



Centro Universitário de Brasília

*Glauber Moreira Rocha*

*Uso inteligente da automatização residencial  
Um estudo de caso sobre o controle de iluminação*

Brasília, DF – Dezembro 2006.

*Glauber Moreira Rocha*

*Uso inteligente da automatização residencial  
Um estudo de caso sobre o controle de iluminação*

Monografia apresentada ao Centro  
Universitário de Brasília – UniCEUB  
como um dos pré-requisitos para  
obtenção do título de Bacharel em  
Engenharia da Computação.

Profº orientador: MC Aderlon Marcelino Queiroz

Brasília, DF – Dezembro 2006.

*Glauber Moreira Rocha*

Uso inteligente da automatização residencial  
Um estudo de caso sobre o controle de iluminação

Monografia apresentada ao Centro Universitário de Brasília – UniCEUB como um dos pré-requisitos para obtenção do título de Bacharel em Engenharia da Computação.

**MEMBROS DA BANCA EXAMINADORA**

<b>MEMBROS DA BANCA</b>	<b>ASSINATURA</b>
<b>1.COORDENADOR DO CURSO</b> Prof °. Abiezer Amarília Fernandes	
<b>2. PROFESSOR ORIENTADOR</b> Prof °. Aderlon M. Queiroz	
<b>3. PROFESSOR EXAMINADOR</b> Prof °. _____	
<b>4. PROFESSOR EXAMINADOR</b> Prof °. _____	
<b>MENSÃO FINAL:</b>	

Brasília, DF – Dezembro 2006.

## **Agradecimentos**

Agradeço a Deus, em primeiro lugar, por estar presente em todas as horas na minha vida, me conduzindo sempre para melhor caminho e me iluminando nas horas difíceis.

Aos meus pais, Dourival Campos Rocha e Maria das Graças Moreira Rocha, pela oportunidade e pela confiança creditada.

Aos meus colegas Henrique, Leonardo, Fabrício e Gustavo pelo incentivo e colaboração mútua nesse projeto final.

Aos monitores do laboratório de eletrônica, Roni e Tiago, pela atenção em esclarecer dúvidas sobre o projeto.

Ao orientador do meu projeto final Aderlon Marcelino Queiroz que foi fundamental com seus conselhos desde a primeira idéia de proposta do projeto em outubro de 2005 à finalização dessa monografia, tendo ótimas idéias principalmente em relação ao controle de intensidade desenvolvido.

## Resumo

Esse projeto aborda a implementação de um software e um hardware de automação residencial de forma web. O usuário poderá cadastrar e conectar dispositivos podendo acioná-los, modificar a intensidade e obter relatórios com a estimativa de consumo.

Foi utilizado para o desenvolvimento do software a tecnologia Java Server Pages e banco de dados PostgreSQL. A conexão entre o computador e o periférico será através da porta paralela. O hardware será composto de um demultiplexador (para indicar qual aparelho será acionado) e registradores com flip-flops (para manter a intensidade escolhida mesmo mudando o aparelho a ser acionado).

Palavras-chaves: Automação residencial, JSP e demultiplexador

## **Abstract**

This project is about a software and hardware that control a house in a web way. The user can registry and connect electrical machines can turn it on, modify the intensity and get the consume estimative.

For the software construction is used Java Server Pages technology and PostgreSQL database. The connection between the computer and the hardware will be by parallel port. The hardware will be compost by a demultiplex (for choose the machine selected) and registers with flip-flops (to stand the intensity selected since changing the machine selected).

Key words: House control, JSP e demultiplex

# Sumário

Lista de figuras.....	6
Lista de tabelas.....	7
Lista de símbolos.....	8
1 Introdução.....	9
1.1 Objetivo.....	10
1.2 Topologia do projeto.....	11
1.3 Apresentação.....	12
2 Hardware.....	13
2.1 Porta paralela.....	15
2.1.2 Extensão do cabo paralelo.....	15
2.1.3 Endereços da porta paralela.....	16
2.2 Multiplexador/Demultiplexador.....	16
2.2.1 Problema Encontrado: Conseguir um demultiplexador de no mínimo 8 portas.....	17
2.3 Flip-Flops.....	17
2.5 Tabela verdade do circuito.....	18
2.7 Controlador de intensidade de iluminação.....	21
2.7.1 Corrente alternada e valor rms.....	23
2.7.2 Divisor de tensão.....	24
2.7.3 Cálculos utilizados para o controle de intensidade.....	24
2.7.3.1 Intensidade máxima.....	25
2.7.3.2 Intensidade média.....	26
2.7.3.3 Intensidade mínima.....	27
3 Software.....	29
3.1 Diagramação UML.....	29
3.3 Linguagem Java.....	35
3.4 Banco de dados PostgreSQL.....	36
3.5 IDE NetBeans.....	37
3.6 Problemas encontrados.....	38
3.6.1 Conseguir fazer a linguagem Java controlar a porta paralela.....	38
3.6.2 Utilizar o endereço 0000 do demultiplexador.....	39
3.7 Montagem.....	39
3.7.1 Banco de dados.....	39
3.7.2 Classes Java.....	41
3.7.3 Telas JSP.....	41
3.8 Estimativa de consumo.....	42
3.9 Telas.....	42
4 Implementação.....	46
4.1 Funcionamento completo do sistema.....	46
4.2 Um exemplo de utilização.....	49
4.3 Resultados encontrados.....	55
No próximo capítulo será mostrado o que foi concluído no trabalho.....	56
5 Conclusão.....	57
5.1 Sugestões para projetos futuros.....	57

Referências.....	59
Apêndice A - Código do aplicativo .....	60
A.1 Classe aparelho.java .....	60
A.2 Conexao.java.....	66
A.3 uso.java .....	68
A.4 index.jsp .....	73
A.5 porta.jsp.....	76
A.6 CadastroPortaAction.jsp .....	79
A.7 AtualizaPortaAction.jsp.....	81
A.8 controle.jsp .....	83
A.9 ControleAction.jsp .....	85
A.10 consumo.jsp .....	88
Apêndice B - Circuitos integrados utilizados .....	92
B.1 Família de circuitos TTL – Transistor-Transistor Logic.....	92
B.2 Família TTL LOW POWER SCHOTTKY (54LS/74LS) .....	92
B.2.1 O decodificador 74LS42 .....	93
B.2.2 Os inversores 74LS04 .....	94
B.2.3 O registrador 7475.....	95



