

**UNIVERSIDADE LUTERANA DO BRASIL
PRÓ-REITORIA DE GRADUAÇÃO
CURSO DE ENGENHARIA ELÉTRICA**

**AUTOMAÇÃO, MONITORAMENTO E CONTROLE DE UMA
RESIDÊNCIA ATRAVÉS DO USO DE UM
MICROCONTROLADOR**

ALEXSANDRO SILVEIRA FLORES

Canoas, novembro de 2005.

ALEXSANDRO SILVEIRA FLORES

Matrícula nº 982102534-0

**AUTOMAÇÃO, MONITORAMENTO E CONTROLE DE UMA
RESIDÊNCIA ATRAVÉS DO USO DE UM
MICROCONTROLADOR**

**Trabalho de Conclusão de Curso
apresentado como requisito para a
obtenção do grau de Engenheiro
Eletricista pela Universidade
Luterana do Brasil – Campus Canoas
– Curso de Engenharia Elétrica.**

Orientador: Prof. M.E. Augusto Alexandre Durganti de Mattos

Canoas, novembro de 2005.

ALEXSANDRO SILVEIRA FLORES

Matrícula nº 982102534-0

**AUTOMAÇÃO, MONITORAMENTO E CONTROLE DE UMA
RESIDÊNCIA ATRAVÉS DO USO DE UM
MICROCONTROLADOR**

BANCA EXAMINADORA:

Prof. M.E. Dalton Luiz Rech Viddor

Prof^ª. M.E. Mirian Cárceres Villamayor

Orientador: Prof. M.E. Augusto Alexandre Durganti de Mattos

Trabalho apresentado e aprovado em:

“... quando fizermos algo, devemos procurar fazê-lo da melhor maneira possível, ou
então não fazê-lo...”

Ayrton Senna da Silva

RESUMO

O conceito de Casa Inteligente originou-se quando o computador pessoal, o modem, o fax e a Internet mudaram não somente a forma de se trabalhar, mas também o local onde se trabalha. Os negócios puderam ser conduzidos através de um SOHO (*Small Office/Home Office*), tão facilmente, como se fossem de um grande escritório. Para profissionais que trabalham em casa, o acesso às facilidades acaba sendo uma necessidade. Para moradores de casas, apartamentos ou condomínios, que desejam segurança, conforto, entre outros benefícios, muitas soluções de automação estão disponíveis. O segredo está nas conectividades corretas, onde todos os dispositivos eletro-eletrônicos devem ser totalmente dependentes e modernos, assim como os de um escritório futurista.

Sendo assim, o presente projeto implementa um sistema de controle e monitoramento de uma residência utilizando um microcontrolador/PC. Através de um micro computador, dotado de um software próprio, têm-se condições de controlar, monitorar e gerenciar as atividades, os aparelhos e os equipamentos que possam estar conectados a essa residência. Porém, cabe ressaltar que o sistema existente poderá funcionar de uma forma autônoma e independente.

Palavras-chave: automação; monitoramento; controle.

ABSTRACT

The conception of an intelligent house started when the personal computer, the modem, fax and Internet changed both, the manner of our work and work environment. The business could be making beyond a SOHO (Small Office/Home Office), as easy as it was a large office. To a professional that work at home, this type of facilities are a necessity. To whom resides in apartments, houses or housing estates and want security, comfort, and the like, different automation solutions are available. The secret is in to implement the right connections, in which all electro-electronic devices might be totally dependents and moderns, such as in a futuristic office.

In such case, the present project Develop a system to control and monitoring a house by using a microcontroler/PC. Making use of a computer and it's own software, it will be possible to control, check and manage activities, devices and equipments that are connected in this house. However, is important to say that the current system could is able to work in an autonomous and independent way.

Key words: automation; monitoring; control.

SUMÁRIO

INTRODUÇÃO.....	12
2 CONCEITOS BÁSICOS.....	14
2.1 Automação Residencial.....	14
2.2 <i>Borland Delphi 5</i>	16
2.3 Kit Microcontrolador ULBEE51.....	19
3 MAQUETE E DESCRIÇÃO DO SISTEMA MICROCONTROLADO.....	21
3.1 Montagem da maquete.....	21
3.2 Sensores utilizados e interligação com a placa I/O.....	22
4 SOFTWARE DE CONTROLE E INTERAÇÃO.....	28
CONCLUSÃO.....	35
REFERÊNCIAS BIBLIOGRÁFICAS.....	37
ANEXO 1.....	39
ANEXO 2.....	40
ANEXO 3.....	41

ANEXO 4.....	42
ANEXO 5.....	45
ANEXO 6.....	61

LISTA DE FIGURAS

Figura nº 1: acionamento de <i>strobe</i>	19
Figura nº 2: diagrama em blocos do sistema implementado.....	21
Figura nº 3: planta baixa da residência.....	22
Figura nº 4: maquete com sensores e acionadores instalados.....	23
Figura nº 5: placa I/O conectada ao Kit ULBEE51.....	24
Figura nº 6: sistema de Automação Residencial.....	24
Figura nº 7: interface de interação usuário/maquete.....	30
Figura nº 8: sensores magnéticos da sala de estar e dormitório.....	31
Figura nº 9: ímãs nas aberturas das janelas da sala de estar e dormitório.....	32
Figura nº 10: sensor de presença da cozinha em estado “Ligado”.....	32
Figura nº 11: sensor IVP acionado na cozinha.....	33

Figura nº 12: temperatura apresentada na tela do PC.....	33
Figura nº13: sensor e equipamento que fornecem controle sobre a temperatura.....	34
Figura nº14: iluminação externa e o ar condicionado ligados.....	34

LISTA DE ABREVIATURAS E SIGLAS

A/D: conversor analógico/digital

A/V: áudio e vídeo

CD: *Compact Disc*

CFTV: circuito fechado de televisão

EPROM: *Erasable Programmable Really Only Memory*

GUI: *Graphics User Interface*

I/O: *input/output*

LDR: *Light Dependent Resistor*

POO: Programação Orientada a Objeto

RAM: *Random Acces Memory*

SOHO: *Small Office/Home Office*

1. INTRODUÇÃO

Sabendo que a sociedade enfrenta, nos dias de hoje, dificuldades em relação à falta de tempo, deixando de lado necessidades básicas para um bom convívio familiar e social, como o lazer, muitas têm procurado adaptar algumas tarefas que necessitam de paciência e organização. Em função disso, buscam uma residência controlada e ao mesmo tempo autônoma em suas propriedades.

A implementação deste projeto oferece a possibilidade de automação em uma residência, de uma maneira mais acessível financeiramente do que as soluções existentes hoje no mercado. Também possibilita aos usuários domésticos o controle e o monitoramento das principais tarefas do cotidiano, que demandam maior disponibilidade de tempo, tais como: o controle da climatização das dependências da residência, o acionamento da iluminação e o monitoramento dos sistemas de segurança.

A elaboração deste projeto visa mais conforto e comodidade às pessoas em seus ambientes residenciais. É importante salientar que o controle e o monitoramento de uma residência proporcionam economia de energia e consequentemente uma redução nos custos do orçamento familiar. Quem optar em possuir uma casa inteligente, também poderá usufruir de uma maior e melhor segurança. Quando se trabalha para a implementação de um projeto como este, busca-se de uma forma sistemática a realização de determinadas tarefas e funções que definirão o perfeito funcionamento desse sistema.

O projeto será capaz de realizar medidas e aquisições de grandezas físicas, como temperatura e presença física. Também irá monitorar as condições ambientais nas dependências e ainda controlar os equipamentos e os aparelhos existentes na residência. Isso será possível através da utilização de um Kit microcontrolador que estará conectado, via canal serial com um PC, que por sua vez possuirá um software de interação entre o usuário e a maquete.

No próximo capítulo tem-se um breve conceito de Automação Residencial, de software *Borland Delphi 5* e do Kit microcontrolador utilizado. No capítulo seguinte segue a descrição do sistema microcontrolado e da montagem da maquete, contendo o *hardware* e os sensores instalados. E por fim no capítulo 4, constam os *softwares* de controle e de interação desenvolvidos para o sistema de automação.

2. CONCEITOS BÁSICOS

2.1 Automação Residencial

A automação residencial utiliza diferentes métodos ou equipamentos, sendo um sistema integrado: telecomunicações, segurança, iluminação, entretenimento, ar condicionado, e muito mais. Trata-se de um sistema inteligente, programável e centralizado, podendo ser controlado e adaptado pelo usuário.

Em uma residência convencional, os aparelhos eletro-eletrônicos possuem seus controles de forma independente, que ao serem manuseados, geram uma certa confusão, acarretando numa ineficiência de aplicação, tornando sua potencialidade real inexplorada. Numa automação residencial, o primeiro benefício que surge é o de integração, que se resume em automatizar os equipamentos que fazem parte do sistema. Um exemplo seria: o proprietário ao sair de sua residência, ativaria a função “Fora de Casa” e com isso, o sistema de segurança estaria acionado, lâmpadas desnecessárias seriam desligadas, o ar condicionado funcionaria em regime de baixo consumo, a secretária eletrônica entraria em funcionamento, uma seqüência de apagar e ascender luzes seria ativada, simulando a presença de moradores, etc.

Instantes antes do retorno à sua propriedade, o morador faria uma ligação telefônica para a sua residência, digitaria um código de acesso, onde ele programaria o sistema para recebê-lo. Com isso, ao chegar em casa, lâmpadas já estariam acesas, o café já estaria preparado, a temperatura do ar condicionado adequada, etc.

Sendo assim, é importante destacar que o grau de automação dependerá sempre do sistema a ser implantado, podendo se ter soluções simples, com baixo custo e soluções complexas, com alto custo de investimento.

