindo que os aplicativos compilados sejam prontamente disponibilizados a placa, eliminando a necessidade

de carregar o sistema de arquivos em uma memória flash, por exemplo, a cada vez que é feita uma correção no código.

Os device drivers compõem a parte do kernel responsável por estabelecer comunicação com os dispositivos de entrada e saída presentes na placa. Em um sistema linux, o acesso a estes geralmente se dá através de uma abstração promovida por arquivos. Ou seja, o arquivo localizado em */dev/ttyUSB0* representa uma interface do tipo USB. Caso seja necessário enviar uma mensagem por esta interface, a aplicação pode simplesmente escrever neste arquivo.

Os dois principais tipos de drivers são os de caracter e os de bloco. Segundo (COBERT JONATHAN; RUBINI; KROAH-HARTMAN, 2005), um driver do tipo caracter é aquele onde a comunicação é dada por rajadas de bytes, típico de uma interface serial ou um console. Já os drivers de bloco são caracterizados por uma massiva transferência de dados, dadas geralmente em unidades de 512 bytes por vez. O disco rígido é um exemplo de um dispositivo de bloco.

Drivers podem estar presentes na imagem do kernel como *built-in* ou como módulos. Os drivers *built-in* são aqueles que compõem o kernel desde a sua inicialização, podendo serem usados a qualquer momento. Já um módulo é carregado dinamicamente, ou seja, só é alocado quando há necessidade, podendo ser removido logo após sua utilização.

4.2 Ferramenta de Desenvolvimento

Indispensáveis para a construção de um bom ambiente de desenvolvimento, as ferramentas de software são responsáveis por compilar e gerar uma imagem que contenha todos os componentes necessários para o funcionamento do SO: Bootloader, Kernel, pacotes de aplicativos e sistema de arquivos. A esse conjunto, damos o nome de *Board Support Package* (BSP).

Através da técnica de cross-compiling, que permite a compilação para uma plataforma diferente da usada no desenvolvimento, a imagem do SO é fornecida ao sistema embarcado, seja via TFTP ou por gravação em uma memória não-volátil da placa. Hoje existem dezenas de ferramentas para linux embarcado, com suporte as mais diversas arquiteturas e plataformas, como ARM, PowerPC, ColdFire, entre outros. Seu uso é um fator impactante no *time to market* de produtos comerciais.

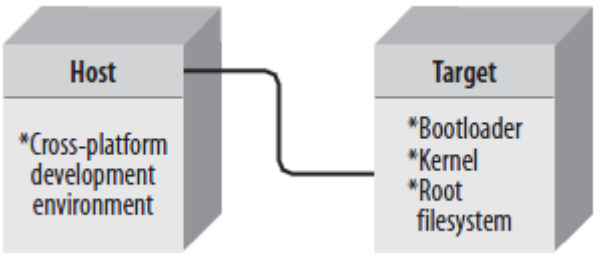


Figura 4.2: Cross-compiling.

4.2.1 LTIB

O projeto *Linux Target Image Builder* (LTIB) é uma iniciativa Open Source que mantém a ferramenta de desenvolvimento de BSPs de mesmo nome. É atualmente um das mais completas ferramentas, contando com mais de 200 pacotes de aplicações e suporte as mais diversas arquiteturas. Seu funcionamento pode ser via interface *ncurses* ou por comandos de linha.

Os pacotes utilizados no LTIB obedecem o formato *RPM Package Manager* (RPM). Assim, para se desenvolver uma aplicação através do LTIB, esta deve estar obedecendo o formato RPM e ser adicionada a árvore de diretórios da ferramenta. Ao mesmo tempo, a adoção deste formato faz com que vários pacotes fornecidos para outras distribuições linux sejam herdados.

Capítulo 5

Sistema RFID Baseado em ARM

Neste capítulo, será apresentada a solução proposta por esta monografia, um sistema RFID baseado em plataforma ARM capaz de lidar com as principais aplicações desta tecnologia, suas características e componentes. Além disso, será descrita a aplicação de validação da arquitetura, um sistema de controle de acesso baseado em RFID.

5.1 Arquitetura da Solução Proposta

Tendo em vista a crescente adoção do RFID, principalmente na cadeia de negócios e logística, cada vez mais por grandes empresas, este trabalho foi proposto. Como vimos no Capítulo 2, o principal impedimento da adoção em massa desta tecnologia é o seu custo de implantação devido aos altos preços de seus componentes.

Pensando nisso, neste projeto propomos uma solução de um sistema RFID utilizando uma plataforma *ARM-based*, com sistema operacional Linux. Estas duas características da solução vão exatamente ao encontro a diminuição de custo dos sistemas RFID, pois temos na arquitetura ARM, um processador extremamente comercial - portanto barato e de fácil obtenção no mercado - de alta performance e baixo consumo de energia.

Unido ao poderoso processador ARM, temos o Linux, um sistema operacional já bastante famoso e cada vez mais crescente no mundo dos sistemas embarcados. Além de fornecer um ambiente estável e comum aos desenvolvedores de software, o Linux tem como principal vantagem a herança de um grande arcabouço de aplicações e bibliotecas, suportadas por vários grupos de desenvolvimento de software livre. Isto

encurta significativamente o tempo de desenvolvimento do produto, possibilitando ainda a rápida adequação a novas aplicações.

A estrutura da solução apresentada nesta monografia tem como cerne a placa de desenvolvimento *Quick Start Board* (QSB) IMX53, da empresa Freescale. Esta é responsável por controlar tanto a interface com o mundo exterior, através de LEDs e da cancela automática, quanto a comunicação com o módulo RFID, que identifica as tags apresentadas. A figura 5.1 apresenta um diagrama de blocos do sistema completo.

