Centro Universitário de Brasília - UniCEUB

Faculdade de Ciências Exatas e Tecnologia - FAET

Curso de Engenharia da Computação

Projeto Final

Marcos Felipe Siqueira Engel

Sistema de Monitoração e Controle de Acesso utilizando Microcontrolador PIC16F877

Brasília
2007

Marcos Felipe Siqueira Engel

Sistema de Monitoração e Controle de Acesso utilizando Microcontrolador PIC16F877

Trabalho apresentado ao Centro Universitário de Brasília (UNICEUB) como pré-requisito para a obtenção de Certificado de Conclusão do Curso de Engenharia da Computação.

Orientador: Prof. José Julimá Bezerra Junior

Brasília
2007

AGRADECIMENTOS

Aos meus pais pela dedicação e pelo incentivo constante. À Ana Carolina pela paciência e compreensão nesta etapa final. Aos meus colegas de classe pelos anos de estudos, em especial, Giuliano Bocucci, Leonardo Lins, Leonardo Morale e Radjalma Costa Júnior. Aos professores que possibilitaram a minha capacitação intelectual e profissional.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Esta monografia refere-se a um projeto de conclusão do curso de Engenharia da Computação, onde foi implementada uma solução de controle de acesso através de teclado, utilizando um microcontrolador como interface entre o microcomputador e o circuito elétrico. O projeto tem como objetivo supervisionar e controlar o acesso de funcionários e colaboradores às áreas de acessos restritos, proporcionando uma redução nas perdas com furtos ou fraudes nos sites monitorados e controlados.
A utilização do teclado, para assegurar a segurança do acesso a locais restritos, se deve ao fato de que esta é a solução mais utilizada e economicamente acessível, pois tecnologias como geometria das mãos, impressão digital, íris ou face, apresentam custo mais elevado de implementação do sistema.

Palavras-chave: Controle de Acesso, Microcontrolador, Linguagem de programação Assembly, Banco de Dados Microsoft Access.

ABSTRACT

This document presents a Computer Engineering conclusion Project, showing an access control system using a keyboard and a microcontroller device as an interface between the microcomputer and the electric circuit.

The Project objectives are to supervise and to control the access of employes and other persons into restrict areas, reducing losses with frauds and burglary in controlled sites.

The use of the keyboard to assure secure access into restrict places is the most adopted solution because its accessible price, in spite of more sophisticated tecnologies, like hand geometry, digital impression, iris or face, more expensive ones.

Keywords: access control, microcontroller device, Assembly programming language, Microsoft Access Data Base.

Capítulo 1 - Introdução

1.1. MOTIVAÇÃO

Apesar do estágio atual de desenvolvimento da segurança dos sistemas empresariais, ainda se tem notícia de roubo de informações e sabotagem em sistemas. Por isso, não se pode descuidar da questão do acesso físico aos servidores (CPDs), tendo em vista que este possui acesso restrito.

Um dos itens de política de segurança é o acesso físico a locais restritos e será este item que o projeto abordará.

Para que o projeto garanta de forma satisfatória a segurança do ambiente e seja economicamente viável, desenvolveu-se um protótipo de acesso via teclado utilizando interfaces de microcontrolador e microcomputador com base de banco de dados atualizável, de fácil manuseio.

Tal protótipo possibilitará que empresas garantam a segurança de determinados locais, controlando o acesso de funcionários via cadastro das matrículas e senhas no banco de dados, fazendo com que informações de grande relevância sejam devidamente resguardadas.

O sistema proposto aprimorará o controle de acesso por meio de uma solução de baixo custo, alta eficiência e fácil manuseio.

1.2. OBJETIVOS

O projeto visa aprimorar o controle de acesso a locais restritos, solicitando a digitação de uma senha no teclado para liberação da entrada no espaço físico e possibilitando a geração de relatórios com a discriminação de cada acesso ao local.

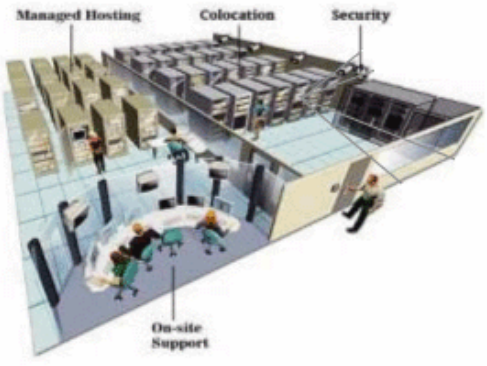


Figura 1.1 - Demonstrativa

Assim, o sistema desenvolvido possibilitará a individualização das pessoas que acessaram o local, por meio da conexão estabelecida entre o teclado, o microcontrolador, o microcomputador e o banco de dados (Microsoft Access).

1.3. METODOLOGIA DE PESQUISA

Para a implementação deste projeto, foram realizadas pesquisas bibliográficas em livros e sites da internet, bem como foram realizados testes para a análise dos componentes a serem utilizados no sistema, para assegurar a sua viabilidade e aplicabilidade.

1.4. ESTRUTURA DA MONOGRAFIA

Além desta introdução, o trabalho contém mais três capítulos.

No capítulo 2 é dada uma visão geral do projeto para identificação da tecnologia na qual se baseia o trabalho e dos componentes que interagirão no sistema.

No capítulo 3 são apresentados o hardwares e as interfaces envolvidos no sistema, com o detalhamento da função dos mesmos, bem como são descritos os softwares desenvolvidos no projeto e a sua interação com os demais componentes do protótipo que possibilitam o funcionamento do sistema e apresentados os resultados obtidos.

No capítulo 4 são apresentadas a conclusão do projeto e as propostas futuras para aprimoramento do sistema.

Capítulo 2 – Referencial Teórico

Os microcontroladores possuem diversas funções e aplicabilidades. Nesse sentido, os microcontroladores são utilizados nos mais variados sistemas devido à melhoria na produção e à diminuição de custos por eles proporcionadas, beneficiando o desenvolvimento do sistema como um todo.

Atualmente, os microcontroladores são aplicados em sistemas como controle de semáforos, balanças eletrônicas, controle de acesso, telefones públicos, taxímetros e eletrodomésticos em geral. [CHAVES; MENDES, 2003]

“O projetista de circuitos eletrônicos microcontrolados tem desempenhado um papel de destaque neste contexto, pois viabiliza o desenvolvimento de soluções personalizadas e de baixo custo. Uma exigência cada vez mais comum entre as empresas modernas. Este é um dos motivos que explica o extraordinário crescimento do uso de microcontroladores no projeto de circuitos eletrônicos, e um número cada vez maior de projetista. Costuma-se dizer que o limite de criação de soluções envolvendo microcontroladores está associado à criatividade do projetista. Quem é projetista sabe quanto dinheiro pode estar por trás de uma boa idéia.” [ZANCO, 2006]

Assim, a solução desenvolvida para o presente projeto, que tratará especificamente da utilização de microcontroladores em sistemas de controle de acesso, é mostrada na Figura 2.1:

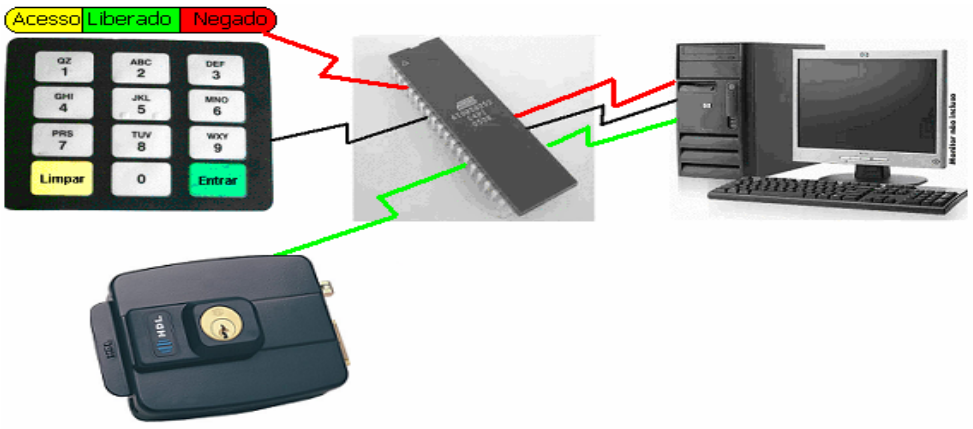


Figura 2.1 - Diagrama geral do projeto

O teclado fará a interface externa entre o usuário e o servidor, atuando como dispositivo de entrada de dados, onde o usuário digitará sua matricula e, posteriormente, sua senha. Esses dados serão recebidos pelo microcontrolador que enviará os dados ao servidor através da porta de comunicação serial que liberará a catraca eletrônica de acordo com o cadastro no banco de dados.

Além disso, o sistema criado possibilitará a fácil inclusão e exclusão de usuários que podem ter acesso ao local, com a utilização de um programa que armazenará no servidor de banco de dados a matrícula e o nome de cada usuário que acessou o local, registrando a hora e a data do acesso, tornando possível a geração de relatórios para controle do administrador do setor.

Assim, para que o projeto garanta de forma satisfatória a segurança no ambiente e seja economicamente viável, foi desenvolvido um protótipo de acesso utilizando microcontrolador PIC16F877 e microcomputador com base de dados atualizável, inseridos através do software de gerenciamento criado.

2.1. CARACTERISTICAS E ESPECIFIDADES DO PROTÓTIPO

O protótipo desenvolvido envolve a interação dos seguintes componentes:

- Teclado: é utilizado para entrada de dados como matrícula e senha;
- Kit LABPIC
  - Microcontrolador PIC16F877: como interface de entrada e saída (I/O) que intermedia o protótipo (teclado/kit/cpu);
  - Display LCD: apresenta a matrícula digitada e resposta do computador do acesso liberado ou negado;
  - Comunicação serial RS232;
  - Relé integrado ao kit para acionamento de fechadura;
  - Fechadura Elétrica: utilizada para liberação do acesso ao local;
- Laptop: utilizado para gerenciar o controle acesso através dos softwares desenvolvidos;
- Softwares desenvolvidos: utilizado para enviar, receber e validar dados, bem como facilitar o gerenciamento dos acessos;
- Banco de dados: armazena os dados dos usuários e acessos;

O teclado de 12 teclas é a interface de entrada responsável pela ligação entre o usuário e o microcontrolador.

Foi utilizado kit LABPIC que é um kit didático para desenvolvimento de projetos utilizando microcontrolador PIC16F877, pois o mesmo possui diversos dispositivos integrados como o display, a comunicação serial e o relé acima descritos, facilitando a implementação deste projeto.

Além disso, foi integrada ao sistema uma fechadura elétrica para restringir o acesso físico ao local. Como o ponto central desse projeto é controlar o acesso a locais restritos, utiliza-se uma fechadura elétrica que só será destravada quando todas as verificações relativas à matrícula e senha forem concluídas. O acionamento dessa fechadura é realizado através de um sinal do microcontrolador que ativa um circuito baseado em relé e este finalmente destrava a fechadura.

O circuito de interface elétrica é apresentado na Figura 2.2 e é composto pelos circuitos do microcontrolador, do relé e da fechadura elétrica. Este circuito libera ou nega o acesso do usuário através da validação dos dados do usuário pelo microcomputador enviando um sinal ao microcontrolador que aciona o relé liberando a fechadura elétrica.

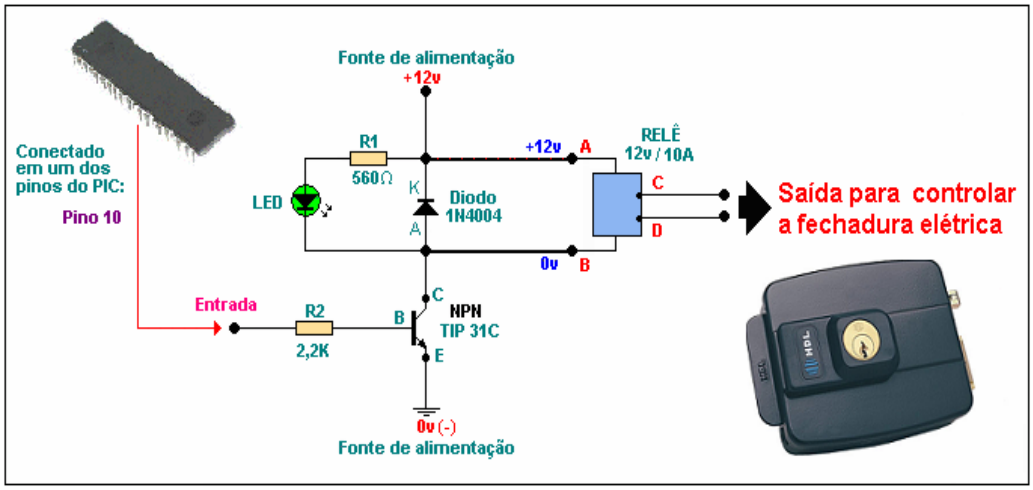


Figura 2.2 - Circuito completo de liberação de acesso

Neste projeto é utilizado ainda um microcomputador que irá tratar os dados de entrada do usuário, sendo que a comunicação entre o microcomputador e o microcontrolador será realizada através da porta serial.