2.1.1 O que pode ser automatizado em uma residência?

Em uma residência, o que pode ser automatizado está dividido em subsistemas como: os Sistemas de Automação, onde estão inseridos os controles do ar (aquecimento/resfriamento, umidade, ar quente e termostato), os controles de dispositivos elétricos (iluminação, aparelhos domésticos, portas e portões, persianas e cortinas, bombas e filtros, ventiladores, irrigadores de jardins, portões de garagem, piscinas e sistema de abastecimento de água), os controles de ambientes (televisão, CFTV – circuito fechado de televisão, som, CD, VCR, *home theater*, *multi-room* A/V, A/V externo e DVD) e o gerenciamento de energia (monitoramento e controle do consumo de energia e o controle remoto da iluminação da casa).

Um outro subsistema importante é o definido como o Sistema de Segurança Residencial, onde constam os alarmes (sensor de movimento – infravermelho, sensor de vibração ou impacto, sensor de entrada/saída de veículos, controle remoto, sensor de vazamento de gás, sensor de fumaça, sensor de calor e sensores magnéticos localizados em portas e janelas). Tem-se também o monitoramento com câmeras (áreas internas, entrada principal, área do bebê, etc.) que podem ser acessadas através da Internet, com inspeções em áreas externas, gravação através de VCR ou via Internet, digitalmente. E finalmente, tem-se a segurança (com luzes de emergência, serviço de monitoramento, controle remoto, botões de socorro e chamada telefônica).

Também destaca-se o subsistema definido como Sistemas de Computador/Internet com o auxílio das comunicações (telefone, correio de voz, ramais internos, correio eletrônico, fax, identificador de chamadas telefônicas, registrador de chamadas telefônicas, secretária eletrônica e videoconferência), a informática (multimídia, TV, receitas, lista de compras, pagamentos de contas, banco de dados, agenda e registros de médicos), e a Internet (correio eletrônico, jornal, rádio, investimentos e etc.).

Com o avanço da ciência e a descoberta de novas tecnologias, novos equipamentos e serviços, que surgem quase que diariamente, a automação da maioria dos subsistemas exemplificados já está disponível para as residências e facilmente poderá vir a ser aplicada [6].

2.2 Borland Delphi 5

Programar em *Windows* sempre foi algo extremamente complicado e acessível apenas a programadores dispostos a investir muito tempo na leitura de livros, em intermináveis testes e análise de programas exemplos que mais confundem do que explicam. Mas porque é tão difícil fazer programas para *Windows*? Para começar, o *Windows* usa o conceito de GUI (*Graphics User Interface*), que embora seja muito familiar para usuários do *Unix*, é novidade para usuários do DOS. O uso de um sistema GUI implica em aprender vários conceitos que são estranhos ao usuário de um sistema baseado em texto como o DOS. Para complicar um pouco mais, o *Windows* é um sistema multitarefa, e as aplicações são orientadas a eventos, o que implica em aprender um novo estilo de programação.

Finalmente, o programador tinha que ter alguma familiaridade com as centenas de funções oferecidas pela API do *Windows*. Por tudo isso, programação em *Windows* é um assunto que costuma provocar desconforto nos programadores.

Felizmente as linguagens visuais chegaram para mudar esta situação. Foi só com elas que o *Windows* conseguiu cumprir sua promessa de ser um sistema amigável e fácil de usar também para os programadores, que sempre tiveram que pagar a conta da facilidade de uso para o usuário.

Entre as linguagens visuais que surgiram, poucas são tão completas quanto a linguagem *Delphi*. Esta vem com um compilador capaz de gerar código diretamente executável pelo *Windows*, proporcionando uma velocidade de execução de 5 à 20 vezes maior que as linguagens interpretadas. Além disso, vem também com um gerenciador de banco de dados completo e um gerador de relatórios. O tempo de

desenvolvimento de qualquer sistema é reduzido a uma fração do tempo que seria necessário usando outras linguagens e o resultado é sempre muito melhor. É por isso que o *Delphi* faz tanto sucesso no mundo inteiro.

2.2.1 Programação Orientada a Objeto (POO)

Embora não seja objetivo deste trabalho comentar a respeito de POO, uma breve introdução é necessária, já que o *Delphi* é essencialmente orientado a objeto. De maneira prática, pode-se pensar no objeto sendo uma estrutura que agrupa dados e funções para manipulação dos mesmos. Como as funções são sempre “íntimas” dos dados, o sistema todo funciona de maneira mais segura e confiável. Além disso, a POO utiliza conceitos como encapsulamento e herança que facilitam muito a programação e a manutenção dos programas.

Neste ponto é oportuno citar que os dados de um objeto costumam ser chamados de variáveis de instância e as funções de métodos. As variáveis de instância definem as propriedades (às vezes chamadas de atributos) do objeto e os métodos definem seu comportamento.

2.2.1.1 Encapsulamento

Como as variáveis e métodos estão na mesma estrutura, pode-se pensar em variáveis e métodos privados, ou seja, dados e funções que só podem ser manipulados pelas funções que estão dentro da estrutura. Desta maneira, é possível formar uma camada protetora nos dados e evitar atribuições desastradas que comprometeriam o funcionamento do programa. Os defensores mais ortodoxos da POO dizem que todos os dados de um objeto deveriam ser privados e o número de funções públicas deveriam ser o menor possível, mas isso nem sempre é viável ou prático. O *Delphi* implementa este conceito e oferece dados/funções públicas (*public*) e privadas (*private*). Outra consequência do encapsulamento é que os objetos podem ser “caixas pretas”. Não é necessário, teoricamente, conhecer detalhes de funcionamento de um objeto para usá-lo, basta enviar as mensagens apropriadas que ele responde com a ação desejada.

2.2.1.2 Classes

A classe representa um tipo ou categoria de objetos, o modelo a partir do qual um objeto pode ser construído. É a estrutura propriamente dita, que define os dados e métodos daquela classe de objetos. O objeto em si é uma instância da classe. Na programação estruturada, pode-se fazer uma analogia com os tipos e variáveis, onde a classe equivale ao tipo e o objeto à variável desse tipo.

2.2.1.3 Herança

É a capacidade que uma classe de objetos tem de herdar variáveis e métodos de outra classe. Esta capacidade permite que o código já escrito seja reutilizado de maneira muito mais eficiente e simples do que na programação estruturada. Um programa orientado a objeto costuma implementar verdadeiras árvores genealógicas de classes, com vários níveis de herança.

Para programar no nível do *designer* não é necessário um conhecimento profundo de POO, mas é preciso conhecer pelo menos a sintaxe. Todas as aplicações para *Windows* precisam de pelo menos uma janela, que no *Delphi* é chamada de *Form*. Cada *form* (assim como todos os objetos visuais) tem um objeto associado a ele e sua representação visual, que se vê na tela. Todos os componentes que se incluem no *form* passam a fazer parte do objeto que define o *form*. Exemplo: coloca-se um botão no *form*, a classe deste *form* será modificada para incluir este botão. Os eventos e métodos deste *form*, também estão na classe. Assim, supondo que o *form* se chame *Form1* (nome default), para desativar o botão que incluiu-se (de nome *Button1*) escreve-se:

```
Form1.Button1.Enabled := false;
```

Note como o *Button1* faz parte da estrutura que define o *form*. Mas se quiser ativar método *RePaint* (Repintar) do *form* se faz:

```
Form1.Repaint;
```

Veja que *Repaint* não é uma variável, tecnicamente é uma *procedure*, mas faz-se referência a ela como parte da estrutura *Form1*. Pode parecer confuso, mas facilita muito a programação [3].

2.3 Kit Microcontrolador ULBEE51

O Kit microcontrolador utilizado, foi desenvolvido na disciplina de Microprocessadores do Curso de Engenharia Elétrica da ULBRA (Universidade Luterana do Brasil). Montado sobre uma placa, conforme esquemático apresentado no anexo 3, o Kit deste trabalho tem como processador central o AT89C52, produzido pela ATMEL.

Este Kit possui 64 Kbytes de memória RAM e 64 Kbytes de memória EPROM. Da maneira como ele é exposto, disponibiliza 16 sinais de habilitação (*strobe*) mapeados em memória, tendo a sua saída em constante nível lógico alto (5V) até que um determinado programa acione um destes sinais, disponibilizando assim, nível lógico baixo (0V) na saída, retornando imediatamente a um nível lógico alto. A figura abaixo apresenta uma disposição de como o sistema de sinais funciona.

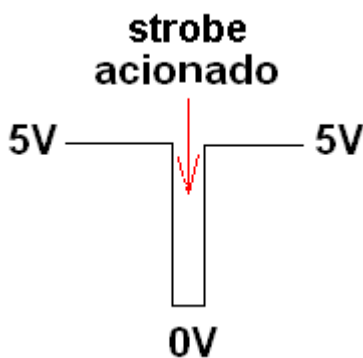


Figura 1 - Instante em que um *strobe* é acionado.

O Kit em uso apresenta um mapeamento para a memória externa, cuja função é atuar como entrada e saída de dados, contendo 8 *bits*. Esta flexibilidade de entrada e saída de dados é devido à montagem e configuração do Kit apresentado. Em

relação à memória RAM externa, a mesma possui 32 Kbytes e serve para salvar os dados fornecidos.

É possível concluir que pode-se controlar componentes eletrônicos, como é o caso dos conversores analógico/digital, bem como os *latches* que serão utilizados no decorrer deste trabalho. Uma vez que, através de instruções, queira-se escrever em alguma saída um dado binário, é possível acionar um *strobe* para controlar o tempo em que o dado deve ser disponibilizado e o fazer via mapeamento de memória.

Também é possível que estes dados sejam manipulados, interna ou externamente ao microprocessador, a fim de que sejam disponibilizados dados de acordo com o esperado. O Kit apresenta porta de entrada e saída de dados binários, cujo acesso é direto e com funções específicas, que não possuem mapeamento em memória. Este recurso não será utilizado neste trabalho, contudo faz-se importante para aplicações futuras e de adaptação de variáveis de entrada e saída [9].