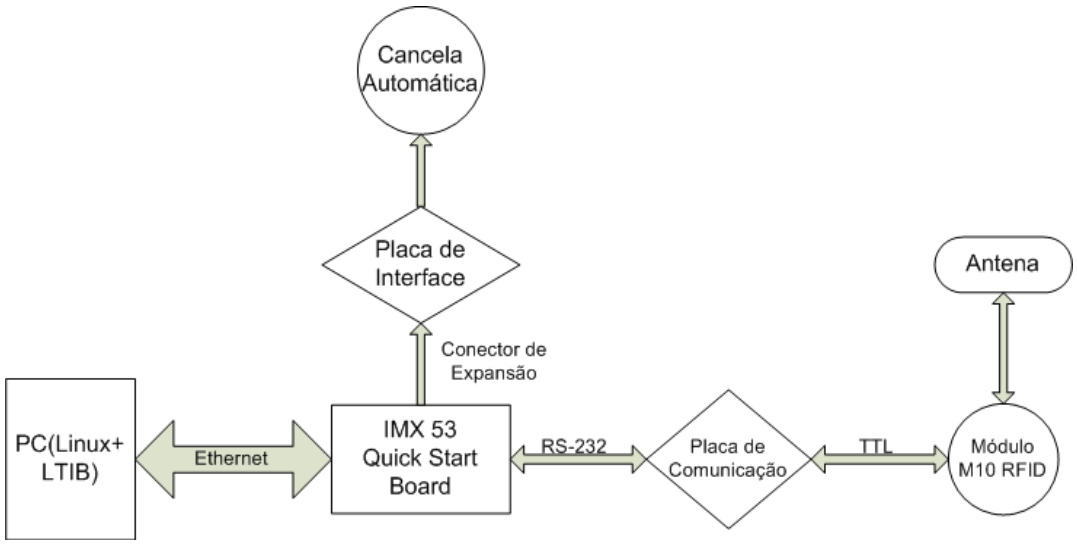


Figura 5.1: Estrutura do projeto.

Desta forma, quando uma tag é apresentada à antena, a leitora RFID envia os dados referentes àquela tag para a placa de desenvolvimento. Esta por sua vez processa estes dados e toma alguma ação através da placa de interface, sinalizando com os LEDs e controlando a cancela automática. A seguir, uma breve explanação acerca de cada componente.

IMX53 QSB

Plataforma alvo do desenvolvimento, a QSB é responsável por gerenciar a solução RFID, processando os dados obtidos da leitora, via comunicação serial, e interagindo com o usuário através da placa de interface. Possui um microprocessador IMX535, baseado em um core ARM e otimizado para aplicações de multimídia. A placa ainda conta com diversas interfaces, como saída e entrada de áudio, processamento de vídeo 2D e 3D, conexão Ethernet, USB e microSD.

Além disso, esta plataforma é suportada pela ferramenta de customização de imagens Linux LTIB, descrita em 4.2. Todas estas características contribuiram para a adoção da QSB na a implementação deste projeto, pois esta tornou possível a estrutura modular da solução e o seu cumprimento em tempo hábil.

Placa de Interface

Placa de Circuito Impresso (PCI) confeccionada para propósito específico da aplicação de controle de acesso. Ela é responsável por prover a interface com o mundo real, através de LEDs indicadores e um motor de passo, que simula a implementação de uma cancela de estacionamento. Foi desenvolvida a partir das informações encontradas em (FREESCALE, 2011). Neste documento, são fornecidas as pinagens de todos os componentes da QSB, inclusive do conector de expansão, de onde são obtidos os sinais de controle para a placa de interface.

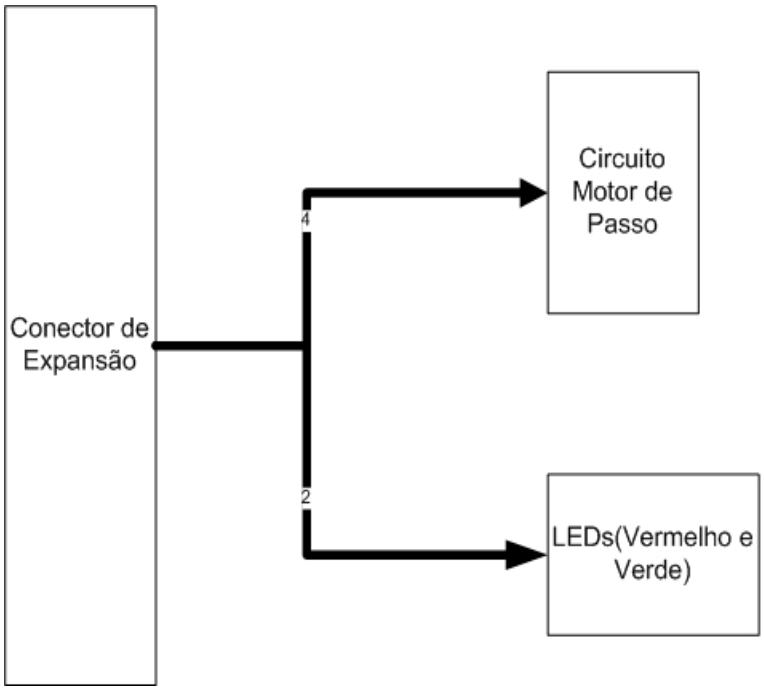


Figura 5.2: Diagrama de blocos da placa de interface.

Para esta placa, foram necessários seis sinais digitais, usados como *General Purpose Input/Output* (GPIO). Destes, quatro são para o controle do motor de passo e os restantes para o acionamento dos dois LEDs. Para o circuito de controle do motor de passo, foram usados transistores bipolares e diodos de roda livre. Esta configuração é necessária para evitar que picos de corrente possam danificar os pinos

do microprocessador. O esquemático da placa encontra-se disponível em anexo. A seguir, um diagrama de blocos da placa de interface.

Placa de Comunicação

Esta placa acomoda a leitora RFID e fornece os circuitos necessários para a integração com a QSB. No desenvolvimento do hardware de comunicação com o módulo RFID M10 Skyetec, foram analisados os requisitos e especificações encontrados em (SKYETEK, 2011), listados na tabela 5.1. Após a análise dos requisitos do módulo, foi constatada a necessidade de uma fonte linear exclusiva para o mesmo, posto que a fonte da QSB não seria suficiente. O circuito de fonte tem como elemento principal o CI LD1085 e é capaz de fornecer até 2.5 Ampère, suportando um valor máximo de tensão de entrada de 30V.