Por fim, o sistema foi desenvolvido com a criação de um banco de dados Microsoft Access que possibilita a armazenagem de dados e geração de relatórios.

Capítulo 3 – Desenvolvimento do Protótipo

Este capítulo apresenta o hardware, as interfaces, os softwares desenvolvidos e os resultados obtidos relativos ao desenvolvimento do protótipo.

Para o desenvolvimento deste protótipo foi necessário a integração de diversos dispositivos que serão apresentados nos tópicos abaixo com as suas devidas descrições e funções dentro do sistema.

3.1. HARDWARE E INTERFACES

Neste tópico, trataremos do hardware e das interfaces utilizadas para a implementação do sistema.

Neste projeto, o hardware proposto é composto por um microcontrolador PIC16F877 e as interfaces com o usuário serão realizadas pelo teclado e pelo display de LCD, bem como a comunicação entre o microcontrolador e o microcomputador será realizada via porta serial.

3.1.1. TECLADO

O teclado numérico utilizado neste sistema foi retirado de um telefone comum.

Na Figura 3.1 é apresentada a configuração interna do teclado, sendo ele composto de quatro linhas por três colunas.

Quando uma tecla é pressionada, a linha e a coluna adjacentes à tecla são curto-circuitadas, dessa maneira pode-se identificar qual tecla foi pressionada.

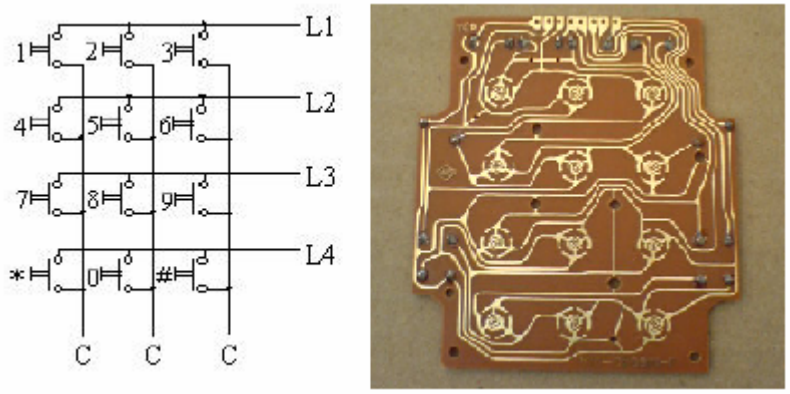


Figura 3.1 - Esquemático do teclado matricial

Além das dez teclas numéricas (de zero a nove), o teclado possui também as teclas ‘*’, que corrige um número digitado pelo usuário e ‘#’, que confirma a seqüência numérica digitada pelo usuário.

Figura 3.2 - Teclado

Assim, por meio do teclado apresentado na Figura 3.2, o usuário se comunicará com o banco de dados em que está cadastrado.

3.1.2. DISPLAY DE LCD

O LCD (Liquid Cristal Display) é o componente responsável pela visualização dos dados que serão inseridos pelo usuário, via teclado, no sistema.

O display de LCD é uma interface de saída útil em sistemas microprocessados. No presente projeto, utilizaremos um módulo de LED backlight já integrado ao KIT LABPIC que passaremos a descrever.

Na Figura 3.3 é mostrado o display de LCD que compõe o sistema desenvolvido:

Figura 3.3 - Display LCD

3.1.3. KIT LABPIC

O kit LABPIC utilizado não possui ligações físicas permanentes entre o microcontrolador e os demais componentes do sistema, possibilitando a melhor adequação do projeto. [LABIT, 2007]

Na Figura 3.4 é mostrado o kit de desenvolvimento utilizado:

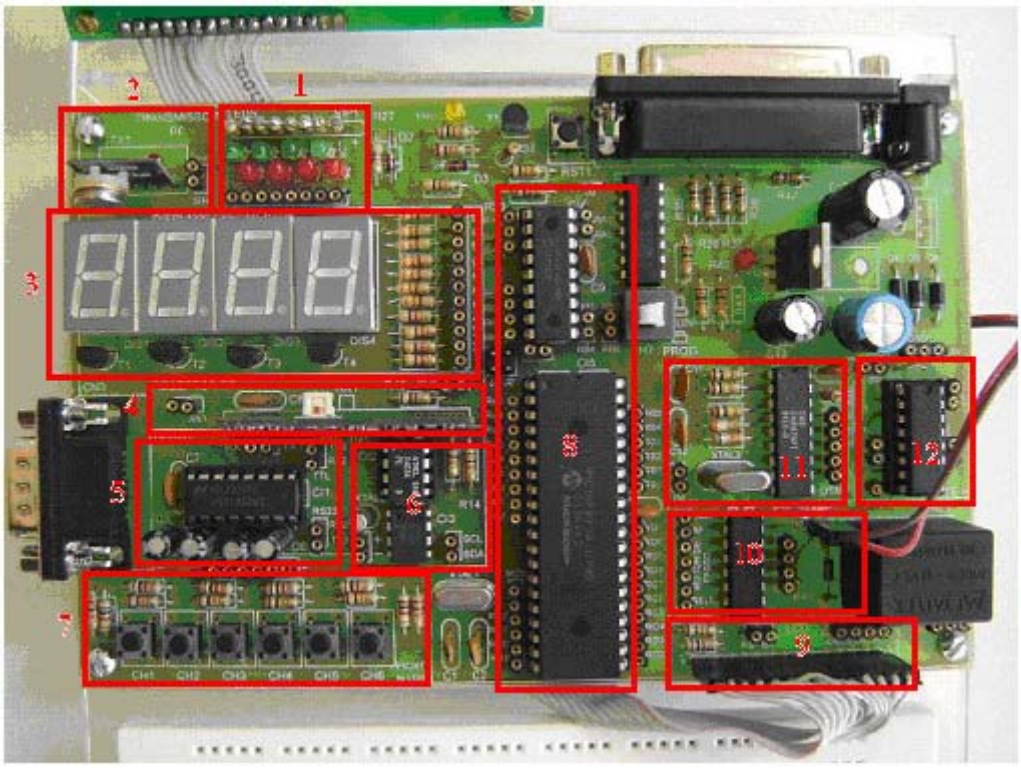


Figura 3.4 – Kit de desenvolvimento LABPIC

O LABPIC é composto de 12 módulos, são eles:

1 – 8 leds, sendo quatro vermelhos e quatro verdes, anôdo comum, com ligações individuais;

2 – Transmissor RF 443MHz;

3 – 4 displays de 7 segmentos, catodo comum, com ligações individuais;

4 – Receptor RF 443MHz;

5 – Interface serial RS-232 e TTL;

6 – Bloco I2C, com memória EEPROM 24C04 e DS1307;

7 – 6 chaves tipo push-button, com circuito fechado para o terra;

8 – 2 microcontroladores PIC: 16F877A e 16F628A, com um cristal de 4MHz, sendo que para fins deste trabalho utilizaremos apenas 0 microcontrolador 16F877A;

9 – LCD ligado em 4 bits;

10 – Lâmpada, alto-falante, relé e driver para motor de passo;

11 – Decodificador DTMF;

12 – Ponte H;

Contudo, serão utilizados apenas os seguintes módulos: 8 leds, sendo quatro vermelhos e quatro verdes, anôdo comum, com ligações individuais; interface serial RS-232 e TTL; microcontrolador PIC 16F877A com um cristal de 4MHz; LCD ligado em 4 bits; e relé.

Assim, o kit em questão possibilitará a programação do sistema na linguagem C e Assembly, sendo que, no presente projeto, será utilizada a linguagem Assembly, dado que não será necessária a utilização da linguagem C.

3.1.4. MICROCONTROLADOR PIC 16F877A

O principal componente do kit LABPIC utilizado é o microcontrolador PIC 16F877A.

Um microcontrolador é um computador programável, em um chip otimizado para controlar dispositivos eletrônicos. É uma espécie de microprocessador, com memória e interfaces de E/S(I/O) integrados, possuindo todas lógicas para se projetar qualquer tipo de circuito encontrado, enfatizando a auto-suficiência, em constraste com um microprocessador de propósito geral, o mesmo tipo usado nos PCs, que requer chips adicionais para prover as funções necessárias. [WIKIPEDIA, 2007]

Notadamente, o microcontrolador PIC 16F877 é um microcontrolador da família de 8 bits e núcleo de 14 bits fabricado pela Microchip Technology. Possui memória flash de programa com 8192 palavras de 14 bits, memória RAM com 368 bytes e memória eeprom com 256 bytes. Sua frequência de operação (clock) vai até 20MHz, resultando em uma velocidade de processamento de 5 MIPS. Seu conjunto de instruções RISC se compõe de 35 instruções. Pode funcionar com alimentação de 2V a 5,5V. Sua pinagem DIP tem 40 pinos. Como periféricos ele possui: 5

conjuntos de portas de entrada e saída (total de 33 portas), conversor analógico-digital de 10 bits de resolução e 8 canais de entrada, periférico de comunicação paralela e serial (USART e MSSP), 2 Módulos CCP (Comparação, Captura e PWM), 3 Timers (1 de 16 bits e 2 de 8 bits) e Watchdog timer. [WIKIPEDIA, 2007]

Para fins deste trabalho, o microcontrolador servirá para interpretar o teclado, enviar mensagens ao display de LCD e realizar a comunicação com o software gerenciador de usuários presente no microcomputador.

A utilização do microcontrolador PIC 16F877 possibilitará que o sistema seja mais eficiente, tendo em vista que este dispositivo utilizado dois barramentos de endereços distintos para acessar instruções e dados, conforme descrito na Figura 3.5 referente à arquitetura Harvard. [CURSO ON LINE PIC, 2007]

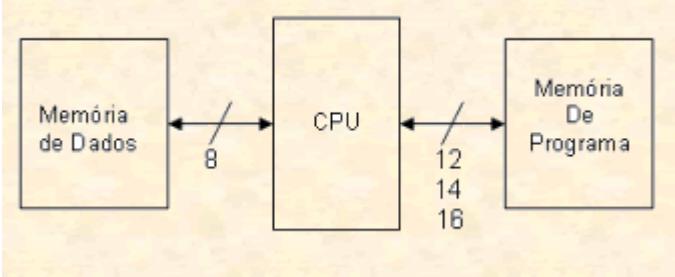


Figura 3.5 – Arquitetura Harvard

As funções a serem executadas por este microcontrolador no projeto serão: receber informações do usuário (matrícula e senha) via teclado e transmitir para o microcomputador, receber informações do usuário (matrícula e senha) via teclado e acionar o display de LCD, bem como controlar a abertura da fechadura elétrica.

3.1.4.1. INTERRUPÇÕES

Contudo, dentro do microcontrolador, existem as chamadas interrupções que devem ser tratadas para que o projeto possa ser implementado.

Uma interrupção pode ser definida como uma parada temporária na operação da rotina principal, que será abordada mais adiante, por um determinado comando, sendo que a operação é reiniciada no mesmo ponto onde ocorreu a parada.

O tratamento que será dado às interrupções neste projeto envolve a comunicação serial para a troca de dados entre o microcontrolador e microcomputador via porta serial.

3.1.5. COMUNICAÇÃO SERIAL NO PROJETO

Dentro desse contexto, importante ressaltar que a transmissão serial no projeto será realizada utilizando-se os pinos de transmissão de dados (TX), pino 3, e recepção de dados (RX) pino 2, que possibilitará o envio de dados entre o microcontrolador e microcomputador. O aterramento, caso ocorra nível de tensão acima de 5 volts, é feito pelo pino 5.

Na Figura 3.6 é mostrada a configuração do cabo utilizado para efetivar a comunicação dos dados.