## Lista de figuras

Figura 1.1 – Diagrama de bloco do projeto .....	10
Figura 1.2 - Topologia do projeto .....	11
Figura 2.2 - Flip-flop tipo D .....	18
Figura 2.3 - Esquema eletrônico inicial do projeto do controle de intensidade ...	22
Figura 2.4 - Esquema eletrônico do controle de intensidade .....	25
Figura 3.1 - Símbolo do ator dos diagramas UML .....	30
Figura 3.2 - Símbolo do uso dos diagramas UML.....	30
Figura 3.3 - Símbolo da classe dos diagramas UML .....	30
Figura 3.4 - Símbolo da Informação dos diagramas UML.....	30
Figura 3.5 - Símbolo da requisição dos diagramas UML .....	31
Figura 3.6 - Símbolo do Componente dos diagramas UML .....	31
Figura 3.7 - Símbolo de Interface dos diagramas UML.....	31
Figura 3.8 - Símbolo de implantação dos diagramas UML .....	32
Figura 3.9 - Casos de usos do sistema.....	32
Figura 3.10 - Caso de uso do cadastro de aparelhos por porta .....	33
Figura 3.11 - Caso de uso do envio de intensidade .....	34
Figura 3.12 - Caso de uso da requisição do relatório de consumo .....	34
Figura 3.13 - Digrama de implantação do sistema .....	35
Figura 3.14 - Tela de administração do PostgreSQL (pgAdmin III) .....	37
Figura 3.15 - Tela da IDE NetBeans .....	38
Figura 3.16 - Tela do menu principal .....	42
Figura 3.17 - Tela do cadastro de aparelhos .....	43
Figura 3.18 - Tela do controle (envio) de intensidade.....	44
Figura 3.19 - Tela do relatório de consumo .....	45
Figura 4.4 – Cadastro da primeira lâmpada.....	50
Figura 4.5 – Cadastro da segunda lâmpada .....	50
Figura 4.6 – Cadastro da terceira lâmpada.....	51
Figura 4.7 – Visualizar os aparelhos cadastrados .....	51
Figura 4.8 – Modificar as intensidades das lâmpadas .....	52
Figura 4.9 – Intensidade na primeira lâmpada.....	52
Figura 4.10 – Intensidade na segunda lâmpada.....	53
Figura 4.11 – Intensidade na terceira lâmpada.....	53
Figura 4.12 – Desligar os aparelhos .....	54
Figura 4.13 – Visualizar o relatório com o total de consumo em W .....	55
Figura B.1 - Portas lógicas do 74LS42 .....	94
Figura B.2 - Portas inversoras no 74LS04 .....	95

## Lista de tabelas

Tabela 2.1 - Endereços da porta paralela .....	16
Tabela 2.2 - Tabela verdade do circuito do demultiplexador .....	19
Tabela 2.2 - Tabela verdade do circuito do demultiplexador (continuação).....	20
Tabela 3.1 - Tabela aparelho do banco de dados .....	40
Tabela 3.2 - Tabela uso do banco de dados.....	40
Tabela 3.3 - Tabela de classes java do aplicativo .....	41
Tabela 3.4 - Telas do aplicativo .....	41
Tabela 4.1 - Tensão medida no teste prático do sistema .....	55
Tabela B.1 - Tabela verdade do 74LS42 .....	94
Tabela B.2 - Tabela verdade do registrador 7475 .....	96

## Lista de símbolos

JSP Java Server Pages (Páginas de Servidor Java)

RMS Root Mean Square (valor médio quadrático)

Vrms Tensão RMS

Irms Corrente RMS

R Resistência

UML Unified Modeling Language (Linguagem unificada de modelagem)

IDE Integrated Development Environment (Ambiente Integrado de Desenvolvimento)

TTL Transistor-Transistor Logic (Transistor- Transistor lógico)

# 1 Introdução

Neste trabalho é proposto um sistema de automação residencial. A principal motivação para esse projeto é facilitar o controle de iluminação em ambientes de grande espaço como ginásios, centro de convenções, auditórios além de residências.

O hardware terá como função acionar e controlar a intensidade de lâmpadas por intermédio de comandos enviados de um computador servidor web, assim será possível controlar a iluminação sem o uso de cabos (por rede sem fio) ou a distâncias geográficas (bairros diferentes e cidades diferentes por internet).

O usuário poderá cadastrar cada lâmpada no sistema informando a qual porta do periférico está conectada, o local do ambiente onde se encontra além do consumo médio. Com essas informações, além de permitir o acionamento, será possível conseguir relatórios de estimativa de consumo de determinados períodos (o consumo não será medido e sim calculado utilizando o tempo que ficou acionado e o consumo cadastrado no software).

Para acionar cada aparelho, o usuário o selecionará em uma tabela informando a intensidade desejada (0, 1, 2 ou 3 que serão respectivamente desligado, intensidade mínima, intensidade média ou intensidade máxima).

Será usado para programação da interface web a linguagem JSP (Java Server Pages), a comunicação do computador servidor com o hardware será pela porta paralela e este será composto de um circuito integrado demultiplexador, 5 circuitos integrados registradores baseados em flip-flops e 2 circuitos integrados compostos por portas inversoras além de controladores de intensidade baseados em resistores acionados por relés.



Figura 1.1 – Diagrama de bloco do projeto

## 1.1 Objetivo

Esse projeto tem o objetivo de aumentar a quantidade de aparelhos controlados pela porta paralela inserindo mais funcionalidades ao controle desses aparelhos.

Para aumentar o número de aparelhos controlados é utilizado um circuito demultiplexador e flip-flops para manter o valor das saídas.

Para inserir mais funcionalidades o trabalho propõe um controle da variação de intensidade, estimativa de consumo e controle a distância.

O controle de intensidade usa uma conversão de 2 bits vindos do circuito demultiplexador para 3 níveis de intensidade em uma lâmpada de 60W mais o estado desligado.

Para a estimativa de consumo o projeto propõe que o software registre o horário do acionamento e multiplique pelo consumo cadastrado no banco de dados.

O controle a distância utilizará como solução um servidor com aplicativo web podendo ser controlado pela internet, rede local ou rede sem-fio.

## 1.2 Topologia do projeto

Ligado ao computador servidor (que possui um aplicativo em formato web que controla os bits enviados pela porta paralela) temos um hardware composto por um circuito demultiplexador e conversores dos bits recebidos em intensidade das lâmpadas de 220 V que serão conectadas ao hardware.

Diagrama do projeto de controle de iluminação



Figura 1.2 - Topologia do projeto

## 1.3 Apresentação

Esse trabalho está dividido em introdução, desenvolvimento do hardware, desenvolvimento do software, implementação, conclusão, referências e apêndice.

### Capítulo 1:

A introdução mostra uma visão geral do trabalho contendo um resumo, a apresentação dos capítulos e o objetivo.

### Capítulo 2:

O capítulo sobre hardware aborda a porta paralela, o circuito demultiplexador e o controle de intensidade. Esse capítulo descreverá os componentes utilizados (bases teóricas, esquemas eletrônicos e cálculos utilizados no projeto).

### Capítulo 3:

O capítulo seguinte aborda o desenvolvimento do software com seus diagramas UML, ferramentas utilizadas, classes, tabelas de banco de dados e desafios encontrados durante o projeto desse aplicativo.

### Capítulo 4:

A implementação abordará a utilização de todo sistema mostrando o objetivo do teste e os resultados obtidos.

### Capítulo 5:

A conclusão resumirá o projeto com os resultados obtidos. Esse capítulo mostrará sugestões de projetos futuros.

Teremos também as referências dos artigos pesquisados e um apêndice do código do aplicativo.

No próximo capítulo é explicado como foi projetado o hardware e seus componentes.