Para a comunicação entre o módulo e a QSB, foi utilizada a interface UART, com *baud rate* configurado para 38400bps. Como a interface fornecida pela QSB opera em nível RS-232, um circuito de conversão de nível - baseado no CI MAX3232 - é usado nesta linha de comunicação. Na figura 5.3, temos um diagrama de blocos da placa de comunicação.

Requisito	Valor/Descrição
Consumo de Corrente	462mA - 1.5A
Tensão de Alimentação	3.3V
Frequência de operação	860MHz - 960MHz
Potência da Antena	10dBm - 30dBm
Interfaces Digitais	USB 2.0, UART TTL, SPI/I2C
Protocolos Suportados	EPC Gen2, IPX EM

Tabela 5.1: Requisitos do módulo M10.

Computador Pessoal(PC)

É a ferramenta utilizada para o desenvolvimento tanto da imagem Linux quanto para a prototipagem das PCIs utilizadas neste projeto. Para este trabalho, foi utilizado um PC com SO Ubuntu 10.04, versão do kernel 2.6.35, equipado com

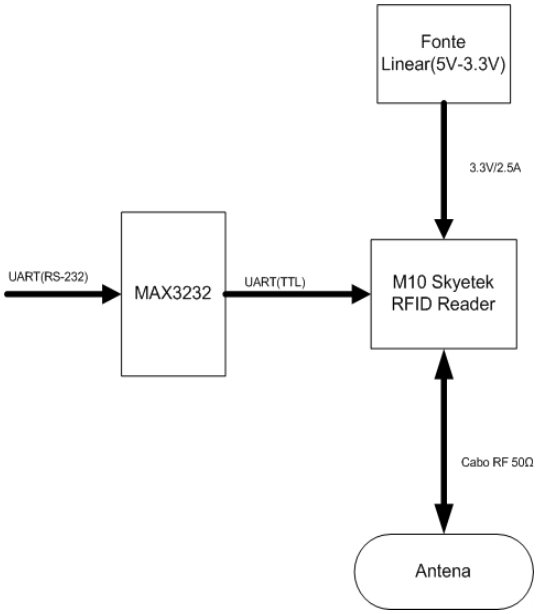


Figura 5.3: Diagrama de blocos da placa de comunicação.

os programas e bibliotecas necessários para a customização da imagem Linux. As principais ferramentas de software são: ambiente de desenvolvimento LTIB, software de comunicação serial por terminal, servidor TFTP e aplicação telnet.

Utilizando a técnica de cross-compiling, descrita em 4.2, o kernel customizado e o sistema de arquivos são fornecidos para a placa de desenvolvimento através do servidor TFTP instalado no computador. Este comportamento duplo do PC de ambiente de desenvolvimento e unidade de armazenamento é crucial para a redução do tempo entre a compilação da imagem e os testes na QSB. Na versão final do sistema computacional proposto, a imagem Linux definitiva é carregada no cartão microSD da placa.

5.2 Aplicação de Teste - Controle de Acesso

Com o intuito de validar as funcionalidades da solução implementada neste projeto, foi criado um software de testes capaz de utilizar o hardware desenvolvido para manter uma aplicação de controle de acesso. Nesta seção, apresentamos o processo de criação desta aplicação, descrevendo os seus passos e características.

5.2.1 Ferramentas Utilizadas

Para a etapa de modificação do kernel e desenvolvimento do software, as seguintes ferramentas foram necessárias:

i. Computador com sistema operacional Linux, distribuição Ubuntu 10.04;

ii. Ferramenta de desenvolvimento de BSPs LTIB;

iii. Kit de desenvolvimento IMX53 QSB Freescale;

iv. Rede Ethernet local;

v. Placas de interface e de comunicação com a leitora.

5.2.2 Inicializando o LTIB

Como o LTIB é uma ferramenta multiplataforma, antes de começar o desenvolvimento da imagem linux, foi necessário a configuração desta ferramenta para se adequar a plataforma deste trabalho, a IMX53 QSB. Após carregar as configurações da placa, deve-se especificar a versão do kernel a ser utilizada. Neste caso, a versão 2.6.35 está sendo usada.