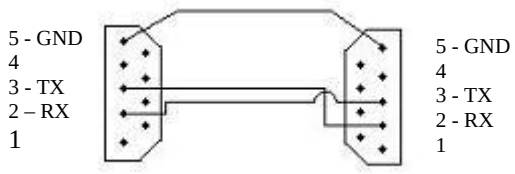


Figura 3.6 – Configuração do cabo de comunicação serial utilizado

A comunicação é feita com 2 vias, a via de TX está ligada ao pino RC6 e a via de RX está ligada ao pino RC7 do microcontrolador, conforme se pode observar na Tabela 3.1:

PIC	**COM.**
RC6	TX (saída)
RC7	RX (entrada)

Tabela 3.1 – Pinagem Microcontrolador PIC

Faz parte também do módulo de comunicação serial. Na Tabela 3.2 é apresentada a pinagem do conector:

Pino	Função
1	-
2	TX (saída)
3	RX (entrada)
4	-
5	Terra (GND)
6	-
7	-
8	-
9	-

Tabela 3.2 – Pinagem Portal Serial

A comunicação será implementada utilizando os recursos do próprio microcontrolador via protocolo serial RS-232 que possibilitará a troca de mensagens entre o microcontrolador e microcomputador.

Segue na Figura 3.7 o conector RS232:

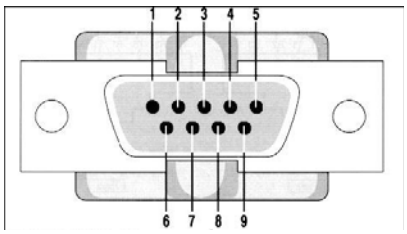


Figura 3.7 – Conector RS232

Para que a integração dos componentes seja feita de forma íntegra e confiável, a taxa de transferência da comunicação será de 9.600 bps, tanto no microcontrolador quanto no microcomputador.

No microcomputador, a comunicação serial foi configurada com base na linguagem de programação em Visual Basic para recepção e tratamento dos dados. Por sua vez, no microcontrolador, a comunicação serial foi configurada em Assembly para recepção e envio de dados.

3.2. SOFTWARES DESENVOLVIDOS

Após a descrição do hardware e das interfaces que compõem o protótipo, apresentaremos o sistema desenvolvido para a implementação do presente projeto.

Serão desenvolvidos dois módulos de softwares: (i) controle de acesso (microcomputador e microcontrolador) e (ii) gerenciamento (banco de dados).

Estes softwares utilizam a linguagem de programação Assembly e Visual Basic com linguagem de banco de dados transact SQL.

3.2.1. MICROCONTROLADOR

Tendo em vista que o microcontrolador utilizado no projeto (PIC16F877) centraliza todas as ações realizadas no sistema, uma vez que todos os dispositivos estão conectados a ele, faz-se necessário o desenvolvimento de um código fonte que englobe todas as possibilidades previstas para o funcionamento do sistema.

São apresentados a seguir os métodos utilizados para alcançar os objetivos. Esses métodos estão de acordo com o código Assembly que está apresentado integralmente em anexo.

Foram utilizadas as seguintes ferramentas para manipulação e compilação do código Assembly: (i) MpLab da Microsystem para desenvolvimento do código em Assembly e compilação do código gerando o hexadecimal a ser gravado no microcontrolado PIC; e (ii) IC-Prog utilizado para gravar o código em hexadecimal na memória flash do PIC.

Ao final da compilação do Código, será criado um arquivo no diretório de trabalho com o mesmo nome do programa fonte original, porém com extensão .HEX. Este arquivo será gravado no microcontrolador PIC, através do programa IC-PROG. [IC-PROG, 2007]

Na Figura 3.8 são mostrados os pinos da porta paralela que é utilizada para a gravação do código hexadecimal no PIC através do kit LABPIC utilizados neste projeto:

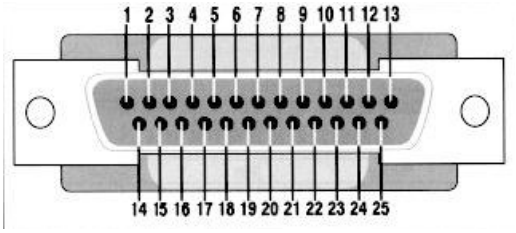


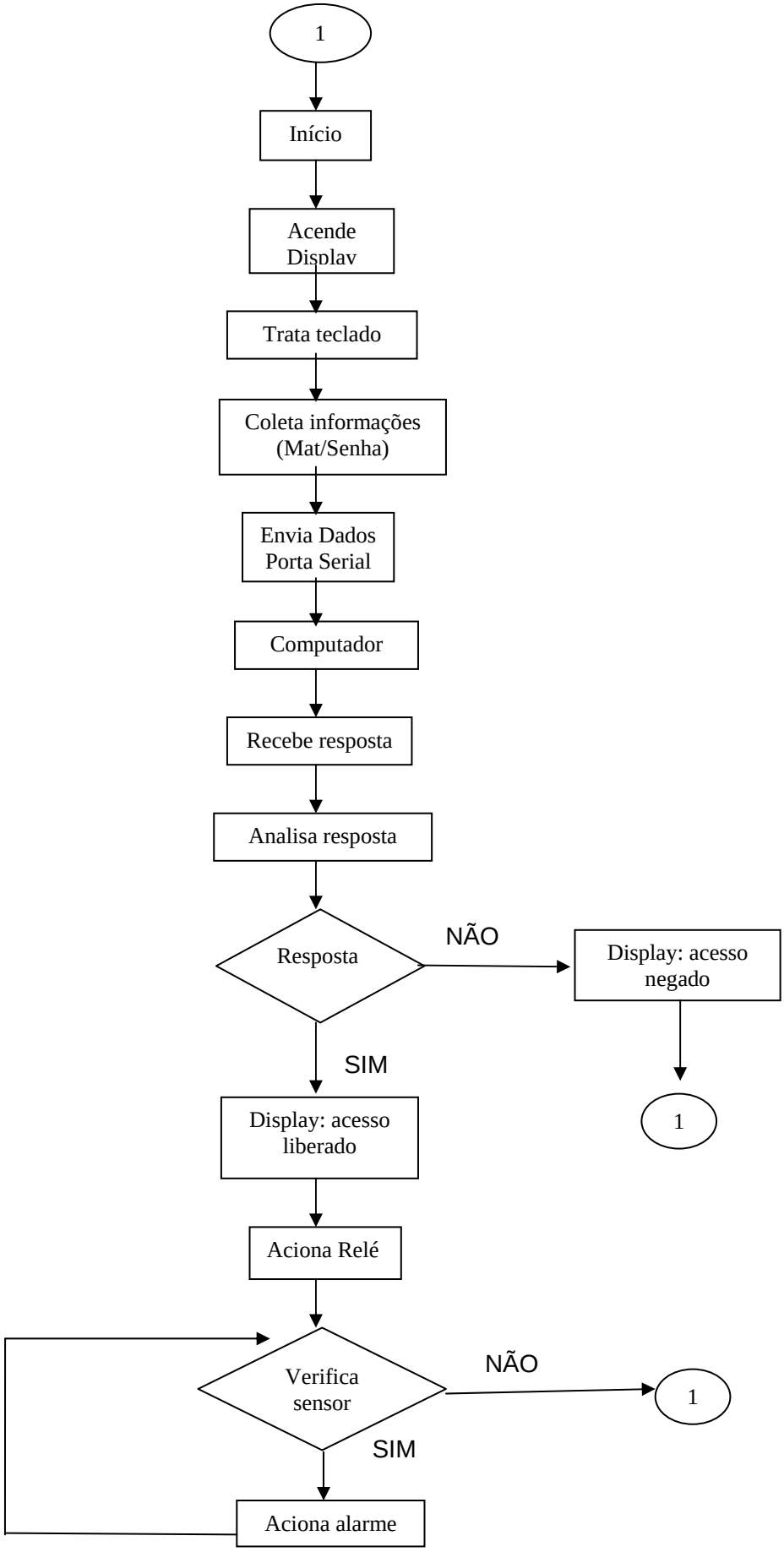
Figura 3.8 - Conector da porta paralela

O código criado terá exemplificadamente as seguintes funções:

- Receber os dados digitados no teclado;
- Enviar os dados ao microcomputador pela porta serial;
- Receber resposta do microcomputador;
- Verificar a resposta para liberar ou negar o acesso ao local;
- Mostrar mensagem no display (liberado/negado);
- Se liberado, acionar relé abrindo a fechadura;
- Se negado, manter a porta fechada;

Para viabilizar tal estrutura, será criado um software com base no seguinte fluxograma que gerenciará a comunicação serial, o sensor, o teclado, os canais para acionamento da fechadura elétrica e o LED.

A seguir fluxograma para facilitar a visualização e a interpretação do desenvolvimento do software:


1
Início
Acende Displav
Trata teclado
Coleta informações (Mat/Senha)
Envia Dados Porta Serial
Computador
Recebe resposta
Analisa resposta
Resposta
NÃO
Display: acesso negado
1
SIM
Display: acesso liberado
Aciona Relé
Verifica sensor
NÃO
1
SIM
Aciona alarme

3.2.2. MICROCOMPUTADOR

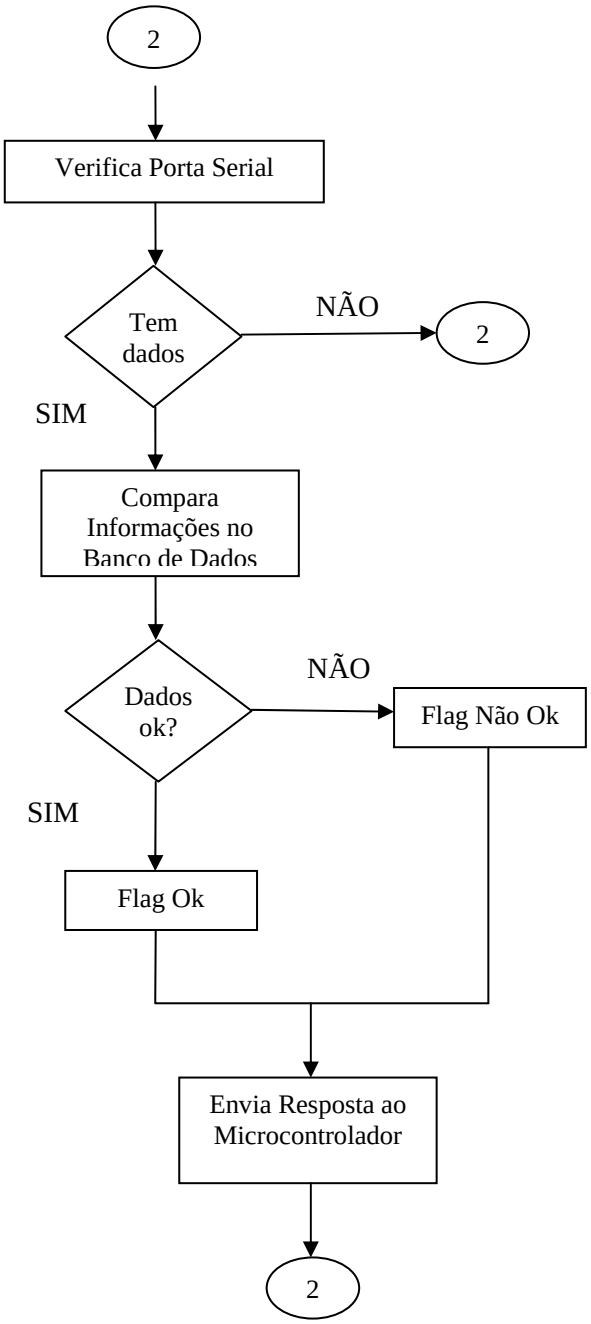
Por meio das rotinas desenvolvidas conforme demonstrado no código anexo, o microcomputador, via porta serial, receberá as informações digitadas pelo usuário (matrícula e senha) enviadas pelo microcontrolador. Essas informações serão tratadas pelo software que foi desenvolvido para validá-las no banco de dados, enviando uma resposta ao microcontrolador, via porta serial, informando se o usuário está apto ou não para acessar ao local.

O software desenvolvido terá as seguintes funções:

- Looping verificando os dados de entrada na porta serial;
- Receber os dados enviados pelo microcontrolador pela porta serial;
- Validar as informações no banco de dados;
- Guardar as informações do usuário que acessou o local;
- Enviar resposta ao microcontrolador PIC;

Para viabilizar tal estrutura, será criado um software com base no seguinte fluxograma que gerenciará a comunicação serial e manipulará os dados recebidos verificando a sua validade no banco de dados.