3. MAQUETE E DESCRIÇÃO DO SISTEMA MICROCONTROLADO

O sistema de automação da residência está sendo demonstrado em forma de diagrama de blocos, na figura abaixo:

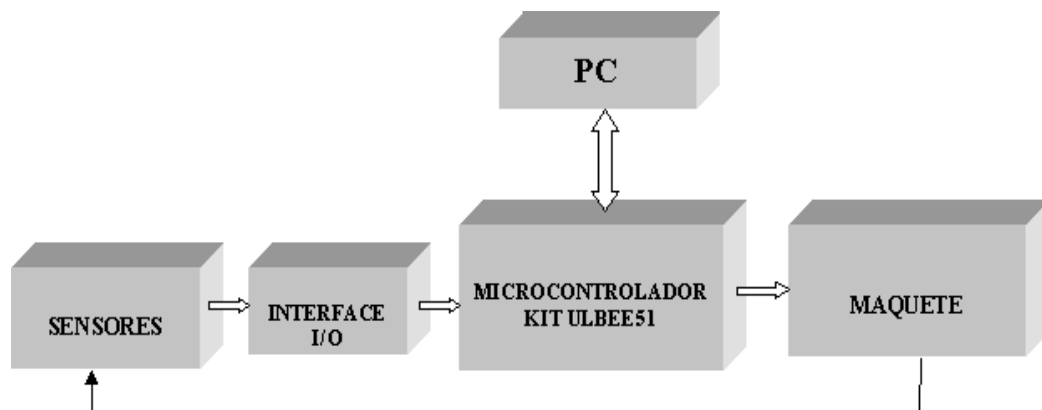


Figura 2 – Diagrama em blocos do sistema implementado.

3.1 Montagem da maquete

Primeiramente foi efetuada a confecção de uma maquete, para que se pudesse simular uma residência. Essa maquete foi montada conforme a planta baixa mostrada pela figura a seguir:



Figura 3 – Planta baixa da residência.

Para a construção dessa maquete, foi utilizado um material chamado de MDF e uma cola apropriada. Construiu-se uma maquete com 8 ambientes bem definidos. Também foi realizada a construção de um compartimento inferior, para que se pudesse inserir as placas eletrônicas que serão descritas no decorrer deste trabalho.

3.2 Sensores utilizados e interligação com a placa I/O

Com a maquete da residência já montada, foi iniciado o processo de instalação dos sensores, acionadores e componentes que seriam fixados conforme a descrição da tabela abaixo:

Descrição	Dependência	I/O	Quant.	Local
sensor magnético	garagem	entrada digital	1	porta
sensor de presença	garagem	entrada digital	1	frontal à porta
sensor magnético	área de serviço	entrada digital	1	janela
sensor de presença	área de serviço	entrada digital	1	frontal à janela
sensor magnético	cozinha	entrada digital	1	janela
sensor de presença	cozinha	entrada digital	1	próximo à janela
sensor de presença	sala de jantar/estar	entrada digital	1	próximo à janela
sensor magnético	sala de jantar/estar	entrada digital	2	porta e janela
acionamento do ar	sala de jantar/estar	saída digital	1	próximo à porta
sensor de temperatura	sala de jantar/estar	entrada analógica	1	centro da área
acionamento da iluminação	sala de jantar/estar	saída digital	1	no interior da área
sensor de presença	dormitórios	entrada digital	2	frontal às janelas
sensor magnético	dormitórios	entrada digital	2	janelas
sensor de presença	banheiro	entrada digital	1	frontal à janela
sensor magnético	banheiro	entrada digital	1	janela
acionamento da iluminação	área externa	saída digital	2	ao redor da residência

Os sensores utilizados foram: sensores de presença (infravermelho passivo) configuração *pull down*; sensores magnéticos tipo *reed-switch*, também na configuração *pull down*; sensor de temperatura LM35; *leds* para o sistema de iluminação e o *cooler*, que foi usado com o intuito de simular um aparelho de ar condicionado. Ainda poderiam ter sido acrescentados outros sensores como: sensor de luminosidade LDR, sensor de presença com ultra-som, sensor de umidade, detector de fumaça óptico, detector de calor termovelocimétrico e etc.

A estrutura da maquete, assim como a instalação dos sensores, podem ser vistos na figura abaixo:



Figura 4 - Maquete com sensores e acionadores instalados.

Para a interligação desses sensores, foi necessária a montagem de um circuito em uma placa padrão, contendo *latches* de entrada, *latches* de saída e um conversor A/D com o seu respectivo circuito de *clock* (ver anexo 1), usado para o correto funcionamento do conversor analógico digital (ver no anexo 2 o esquemático do circuito de I/O) [4]. Esse A/D foi utilizado para medir os valores de temperatura existentes em uma dependência da residência. Esses valores serão utilizados para um posterior gerenciamento de conforto para o ambiente.

Essa placa I/O foi conectada ao Kit microcontrolador, através dos conectores de entrada e saída de dados, assim como o conector de *strobe*. A interligação desta placa ao Kit, assim como o agrupamento das mesmas na maquete da residência, podem ser vistas na figura abaixo:

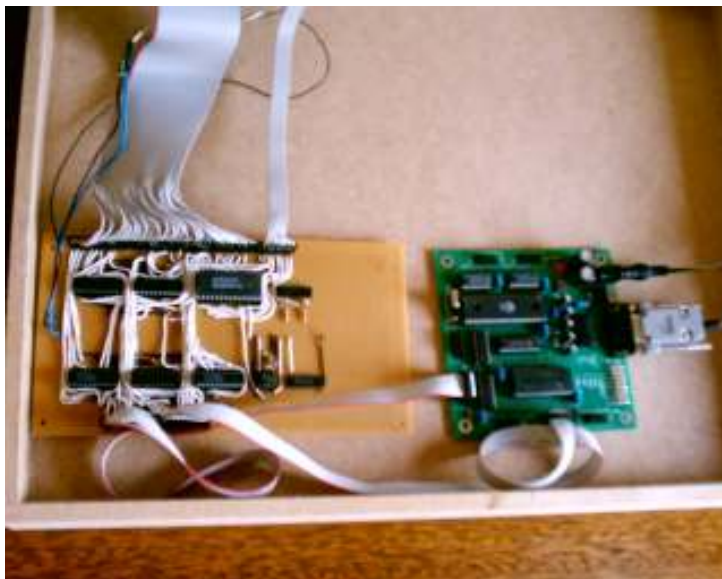


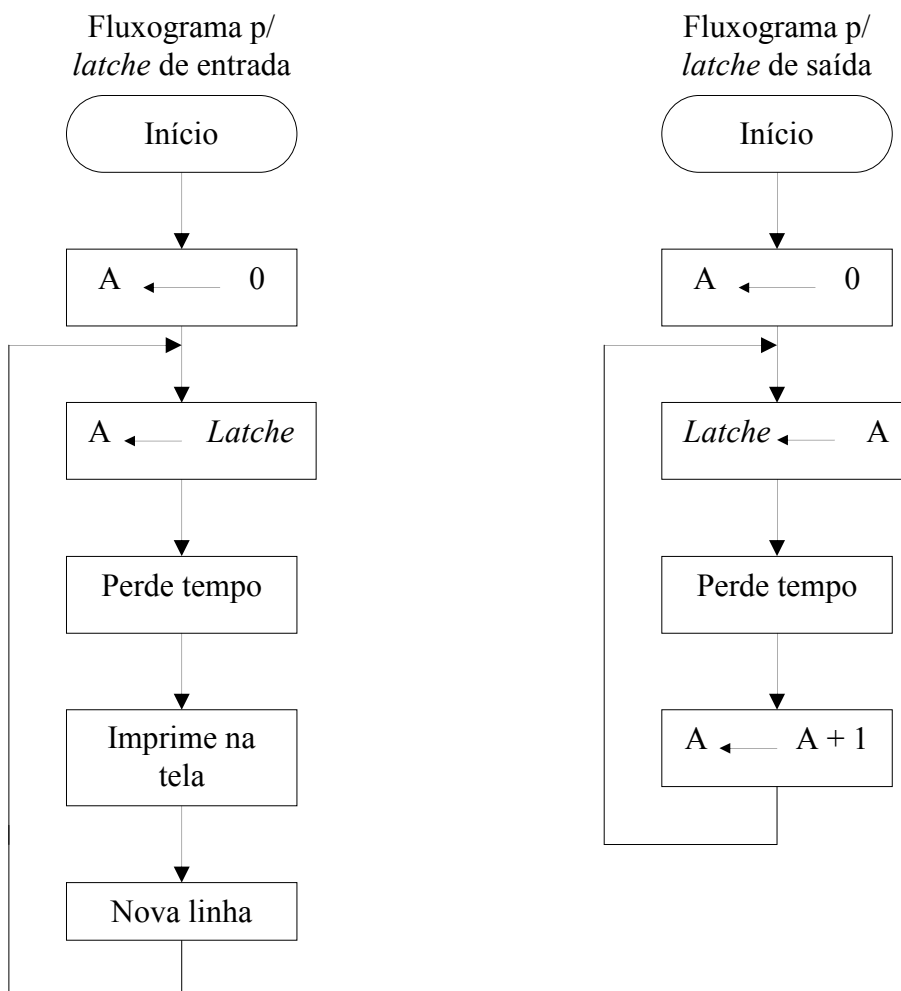
Figura 5 - Placa de I/O conectada ao Kit ULBEE51.

O sistema de automação da maquete será alimentado por uma fonte de energia simétrica, de tensão contínua (5V, 12 V e -12 V). Na figura abaixo, pode-se observar o sistema montado:



Figura 6 – Montagem completa do sistema de Automação Residencial proposto.