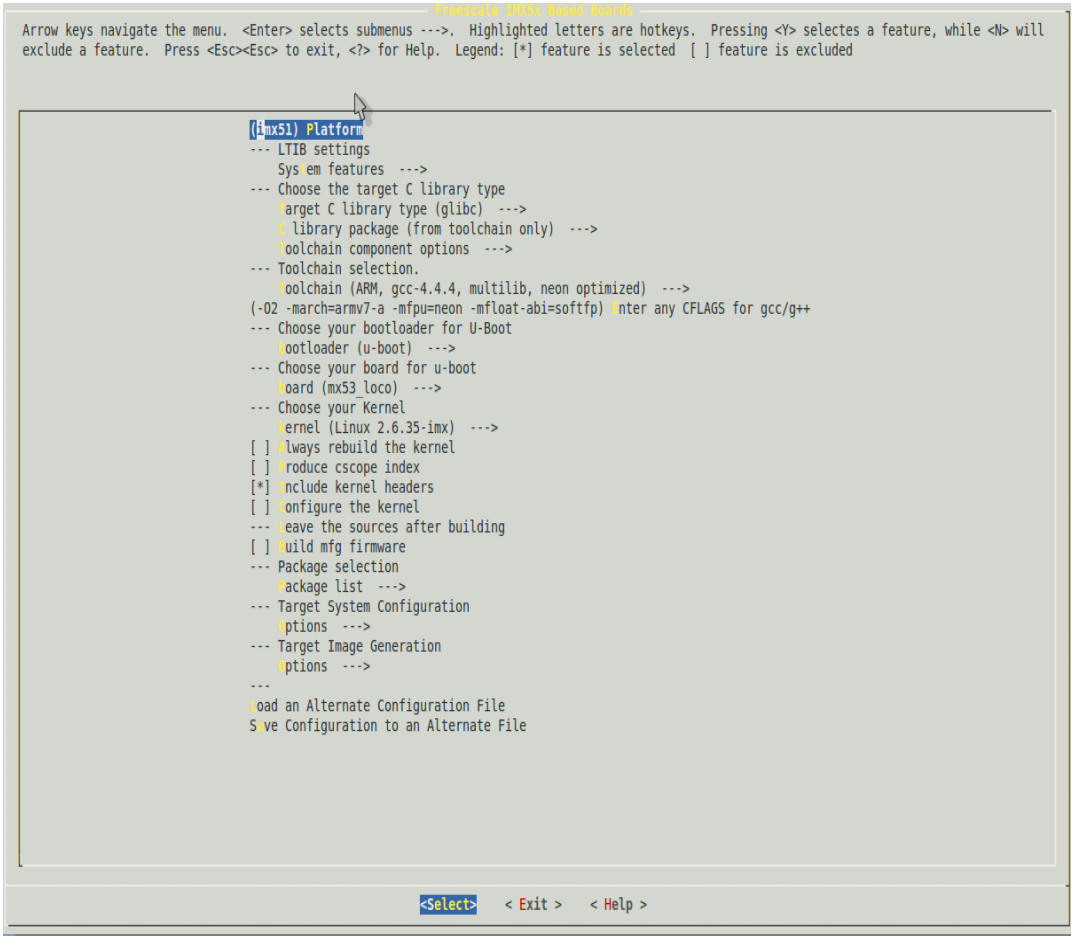


Figura 5.4: Tela inicial do LTIB.

Na figura 5.4, podemos observar a interface do LTIB. Neste menu, temos várias configurações a fazer, como escolha da plataforma alvo(neste caso, IMX51), bootloader utilizado e seleção de pacotes de aplicações.

Como descrito em 4.1, o bootloader presente na QSB está configurado para buscar o kernel e o sistema de arquivos em um servidor tftp. Este servidor está rodando em background no computador de desenvolvimento e aponta para a árvore de diretórios do LTIB.

5.2.3 Driver da Placa de Interface

Após a configuração do ambiente, iniciou-se o desenvolvimento do driver responsável por manipular a placa de interface. Este driver possui dois estados: cancela abrindo(A1) e cancela fechando(A2). Durante o primeiro estado, o motor

de passo é acionado e desloca-se 90°no sentido anti-horário, provocando a abertura da cancela. O LED verde é acionado durante este estado.

No segundo estado, o motor desloca-se 90°no sentido horário e o LED vermelho é acionado, fechando a cancela. A seguir, um diagrama de fluxo do funcionamento deste driver.

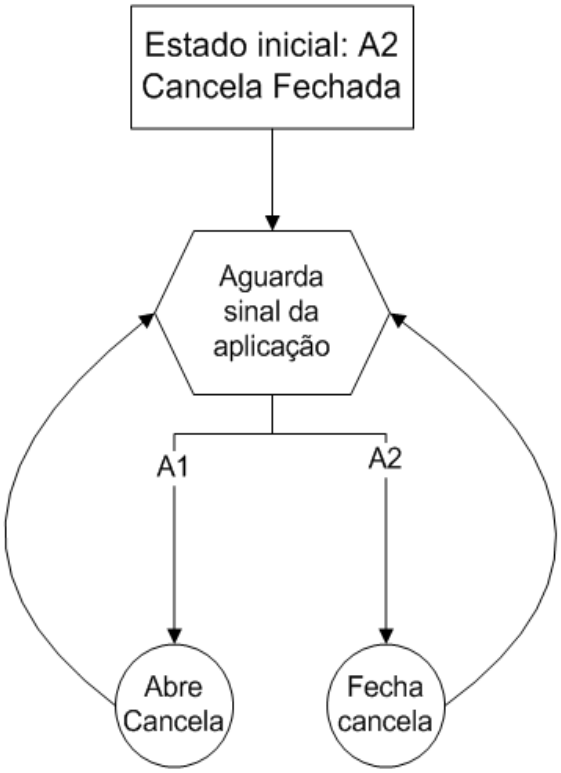


Figura 5.5: Diagrama de fluxo da placa de interface.

5.2.4 Aplicação de Controle de Acesso

Para o desenvolvimento da aplicação de controle de acesso, um pacote foi criado e adicionado à árvore de diretórios do LTIB. Este pacote, batizado Sentinel, tem como função básica administrar a comunicação com a leitora RFID, processar as informações obtidas, relativas as tags apresentadas e a partir delas enviar um sinal ao driver da placa de interface, liberando ou não o acesso.

Além disso, o módulo Sentinel administra um arquivo que possui os identificadores válidos, ou seja, aqueles que permitirão acesso quando apresentados a

leitora. Para isso, o programa desenvolvido trabalha em dois modos: Administração e Controle de Acesso.

No modo de administração, duas funções foram implementadas: adicionar tag e remover tag. Para adicionar uma tag ao banco de dados, basta entrar neste modo e apresentá-la a leitora. Quando a tag é detectada, uma confirmação é requerida e então o identificador é adicionado ao banco de dados. Quando o usuário deseja retirar uma tag, este deve digitar a opção referente ao identificador listado na tela.

Com o banco de dados construído, o programa pode funcionar em modo controle de acesso. Neste modo, o sistema aguarda a apresentação de uma tag e verifica se esta pertence ou não ao conjunto de tags permitidas. O funcionamento da aplicação de controle de acesso obedece ao fluxograma mostrado na figura 5.6.