A seguir fluxograma para facilitar a visualização e a interpretação do desenvolvimento do software:


2
Verifica Porta Serial
Tem dados
NÃO
2
SIM
Compara Informações no Banco de Dados
Dados ok?
NÃO
Flag Não Ok
SIM
Flag Ok
Envia Resposta ao Microcontrolador
2

3.2.3 BANCO DE DADOS

Para a monitoração do controle destes acessos, foi igualmente desenvolvido um software para gerenciar o armazenamento das informações recebidas no sistema.

Este software terá as seguintes funções:

- Possibilitar o cadastro, a remoção e a edição de informações dos usuários;
- Gerar relatórios, a partir das informações gravadas pelos softwares de acesso, das pessoas que acessaram o local;

A interação dos softwares de acesso e do software de gerenciamento de controle de acesso em conjunto com o circuito montado garantirá a segurança do local restrito.

3.2.3.1. CONCEITO DE BANCO DE DADOS

O banco de dados é um sistema computadorizado de arquivamento de registros, onde informações são armazenadas e identificadas por uma característica comum. Exemplo:

Um arquivo tipo texto delimitado por ponto e vírgula com várias linhas pode ser considerado um sistema de arquivamento e controle de registros, que, por sua vez, pode ser considerado uma pequena base de dados, ou, ainda, em uma tabela simples [ALBUQUERQUE, 2005]:

Nome; Cargo; Área; Matrícula.
José da Silva; Gerente; Financeiro; 1212.
Maria Pereira; Analista; Comercial; 3455.
Márcio Ferreira; Técnico; Informática; 2347.

Segue na Tabela 3.3 exemplo de tabela de dados:

Nome	Cargo	Área	Matrícula
José da Silva	Gerente	Financeiro	1212
Maria Pereira	Analista	Comercial	3455
Márcio Ferreira	Técnico	Informática	2347

Tabela 3.3 – Exemplo de tabela de dados

No exemplo da Tabela 3.3, temos uma situação de controle e identificação de funcionários em que seu nome, cargo, área e matrícula são gravados em um arquivo no microcomputador. Desta forma, acessa-se este arquivo e filtra-se, conforme necessidade.

Para realizar tais consultas e até mesmo a atualização dos dados existentes neste arquivo, é necessário uma série de controles e procedimentos. É nesta parte que começamos a discutir o Sistema de Gerenciamento de Controle de Dados (SGBD).

O SGDB é uma coleção de programas que possibilita que os usuários criem ou mantenham um banco de dados. O SGBD é, portanto, um sistema de software de finalidade genérica que facilita o processo de definição, construção e manipulação de banco de dados para várias aplicações.

Definir um banco de dados envolve especificar os tipos de dados, as estruturas e as restrições para as informações que serão armazenadas no banco de dados. Construir o Banco de dados é o processo de armazenar as referidas informações em algum meio de armazenamento que seja controlado pelo SGBD. Manipular o banco de dados inclui funções como fazer consultas ao banco de dados para recuperar dados específicos, atualizar o banco de dados para refletir alterações e gerar relatórios a partir dos dados.

O sistema aqui desenvolvido necessita manipular e gerenciar dados, por isso utiliza o Sistema Gerenciador de Bancos de Dados (SGBD), utilizando o modelo relacional que é comumente utilizado devido sua facilidade de manuseio.

3.2.3.2 LINGUAGEM DE CONSULTA E MANIPULAÇÃO DE DADOS

Conforme mencionado, um SGBD faz operações de recuperação, modificação, remoção e inserção que são realizadas de acordo com o sistema de banco de dados a ser utilizado.

Atualmente, a maioria dos bancos de dados funciona com um padrão SQL (Structured Query Language) que é a linguagem utilizada para a manipulação de dados por comandos conhecidos em inglês por *statments,* conforme descrito a seguir:

- Recuperação: SELECT;
- Modificação; UPDATE;
- Remoção: DELETE;
- Inserção; INSERT

“Assim, temos um padrão fortemente definido, que é de grande utilização em diversos sistemas SGBD’s. A SQL é uma linguagem de banco de dados ampla; possui instruções para definição de dados, consulta e atualizações. Portanto, ela é uma DDL (Linguagem de Definição de Dados) e uma DML (Linguagem de Manipulação de Dados)”. [GOUVEIA, 1985]

3.2.3.3 MODELO DE BANCO DE DADOS DO PROJETO

As configurações do banco de dados estão fortemente ligadas às características dos dados enviados pelo microcontrolador, pois o sistema deve ser eficiente em tempo de resposta e processamento, procurando agilizar o acesso do usuário, mas garantindo a segurança e a validade dos dados digitados.

Neste projeto, foi criada uma base de dados simples, para fácil manipulação das informações. Contudo, o SGBD utilizado permite consultas e/ou acessos, como acontece em sistemas empresariais reais que possuem bancos de dados como Oracle, SQL Server, ou DB2.

A função do banco de dados neste projeto é individualizar o acesso ao local possibilitando a geração de relatórios. Para tanto, foi utilizado o Microsoft Access, que suporta a linguagem SQL, tendo em vista que este é um banco de dados de simples configuração e de fácil acesso por aplicações cliente-servidor.

Assim, para controlar e permitir a entrada apenas de pessoas cadastradas no banco de dados e com acesso autorizado, é desenvolvido um software de banco de dados, com base no quanto aqui descrito.

A primeira etapa da elaboração deste software consiste na elaboração de uma tabela FUNCIONARIO que tem como finalidade o cadastro principal dos usuários que podem acessar o ambiente restrito.

Nesta Tabela há as seguintes informações relativas ao funcionário: código (matrícula), nome do usuário, setor, cargo e senha, conforme ilustrado na Figura 3.9.

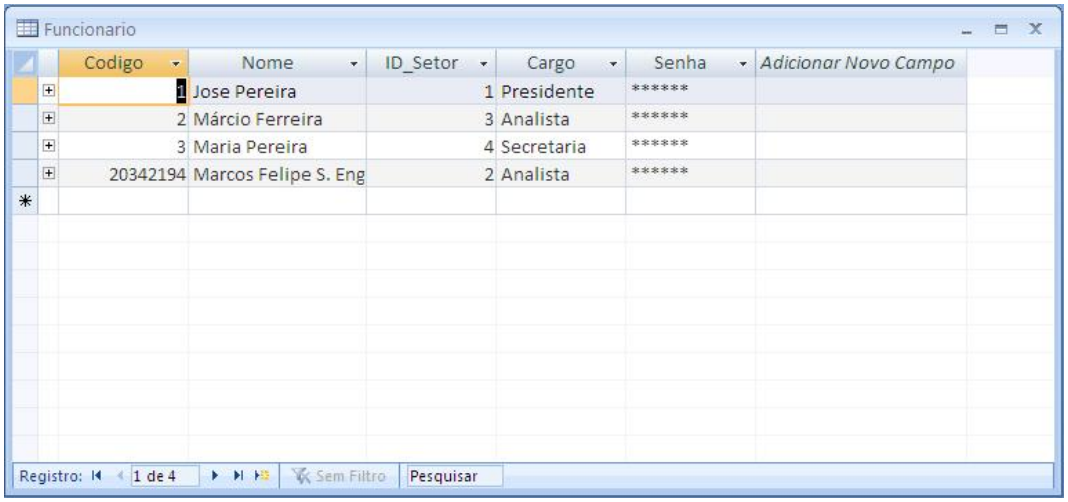


Figura 3.9 – Tela da tabela de dados FUNCIONARIO

Após a criação da Tabela, verificou-se a necessidade de delimitar a informação relativa ao setor para que não houvesse a multiplicação de dados digitados indevidamente. Assim, foi elaborada uma Tabela relacionando a descrição do setor com um número de identificação de área vinculada à Tabela FUNCIONARIO, limitando a escolha das áreas descritas na Tabela SETOR quando do cadastro de um usuário.

Na Figura 3.10 é mostrada a Tabela de setores desenvolvida:

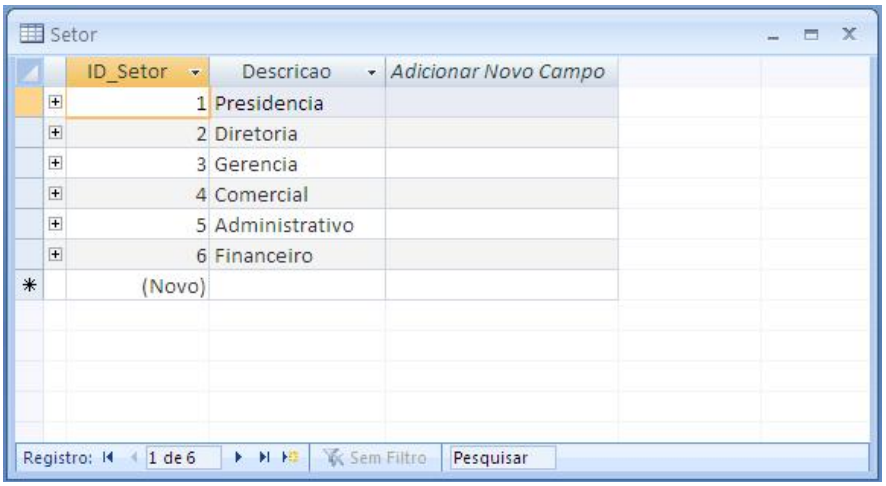


Figura 3.10 – Tela da tabela de dados SETOR

Além disso, é elaborada uma Tabela para armazenar as informações relativas à data e ao horário de acesso ao local restrito pelas pessoas cadastradas. Esta

Tabela relacionará o código (matrícula) com a data de entrada e saída do funcionário.

Na Figura 3.11 é visualizada a Tabela de dados de controle desenvolvida:

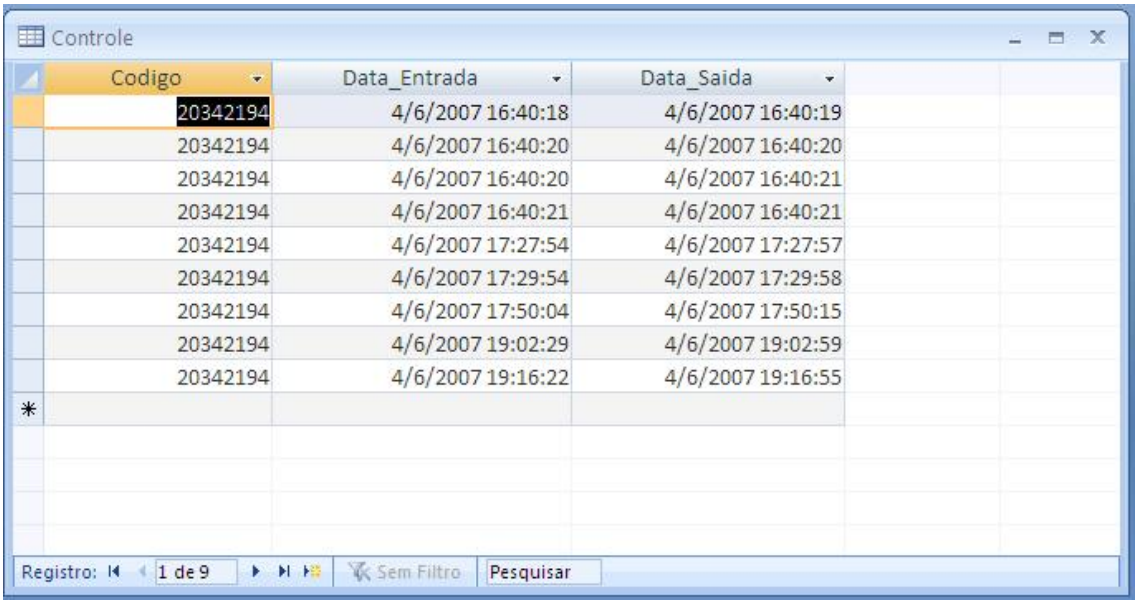


Figura 3.11 – Tela da Tabela de dados CONTROLE

Em seguida, é desenvolvido o relacionamento entre as Tabelas ilustradas nas Figuras. O relacionamento da Tabela FUNCIONARIO para a tabela SETOR será de 1 para 1 que significa que um usuário só pode pertencer a um setor. O relacionamento da Tabela FUNCIONARIO para a Tabela CONTROLE será de 1 para n, pois um determinada usuário pode acessar várias vezes ao local restrito.