## 2 Hardware

O hardware é constituído por um buffer, demultiplexador, flip-flop e portas inversoras. O hardware também é constituído por um controlador de intensidade formado por relés, acopladores ópticos e resistores de 15W.

O buffer (74LS244) é responsável por proteger a porta paralela de variações de corrente que possam vir do aparelho. Possui 8 entradas/saídas onde serão utilizadas apenas 6 para transmitir o pacote de dados.

O primeiro componente é um decodificador binário decimal (74LS42) com 4 entradas (vindas do 74LS244) e 10 saídas que controlarão os registradores (descritos posteriormente). Um problema encontrado foi o fato de a saída selecionada ficar em nível baixo e as demais em nível alto, para corrigir isso são necessários utilizar portas lógicas inversoras (74LS04) levando-se em conta que os registradores 7475 (descritos posteriormente) precisam de nível alto para mudar seus dados e nível baixo de entrada para manter.

Os registradores 7475 possuem 4 flip-flops tipo D e duas entradas de controle (cada uma controla 2 flip-flops). Os flip-flops receberão os bits de dados enviados, mas apenas os que tiverem nível alto em sua entrada de controle mudarão seus dados para os atuais, os demais manterão os bits gravados anteriormente.



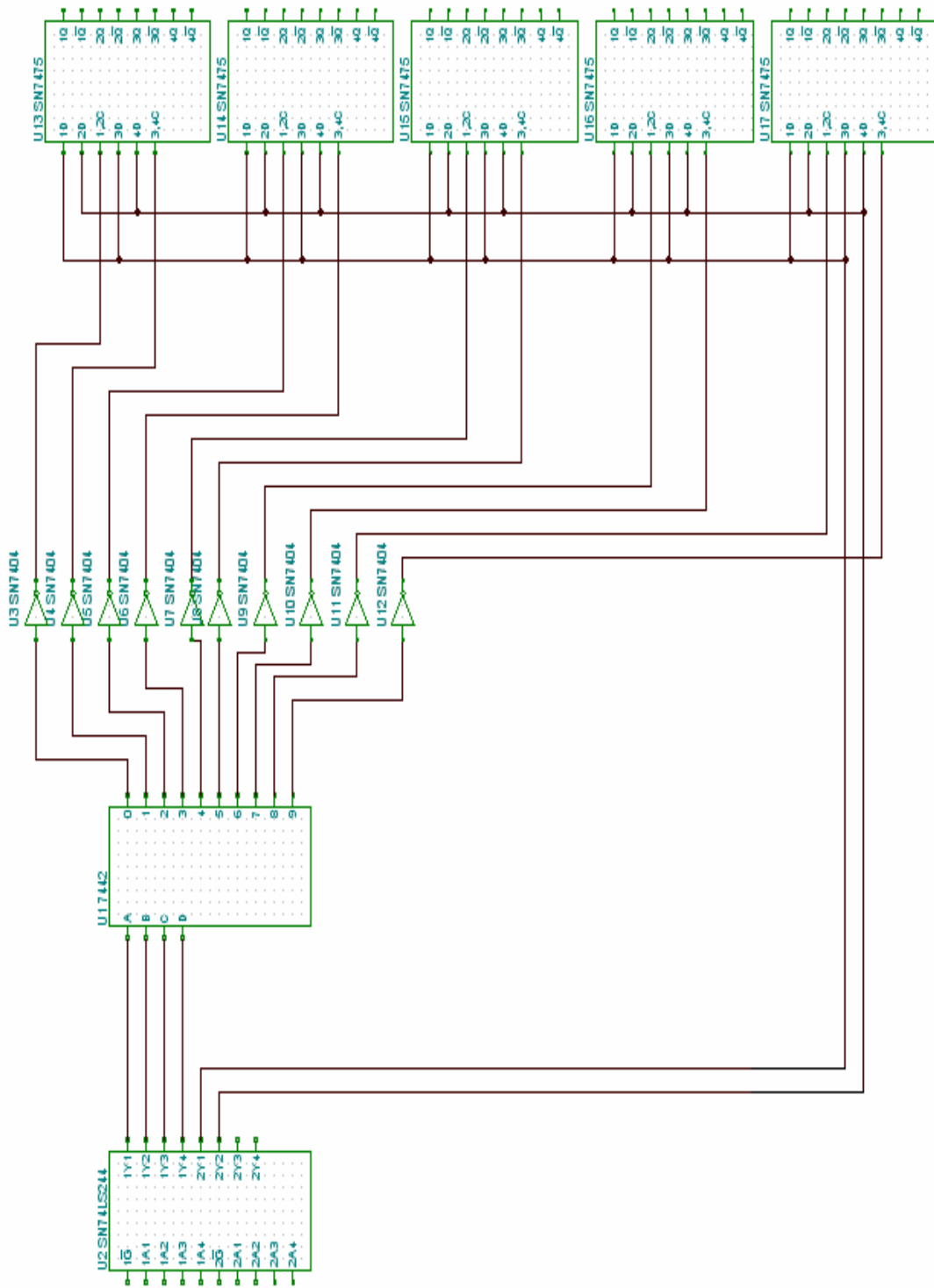


Figura 2.1 - Esquema do circuito do demultiplexador

A porta paralela enviará para o buffer (74LS244) um pacote composto pelo dado (nível de intensidade) a ser enviado (2 bits) e o endereço da porta (com 4 bits).

Exemplo: “110010”

Onde os dois primeiros bits “11” indicam mandar o dado 3 e os outros quatro bits “0010” indicam o endereço da porta 2. Os dois primeiros bits serão enviados direto para os registradores e os 4 últimos serão enviados para o demultiplexador para converter em acionamento de uma de suas 10 saídas.

Esta seção explica quais componentes foram utilizados na montagem do hardware responsável por enviar a intensidade selecionada no software (explicado no capítulo 3) às lâmpadas conectadas em cada uma das suas 10 portas.

## **2.1 Porta paralela**

A porta paralela é uma interface de comunicação entre o computador e um periférico. Quando a IBM criou seu primeiro PC (Personal Computer) ou Computador Pessoal, a idéia era conectar a essa porta uma impressora, mas atualmente, são vários os periféricos que se utilizam desta porta para enviar e receber dados para o computador (exemplos: scanners, câmeras de vídeo, unidade de disco removível e outros).

### **2.1.2 Extensão do cabo paralelo**

A extensão do cabo para interligar um computador a um periférico, é de no máximo 8m. Na prática, utiliza-se um cabo com extensão menor. Quanto maior a extensão do cabo, maior é a interferência na transmissão dos dados.

No projeto isso implica em ter que limitar a distância do hardware para o computador servidor web a uma distância pequena (do tamanho do cabo paralelo).

### 2.1.3 Endereços da porta paralela

O computador nomeia as portas paralelas, chamando-as de LPT1, LPT2, LPT3..., mas, a porta física padrão do computador é a LPT1, e seus endereços são: 378h (para enviar um byte de dados pela Porta), 378+1h (para receber um valor através da Porta) e, 378+2h (para enviarem dados). Às vezes pode está disponível a LPT2, e seus endereços são: 278h, 278+1h e 278+2h, com as mesmas funções dos endereços da porta LPT1 respectivamente.