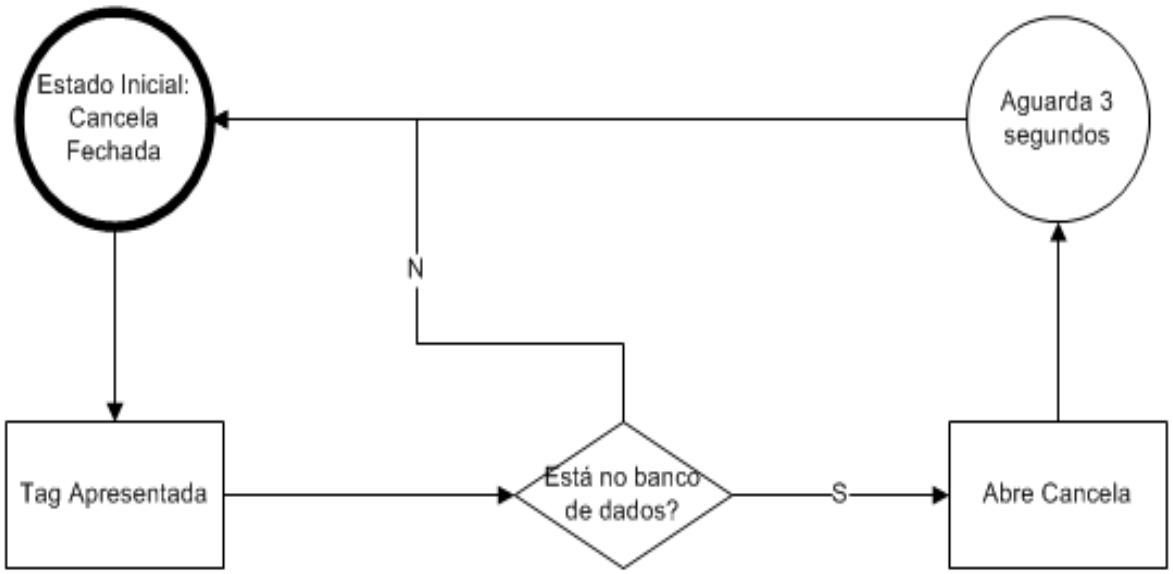


Figura 5.6: Loop de controle de acesso do programa Sentinel.

5.3 Prototipagem

Nesta fase do projeto, as PCIs necessárias tanto para a integração entre a leitora e a QSB quanto para a interface com o usuário foram projetadas, confeccionadas e montadas. Nesta seção, apresentaremos todas as etapas deste processo.

5.3.1 Ferramentas Utilizadas

Para as etapas de desenvolvimento do layout e prototipagem, as ferramentas utilizadas são listadas a seguir:

i. Computador com suíte de design de circuitos elétricos Cadence Allegro PCB Design;

ii. Linha de Prototipagem de PCIs LPKF;

iii. Estação de solda e ferramentas de montagem de componentes eletrônicos;

iv. Osciloscópio digital e multímetro;

5.3.2 Layout

Tendo em mãos os dados do esquema elétrico, os layouts das duas placas foram desenvolvidos no programa Allegro PCB Editor. Terminado o design do circuito, foi feita uma última verificação, procurando por falhas de *Design Rule Check* (DRC). Tais falhas podem ser ocasionadas por um espaçamento entre duas trilhas de sinais que estejam muito próximas, por componentes invadindo a área de outro, ou por dimensões fora de escala. Ambas as placas possuem duas camadas, sendo uma delas um plano de GND. Esta é uma prática bastante comum em layouts de mais de uma camada, pois possibilita a presença de um certo sinal - neste caso o GND - em toda a extensão da placa, diminuindo o número de trilhas referentes a este sinal.

Feita a verificação do layout, são gerados os arquivos Gerber a partir do layout. Estes arquivos contém todas as dimensões e formas que compõem o layout das placas, inclusive os furos e vias. Todas estes arquivos já estão formatados de acordo com o padrão das ferramentas que serão utilizadas no próximo passo da prototipagem, a confecção das PCIs.

5.3.3 Confecção e Montagem

Para que seja possível utilizar a linha de prototipagem LPKF, os arquivos Gerber, gerados na etapa anterior devem ser exportados para um formato que o software de controle do maquinário compreenda. Neste caso, devemos gerar um arquivo JOB, que será utilizado pela máquina de prototipagem Protomat M60, responsável por

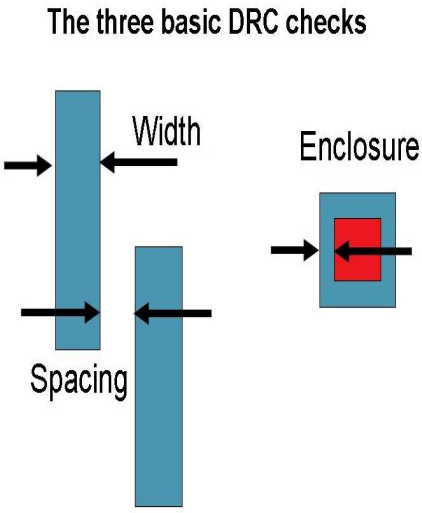


Figura 5.7: Principais pontos de verificação do DRC.

fazer os furos e "desenhar" as trilhas através da retirada do cobre da placa base. Na figura 5.8, pode-se observar o fluxograma do processo.

Basicamente, os dados de um layout de duas camadas necessários para a prototipagem são separados em quatro classes: Board, Top, Bottom e Drill. A classe Board contém os limites do layout desenvolvido, enquanto Top e Bottom possuem todas as vias, trilhas e pads que fazem parte da placa. Drill contém os furos que serão feitos na placa.

Após a importação dos dados referentes ao layout, a placa base foi posicionada no equipamento e a classe Drill é executada. Após ser furada, a placa base passou por um processo de metalização na máquina Contac III. Nesta fase, é estabelecida uma continuidade elétrica entre as duas camadas da placa, através da deposição por corrente reversa de cobre nos furos da mesma. Terminada a metalização, a placa é posicionada novamente na Protomat M60 e as classes Top e Bottom são executadas, terminando o processo de confecção da PCI.

Na fase de montagem, os componentes foram soldados na placa e a continuidade das conexões foi testada, sendo feito os retrabalhos e testes necessários para o devido funcionamento dos circuitos. A lista dos componentes utilizados está disponível em anexo.