Segue na Figura 3.12 a demonstração do relacionamento descrito:

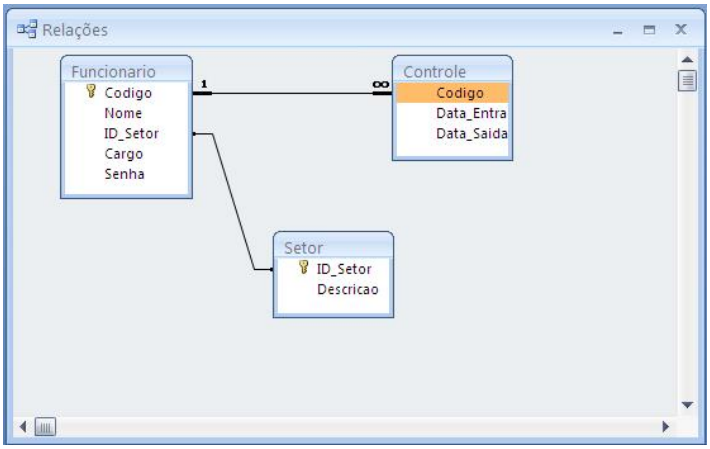
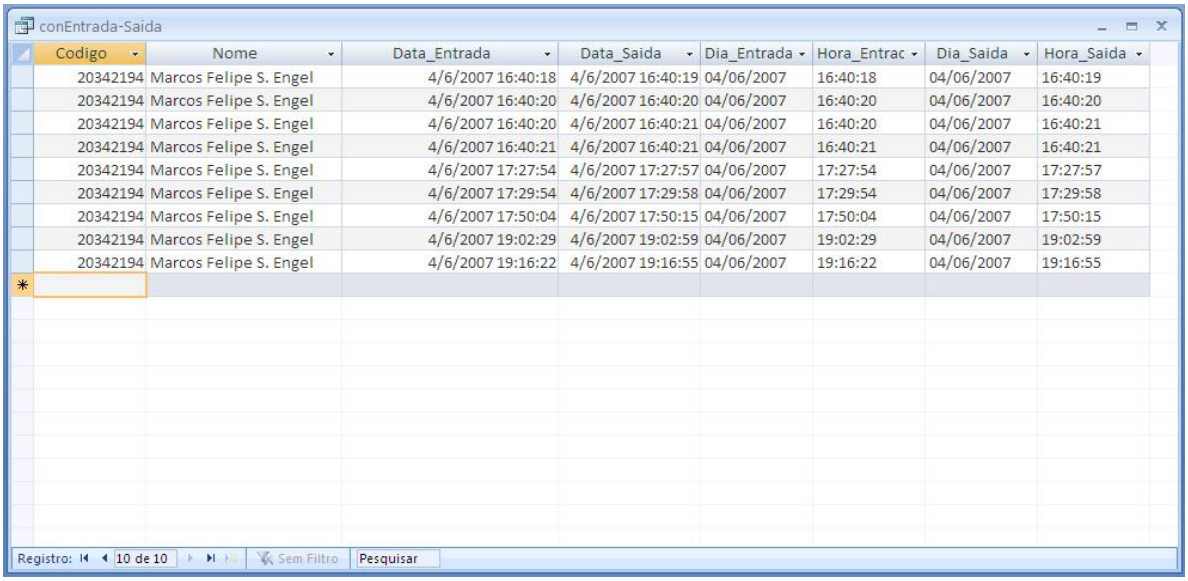


Figura 3.12 – Modelo entidade relacionamento

Tendo em vista os relacionamentos criados, será possível gerar relatórios com todas as informações do usuário alinhadas com as informações sobre o acesso ao local, conforme relatório apresentado na Figura 3.13:


conEntrada-Saida

Codigo	Nome	Data_Entrada	Data_Saida	Dia_Entrada	Hora_Entrac	Dia_Saida	Hora_Saida
20342194	Marcos Felipe S. Engel	4/6/2007 16:40:18	4/6/2007 16:40:19	04/06/2007	16:40:18	04/06/2007	16:40:19
20342194	Marcos Felipe S. Engel	4/6/2007 16:40:20	4/6/2007 16:40:20	04/06/2007	16:40:20	04/06/2007	16:40:20
20342194	Marcos Felipe S. Engel	4/6/2007 16:40:20	4/6/2007 16:40:21	04/06/2007	16:40:20	04/06/2007	16:40:21
20342194	Marcos Felipe S. Engel	4/6/2007 16:40:21	4/6/2007 16:40:21	04/06/2007	16:40:21	04/06/2007	16:40:21
20342194	Marcos Felipe S. Engel	4/6/2007 17:27:54	4/6/2007 17:27:57	04/06/2007	17:27:54	04/06/2007	17:27:57
20342194	Marcos Felipe S. Engel	4/6/2007 17:29:54	4/6/2007 17:29:58	04/06/2007	17:29:54	04/06/2007	17:29:58
20342194	Marcos Felipe S. Engel	4/6/2007 17:50:04	4/6/2007 17:50:15	04/06/2007	17:50:04	04/06/2007	17:50:15
20342194	Marcos Felipe S. Engel	4/6/2007 19:02:29	4/6/2007 19:02:59	04/06/2007	19:02:29	04/06/2007	19:02:59
20342194	Marcos Felipe S. Engel	4/6/2007 19:16:22	4/6/2007 19:16:55	04/06/2007	19:16:22	04/06/2007	19:16:55

Registro: 10 de 10 Sem Filtro Pesquisar

Figura 3.13 – Relatório de Acesso

Assim, o microcomputador armazena os dados necessários para garantir ou restringir o acesso dos funcionários, bem como grava os seus dados de acesso, como data e hora.

Após criada a base de dados, é necessário a elaboração de um software para gerenciar esta base e cadastrar novos usuários. Na Figura 3.14 é apresentada a tela inicial do software criado:

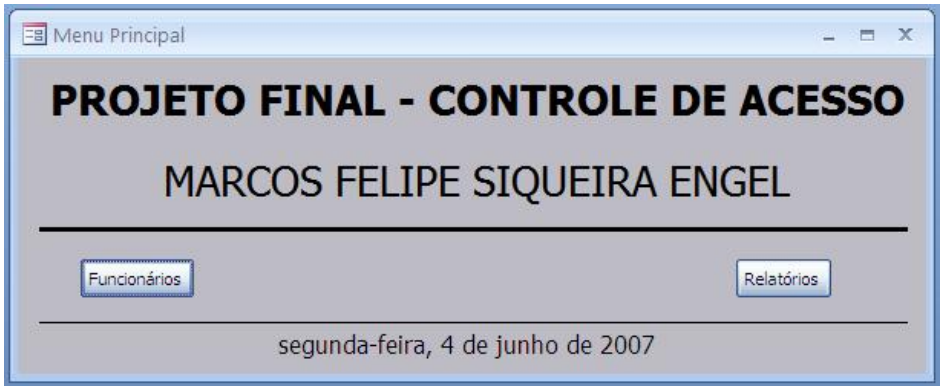


Figura 3.14 – Tela inicial do Software

Para a manipulação dos dados dos funcionários, foi desenvolvida a interface apresentada na Figura 3.15 que facilita o cadastro das informações da base de dados.

Na Figura 3.15 é apresentada a interface desenvolvida:

Figura 3.15 – Tela para manipulação de dados dos usuários

O banco de dados é consultado mediante a utilização do software de gerenciamento acima descrito, o qual possibilita a obtenção de relatórios sobre o acesso ao local restrito.

Tanto a base de dados quanto o software de gerenciamento serão desenvolvidos por meio da ferramenta Microsoft Access que possibilita a criação desde a base de dados até a interface de gerenciamento.

A interação do banco de dados a ser utilizado com os demais componentes do sistema se dará através do software de gerenciamento descrito, onde as informações do usuário encaminhadas pelo microcontrolador são recebidas, validadas e armazenadas, possibilitando a geração de relatórios com data e hora de acesso.

3.4. RESULTADOS E SIMULAÇÕES

3.4.1. CONFIGURAÇÃO DO EQUIPAMENTO UTILIZADO

As simulações relativas ao projeto foram realizadas em um microcomputador (laptop) com a seguinte configuração:

1. *Hardware*:
    - AMD Athlon XP 3.2 Ghz
    - 512 MB de RAM
    - Fabricante HP
    - Sistema Operacional: Windows XP Home Edition Service Pack2

2. *Softwares* utilizados:
    - MPLAB V 7.6
    - ICPROG
    - MS Access 2003

3.4.2. RESULTADOS OBTIDOS

A utilização do microcontrolador PIC neste projeto permitiu uma maior flexibilidade na expansão de memória e armazenamento.

Além disso, verificou-se que a arquitetura Harvard utilizada pelo PIC em questão é mais desenvolvida do que a arquitetura Von Newmann utilizada, por exemplo, no microcontrolador da família 8051. Isso porque a transmissão no PIC 16F877A utiliza um barramento para memória de programa e outro para memória de dados, já o 8051 utiliza o mesmo barramento para memória de programa e de dados, o que pioraria o desempenho do sistema. [CURSO PIC ON LINE, 2007]

Os resultados foram obtidos a partir da integração de diversos componentes, sendo que, durante a implementação do projeto, houve diversas dificuldades que levaram à necessidade de um estudo aprofundado sobre a linguagem Assembly e sobre a comunicação serial utilizada no sistema.

O desenvolvimento do sistema com a Linguagem Assembly possibilitou um maior desempenho de processamento do microcontrolador, minimizando as instruções necessárias para o desenvolvimento do código.

O código do PIC foi desenvolvido com a intenção de prever o maior número de situações possíveis e apresentou um desempenho satisfatório.

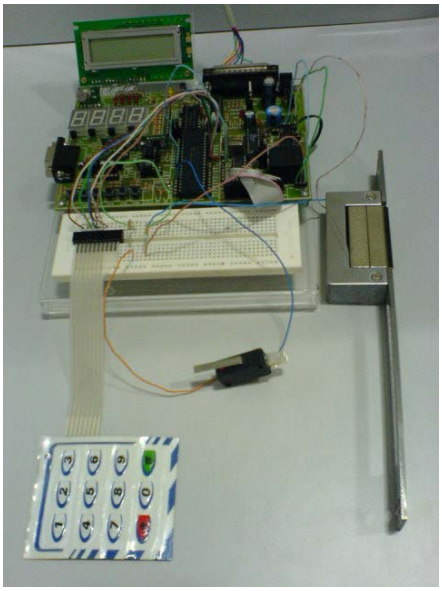


Figura 3.16 - Foto tirada durante o desenvolvimento do protótipo

Procurou-se, durante o desenvolvimento do sistema, adotar medidas que o tornasse realizável em qualquer ambiente onde se deseja restringir acessos e obter informações sobre esses acessos.

Assim, durante o desenvolvimento do protótipo, as simulações realizadas culminaram no sistema reproduzido na Figura 3.17:

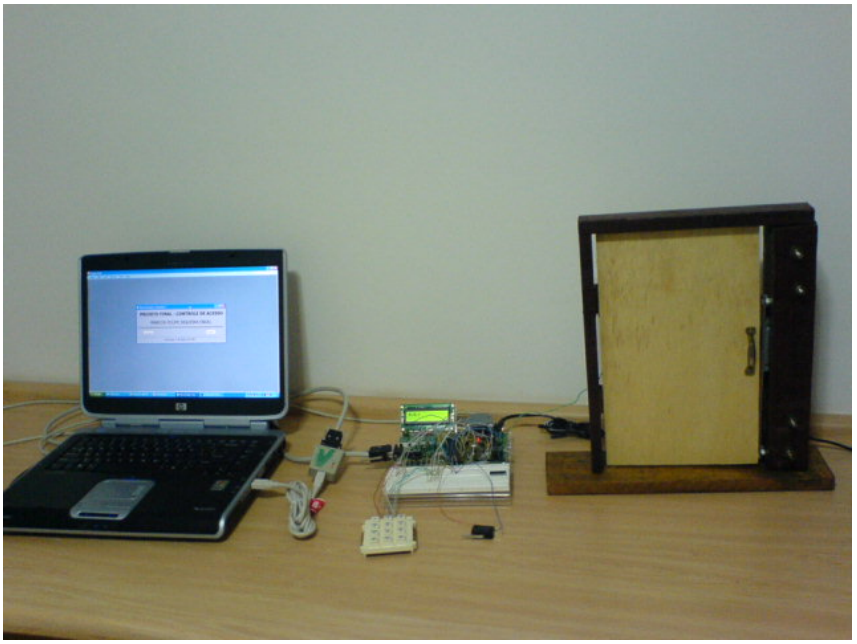


Figura 3.17 - Foto tirada durante as simulações realizadas

Esse projeto representa uma maneira eficiente de realizar um controle de acesso de um ambiente desejado. O cadastramento remoto de usuários permite que mesmo uma pessoa que não tenha conhecimentos sobre microcontroladores ou coisas afins possa usufruir de todas as funcionalidades que o sistema oferece. Isso porque através do software de gerenciamento (Microsoft Access) poder-se-á excluir e cadastrar usuários, bem como gerar relatórios de acessos.