Com a intenção de realizar testes e comprovar o correto funcionamento do circuito citado acima, foi realizado um pequeno programa no compilador MIDE51, que utiliza como base a linguagem de programação em “C” (ver abaixo o fluxograma, bem como no anexo 4 o código fonte do mesmo) [12].



3.2.1 Teste dos sensores magnéticos

Através de um circuito de *reed-switch* na configuração *pull down*, ligou-se os sensores magnéticos no *latche* de entrada 1. Esse *latche* está localizado na placa I/O, que conforme foi visto anteriormente, está ilustrada no anexo 2.

Compilou-se no software MIDE51 um programa de teste que apresenta na tela do computador o *status* do sensor magnético utilizado. Esse pequeno programa,

após ter sido compilado, foi transferido via *Hyper Terminal* para o Kit do microcontrolador.

Ao aproximar-se os ímãs dos sensores magnéticos, verificou-se na tela a condição de 255 em decimal para todas as portas e janelas fechadas e de 0 em decimal, pois utilizava-se todas as entradas do *latche*, para simular a abertura das portas e janelas da residência. Os testes foram feitos com todas as combinações possíveis de janelas e portas abertas ou fechadas, demonstrando o correto funcionamento dos sensores.

3.2.2 Teste dos sensores de presença

A configuração *pull down* também foi utilizada para conectar os sensores de presença no *latche* de entrada 2. A localização desse *latche* na placa de I/O, está ilustrada no anexo 2.

Compilou-se no software MIDE51 um programa de teste que apresenta na tela do computador a presença ou não de uma condição física. Esse programa, após ter sido compilado, foi transferido via *Hyper Terminal* para o Kit do microcontrolador. O programa de teste citado acima foi o mesmo utilizado na verificação dos sensores magnéticos, porém foi alterado o *strobe* de acionamento do *latche*.

Após a execução desse programa no Kit, verificou-se na tela do PC a condição de 255, definido em números decimais, para a não presença física em nenhum dos sensores IVP's. E para a presença física em todos os sensores de presença, 0 em decimal. Foram feitos os testes com todas as combinações possíveis de presença e não presença física dentro da residência, mostrando o correto funcionamento desses sensores.

3.2.3 Teste dos acionadores do sistema de iluminação

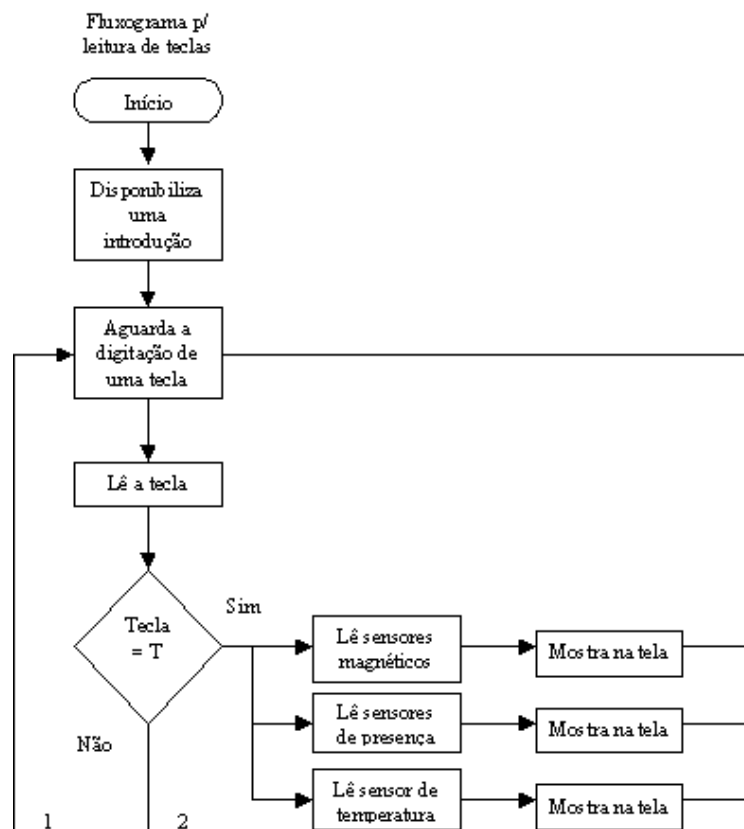
Em relação ao controle da iluminação da sala de jantar/estar e das áreas externas, foi possível realizar os testes através do *latche* de saída 1, também localizado na placa I/O (ver no anexo 2). Conectou-se os *leds* que simulam a iluminação da residência na entrada D0 à D2 do primeiro *latche* de saída. Através de um programa de teste, também compilado no MIDE51, foi possível ligá-los e desligá-los. O programa compilado foi enviado via *Hyper Terminal* ao Kit e consequentemente executado no mesmo.

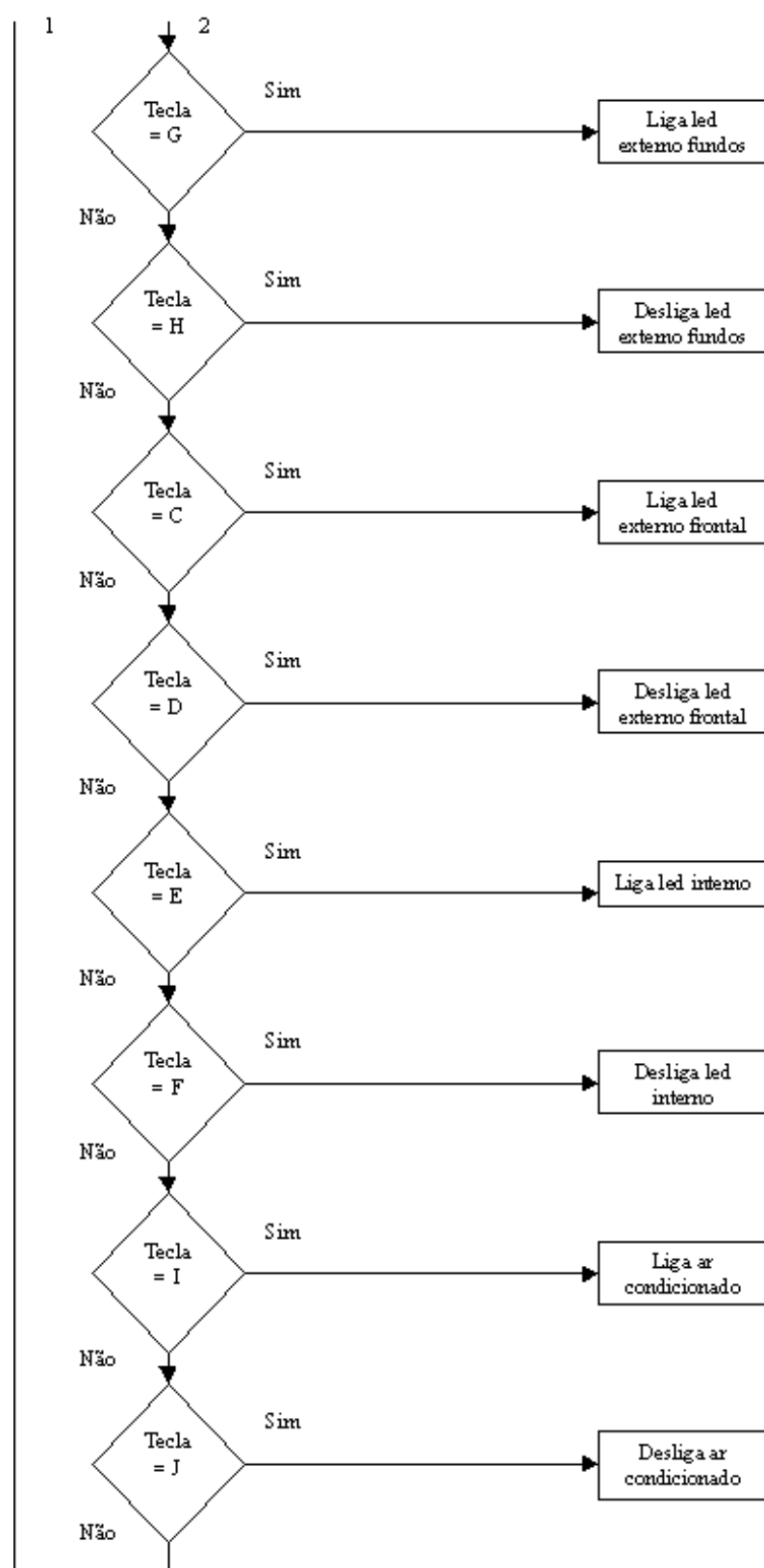
Os controles dos *latches* de entrada e saída foram feitos através dos pinos de *strobe*, localizados no conector 8 do Kit ULBEE51. Também houve a conexão do *flat cable* de dados da placa I/O ao conector 7 do respectivo Kit microcontrolador.

É importante salientar que o sistema poderá ser ampliado, pois o conversor A/D utilizado dispõe de mais 7 entradas analógicas, bem como o circuito de I/O implementado possui mais um *latche* de entrada disponível e um *latche* de saída, prontos para uma utilização futura. Em relação ao Kit do microcontrolador, ainda restaram 9 *strobes* para serem utilizados.

4. SOFTWARE DE CONTROLE E INTERAÇÃO

Após a montagem e testes do hardware, partiu-se para a elaboração do software de controle da residência. Para o desenvolvimento desse programa, foi novamente utilizado o programa MIDE51, só que a linguagem de programação usada agora, denomina-se de *Assembly* [10 e 11]. O fluxograma do programa pode ser visto na figura abaixo. No anexo 5 encontra-se o código fonte do mesmo.





Ao ser executado o aplicativo do software *Delphi*, que coloca em funcionamento a interface de interação, tem-se a visualização da maquete da residência, exposta na tela do PC.