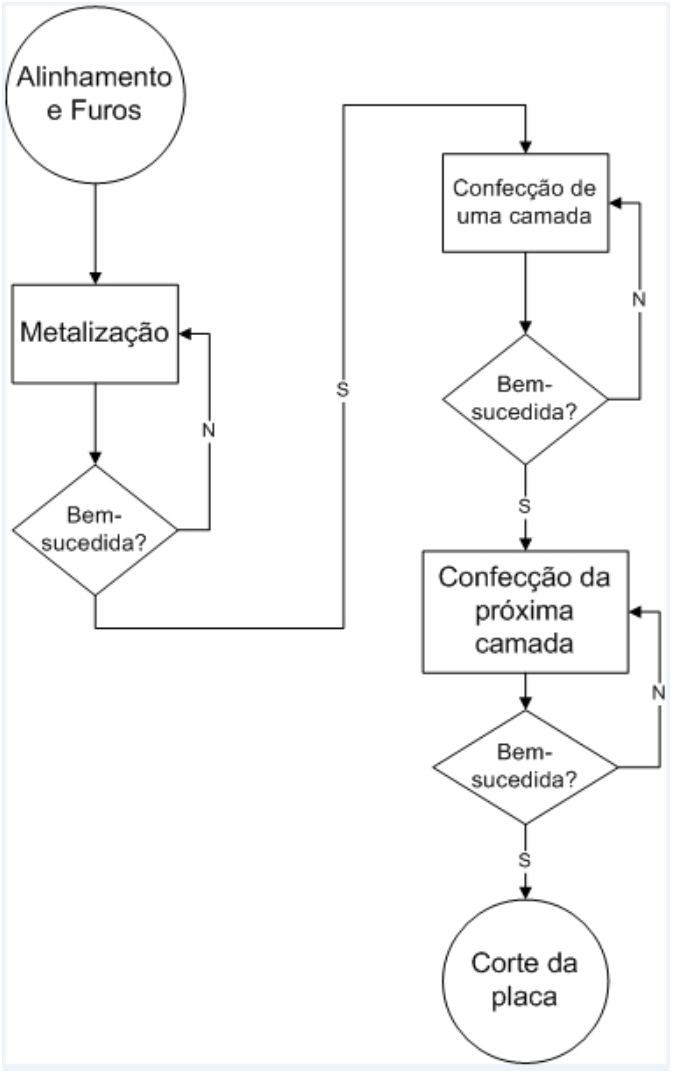


Figura 5.8: Fluxograma do processo de prototipagem.

Capítulo 6

Resultados

6.1 Introdução

Neste capítulo serão apresentados os resultados obtidos nas diversas fases da implementação deste projeto. Também analisaremos os dados obtidos a partir de testes funcionais realizados com a solução de controle de acesso, que medem confiabilidade da aplicação e estabilidade do sistema.

6.2 Prototipagem e Customização

No processo de prototipagem das PCIs utilizadas neste projeto, os resultados obtidos seguindo a metodologia descrita em 5.8 foram positivos. Não houveram falhas que resultassem em inoperabilidade dos componentes do sistema, quer seja por erros de projeto ou imperfeições na montagem.

O circuito de fonte, que representava uma das maiores preocupações do sistema, mostrou-se eficaz para a aplicação, fornecendo a corrente necessária para o correto funcionamento do módulo RFID. A única modificação necessária na placa de interface foi a substituição dos transistores BC547C. Estes mostraram-se insuficientes para fornecer a correte necessária para o acionamento das bobinas do motor.

No desenvolvimento do módulo Sentinel e do driver de controle da placa de interface, após alguns ajustes necessários para a utilização da interface serial da QSB, por padrão definida como serial de *debug*, os resultados também se mostraram satisfatórios, pois todas as funcionalidades previstas para o sistema foram

implementadas.

6.3 Controle de Acesso

Com o intuito de testar as funcionalidades do controle de acesso do programa Sentinel, foi organizado um cenário composto de 20 tags, do tipo smartlabel passiva UHF. Usando o modo de administração do Sentinel, três destas tags foram adicionadas ao banco de dados. Após isso, a aplicação foi iniciada em modo operacional e as tags foram apresentadas uma a uma a leitora.

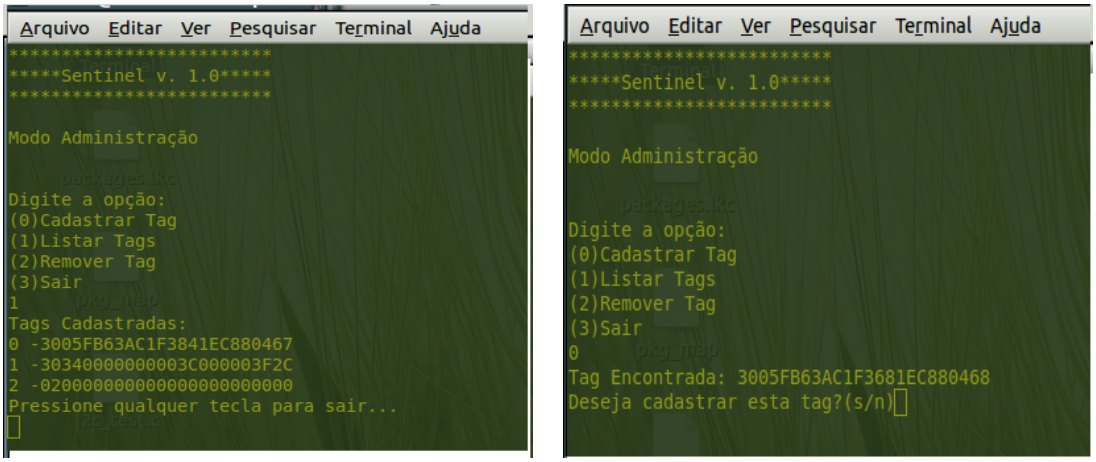


Figura 6.1: Sentinel em modo de gerenciamento.