Capítulo 4 - Conclusão

Neste projeto foi desenvolvido um protótipo de controle de acesso seguro para ambientes empresariais.

Durante o desenvolvimento do sistema, priorizou-se a implementação física da proposta teórica, sendo que o resultado obtido, apesar das dificuldades encontradas devido à complexidade envolvida na integração dos componentes, foi satisfatório e contou com a aplicação de conhecimentos adquiridos ao longo do curso sobre banco de dados, programação, lógica digital, segurança, microcontrolador e microprocessador.

O projeto possibilita o desenvolvimento de um circuito de baixo custo para o controle de acesso que atenderá perfeitamente soluções empresariais variadas. Isso porque criou-se uma opção segura e adequável a cada caso, conforme solicitado pela empresa interessada.

Utilizou-se uma série referências teóricas e técnicas relacionadas ao campo de microcontroladores e ao estudo da linguagem Assembly necessários para a implementação do sistema.

O estudo dos princípios básicos de manipulação de bancos de dados foi utilizado para a construção das tabelas responsáveis por armazenar todos os dados inseridos ou recebidos, bem como na elaboração dos softwares.

Assim, esse projeto, com a aquisição de fechadura elétrica, confecção do circuito e instalação dos softwares desenvolvidos, foi devidamente desenvolvido e concluído.

4.1. PROPOSTAS FUTURAS

Como proposta futura para este projeto, pode-se citar a melhoria deste protótipo utilizando como interface de entrada leitor biométrico para digital. Ainda, poderia se pensar no desenvolvimento deste sistema melhorado com a integração de câmeras de vídeo com gravação simultânea do local acessado.

REFERÊNCIAS BIBLIOGRÁFICAS

ALBUQUERQUE, Júlio César Nascimento de. *Protótipo de Acesso por meio de Cartão para ambientes restritos.* Monografia de Final de Curso de Engenharia da Computação da Faculdade de Ciências Exatas e Tecnologia - FAET - do Centro Universitário de Brasília – UniCEUB, 2005.

CHAVES, Cristian Rodrigues e MENDES, Marcelo Pereira. *Sistema de Controle de Acessos Microncontrolado – Teclado.* Monografia de Final de Curso de Engenharia Elétrica da Universidade Federal de Góias, 2003.

GOUVEIA, Hélio Auro. *Tradução de: Database: a Primer. Banco de dados: Fundamentos/ CJ Date.* Rio de Janeiro: Campus, 1985.

Microchip Technology Inc. *Datashett PIC16F87X*. Jan 2001.

LABIT. Manual do Kit de Desenvolvimento LABPIC – V.2.21. 2007

NICOLOSI, Denys Emílio Campion. *Microcontrolador 8051 Detalhado*. 5. ed. São Paulo: Érica, 2000.

SÁ, Maurício Cardoso de. *Programação C para Microcontroladores 8051*. 1. ed. São Paulo: Érica, 2005.

VASCONCELOS, Laércio. *Hardware Total*. São Paulo: Makron Books, 2002.

ZANCO, Wagner. *Microcontroladores Uma Abordagem Pratica e Objetiva*. São Paulo, Editora ERICA, 2006.

_____. *Microcontroladores PIC, Técnicas de Software e Hardware para Projetos de Circuitos Eletrônicos com base no PIC 16F877A*. São Paulo: Editora ERICA, 2006.

SITES:

CURSO BÁSICO DE PIC – ONLINE

http://www.edutecbauru.com.br/cursopic/

Acesso em: 16.03.2007

IC-Prog version 1.05D

http://www.ic-prog.com/index1.htm

Acesso em: 28.02 2007.

Microchip Technology Incorporated, MPLab v7.6

http://www.microchip.com

Acesso em: 28.02. 2007.

Rogercom

http://www.rogercom.com

Acesso em: 28.02. 2007.