Neste software de interação entre o usuário e a maquete (ver figura nº 4), é possível obter o *status* dos sensores localizados nas dependências da residência. A tela de interação disponibiliza, a cada 5 segundos, em tempo real, as condições apresentadas pelos sensores de presença, magnéticos e de temperatura. Essa atualização é feita através do componente chamado de *timer*, que foi inserido durante a programação no software *Delphi* [2].

Pode-se observar, através da figura nº 4, que todos os sensores magnéticos da residência estão em estado “Ligado”. Isso quer dizer que a maquete encontra-se com todas as portas e janelas abertas, ou seja, os ímãs dos sensores magnéticos não estão próximos das aberturas.

Realizando uma nova observação, verifica-se que os sensores magnéticos da sala de estar e do dormitório apresentam a condição de “Desligado”. Isto pode ser notado na figura abaixo:

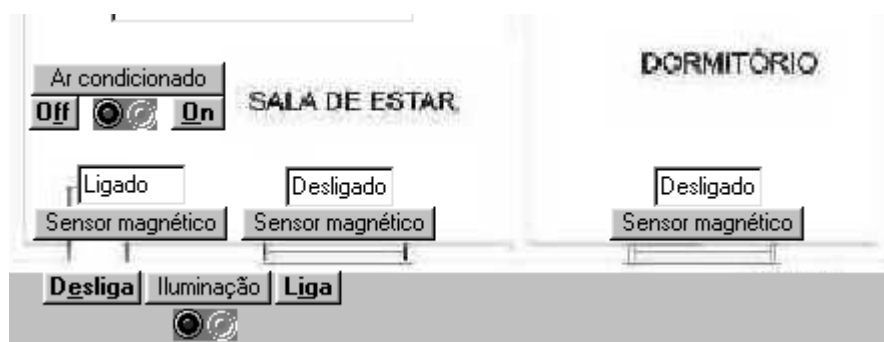


Figura 8 – Status dos sensores magnéticos da sala de estar e do dormitório.

Quando no software de interação demonstrado acima, a condição de “Desligado” é apresentada, pode-se verificar através da figura abaixo, que os sensores magnéticos das respectivas dependências da residência estão com os ímãs aproximados das aberturas das janelas, simulando a situação de janelas fechadas.

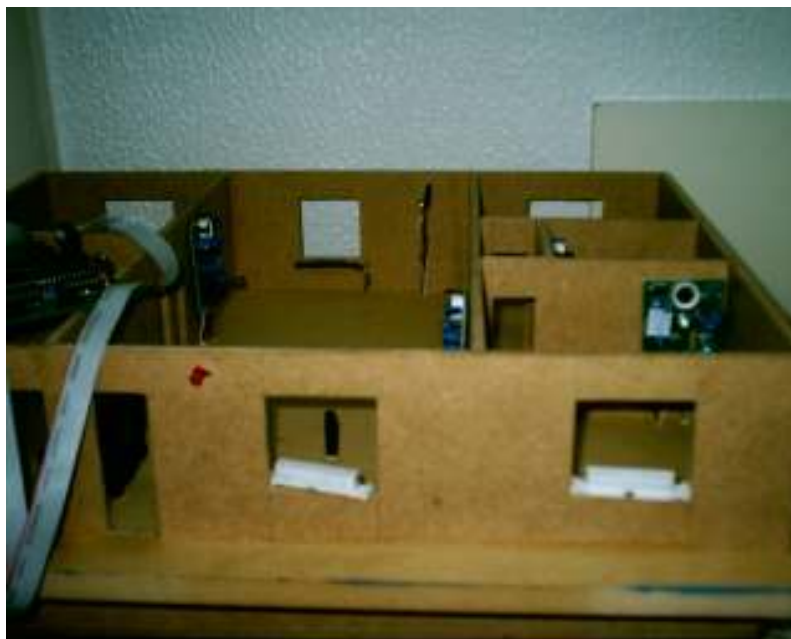


Figura 9 – Ímãs próximos às aberturas das janelas da sala de estar e do dormitório.

Em relação aos sensores de presença, percebe-se a mesma situação encontrada nos sensores magnéticos. Com a tela sendo atualizada a cada 5 segundos, verifica-se a presença ou não de uma condição física.

Um exemplo para facilitar o entendimento, seria a situação em que a figura abaixo demonstra um determinado sensor de presença ativado:

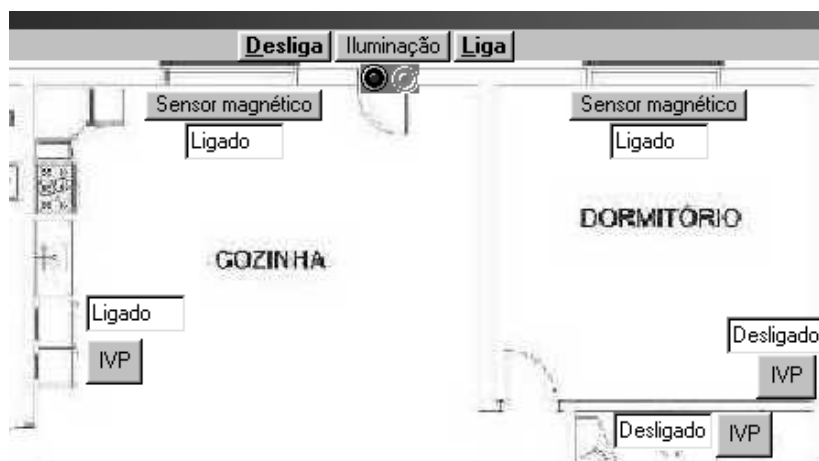


Figura 10 – Sensor de presença da cozinha em estado “Ligado”.

Neste caso, pode-se constatar que o sensor IVP localizado na cozinha da residência identificou uma presença física no local. Por consequência, na maquete

da residência é comprovada a condição de detecção de presença no ambiente. Na figura abaixo se tem o sensor acionado na própria maquete.



Figura 11 – Sensor IVP acionado na cozinha.

O software de interação usuário/maquete também fornece as condições de conforto de um ambiente. Neste caso, é disponibilizado na tela do PC as condições de temperatura na sala de estar e de jantar, possibilitando assim o acionamento ou não do *cooler* do projeto, que é o equipamento responsável por simular um aparelho de ar condicionado no mesmo. A figura abaixo demonstra como é fornecida a temperatura na dependência da residência em questão.



Figura 12 – Temperatura apresentada na tela do PC.

É importante salientar que para o acionamento do ar condicionado na residência, basta apenas um clique com o botão esquerdo do mouse sobre os comandos de ligar (*On*) ou desligar (*Off*). Ou ainda pode-se pressionar as teclas *ALT* + *O* e *ALT* + *F* para ligar ou desligar respectivamente o aparelho responsável pela climatização do ambiente da dependência. A figura a seguir mostra o sensor de temperatura, localizado no centro da área e o *cooler* simbolizando o aparelho de ar condicionado.



Figura 13 – Sensor e equipamento que fornecem controle sobre a temperatura.

Esse software de interação também permite ao usuário clicar sobre um botão relacionado, por exemplo, à iluminação externa, mudando o seu estado, ou seja, possibilita o acionamento ou não do componente presente na maquete da residência.



Figura 14 – Iluminação externa ligada, bem como o ar condicionado.

5. CONCLUSÃO

Quando se trabalha em um projeto como este que foi exposto, procura-se atender às necessidades de uma parcela da sociedade, que em função de sua intensa atividade diária, busca recursos que facilitem e agilizem suas tarefas cotidianas. Também procura-se desenvolver novas tecnologias que venham a contribuir com a diminuição do consumo de energia elétrica, visando a preservação do meio ambiente e a redução dos custos no orçamento familiar.

Outro fator relevante que procurou-se atender foi a segurança da residência e de seus usuários, questão bastante presente, em função da crescente onda de violência e falta de segurança pública. A intenção de se monitorar, controlar e gerenciar uma residência através do uso de um PC e de um microcontrolador, que juntos podem realizar as diversas ações pretendidas pelo usuário em uma residência, destaca-se como sendo o principal motivo que incentivou a implementação desse sistema de automação,.

A partir dos objetivos traçados, as metas alcançadas e o estágio em que o projeto foi finalizado, concluiu-se que o mesmo foi elaborado e construído conforme o planejado. Também foi visto ao longo deste trabalho, que conseguiu-se estabelecer um controle e um gerenciamento sobre equipamentos e aparelhos encontrados na residência, bem como verificou-se o perfeito monitoramento das condições ambientais nas dependências da mesma.

Para estudos futuros, esse projeto pode vir a atender a outras necessidades e também vir a possuir novas características, como por exemplo, um monitoramento e um gerenciamento de funções e utilidades em uma residência à distância. Para isso, poderia se devolver uma comunicação com um modem, através de uma linha telefônica fixa.

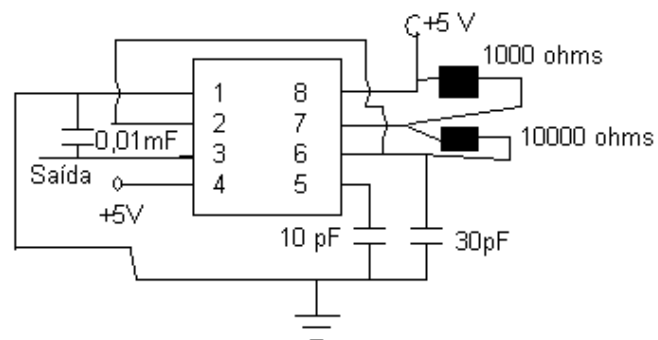
E com o avanço da tecnologia usada em celulares, outra opção que surge, seria uma automação realizada pela telefonia móvel. Também poderia ser desenvolvido um projeto de Automação Residencial que utilizasse horários pré definidos, para o acionamento de determinados equipamentos ou aparelhos, localizados em uma residência.