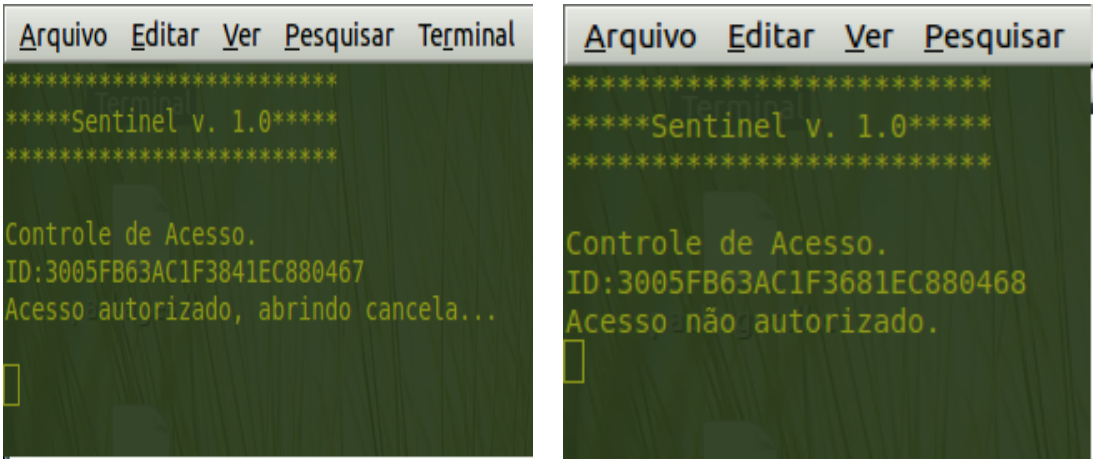


Figura 6.2: Sentinel em modo operacional.

Nas figuras 6.1 e 6.2 podemos observar o menu presente no modo de administração e a rotina de controle de acesso operando. Após 10 repetições do

ciclo completo de teste do grupo de tags, foi obtida a taxa de 100% de acerto, sem a geração de falso positivo ou negativo quando as tags eram apresentadas. Assim, pode-se considerar que a aplicação funcionou a contento.

Capítulo 7

Conclusão e Trabalhos Futuros

Este trabalho propôs a criação de um sistema capaz de lidar com as principais aplicações da tecnologia RFID. Neste aspecto, a implementação foi um sucesso, demonstrando bons resultados e agregando diversos conhecimentos em áreas distintas.

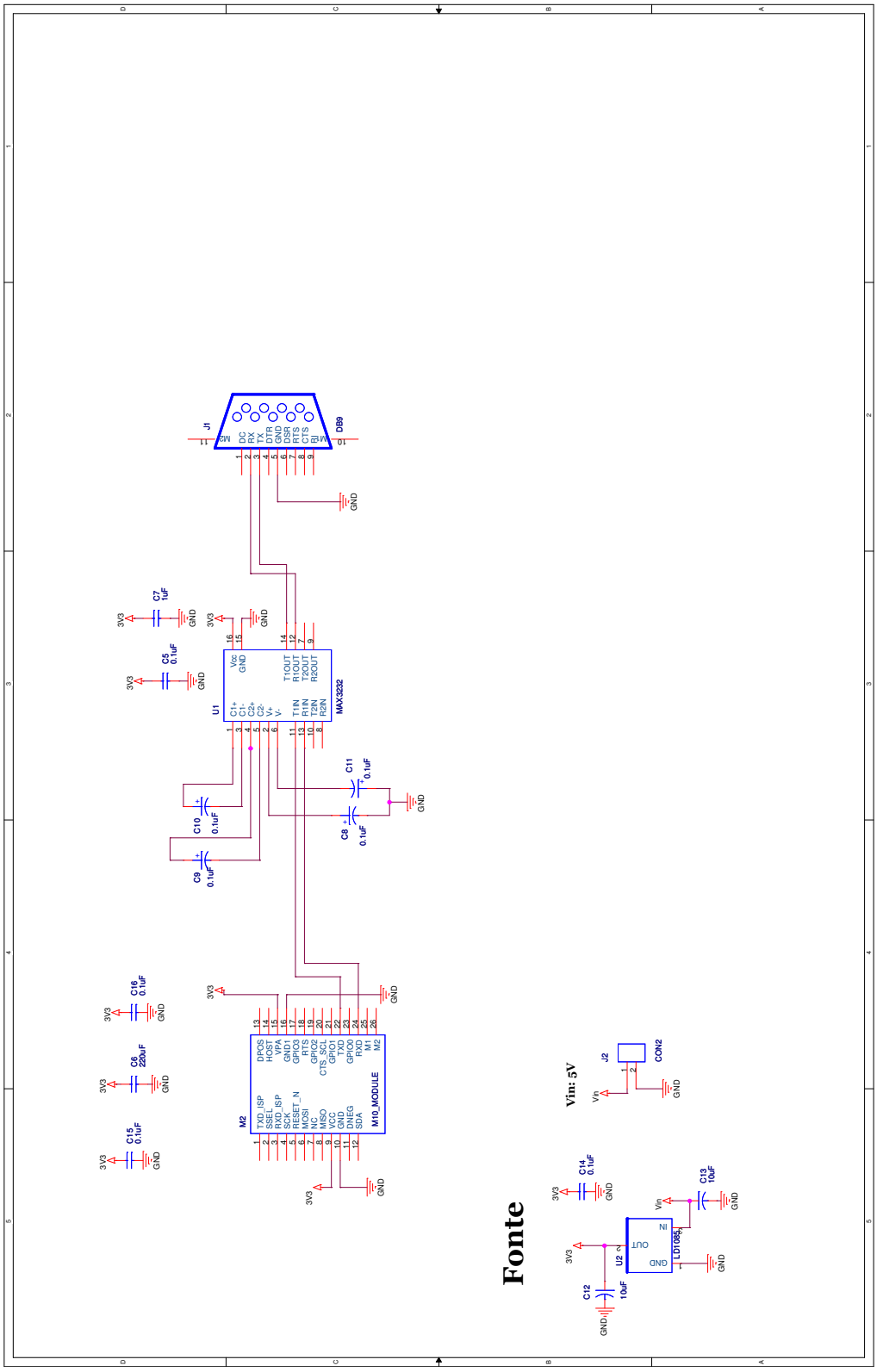
Implementando a aplicação de controle de acesso, a solução mostrou-se eficaz e robusta. Porém, o sistema tem capacidade de lidar como aplicações mais sofisticadas, com rastreamento ou controle de estoque. Aqui não abordamos tais aplicações devido a fatores como formato da antena, alcance de leitura, infra-estrutura e tempo de projeto.