Tabela 2.1 - Endereços da porta paralela

Nome da Porta	Endereço de memória	Endereço da Porta		Descrição
LPT1	0000:0408	378 hexadecimal	888 decimal	Endereço base
LPT2	0000:040A	278 hexadecimal	632 decimal	Endereço base

Fonte: Antônio Rogério Messias

## 2.2 Multiplexador/Demultiplexador

O circuito multiplexador é utilizado para enviarmos as informações contidas em vários canais (fios), a um só canal (fio).

Entende-se por demultiplexador como sendo o bloco que efetua a função inversa ao multiplexador, ou seja, a de enviar informações contidas em um canal a vários canais de saída.

As entradas de seleção têm como finalidade escolher qual o canal de informação de saída que deve ser conectado à entrada, ou seja, devem endereçar o canal de saída, ao qual a informação deve se dirigir [IDOETA, 1998].

### **2.2.1 Problema Encontrado: Conseguir um demultiplexador de no mínimo 8 portas**

Foi difícil encontrar comercialmente um demultiplexador que controlasse no mínimo 8 portas (que é a quantidade de bits da porta paralela assim sem nenhum circuito de intermediação podemos controlar 8 relés que controlam 8 aparelhos utilizando somente a porta paralela e nenhum circuito demultiplexador e foi preciso de mais portas para ampliar sua capacidade de controle).

Primeiramente foi testado um demultiplexador de 8 portas utilizando portas lógicas mas essa solução ocupou muito espaço no protoboard tornando inviável para continuação do projeto (não tinha espaço para montagem dos outros componentes).

A solução foi conseguir um circuito integrado que possuía um decodificador binário decimal sendo na verdade um demultiplexador de 10 portas. Assim ampliou-se a capacidade de controle da porta paralela utilizando menor espaço no protoboard.

### **2.3 Flip-Flops**

De forma geral pode-se representar o flip-flop como um bloco onde temos duas saídas: Q e Q' (inverso de Q), entradas para as variáveis e uma entrada de controle (clock). A saída Q será a principal do bloco.

Este dispositivo possui basicamente dois estados de saída. Para o flip-flop assumir um destes estados é necessário que haja uma combinação das variáveis e do pulso de controle (clock). Após este pulso, o flip-flop permanecerá neste estado até a chegada de um novo pulso de clock e, então, de acordo com as variáveis de entrada, mudará ou não de estado.

O flip-flop tipo D envia a entrada direta para a entrada J e invertida para a entrada K [IDOETA, 1998].

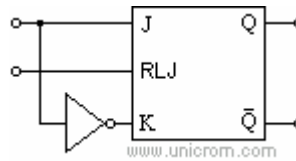


Figura 2.2 - Flip-flop tipo D

Fonte: [www.alldatasheets.com](http://www.alldatasheets.com)

## 2.5 Tabela verdade do circuito

Esta seção tem o objetivo de mostrar a intensidade e qual saída foi selecionada de acordo com os bits enviados pela porta paralela. Para isso é utilizada uma tabela verdade.

As 6 colunas da esquerda representam os dados enviados pela porta paralela com os 2 primeiros bits sendo a intensidade e os outros 4 para o endereço da porta.

As outras 10 colunas mostrarão o nível de intensidade da porta relativo aos bits enviados pela porta paralela (0=desligado, 1=intensidade mínima, 2=intensidade média, 3=intensidade máxima).

Tabela 2.2 - Tabela verdade do circuito do demultiplexador

Bits enviados pela porta paralela						Portas														
						1	2	3	4	5	6	7	8	9	10					
0	0	0	0	0	0	0														
0	0	0	0	0	1		0													
0	0	0	0	1	0			0												
0	0	0	0	1	1				0											
0	0	0	1	0	0					0										
0	0	0	1	0	1						0									
0	0	0	1	1	0							0								
0	0	0	1	1	1								0							
0	0	1	0	0	0													0		
0	0	1	0	0	1														0	
0	0	1	0	1	0															
0	0	1	0	1	1															
0	0	1	1	0	0															
0	0	1	1	0	1															
0	0	1	1	1	0															
0	0	1	1	1	1															
0	1	0	0	0	0	1														
0	1	0	0	0	1		1													
0	1	0	0	1	0				1											
0	1	0	0	1	1					1										
0	1	0	1	0	0						1									
0	1	0	1	0	1							1								
0	1	0	1	1	0								1							
0	1	0	1	1	1									1						
0	1	1	0	0	0														1	
0	1	1	0	0	1															1
0	1	1	0	1	0															
0	1	1	0	1	1															
0	1	1	1	0	0															
0	1	1	1	0	1															
0	1	1	1	1	0															
0	1	1	1	1	1															
0	1	1	1	1	1															

Tabela 2.2 - Tabela verdade do circuito do demultiplexador (continuação)

Bits enviados pela porta paralela						Portas														
						1	2	3	4	5	6	7	8	9	10					
1	0	0	0	0	0	2														
1	0	0	0	0	1		2													
1	0	0	0	1	0			2												
1	0	0	0	1	1				2											
1	0	0	1	0	0					2										
1	0	0	1	0	1						2									
1	0	0	1	1	0							2								
1	0	0	1	1	1								2							
1	0	1	0	0	0													2		
1	0	1	0	0	1														2	
1	0	1	0	1	0															
1	0	1	0	1	1															
1	0	1	1	0	0															
1	0	1	1	0	1															
1	0	1	1	1	0															
1	0	1	1	1	1															
1	1	0	0	0	0	3														
1	1	0	0	0	1		3													
1	1	0	0	1	0			3												
1	1	0	0	1	1				3											
1	1	0	1	0	0					3										
1	1	0	1	0	1						3									
1	1	0	1	1	0							3								
1	1	0	1	1	1								3							
1	1	1	0	0	0													3		
1	1	1	0	0	1														3	
1	1	1	0	1	0															
1	1	1	0	1	1															
1	1	1	1	0	0															
1	1	1	1	0	1															
1	1	1	1	1	0															
1	1	1	1	1	1															

## 2.7 Controlador de intensidade de iluminação

Como resultado final, o sistema tem a intensidade em uma lâmpada de acordo com o comando enviado pelo software. Cada controlador de intensidade é composto por 2 acopladores ópticos que receberão os dois bits vindo do demultiplexador, 2 relés que são acionados pelos acopladores ópticos e ligarão um circuito de corrente alternada correspondente onde um é a lâmpada mais 2 resistores de  $330 \Omega$  (para intensidade média) ligados em paralelo, o outro é a lâmpada mais um resistor de  $470 \Omega$  (para intensidade mínima) e se os dois estiverem acionados ligam um circuito sem resistências onde é observada a intensidade máxima da lâmpada, por último estará a lâmpada de 60 W (onde a mudança de intensidade é mais visível do que uma lâmpada de 15 W conforme testes realizados no projeto) ligadas em tensão alternada de 220 V.