REFERÊNCIAS BIBLIOGRÁFICAS

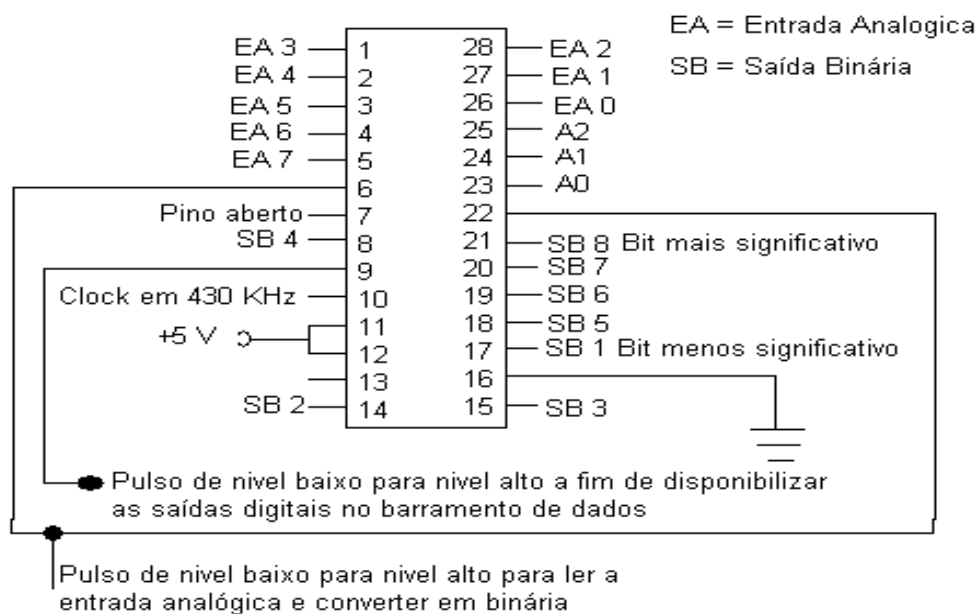
1. Associação Brasileira de Automação Residencial, <http://www.aureside.org.br>
2. BRAGA, Willian. Borland Delphi 6 Inprise – Série IT Educacional, Essencial, Rápido e Didático. Rio de Janeiro: Alta Books, 2002.
3. Clube Delphi, <http://www.clubedelphi.com.br/>.
4. *DatasheetArchive*, <http://datasheetarchive.com/>.
5. Elektor, Eletrônica & Microinformática. Revista nº 41, ano 4, edição brasileira.
6. Eletrônica Total. Revista Saber nº 85, julho de 2002.
7. FACUNTE, Emerson. Delphi 5 - Desenvolvendo Aplicações Cliente/Servidor. Editora Brasport.
8. *Home-page* de ROGERCOM – Pesquisa e Desenvolvimento, <http://www.rogercom.com/PortaSerial/ControleAcesso/ControlePag4.htm/>.
9. *Home-page* de Augusto Alexandre de Mattos, <http://www.ulbra.tcche.br/~augusto/>.

10. NICOLOSI, Denys E. C.. Microcontrolador 8051 Detalhado. São Paulo: Érica, 4^o edição, 2000.
11. Saber Eletrônica Especial. Mini-Curso de Microcontrolador. Revista n^o 2, janeiro 2001.
12. SANTOS, Henrique José dos. Curso de Linguagem C da UFMG (Universidade Federal de Minas Gerais). Santos – SP. <http://www.ead.eee.ufmg.br/cursos/c/>.

ANEXO 1

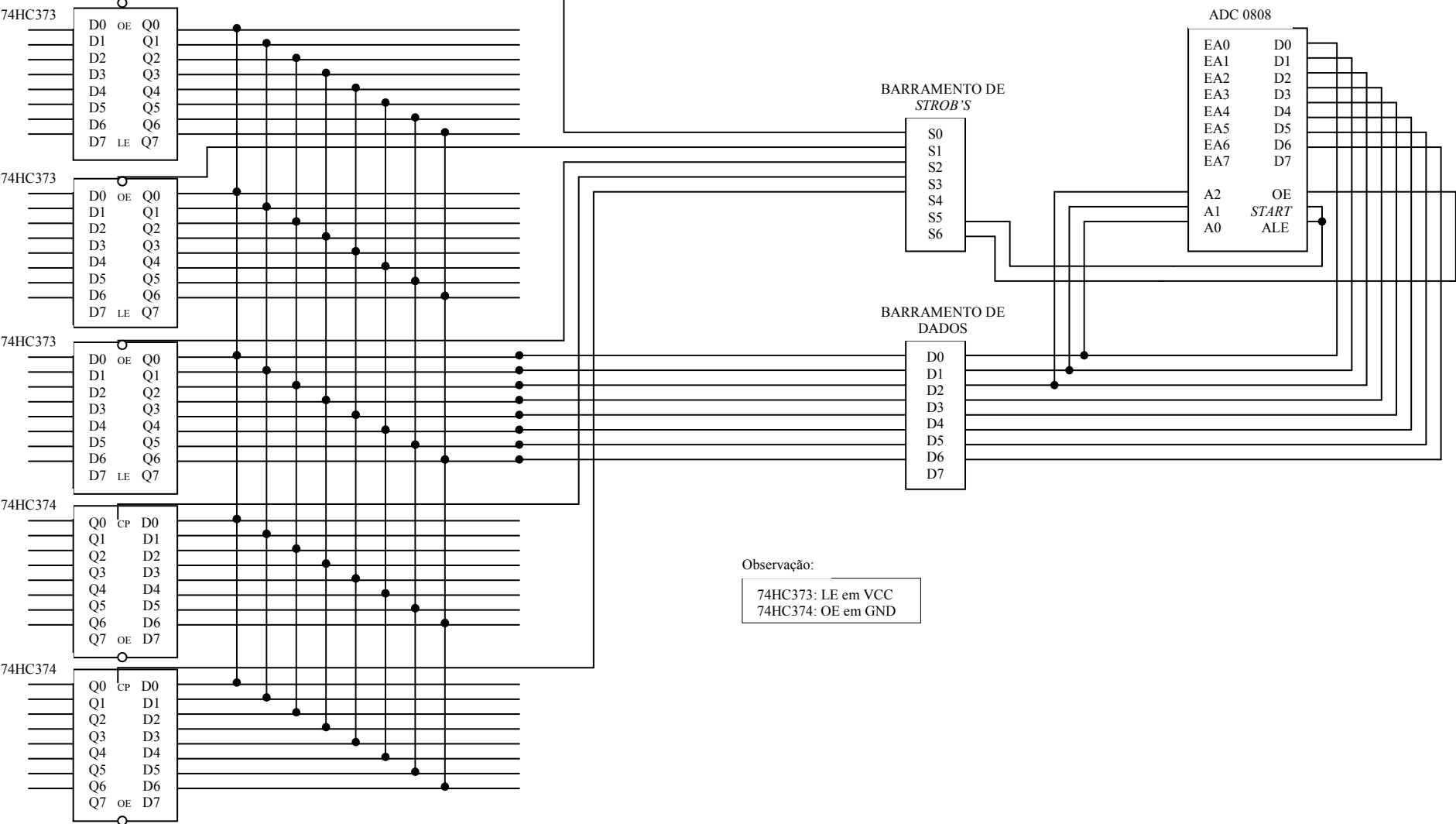


Esquema de ligação para o clock usando um LM 555 (oscilador)