WIKIPEDIA – **"MICROCONTROLADOR"**

http://pt.wikipedia.org/wiki/microcontrolador

Acesso em: 06.06.2007

APÊNDICE I – CÓDIGO DO MICROCONTROLADOR

```
#INCLUDE P16F877A.INC

  __CONFIG _XT_OSC & _WDT_OFF & _PWRTE_ON & _CP_OFF & _BODEN_ON
& _LVP_OFF

#DEFINE BANK0 BCF STATUS,RP0   ; Declarando comando para mudar de banco
do microcontrolador PIC
#DEFINE BANK1 BSF STATUS,RP0   ; Declarando comando para mudar de banco
do microcontrolador PIC
#DEFINE ENB  PORTD,5    ; Definindo porta do display
#DEFINE RS  PORTD,4     ; Definindo porta do display
#DEFINE AF  PORTA,0     ; Definindo porta do Alto-falante
#DEFINE LED_VERMELHO PORTA,3  ; Definindo porta do Led Verde
#DEFINE LED_VERDE    PORTA,4  ; Definindo porta do Led Vermelho
#DEFINE PORTA_RELE   PORTA,2  ; Definindo porta do Rele
#DEFINE BOTAO_SENSOR PORTA,5  ; Definindo porta do Sensor

  CBLOCK  30H
    TEMPO1  ; Variável usada nas rotinas de tempo
    TEMPO2  ; Variável usada nas rotinas de tempo
    CHAR    ; Caracter ou comando a ser enviado p/ LCD
    SEL_MSG  ; Seleciona mensagem a ser exibida no display
    PONT    ; Ponteiro das tabelas de mensagens do LCD
    Tecla    ; Variável usada para verificação da tecla pressionada
POS  ; Variável para verificação da quantidade de teclas pressionadas
FLAG  ; Variável para indicar se as teclas pressionadas são referentes ao RA ou
Senha
DADO  ; Variável para receber os valores transmitidos pela porta serial
TIMEOUTDB  ; Variável contador para verificar tempo de reposta das informações
do computador
    FREQ    ; Variável da freqüência utilizada para o alarme sonoro
    SALVA_W  ; Salva conteúdo de W nas interrupções
    SALVA_S  ; Salva conteúdo de STATUS nas interrupções
  ENDC

  ORG    000H
  GOTO  INICIO

  ORG  004H     ; Endereço reservado para interrupções
  MOVWF  SALVA_W    ; Salva variável W (work) na variável SALVA_W
  SWAPF    STATUS,W    ; troca Nibble inferior com Nibble superior da variável
STATUS e atribui a W
  MOVWF  SALVA_S    ; Salva variável de status na variável SALVA_S
  MOVF   RCREG,W     ; Lê dado recebido pela porta serial e coloca na variável W
(work)
  MOVWF  DADO    ; Atribui o valor de W para a variável DADO

  FIMINT
```

```
  SWAPF   SALVA_S,W   ; troca Nibble inferior com Nibble superior da variável
SALVA_S e atribui a W
  MOVWF  STATUS    ; Atribui o valor de W para a variável STATUS
  SWAPF   SALVA_W,F   ; troca Nibble inferior com Nibble superior da variável
SALVA_W e atribui a SALVA_W
  SWAPF   SALVA_W,W   ; troca Nibble inferior com Nibble superior da variável
SALVA_W e atribui a W
  RETFIE

INICIO
  BANK1
; Inicialização do LCD
  MOVLW  0C0H      ; coloca o valor b'11000000' na variável W
  MOVWF  TRISD      ; coloca o valor da variável W no TRISD

; Inicialização TRANSMISSÃO SERIAL RS-232
  MOVLW  0BFH      ; RC7 = RX, RC6 = TX
  MOVWF  TRISC      ; coloca o valor da variável W no TRISC
  MOVLW  B'00000111'    ; Porta A como sinais digitais
  MOVWF  ADCON1      ; coloca o valor da variável W no ADCON1
  MOVLW  24
  MOVWF  TXSTA      ; Modo assíncrono, 8 bits, TX habilitada
  MOVLW  .25
  MOVWF  SPBRG      ; Baud rate = 9600 bps
  MOVLW  B'11000000'   ; gie desligado 0xxxxxxxx
  MOVWF  INTCON      ; Habilita interrupção de periféricos
  MOVLW  20
  MOVWF  PIE1      ; Habilita interrupção RX serial
  CLRF  PIE2      ; Limpa a variável PIE2
  BANK0
  MOVLW  90
  MOVWF  RCSTA      ; Serial habilitada, recepção continua, 8 bits
  MOVF  RCREG,W      ; Limpa registrador e flags de recepção

; Inicialização do TECLADO
  BANK1
  MOVLW   B'11110000'    ;configura PB5:7 como entradas
  MOVWF   TRISB      ;configura PB1:4 como saídas

  BANK0
  CLRF    PORTB      ; Limpa os bits do PORTB
  CLRF    PORTA      ; Limpa os bits do PORTA
  CLRF   POS        ; Limpa os bits da variável POS
  BSF     PORTA,3      ; Seta o terceiro bit do PORTA (xxxx.1xxx)
  BCF    FLAG,0      ; Limpa o bit 0 da variável FLAG

; Inicialização RELE E SOM E SENSOR
  BANK1
  MOVLW  B'00110000'
  MOVWF  TRISA      ; coloca o valor da variável W no TRISA
```

```
  MOVLW  B'00000111'    ; Porta A como sinais digitais
  MOVWF  ADCON1      ; coloca o valor da variável W no ADCON1
  BANK0

; Reservado espaço de memória para guardar as teclas pressionadas, de 0X20 a 0X2F, por isso cblock começa em 0X30, uma apos posição 0X2F
  MOVLW 0X20
  MOVWF FSR        ; coloca o valor da variável W no FSR

; Setando Propriedades do Display
  CALL  LP_10MS      ; Chama função LP_10MS , para aguardar 10 milisegundos

  MOVLW   28        ; Interface 4 bits, 2 linhas, caracter 5x7
  CALL  SEND_CMD    ; Chama função SEND_CMD para enviar comando ao display

  MOVLW  0C        ; Cursor desligado
  CALL  SEND_CMD    ; Chama função SEND_CMD para enviar comando ao display

  MOVLW  06        ; Deslocamento para a direita
  CALL  SEND_CMD    ; Chama função SEND_CMD para enviar comando ao display


  CALL INICIALIZA_RA

MAIN            ; MUDAR
  CALL  VERIFICA_TECLADO    ; Chama função para verificação do teclado (VERIFICA_TECLADO)
  GOTO MAIN

; ************ INICIO TECLADO ************
VERIFICA_TECLADO
  CLRF     Tecla         ; Variável que armazena a tecla pressionada
  INCF     Tecla,f    ; Incrementa variável Tecla e armazena nela mesma
  MOVLW    0Eh           ;
  MOVWF    PORTB      ; Habilita primeira linha para verificar tecla pressionada

  BTFSS   PORTB,4        ; Verifica se a tecla pressionada primeira linha e primeira coluna
  GOTO   VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
  INCF  Tecla,f      ; Incrementa a variável Tecla

  BTFSS   PORTB,5        ; Verifica se a tecla pressionada primeira linha e segunda coluna
  GOTO   VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
  INCF  Tecla,f      ; Incrementa a variável Tecla

  BTFSS   PORTB,6        ; Verifica se a tecla pressionada primeira linha e terceira coluna
```

```
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 MOVLW  0Dh          ;
 MOVWF  PORTB      ; Habilita segunda linha para verificar tecla pressionada

 BTFSS  PORTB,4       ; Verifica se a tecla pressionada segunda linha e primeira coluna
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 BTFSS  PORTB,5       ; Verifica se a tecla pressionada segunda linha e segunda coluna
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 BTFSS  PORTB,6       ; Verifica se a tecla pressionada segunda linha e terceira coluna
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 MOVLW  0Bh       ;
 MOVWF  PORTB      ; Habilita terceira linha para verificar tecla pressionada

 BTFSS  PORTB,4       ; Verifica se a tecla pressionada terceira linha e primeira coluna
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 BTFSS  PORTB,5       ; Verifica se a tecla pressionada terceira linha e segunda coluna
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 BTFSS  PORTB,6       ; Verifica se a tecla pressionada terceira linha e terceira coluna
 GOTO  VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
 INCF  Tecla,f     ; Incrementa a variável Tecla

 MOVLW  07h       ;
 MOVWF  PORTB      ; Habilita quarta linha para verificar tecla pressionada
```

```
  BTFSS   PORTB,4         ; Verifica se a tecla pressionada quarta linha e primeira coluna
  GOTO   VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
  INCF  Tecla,f      ; Incrementa a variável Tecla

  BTFSS   PORTB,5         ; Verifica se a tecla pressionada quarta linha e segunda coluna
  GOTO   VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
  INCF  Tecla,f      ; Incrementa a variável Tecla

  BTFSS  PORTB,6      ; Verifica se a tecla pressionada quarta linha e terceira coluna
  GOTO   VERIFICA_TECLA  ; Caso operação acima seja ok, vai para LABEL -> VERIFICA_TECLA
  INCF  Tecla,f      ; Incrementa a variável Tecla

; Verificou todas linhas e nenhuma tecla foi pressionada
  CLRF  Tecla      ; limpa valor da variável Tecla
  RETURN          ; return da chamada da função realizada rotina MAIN

VERIFICA_TECLA
  BTFSC   FLAG,0         ; Verifica se o bit 0 da variável FLAG, para guardar teclas pressionadas como RA ou Senha
  GOTO  GUARDA_SENHA    ; Vai para o LABEL -> GUARDA_SENHA

  MOVLW   d'8'      ; seta variável work para 8 (b'1000')
  SUBWF   POS,W       ; subtrai variável POS de 8, para verificação do STATUS na operação abaixo
  BTFSC   STATUS,Z     ; verifica resultado do status Z esta clear: se sim vai (goto) label SENHA
  GOTO   SENHA      ; vai para label Senha pois já foi teclado os 8 dígitos do RA
  CALL   LP_1MS

  MOVF    Tecla,w       ; coloca o valor da Tecla pressionada em W ; com quatro bits (xxxx????) ex: Tecla 1 = xxxx0001
  ADDLW   .48          ; faz um AND de W com .48 ou b(00110000) -> para termos o valor da tecla pressionada no valor ASCII
  CALL   TRATA_ULTIMALINHA
  MOVWF   INDF      ; coloca o valor de INDF em W
  INCF   FSR,F      ; incrementa o FSR com valor de W+1 -> desloca a memória para próximo ponto armazenando a Tecla pressionada
  CALL   SEND_CHAR    ; envia o conteúdo de W a função SEND_CHAR para enviar ao LCD
  INCF   POS,F        ; incrementa a variável POS para verificação da quantidade de teclas pressionadas

Espera1   BTFSS   PORTB,4
        GOTO    Espera1
Espera2   BTFSS   PORTB,5
```

```
        GOTO    Espera2
Espera3   BTFSS   PORTB,6
        GOTO    Espera3
  RETURN

GUARDA_SENHA
  MOVLW   d'8'    ; seta variável work para 8 (b'1000')
  SUBWF    POS,W     ; subtrai variável POS de 8, para verificação do STATUS na operação abaixo
  BTFSC    STATUS,Z  ; verifica resultado do status Z esta clear: se sim vai (goto) label ACABOU
  GOTO    ACABOU     ; vai para label ACABOU para informar que ja foram tecladas as informações do RA e Senha
  CALL   LP_1MS

  MOVF     Tecla,w     ; coloca o valor da Tecla pressionada em W ; com quatro bits (xxxx????) ex: Tecla 1 = xxxx0001
  ADDLW    .48        ; faz um AND de W com .48 ou b(00110000) -> para termos o valor da tecla pressionada no valor ASCII
  CALL   TRATA_ULTIMALINHA
  MOVWF   INDF    ; coloca o valor de INDF em W
  INCF    FSR,F     ; incrementa o FSR com valor de W+1 -> desloca a memória para próximo ponto armazenando a Tecla pressionada
  CALL    SEND_CHAR  ; envia o conteúdo de W a função SEND_CHAR para enviar ao LCD
  INCF   POS,F      ; incrementa a variável POS para verificação da quantidade de teclas pressionadas

Espera11  BTFSS   PORTB,4
     GOTO    Espera11
Espera22  BTFSS   PORTB,5
     GOTO    Espera22
Espera33  BTFSS   PORTB,6
     GOTO    Espera33
  RETURN

; Escreve SENHA no LCD
SENHA
  MOVLW   0xC0      ; Endereça a DDRAM do LCD para linha 2 e coluna 1 do display
  CALL  SEND_CMD
    MOVLW  'S'
  CALL  SEND_CHAR
    MOVLW   'E'
  CALL  SEND_CHAR
    MOVLW   'N'
  CALL  SEND_CHAR
    MOVLW   'H'
  CALL  SEND_CHAR
    MOVLW   'A'
  CALL  SEND_CHAR
```

```
   MOVLW   ':'
  CALL  SEND_CHAR
  CLRF   POS       ; limpa da variável POS para verificação da quantidade de teclas pressionadas (8 dígitos)
  BSF      FLAG,0      ; seta o bit 0 da variável FLAG para verificação se a tecla pressionada e referente a senha
RETURN

TRATA_ULTIMALINHA

  MOVWF  Tecla

  MOVLW   ':'          ; coloca o valor da tabela ASCII referente ao : em W
  SUBWF   Tecla,W     ; subtrai variável DADO mesmos Valor da variável W work
  BTFSC    STATUS,Z       ; verifica se o status Z esta clear: se sim vai (goto) label ASTERISCO
  CALL   ASTERISCO  ;

  MOVLW   ';'          ; coloca o valor da tabela ASCII referente ao ; em W
  SUBWF   Tecla,W     ; subtrai variável DADO mesmos Valor da variável W work
  BTFSC    STATUS,Z       ; verifica se o status Z esta clear: se sim vai (goto) label NUM_ZERO
  CALL   NUM_ZERO  ;

  MOVLW   '<'          ; coloca o valor da tabela ASCII referente ao < em W
  SUBWF   Tecla,W     ; subtrai variável DADO mesmos Valor da variável W work
  BTFSC    STATUS,Z       ; verifica se o status Z esta clear: se sim vai (goto) label CERQUILHA
  CALL   CERQUILHA  ;

  MOVF    Tecla,w    ; coloca o valor da variável Tecla em W
RETURN

ASTERISCO
  MOVLW   '*'          ; coloca o valor da tabela ASCII referente ao * em W
  MOVWF   Tecla    ; coloca o valor da variável W na variável Tecla
RETURN
NUM_ZERO
  MOVLW   '0'          ; coloca o valor da tabela ASCII referente ao 0 em W
  MOVWF   Tecla    ; coloca o valor da variável W na variável Tecla
RETURN
CERQUILHA
  MOVLW   '#'          ; coloca o valor da tabela ASCII referente ao # em W
  MOVWF   Tecla    ; coloca o valor da variável W na variável Tecla
RETURN
;    FIM  TECLADO

ACABOU
; ESCREVE VALIDANDO DADOS
  MOVLW   0xC0      ; Endereça a DDRAM do LCD para linha 2 e coluna 1 do display
```

```
  CALL  SEND_CMD
  MOVLW  'V'
  CALL  SEND_CHAR
  MOVLW  'A'
  CALL  SEND_CHAR
   MOVLW   'L'
  CALL  SEND_CHAR
   MOVLW   'I'
  CALL  SEND_CHAR
   MOVLW   'D'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   'N'
  CALL  SEND_CHAR
   MOVLW   'D'
  CALL  SEND_CHAR
   MOVLW   'O'
  CALL  SEND_CHAR
   MOVLW   ' '
  CALL  SEND_CHAR
   MOVLW   'D'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   'D'
  CALL  SEND_CHAR
   MOVLW   'O'
  CALL  SEND_CHAR
   MOVLW   'S'
  CALL  SEND_CHAR
   MOVLW   '.'
  CALL  SEND_CHAR

; ************ ENVIA DADOS SERIAL ************

    CALL  LP_1MS
  MOVLW  0X1F
  MOVWF   FSR         ; coloca o valor de W setado na instrução acima em FSR
(endereço da memória onde estão armazenadas as teclas pressionadas)

ENVIA_SERIAL
  INCF   FSR,F    ; incrementa o FSR com valor de W+1 -> desloca a memória para
próximo ponto armazenando a Tecla pressionada
  MOVF  INDF,W    ; coloca o valor de INDF em W
  MOVWF   TXREG     ; coloca o valor de W em TXREG (responsável pelo envio do
byte a serial)
  CALL   LP_2MS    ; aguarda 2 milisegundos
  MOVLW  0X30    ; coloca em W o valor 0x30 referente a ultima posição da memória
onde contem dados das teclas pressionadas
```

```
  XORWF  FSR,W    ; faz um XOR do valor de FSR com valor de W pra verificar se chegou ao final das teclas pressionadas
  BTFSS  STATUS,Z  ; verifica se o status Z (retornado da instrução acima) esta setado: se sim vai para label -> ENVIA_SERIAL
  GOTO  ENVIA_SERIAL
  MOVLW   0X20     ; coloca em W o valor 0x20 referente a primeira posição da memória onde são armazenados os dados das teclas pressionadas
  MOVWF   FSR      ; coloca o valor de W setado na instrução acima em FSR (endereço da memória onde estão armazenadas as teclas pressionadas)
  CALL   LP_2MS    ; aguarda 2 milisegundos
  MOVLW  '&'
  MOVWF  TXREG     ; Transmite caracter "&" para informar fim da transmissão das informações ao banco de dados no microcomputador

; ********** RECEBE DADOS DA PORTA SERIAL e ANALISA A RESPOSTA ENVIADA **********

  MOVLW   d'0'    ; coloca o valor de 0 na variável W
  MOVWF   TIMEOUTDB  ; coloca o valor de W na variável TIMEOUTDB

RECEBE_SERIAL
  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
  MOVLW   '1'     ; coloca o valor de 1 na variável W
  SUBWF   DADO,W      ; subtrai variável DADO do Valor da variável W work
  BTFSC   STATUS,Z  ; verifica se o status Z esta clear: se sim vai (goto) função VALIDACAO_OK
  GOTO   VALIDACAO_OK

  MOVLW   '2'     ; coloca o valor de 2 na variável W
  SUBWF   DADO,W      ; subtrai variável DADO do Valor da variável W work
  BTFSC   STATUS,Z  ; verifica se o status Z esta clear: se sim vai (goto) função USUARIO_INVALIDO
  GOTO   USUARIO_INVALIDO

  MOVLW   '3'     ; coloca o valor de 3 na variável W
  SUBWF   DADO,W      ; subtrai variável DADO do Valor da variável W work
  BTFSC   STATUS,Z  ; verifica se o status Z esta clear: se sim vai (goto) função SENHA_INVALIDA
  GOTO   SENHA_INVALIDA

  MOVLW   d'5'     ; coloca o valor de 5 na variável W
  SUBWF   TIMEOUTDB,W    ; subtrai variável TIMEOUTDB do Valor da variável W work
  BTFSC   STATUS,Z    ; verifica se o status Z esta clear: se sim vai (goto) função TENTE_DE_NOVO
  GOTO   TENTE_DE_NOVO
  INCF   TIMEOUTDB,F    ; incrementa o valor da variável TIMEOUTDB

  GOTO   RECEBE_SERIAL
```

```
; ********** ENVIA MENSAGEM AO DISPLAY CONFORME TRATADA NA ROTINA ACIMA **********

TENTE_DE_NOVO ; escreve TENTE DE NOVO , TIME OUT
  BSF    LED_VERMELHO  ; acende led vermelho
    MOVLW   01          ; coloca o valor 1 na variável W para ser enviado a função SEND_CMD para Apagar display
     CALL  SEND_CMD
    MOVLW  0x80      ; Endereça a DDRAM do LCD para linha 1 e coluna 1 do display
  CALL  SEND_CMD

    MOVLW   'T'
  CALL  SEND_CHAR
    MOVLW   'E'
  CALL  SEND_CHAR
    MOVLW   'N'
  CALL  SEND_CHAR
    MOVLW   'T'
  CALL  SEND_CHAR
    MOVLW   'E'
  CALL  SEND_CHAR
    MOVLW   ' '
  CALL  SEND_CHAR
    MOVLW   'D'
  CALL  SEND_CHAR
    MOVLW   'E'
  CALL  SEND_CHAR
    MOVLW   ' '
  CALL  SEND_CHAR
    MOVLW   'N'
  CALL  SEND_CHAR
    MOVLW   'O'
  CALL  SEND_CHAR
    MOVLW   'V'
  CALL  SEND_CHAR
    MOVLW   'O'
  CALL  SEND_CHAR

  MOVLW   0xC0      ; Endereça a DDRAM do LCD para linha 2 e coluna 1 do display
  CALL  SEND_CMD

    MOVLW   'T'
  CALL  SEND_CHAR
    MOVLW   'I'
  CALL  SEND_CHAR
    MOVLW   'M'
  CALL  SEND_CHAR
    MOVLW   'E'
  CALL  SEND_CHAR
```

```
   MOVLW   ' '
  CALL  SEND_CHAR
   MOVLW   'O'
  CALL  SEND_CHAR
   MOVLW   'U'
  CALL  SEND_CHAR
   MOVLW   'T'
  CALL  SEND_CHAR
   MOVLW   ' '
  CALL  SEND_CHAR

  CLRF  DADO          ; limpa da variável DADO

  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
GOTO INICIALIZA_RA


SENHA_INVALIDA ; escreve SENHA INVALIDA
  BSF    LED_VERMELHO  ; acende led vermelho
   MOVLW   01          ; coloca o valor 1 na variável W para ser enviado a função
SEND_CMD para Apagar display
    CALL  SEND_CMD
   MOVLW  0x80      ; Endereça a DDRAM do LCD para linha 1 e coluna 1 do display
  CALL  SEND_CMD

   MOVLW  'S'
  CALL  SEND_CHAR
   MOVLW   'E'
  CALL  SEND_CHAR
   MOVLW   'N'
  CALL  SEND_CHAR
   MOVLW   'H'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   ' '
  CALL  SEND_CHAR
   MOVLW   'I'
  CALL  SEND_CHAR
   MOVLW   'N'
  CALL  SEND_CHAR
   MOVLW   'V'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   'L'
  CALL  SEND_CHAR
   MOVLW   'I'
```

```
  CALL  SEND_CHAR
    MOVLW   'D'
  CALL  SEND_CHAR
    MOVLW   'A'
  CALL  SEND_CHAR

  CLRF  DADO          ; limpa da variável DADO

  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
GOTO INICIALIZA_RA


USUARIO_INVALIDO ; escreve USUARIO INVALIDO
  BSF    LED_VERMELHO  ; acende led vermelho
    MOVLW   01          ; coloca o valor 1 na variável W para ser enviado a função
SEND_CMD para Apagar display
     CALL  SEND_CMD
    MOVLW  0x80      ; Endereça a DDRAM do LCD para linha 1 e coluna 1 do display
  CALL  SEND_CMD

    MOVLW   'U'
  CALL  SEND_CHAR
    MOVLW   'S'
  CALL  SEND_CHAR
    MOVLW   'U'
  CALL  SEND_CHAR
    MOVLW   'A'
  CALL  SEND_CHAR
    MOVLW   'R'
  CALL  SEND_CHAR
    MOVLW   'I'
  CALL  SEND_CHAR
    MOVLW   'O'
  CALL  SEND_CHAR
    MOVLW   ' '
  CALL  SEND_CHAR
    MOVLW   'I'
  CALL  SEND_CHAR
    MOVLW   'N'
  CALL  SEND_CHAR
    MOVLW   'V'
  CALL  SEND_CHAR
    MOVLW   'A'
  CALL  SEND_CHAR
    MOVLW   'L'
  CALL  SEND_CHAR
    MOVLW   'I'
  CALL  SEND_CHAR
```

```
   MOVLW   'D'
  CALL  SEND_CHAR
   MOVLW   'O'
  CALL  SEND_CHAR

  CLRF  DADO          ; limpa da variável DADO

  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
  CALL   ESPERA_1SEG
GOTO INICIALIZA_RA

VALIDACAO_OK ; escreve VALIDACAO OK
  BSF    LED_VERDE    ; acende led verde
   MOVLW   01          ; coloca o valor 1 na variável W para ser enviado a função SEND_CMD para Apagar display
    CALL  SEND_CMD
   MOVLW  0x80     ; Endereça a DDRAM do LCD para linha 1 e coluna 1 do display
  CALL  SEND_CMD

   MOVLW   'V'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   'L'
  CALL  SEND_CHAR
   MOVLW   'I'
  CALL  SEND_CHAR
   MOVLW   'D'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   'C'
  CALL  SEND_CHAR
   MOVLW   'A'
  CALL  SEND_CHAR
   MOVLW   'O'
  CALL  SEND_CHAR
   MOVLW   ' '
  CALL  SEND_CHAR
   MOVLW   'O'
  CALL  SEND_CHAR
   MOVLW   'K'
  CALL  SEND_CHAR

  CLRF  DADO          ; limpa da variável DADO
  CALL   LP_2MS

; ************ ROTINA PARA ACIONAR O RELE QUE IRA ABRIR A FECHADURA ELETRICA ************
```

```
RELE
  BSF    PORTA_RELE    ; seta a porta para acionar o rele
  CALL   LP_100MS    ; fica 100 milisegundos alimentando o rele
  BCF    PORTA_RELE    ; clear na porta de acionamento rele
  CALL   ESPERA_5SEG

; ********** VERIFICA SE A PORTA ESTA ABERTA E ACIONA ALARME SONORO **********

SENSOR
  BTFSS   BOTAO_SENSOR    ; verifica se o SENSOR esta setado: caso esteja vai para rotina de Inicialização, se estiver aberta a mais de 5 segundos aciona alarme sonoro
  GOTO   INICIALIZA_RA

    MOVLW  .255    ; Nota SOL -> Freq. = 392 Hz
    MOVWF  FREQ
    CALL  SOM

  GOTO   SENSOR

SOM
    BSF    AF
    MOVF  FREQ,W
    CALL  LP_FREQ
    BCF    AF
    MOVF  FREQ,W
    GOTO  LP_FREQ

LP_FREQ
    MOVWF  TEMPO1
LOOP_FREQ
    NOP
    NOP
    DECFSZ  TEMPO1,F
    GOTO  LOOP_FREQ
RETURN

; ********** ROTINA DE INICIALIZAÇÃO (limpa variáveis utilizadas e escreve RA: no LCD) **********

INICIALIZA_RA
  BCF   BOTAO_SENSOR  ; coloca modo do Sensor como fechado
  BCF    PORTA_RELE    ; coloca porta do rele como clear (rele sem alimentação)
  BSF    LED_VERMELHO  ; apaga led_vermelho
  BSF    LED_VERDE    ; apaga led_verde
  BCF    FLAG,0      ; seta bit 0 da variável flag para 0
  MOVLW  01        ; apaga display
  CALL  SEND_CMD
```

```
; Escrevendo no display (R.A.:)
    MOVLW  0x80      ; Endereça a DDRAM do LCD para linha 1 e coluna 1 do display
  CALL  SEND_CMD
    MOVLW  'R'
  CALL  SEND_CHAR
    MOVLW   '.'
  CALL  SEND_CHAR
  MOVLW   'A'
  CALL  SEND_CHAR
    MOVLW   '.'
  CALL  SEND_CHAR
    MOVLW   ':'
  CALL  SEND_CHAR
  CLRF   POS
  CLRF   DADO
RETURN

; *************** ENVIA COMANDOS e DADOS p/ LCD *****************

; -> Envia comando para o LCD
SEND_CMD
  MOVWF  CHAR    ; Salva comando
  BCF     RS       ; Coloca o LCD em modo comando
  GOTO  ENVIA

; -> Envia dado para o LCD
SEND_CHAR
  MOVWF  CHAR    ; Salva caracter
  BSF     RS       ; Coloca o LCD em modo dados
ENVIA
  MOVLW  0F0
  ANDWF  PORTD,F
  SWAPF  CHAR,W
  ANDLW  0F      ; Nibble mais significativo
  IORWF  PORTD,F

  BSF    ENB     ; Gera sinal para o LCD
  NOP
  NOP
  BCF    ENB

  MOVLW  0F0
  ANDWF  PORTD,F
  MOVF  CHAR,W
  ANDLW  0F      ; Nibble menos significativo
  IORWF  PORTD,F

  BSF    ENB     ; Gera sinal para o LCD
  NOP
```

```
  NOP
  BCF    ENB
  GOTO  LP_2MS

; Laços de tempo
ESPERA_5SEG
   CALL ESPERA_1SEG
   CALL ESPERA_1SEG
   CALL ESPERA_1SEG
   CALL ESPERA_1SEG
   CALL ESPERA_1SEG
RETURN

ESPERA_1SEG
   CALL LP_250MS
   CALL LP_250MS
   CALL LP_250MS
   CALL LP_250MS
RETURN

LP_250MS  MOVLW  .250
     MOVWF  TEMPO2
     GOTO  LP_1MS
LP_100MS  MOVLW  .100
     MOVWF  TEMPO2
     GOTO  LP_1MS
LP_10MS    MOVLW  .10
     MOVWF  TEMPO2
     GOTO  LP_1MS
LP_2MS    MOVLW  .2
     MOVWF  TEMPO2
LP_1MS    MOVLW  .250
     MOVWF  TEMPO1
LOOP    NOP
     DECFSZ  TEMPO1,F
     GOTO  LOOP
     DECFSZ  TEMPO2,F
     GOTO  LP_1MS
RETURN

END
```