O projeto inicial do controlador de intensidade foi projetado onde a bateria era de corrente contínua com 220 V apenas para ter noção do valor do resistores para teste.



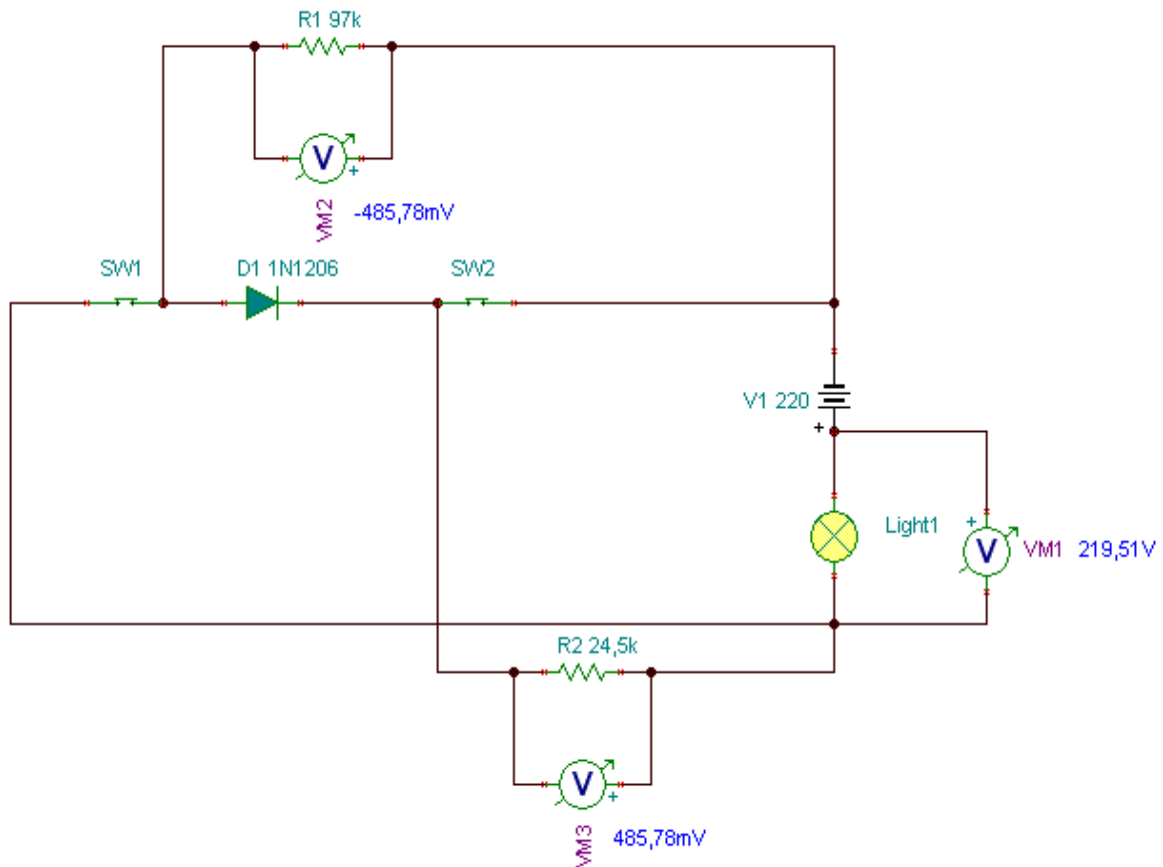


Figura 2.3 - Esquema eletrônico inicial do projeto do controle de intensidade

Nos testes utilizando os componentes no protoboard, por não conseguir no mercado os resistores com o valor da simulação por software, foi utilizado o mais aproximado que foi o de 18 k $\Omega$  de 15 W. Com isso a lâmpada não chegou a acender.

Utilizando potenciômetro descobri que a resistência máxima onde a lâmpada acenderia seria de 520  $\Omega$  (muito abaixo do cálculo com o software). Baseado nisso e na disponibilidade de componentes no mercado, foi utilizado um resistor 470  $\Omega$  para intensidade mínima e 2 resistores de 330  $\Omega$  em paralelo para intensidade média. Para o teste do seletor de intensidade usei interruptores ao invés de relés.

É utilizado no projeto relés que precisam de 5 V (mas nos testes 3 V foram suficientes) para fecharem um circuito de 220 V testados com sucesso. Para ter mais segurança isolando o relé (que já isola o circuito de corrente alternada do de corrente contínua) do circuito digital e obter uma tensão maior para acionamento, são utilizados os acopladores ópticos 4N33 que são acionados com a tensão de 1,8 V vindas do circuito demultiplexador.

Para os cálculos de projeto do circuito responsável por controlar a intensidade são usados o valor RMS de corrente e tensão além de divisor de tensão.

### **2.7.1 Corrente alternada e valor RMS**

A corrente utilizada em residências é alternada, 220 V e 60 Hz (em Brasília). É utilizada pela sua facilidade de transporte em longas distâncias (gerador ou usina para residências).

Existem várias técnicas para gerar tensões alternadas. A mais comum é aquela que alimenta as tomadas domiciliares, ou seja, a usina geradora: estas usinas são em geral alimentadas por quedas d'água, óleo, gás ou fissão nuclear. Em todos os casos o componente mais importante é um gerador de corrente alternada.

A energia oriunda de uma das fontes citadas acima é utilizada para fazer girar um rotor (construído com pólos magnéticos alternados) envolvido pelos enrolamentos do estator (a parte estacionária do gerador).

Utilizando um gerador projetado apropriadamente, é obtido nos terminais de saída uma tensão alternada que, com o auxílio de transformadores, pode ter sua amplitude consideravelmente aumentada para ser distribuída através das linhas de transmissão até chegar ao consumidor.

A corrente alternada atinge valores máximos negativos e positivos (valor de pico), onde (em Brasília por exemplo) 220 V não é o valor máximo atingido. É possível calcular corrente, tensão e potência de forma parecida a corrente contínua, para isso é utilizado um valor onde a potência tanto de corrente

contínua e alternada são parecidas, o valor eficaz ou RMS. Nesse valor são encontrados os 220 V que descrevem a rede elétrica de Brasília.

A maioria dos amperímetros e voltímetros de corrente alternada é projetada para medir o valor médio quadrático (valor RMS) das tensões e correntes alternadas, e não o valor de pico. O valor RMS de uma corrente,  $I_{rms}$ , é definido através da relação:

$I_{rms}$  é igual  $I_{max}$  dividido por  $\sqrt{2}$

O valor RMS de qualquer função senoidal é igual ao valor máximo dessa função dividido por  $\sqrt{2}$  [BOYLESTAD, 1998].

### 2.7.2 Divisor de tensão

Se colocar 2 ou mais resistências em série a tensão de entrada será dividida pelas duas de acordo com o valor de sua resistência. O divisor de tensão será responsável pela variação de intensidade observada na lâmpada porque foi montado um circuito contendo uma resistência e a lâmpada ligadas em série, ou seja, quanto maior a resistência, menor será a intensidade observada na lâmpada.