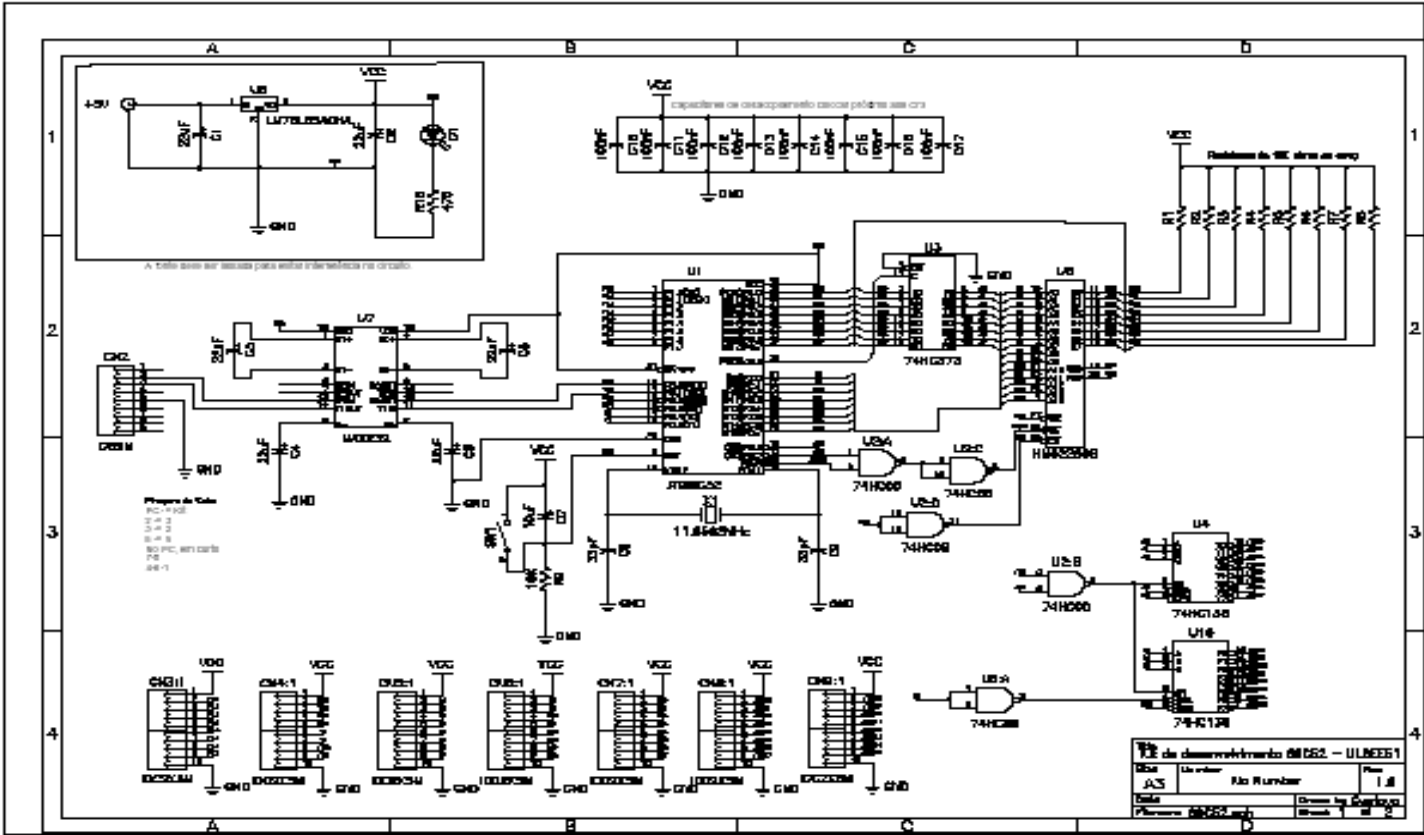


Esquema de ligação do ADC 0808

ANEXO 2



ANEXO 3



ANEXO 4

```

/*****
/*          AUTOMAÇÃO, MONITORAMENTO E CONTROLE          */
/*          DE UMA RESIDÊNCIA                            */
/*          ATRAVÉS DO USO DE UM MICROCONTROLADOR       */
*****/

#include <at89x52.h>
#include <paulmon2.h>

#define ON      1
#define OFF     0

#define strobe0    0x0000    //endereço do strobe zero
#define strobe1    0x0001    //endereço do strobe um
#define strobe2    0x0002    //endereço do strobe dois
#define strobe3    0x0003    //endereço do strobe três
#define strobe4    0x0004    //endereço do strobe quatro
#define strobe5    0x0005    //endereço do strobe cinco
#define strobe6    0x0006    //endereço do strobe seis

/*****Variáveis Globais *****/

volatile xdata at strobe0 unsigned char latch1;
volatile xdata at strobe1 unsigned char latch2;
volatile xdata at strobe2 unsigned char latch3;
volatile xdata at strobe3 unsigned char latch4;
volatile xdata at strobe4 unsigned char latch5;
```

```

volatile xdata at strobe5 unsigned char ad_ale_start;
volatile xdata at strobe6 unsigned char ad_oe;

/***** Rotina de delay *****/

void wait_200ms(void)
{
    _asm

        mov r0,#200
0100$:
        mov r2,#8
0200$:
        mov r1,#55
0300$:
        djnz r1,0300$
        djnz r2,0200$
        djnz r0,0100$

    _endasm;
}

/***** Rotina de Serviço - Timer 0 *****/

        //latch1 = 0000;           //carrega/inicia conversão canal 0
        //nibble_baixo = latch1;   //salva nibble baixo

/***** Rotina principal p/ latch's de saída *****/

void main(void)
{
    unsigned char a=0;
    while(ON)
    {
        latch4=a;
        wait_200ms();
        wait_200ms();
        wait_200ms();
        a++;
    }
}

```

```
}

/***** Rotina principal p/ latch's de entrada *****/

/*void main(void)
{
    unsigned char a=0;

    while(ON)
    {
        a=latch1;
        wait_200ms();
        wait_200ms();
        wait_200ms();
        pm2_pint8u(a);
        pm2_newline();
    }
}*/
```

ANEXO 5

```
cr      equ  0dh
lf      equ  0ah
nul     equ  00h
pint8u      equ  004dh
newline equ  0048h
upper      equ  0040h
cin      equ  0032h
pstr     equ  0038h

org 8000h

mov a,#00h
mov dptr,#0003h
movx @dptr,a

mov dptr,#menu3
lcall pstr      ;exibe na tela a mensagem menu3

lcall newline
lcall newline

mov dptr,#menu4
lcall pstr      ;exibe na tela a mensagem menu4
lcall newline

mov dptr,#menu5
```

```

lcall pstr          ;exibe na tela a mensagem menu5

lcall newline
lcall newline
lcall newline
mov 05h,#00h

inicio:

;mov dptr,#menu1
;lcall pstr          ;exibe na tela a mensagem menu1
;lcall newline

;mov dptr,#menu11
;lcall pstr          ;exibe na tela a mensagem menu11
;lcall newline

;mov dptr,#menu2
;lcall pstr          ;exibe na tela a mensagem menu2
;lcall newline

lcall cin           ;espera um valor do teclado
lcall upper

mov 01h,a
cjne a,#"A",pulo1
lcall next1

pulo1:
cjne a,#"B",pulo2
lcall next2
pulo2:
cjne a,#"C",pulo3
lcall next3

pulo3:
cjne a,#"D",pulo4
lcall next4

```

```
pulo4:
cjne a, #"E",pulo5
lcall next5
```

```
pulo5:
cjne a, #"F",pulo6
lcall next6
```

```
pulo6:
cjne a, #"G",pulo7
lcall next7
```

```
pulo7:
cjne a, #"H",pulo8
lcall next8
```

```
pulo8:
cjne a, #"I",pulo9
lcall next9
```

```
pulo9:
cjne a, #"J",pulo10
lcall next10
```

```
pulo10:
cjne a, #"L",pulo11
lcall next11
```

```
pulo11:
cjne a, #"T",pulo12
lcall next12
```

```
pulo12:
ljmp aqui
```

```
next1:
lcall newline
mov dptr,#0000h ;corresponde ao strob 0
movx a,@dptr
lcall pint8u ;coloca na tela o valor dos sensores magneticos
```

```
ljmp aqui

next2:
lcall newline
mov dptr,#0001h    ;corresponde ao strob 1
movx a,@dptr
lcall pint8u        ;coloca na tela o valor dos sensores de presenca
ljmp aqui

next3:
lcall led1
ljmp aqui

next4:
lcall led2
ljmp aqui

next5:
lcall led3
ljmp aqui

next6:
lcall led4
ljmp aqui

next7:
lcall led5
ljmp aqui

next8:
lcall led6
ljmp aqui

next9:
lcall cooler1
ljmp aqui

next10:
lcall cooler2
ljmp aqui
```



```

next11:
lcall temp
ljmp aqui

next12:
lcall temp

aqui0:
lcall newline
mov dptr,#0000h ;corresponde ao strob 0
movx a,@dptr
anl a,#00000001b
cjne a,#00h,puloa
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui1

puloa:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui1

aqui1:
mov dptr,#0000h ;corresponde ao strob 0
movx a,@dptr
anl a,#00000010b
rr a
cjne a,#00h,pulob
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui2

pulob:
mov dptr,#desligado
lcall pstr
lcall newline

```

```
ljmp aqui2
```

```
aqui2:
```

```
mov dptr,#0000h ;corresponde ao strob 0
```

```
movx a,@dptr
```

```
anl a,#00000100b
```

```
rr a
```

```
rr a
```

```
cjne a,#00h,puloc
```

```
mov dptr,#ligado
```

```
lcall pstr
```

```
lcall newline
```

```
ljmp aqui3
```

```
puloc:
```

```
mov dptr,#desligado
```

```
lcall pstr
```

```
lcall newline
```

```
ljmp aqui3
```

```
aqui3:
```

```
mov dptr,#0000h ;corresponde ao strob 0
```

```
movx a,@dptr
```

```
anl a,#00001000b
```

```
rr a
```

```
rr a
```

```
rr a
```

```
cjne a,#00h,pulod
```

```
mov dptr,#ligado
```

```
lcall pstr
```

```
lcall newline
```

```
ljmp aqui4
```

```
pulod:
```

```
mov dptr,#desligado
```

```
lcall pstr
```

```
lcall newline
```

```
ljmp aqui4
```

```

aqui4:
mov dptr,#0000h    ;corresponde ao strob 0
movx a,@dptr
anl a,#00010000b
rr a
rr a
rr a
rr a
cjne a,#00h,puloe
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui5

```

```

puloe:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui5

```

```

aqui5:
mov dptr,#0000h    ;corresponde ao strob 0
movx a,@dptr
anl a,#00100000b
rr a
rr a
rr a
rr a
rr a
cjne a,#00h,pulof
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui6

```

```

pulof:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui6

```

```

aqui6:
mov dptr,#0000h    ;corresponde ao strob 0
movx a,@dptr
anl a,#01000000b
rr a
rr a
rr a
rr a
rr a
rr a
cjne a,#00h,pulog
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui7

pulog:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui7

aqui7:
mov dptr,#0000h    ;corresponde ao strob 0
movx a,@dptr
anl a,#10000000b
rr a
rr a
rr a
rr a
rr a
rr a
rr a
cjne a,#00h,puloh
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui8

```

```

puloh:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui8

aqui8:
mov dptr,#0001h ;corresponde ao strob 1
movx a,@dptr
anl a,#00000001b
cjne a,#00h,puloi
mov dptr,#ligado ;logica inversa nos sensores
lcall pstr
lcall newline
ljmp aqui9

puloi:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui9

aqui9:
mov dptr,#0001h ;corresponde ao strob 1
movx a,@dptr
anl a,#00000010b
rr a
cjne a,#00h,puloj
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui10

puloj:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui10

```

```

aqui10:
mov dptr,#0001h    ;corresponde ao strob 1
movx a,@dptr
anl a,#00000100b
rr a
rr a
cjne a,#00h,pulol
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui11

```

```

pulol:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui11

```

```

aqui11:
mov dptr,#0001h    ;corresponde ao strob 1
movx a,@dptr
anl a,#00001000b
rr a
rr a
rr a
cjne a,#00h,pulom
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui12