APÊNDICE II – CÓDIGO DO BANCO DE DADOS

'Modulo responsável pela comunicação Serial:

```
Option Explicit
Dim MsComm1 As New MsComm

Public Sub Abre_Conexao(n As Boolean)
    On Error Resume Next
    If n = True Then
        MsComm1.CommPort = 9
        MsComm1.Settings = "9600,N,8,1"
        MsComm1.PortOpen = True
    Else
        MsComm1.PortOpen = False
    End If
    On Error GoTo 0
End Sub

Public Sub EnviaTxt(txt As String)
    Dim letra As String
    'Envia o texto para a porta COM
    If MsComm1.PortOpen = True Then
        Do Until Len(txt) = 0
            letra = Left(txt, 1)
            txt = Right(txt, Len(txt) - 1)
            MsComm1.Output = letra + Chr(13)
        Loop
    Else
        MsgBox "É necessário fazer a conexão primeiro.", vbInformation, "Conexão"
    End If
End Sub

Public Function Recebe()
    Dim var As String, cod As String, pwd As String
    'Recebe informações da porta COM
    var = Left(MsComm1.Input, 17)
    If Right(var, 1) = "&" Then
        cod = Left(var, 8)
        pwd = Mid(var, 9, 16)
    End If
    Controle.Verifica cod, pwd
    MsgBox cod & " - " & pwd, vbInformation, ""
End Function
```

```
'Modulo de validação das informações:

Option Explicit
Option Compare Database

Public Sub Verifica(cod As String, pwd As String)
   Dim SQL As String, rs As Recordset, db As Database
   Dim rsCon As Recordset
   On Error GoTo Error
   If pwd = "" Then pwd = InputBox("Senha", "")
   Set db = CurrentDb()
   SQL = "SELECT Codigo, Senha FROM Funcionario WHERE Codigo = " & cod & "
AND Senha = '" & pwd & "'"
   Set rs = db.OpenRecordset(SQL)
   If Not IsNull(rs("Codigo")) Then
      SQL = "SELECT Codigo, Data_Entrada, Data_Saida " _
         & "FROM Controle " _
         & "WHERE isnull(Data_Saida) and Codigo = " & rs("Codigo") & ""
      Set rsCon = db.OpenRecordset(SQL)
      If IsNull(rsCon("Data_Saida")) Then
         rsCon.Edit
         rsCon("Data_Saida") = Now
         rsCon.Update
      Else
         rsCon.AddNew
         rsCon("Codigo") = cod
         rsCon("Data_entrada") = Now
         rsCon.Update
      End If
   End If
   Serial.EnviaTxt "1"
   Set rs = Nothing
   Set rsCon = Nothing
   Set db = Nothing
   Form_Entrada_Saida.Requery
   Exit Sub
Error:
   If Err.Number = 3021 Then
      Serial.EnviaTxt "2"
   Else
      MsgBox Err.Number & " - " & Err.Description
   End If
   Set rs = Nothing
   Set rsCon = Nothing
   Set db = Nothing
End Sub
```