Com este teste de conceito bem-sucedido, esperam-se como próximos passo o projeto de uma placa dedicada, testes de outros layouts de antenas e leitoras RFID. Tudo isto contribuiria para a criação de uma solução bem mais completa e competitiva para o mercado.

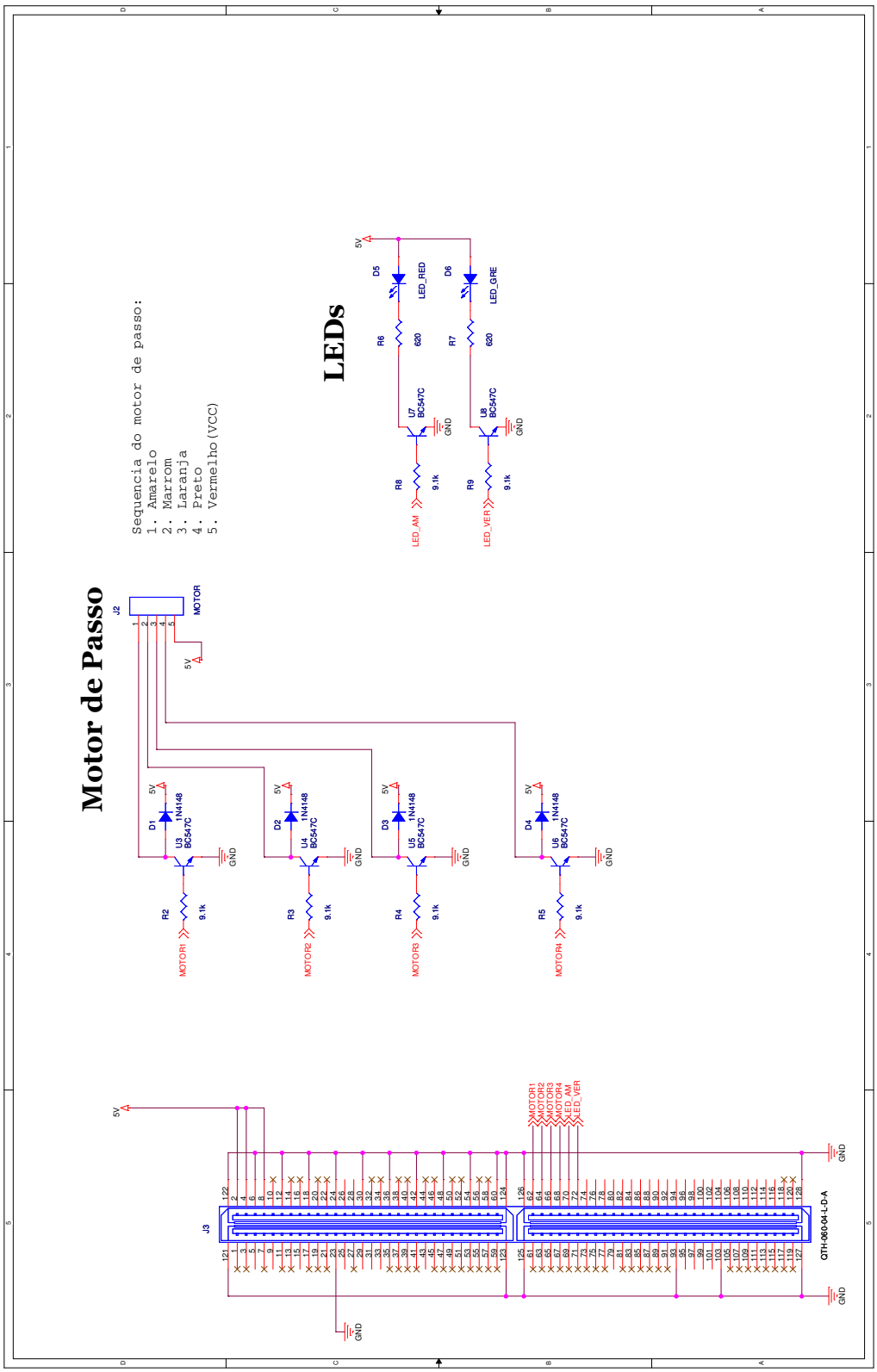
Apêndice A

Esquemáticos Desenvolvidos

A.1 Placa de Comunicação



A.2 Placa de Interface



Referências Bibliográficas

COBERT JONATHAN; RUBINI, A.; KROAH-HARTMAN, G. *Linux Device Drivers (3th ed.)*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2005. ISBN 0-596-00590-3.

EPCGLOBAL. *Class 1 Generation2 UHF Air Interface Protocol Standard*. November 2011. Http://www.epcglobalus.org.

FREESCALE. *HArdware Reference Manual for i.MX53 Quick Start*. November 2011. Http://cache.freescale.com/files/32bit/doc/userguide/IMX53QSBRM.pdf.

GLOVER, B.; BHATT, H. *RFID Essential (1th ed.)*. Sebastapol, CA, USA: O'Reilly Media, Inc., 2006. ISBN 0-596-00944-5.

HESSEL, F. e. a. *Implementando RFID na cadeia de Negócios: Tecnologia a serviço da Excelência. (1th ed.)*. Porto Alegre, RS, BRA: EDIPUCRS, 2009. ISBN 978-85-7430-897-5.

HOLLABAUGH, C. *Embedded Linux: Hardware, Software and Interfacing*. Indianapolis, IN, USA: Addison-Wesley, Pearson Education, 2002. ISBN 0-672-32226-9.

SKYETEK. *SkyeModule M10 Reference Guide*. acessado em 09/07/2011 2011. Disponível em: http://virtual2002.tau.ac.il/taglink_guide/M5eFamilyDevGuide_Nov08.pdf.

SLOSS, A. N.; SYMES, D. *ARM System Developer's Guide*. San Francisco, CA, USA: Elsevier, Inc., 2004. ISBN 1-55860-874-5.