$V_{rms\ total}$  é igual à Tensão RMS total

$V_{rms\ total}$  é igual À  $V_{rms\ lâmpada}$  mais  $V_{rms\ resistência}$

### 2.7.3 Cálculos utilizados para o controle de intensidade

Tendo em vista que já é conhecida a tensão RMS das residências em Brasília (220 V) e não o valor de pico, além de o multímetro utilizado na implementação medir o valor das tensões RMS do circuito, não existe necessidade de calcular os valores RMS da tensão e corrente pois eles já são fornecidos como constantes. O objetivo é obter a tensão nos resistores e na

lâmpada, visto que já possuindo o valor RMS do gerador de tensão alternada, podemos fazer os cálculos da tensão como em corrente contínua.

$V_{rms}$ (Tensão) é igual à  $I_{rms}$ (corrente) multiplicado por R(resistência)

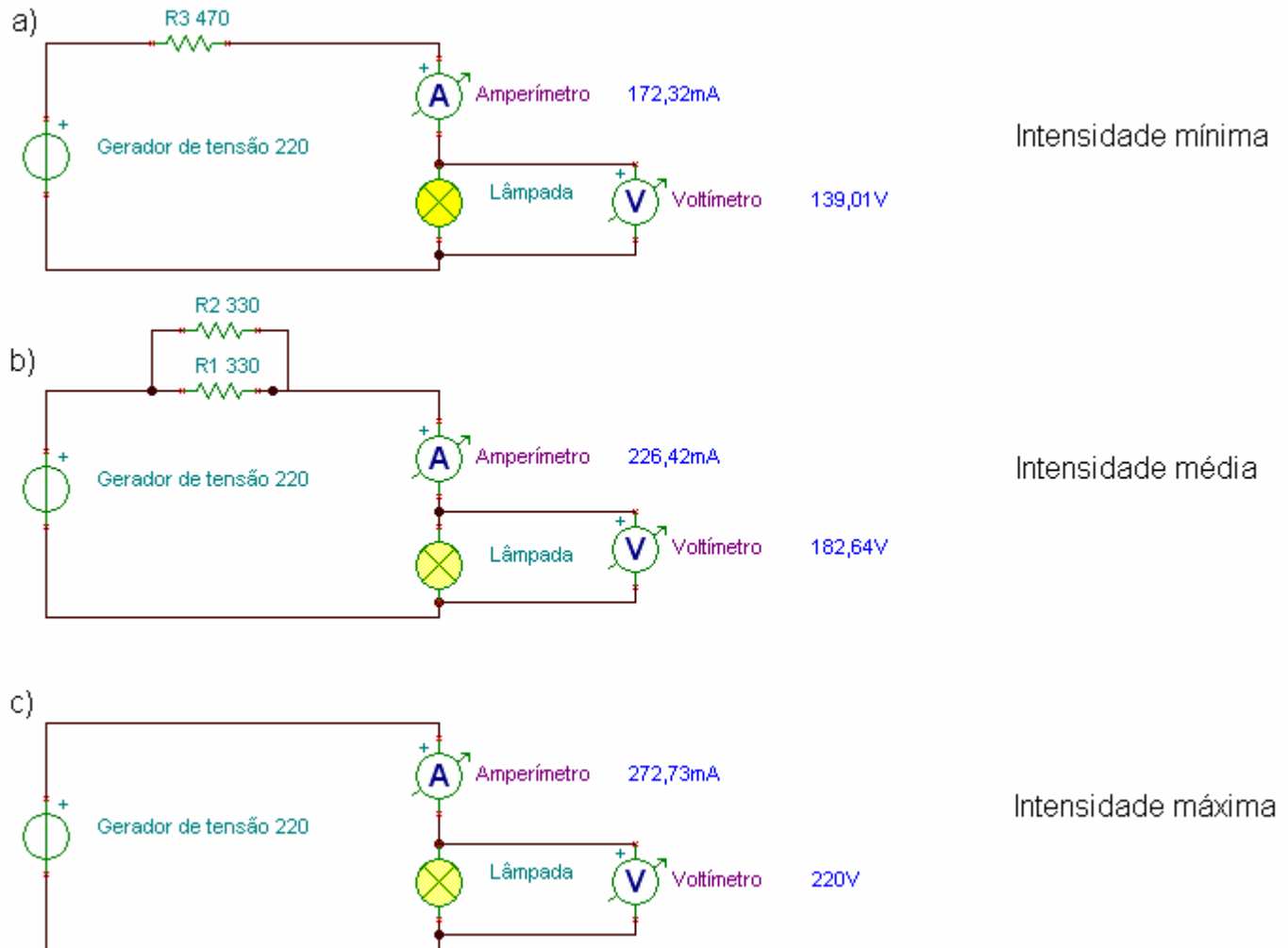


Figura 2.4 - Esquema eletrônico do controle de intensidade

Nas seções seguintes são mostrados os cálculos utilizados para obter variações de intensidade nos dispositivos conectados ao sistema.

### 2.7.3.1 Intensidade máxima (c)

Essa intensidade é observada quando é acionado o circuito sem resistência ligada em série com a lâmpada.

Obs.: Potência ativa é igual à potência aparente em circuitos resistivos.

Potência aparente RMS = corrente RMS multiplicado pela tensão rms

Potência aparente RMS (Pot) = 60 W

Tensão RMS ( $V_{rms}$ ) = 220V

Corrente RMS ( $I_{rms}$ ) = ?

Corrente RMS ( $I_{rms}$ ) = Potência dividida pela tensão

corrente = 60 W / 220V

Corrente RMSs = 272,73mA

Tensão ( $V_{rms}$  lâmpada) na lâmpada = 220 V (A tensão sem passar por resistência)

### 2.7.3.2 Intensidade média (b)

Essa intensidade é observada quando é acionado o circuito com duas resistências de 330  $\Omega$  (ligadas em paralelo entre si) ligadas em série com a lâmpada. É observada uma intensidade na lâmpada menor do que quando é acionada a intensidade máxima.

$$1 / R_{Total} = 1 / 330 + 1 / 330$$

$$1 / R_{Total} = 2/330$$

$$R_{Total} = 330 / 2$$

$$R_{\text{Total}} = 175 \, \Omega$$

$$V_{\text{rms Total}} = 220 \, \text{V}$$

$$V_{\text{rms total}} = V_{\text{rms resistência}} + V_{\text{rms lâmpada}}$$

$V_{\text{rms total}} = \text{Resistência multiplicada pela corrente mais a potência da lâmpada dividida pela corrente}$

$$220\text{V} = 175 \, \Omega \times I + 60 \, \text{W} / I$$

$$\text{Corrente RMS (I)} = 226,42 \, \text{mA}$$

Tensão RMS ( $V_{\text{rms resistência}}$ ) nos resistores em paralelo = 37,36 V

Tensão RMS ( $V_{\text{rms lâmpada}}$ ) na lâmpada = 182,64 V

### 2.7.3.3 Intensidade mínima (a)

Essa intensidade é observada quando é acionado o circuito com uma resistência de  $470 \, \Omega$  ligada em série com a lâmpada. É observada uma intensidade na lâmpada menor do que quando é acionada a intensidade média.

$$R_{\text{Total}} = 470 \, \Omega$$

$$V_{\text{rms Total}} = 220 \, \text{V}$$

$$V_{\text{rms total}} = V_{\text{rms resistência}} + V_{\text{rms lâmpada}}$$

$V_{\text{rms total}} = \text{Resistência multiplicada pela corrente mais potência rms da lâmpada dividida pela corrente.}$

220V é igual à  $470 \Omega$  multiplicado pela corrente mais 60 W dividido pela corrente

Corrente RMS (I) = 172,32 mA

Tensão RMS ( $V_{\text{rms resist\~{e}ncia}}$ ) no resistor = 80,99 V

Tensão RMS ( $V_{\text{rms lâmpada}}$ ) na lâmpada = 139,01 V

No próximo capítulo será descrito o projeto do aplicativo que controla os bits enviados para o hardware descrito. Será demonstrado os diagramas UML, a montagem das classes e banco de dados.