```

```

pulom:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui12

```

```

aqui12:
mov dptr,#0001h    ;corresponde ao strob 1
movx a,@dptr

```

```

anl a,#00010000b
rr a
rr a
rr a
rr a
cjne a,#00h,pulon
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui13

```

```

pulon:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui13

```

```

aqui13:
mov dptr,#0001h ;corresponde ao strob 1
movx a,@dptr
anl a,#00100000b
rr a
rr a
rr a
rr a
rr a
cjne a,#00h,puloo
mov dptr,#ligado
lcall pstr
lcall newline
ljmp aqui14

```

```

puloo:
mov dptr,#desligado
lcall pstr
lcall newline
ljmp aqui14

```

```

aqui14:
mov dptr,#0001h    ;corresponde ao strob 1
movx a,@dptr
anl a,#01000000b
rr a
rr a
rr a
rr a
rr a
rr a
cjne a,#00h,pulop
mov dptr,#ligado
lcall pstr
;lcall newline
ljmp aqui15

pulop:
mov dptr,#desligado
lcall pstr
;lcall newline
ljmp aqui15

aqui15:
ljmp aqui

tempo:
push 01h
push 02h
dela05:
mov 02h,#08h
dela10:
mov 01h,#55h
dela20:
djnz 01h,dela20
djnz 02h,dela10
djnz 00h,dela05
pop 02h
pop 01h
ret

```



```
led1:
mov a,05h
anl a,#11111101b ;liga led do d1
orl a,#00000010b
mov 05h,a
mov dptr,#0003h
movx @dptr,a

ljmp aqui

led2:
mov a,05h
anl a,#11111101b ;desliga led do d1
mov 05h,a
mov dptr,#0003h
movx @dptr,a

ljmp aqui

led3:
mov a,05h
anl a,#11111110b ;liga led do d0
orl a,#00000001b
mov 05h,a
mov dptr,#0003h
movx @dptr,a

ljmp aqui

led4:
mov a,05h
anl a,#11111110b ;desliga led do d0
mov 05h,a
mov dptr,#0003h
movx @dptr,a
ljmp aqui

led5:
mov a,05h
anl a,#11111011b ;liga led do d2
```

```

    orl a,#00000100b
    mov 05h,a
    mov dptr,#0003h
    movx @dptr,a

```

```

    ljmp aqui

```

```

led6:
    mov a,05h
    anl a,#11111011b ;desliga led do d2
    mov 05h,a
    mov dptr,#0003h
    movx @dptr,a

```

```

    ljmp aqui

```

```

cooler1:
    mov a,05h
    anl a,#11110111b ;liga ar do d3
    orl a,#00001000b
    mov 05h,a
    mov dptr,#0003h
    movx @dptr,a

```

```

    ljmp aqui

```

```

cooler2:
    mov a,05h
    anl a,#11110111b ;desliga ar do d3
    mov 05h,a
    mov dptr,#0003h
    movx @dptr,a
    ljmp aqui

```

```

temp:
    mov dptr,#temperatura
    lcall pstr
    mov dptr,#0005h ;seleciona strob6
    movx a,@dptr ;manda capturar e converter
    mov 00h,02h ;rotina para tempo de 1ms

```

```

lcall tempo      ;curte o delay
mov dptr,#0006h  ;seleciona strob5
movx a,@dptr     ;armazena no acumulador valor digital da temp
lcall pint8u     ;imprime na tela o valor da temperatura
mov dptr,#graus
lcall pstr
ljmp aqui0

```

```

aqui:
mov 00h,#0feh
lcall tempo
lcall newline
lcall newline
ljmp inicio

```

```

menu1:
db "Digite A para ler os sensores magneticos:",nul

```

```

menu2:
db "Digite T para ler a Temperatura:",nul

```

```

menu11:
db "Digite B para ler os sensores de presenca:",nul

```

```

menu3:
db "TCC Alessandro",nul

```

```

menu4:
db "Automacao Monitoramento e Controle",nul

```

```

menu5:
db "de uma Residencia com Microcontrolador",nul
temperatura:

```

```

db "Temperatura:",nul
graus:
db " Graus Celsius",nul

```

```

ligado:
db "Ligado",nul

```

```

desligado:

```

```
db "Desligado",nul
```

```
end
```

ANEXO 6

```
unit Casa;

interface

uses
    Windows, Messages, SysUtils, ComPort, StdCtrls, Controls, jpeg,
    Classes,
    ExtCtrls, Graphics, Forms,
    Dialogs;

type
    TForm1 = class(TForm)
        Image1: TImage;
        Button1: TButton;
        Button3: TButton;
        Button4: TButton;
        Button5: TButton;
        Button8: TButton;
        Button9: TButton;
        Button10: TButton;
        Button13: TButton;
        Button14: TButton;
        Button15: TButton;
        Button16: TButton;
        Button7: TButton;
        Button18: TButton;
        Button19: TButton;
```

```
Button20: TButton;  
Button21: TButton;  
Button22: TButton;  
Button2: TButton;  
Button6: TButton;  
Button11: TButton;  
Button12: TButton;  
Button17: TButton;  
Edit1: TEdit;  
ComPort1: TComPort;  
Status: TMemo;  
Image2: TImage;  
Image3: TImage;  
Button23: TButton;  
Button24: TButton;  
Button25: TButton;  
Button26: TButton;  
Image4: TImage;  
Image5: TImage;  
Button27: TButton;  
Button28: TButton;  
Image6: TImage;  
Image7: TImage;  
Button29: TButton;  
Button30: TButton;  
Image8: TImage;  
Image9: TImage;  
Button31: TButton;  
Temp: TEdit;  
Image10: TImage;  
Image11: TImage;  
Image12: TImage;  
Image13: TImage;  
Image14: TImage;  
Image15: TImage;  
Image16: TImage;  
Timer1: TTimer;  
Edit2: TEdit;  
Edit3: TEdit;  
Edit4: TEdit;
```

```

    Edit5: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    Edit9: TEdit;
    Edit10: TEdit;
    Edit11: TEdit;
    Edit12: TEdit;
    Edit13: TEdit;
    Edit14: TEdit;
    Edit15: TEdit;
    Edit16: TEdit;
    procedure Button18Click(Sender: TObject);
    procedure Button19Click(Sender: TObject);
    procedure Button22Click(Sender: TObject);
    procedure ComPort1ReceiveCallBack(Data: String);
    procedure Button30Click(Sender: TObject);
    procedure Button29Click(Sender: TObject);
    procedure Button27Click(Sender: TObject);
    procedure Button28Click(Sender: TObject);
    procedure Button25Click(Sender: TObject);
    procedure Button26Click(Sender: TObject);
    procedure Button23Click(Sender: TObject);
    procedure Button24Click(Sender: TObject);
    procedure Button31Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
private
    { Private declarations }
public
    { Public declarations }
end;

var
    Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Button18Click(Sender: TObject);

```

```

begin
comport1.open;
end;

procedure TForm1.Button19Click(Sender: TObject);
begin
comport1.close;
end;

procedure TForm1.Button22Click(Sender: TObject);
begin
status.clear;
comport1.send(edit1.text+#13#10);
end;

procedure TForm1.ComPort1ReceiveCallBack(Data: String);
begin
Status.Lines.Add( Data );
temp.text:=Status.lines[4];
edit2.text:=Status.lines[6];
edit3.text:=Status.lines[8];
edit4.text:=Status.lines[10];
edit5.text:=Status.lines[12];
edit6.text:=Status.lines[14];
edit7.text:=Status.lines[16];
edit8.text:=Status.lines[18];
edit9.text:=Status.lines[20];
edit10.text:=Status.lines[22];
edit11.text:=Status.lines[24];
edit12.text:=Status.lines[26];
edit13.text:=Status.lines[28];
edit14.text:=Status.lines[30];
edit15.text:=Status.lines[32];
edit16.text:=Status.lines[34];
end;

procedure TForm1.Button30Click(Sender: TObject);
begin
comport1.send('f'+#13#10);

```



```
image9.visible:=true;
image8.visible:=false;
end;

procedure TForm1.Button29Click(Sender: TObject);
begin
comport1.send('e'+#13#10);
image9.visible:=false;
image8.visible:=true;
end;

procedure TForm1.Button23Click(Sender: TObject);
begin
comport1.send('d'+#13#10);
image2.visible:=true;
image3.visible:=false;
end;

procedure TForm1.Button24Click(Sender: TObject);
begin
comport1.send('c'+#13#10);
image2.visible:=false;
image3.visible:=true;
end;

procedure TForm1.Button27Click(Sender: TObject);
begin
comport1.send('j'+#13#10);
image7.visible:=true;
image6.visible:=false;
end;

procedure TForm1.Button28Click(Sender: TObject);
begin
comport1.send('i'+#13#10);
image7.visible:=false;
image6.visible:=true;
end;

procedure TForm1.Button25Click(Sender: TObject);
```

```

begin
comport1.send('h'+#13#10);
image5.visible:=true;
image4.visible:=false;
end;

procedure TForm1.Button26Click(Sender: TObject);
begin
comport1.send('g'+#13#10);
image5.visible:=false;
image4.visible:=true;
end;

procedure TForm1.Button31Click(Sender: TObject);
begin
status.clear;
comport1.send('t'+#13#10);
temp.text:=Status.lines[4];
edit2.text:=Status.lines[6];
edit3.text:=Status.lines[8];
edit4.text:=Status.lines[10];
edit5.text:=Status.lines[12];
edit6.text:=Status.lines[14];
edit7.text:=Status.lines[16];
edit8.text:=Status.lines[18];
edit9.text:=Status.lines[20];
edit10.text:=Status.lines[22];
edit11.text:=Status.lines[24];
edit12.text:=Status.lines[26];
edit13.text:=Status.lines[28];
edit14.text:=Status.lines[30];
edit15.text:=Status.lines[32];
edit16.text:=Status.lines[34];
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
form1.button31.Click;
end;
end.

```