## 3 Software

O sistema web foi desenvolvido utilizando a linguagem Java com interface para o usuário em HTML (trabalhando com os dois pela técnica provida pelas JSPs - Java Server Pages), guardando as informações (descritas posteriormente) geradas em um banco de dados PostgreSQL e usando como interface com a porta paralela os componentes parport e userport.

### 3.1 Diagramação UML

A diagramação UML é utilizada para documentar e modelar sistemas de informação (aplicativos de computador). É focada na programação orientada a objetos onde pode-se relacionar as classes (os objetos) ao sistema de uma forma que outras pessoas que não conheçam UML possam entender.

Os diagramas UML desse capítulo mostram respectivamente o caso de uso geral do sistema mostrando as principais ações que o usuário pode fazer, em seguida temos o diagrama de seqüência do cadastro das portas no sistema, o diagrama de seqüência do controle de intensidade onde serão mostradas todas as portas cadastradas e as opções de mudança e envio de intensidade, o diagrama de seqüência da visualização da estimativa de consumo por período e por último o diagrama de implementação e componentes mostrando a interligação dos componentes utilizados em todo o projeto.

Esses diagramas UML têm o objetivo de ilustrar o funcionamento do projeto.

#### Legenda:

Ator: É quem interage com o sistema mandando e obtendo ações. Pode ser um usuário ou outro sistema automatizado.



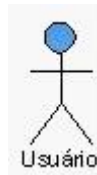


Figura 3.1 - Símbolo do ator dos diagramas UML

Uso: É a ação feita pelo ator. É utilizado para especificar os objetivos do sistema, quais ações o sistema terá que cumprir.



Figura 3.2 - Símbolo do uso dos diagramas UML

Classe: É o molde do objeto (para linguagens orientadas a objetos) onde especificará seus atributos e métodos.



Figura 3.3 - Símbolo da classe dos diagramas UML

Informação: É uma requisição ou dado enviado para uma classe.

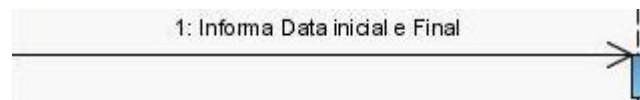


Figura 3.4 - Símbolo da Informação dos diagramas UML

Retorno: É a ação ou informação repassada pela classe ao ator ou outra classe que enviou uma requisição.

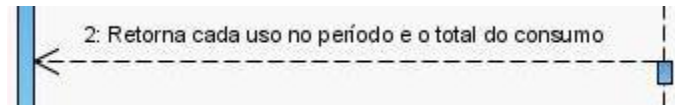


Figura 3.5 - Símbolo da requisição dos diagramas UML

Componente: Representa uma parte da composição estrutural do sistema.



Figura 3.6 - Símbolo do Componente dos diagramas UML

Interface: Representa um dispositivo intermediário entre componentes para sua comunicação.



Figura 3.7 - Símbolo de Interface dos diagramas UML

Implantação: Conjunto de componentes de um sistema:

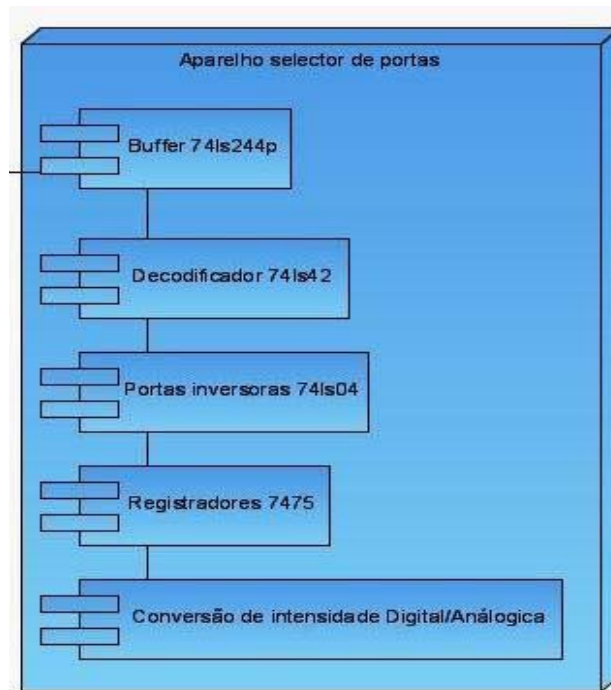


Figura 3.8 - Símbolo de implantação dos diagramas UML

O diagrama de casos de uso do sistema onde o ator "Usuário" terá as ações de cadastrar aparelhos ligados em cada porta, selecionar a intensidade de cada aparelho a ser enviada e visualizar o relatório de consumo:

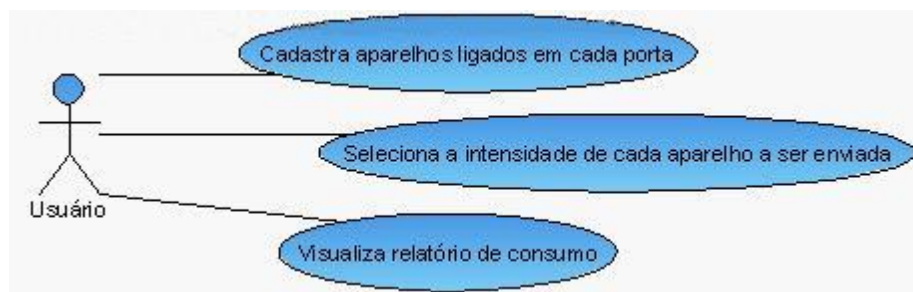


Figura 3.9 - Casos de usos do sistema

A sequência de acontecimentos quando o usuário quiser cadastrar um novo aparelho no sistema. Ele selecionará a porta, a classe aparelho verificará se existe algum registro nessa porta escolhida retornando os dados caso exista, o usuário inserirá novos dados e quando os enviar receberá uma mensagem de confirmação:

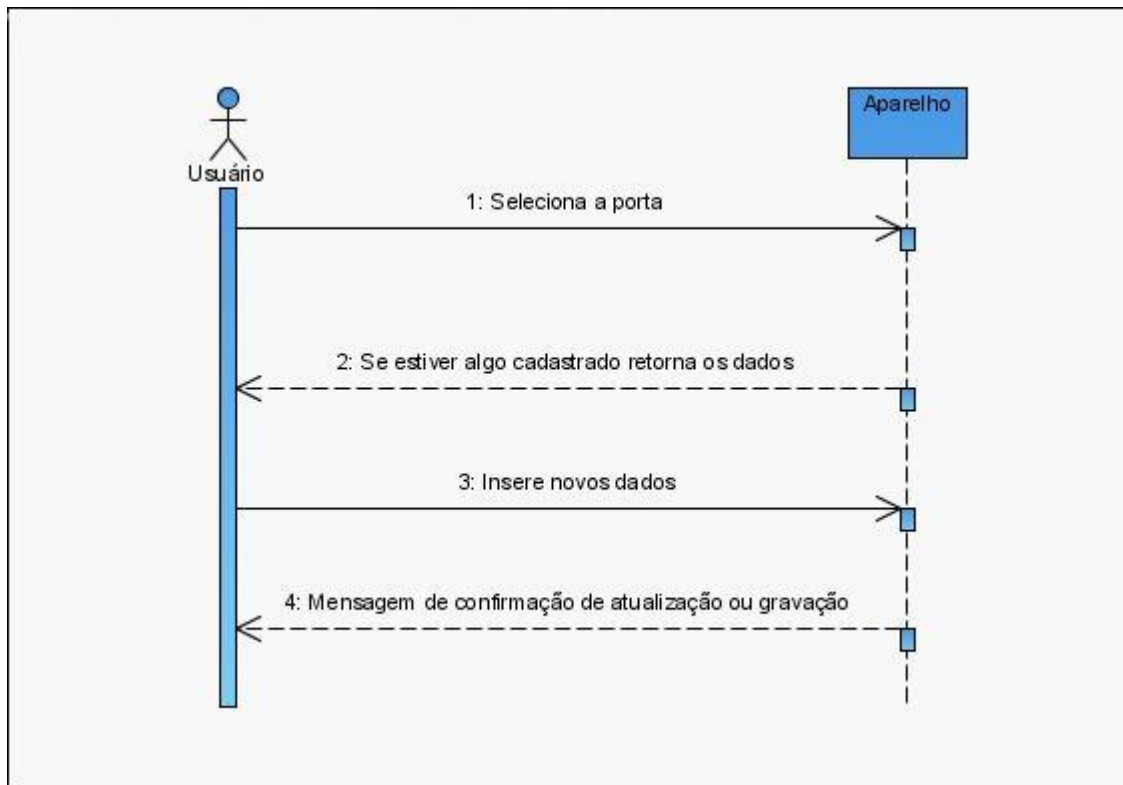


Figura 3.10 - Caso de uso do cadastro de aparelhos por porta

O uso principal do projeto, o envio de intensidade (discreta em 4 níveis: desligado, baixa, média e máxima).

O usuário faz requisição de uma lista de dispositivos para a classe aparelho, ao retornar a lista o usuário seleciona a intensidade em algum dispositivo cadastrado, será cadastrado um novo uso do dispositivo registrando a hora, data e intensidade que foi acionado, por último será enviado os 6 bits (endereço da porta e intensidade) para o hardware (demultiplexador com conversor de bits para intensidade na lâmpada):

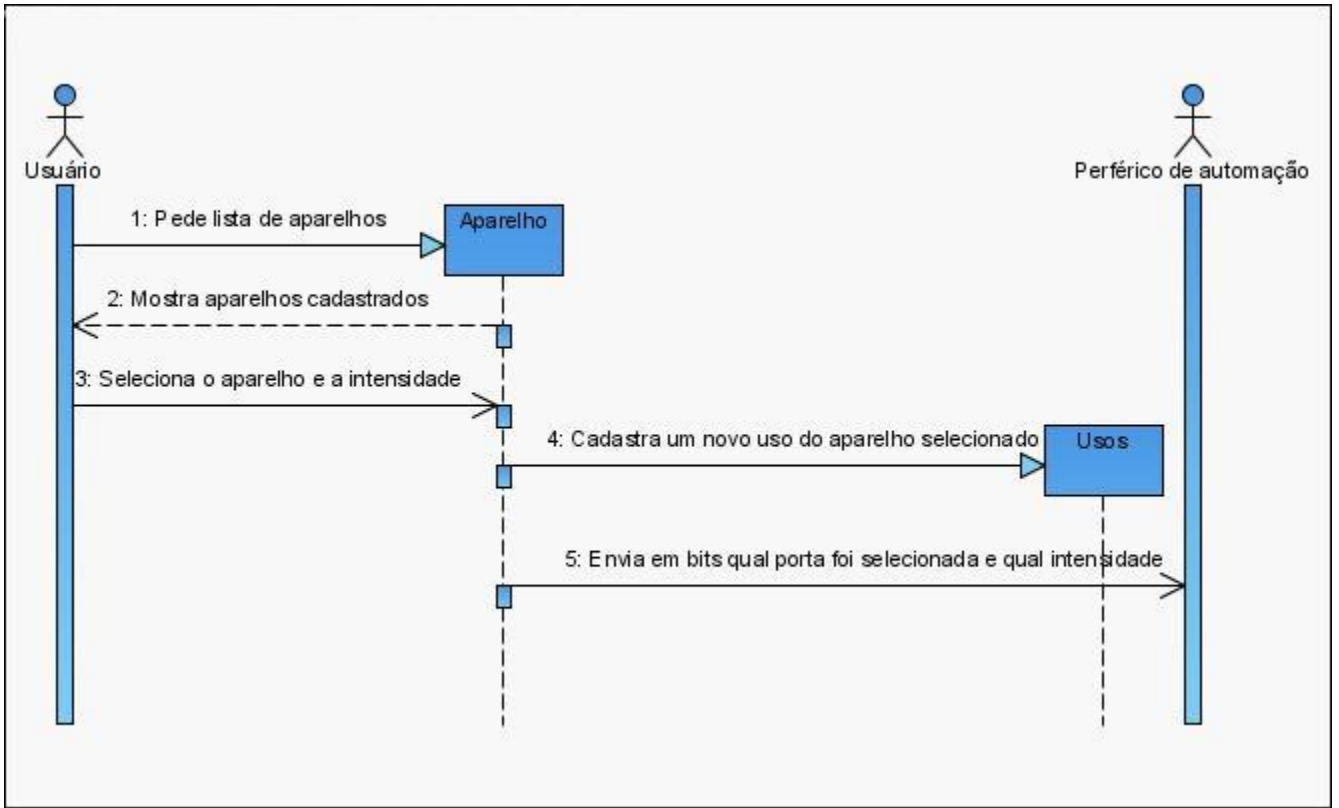


Figura 3.11 - Caso de uso do envio de intensidade

O requerimento do relatório de consumo onde o usuário informa data inicial e final obtendo uma lista e total do consumo nesse período.

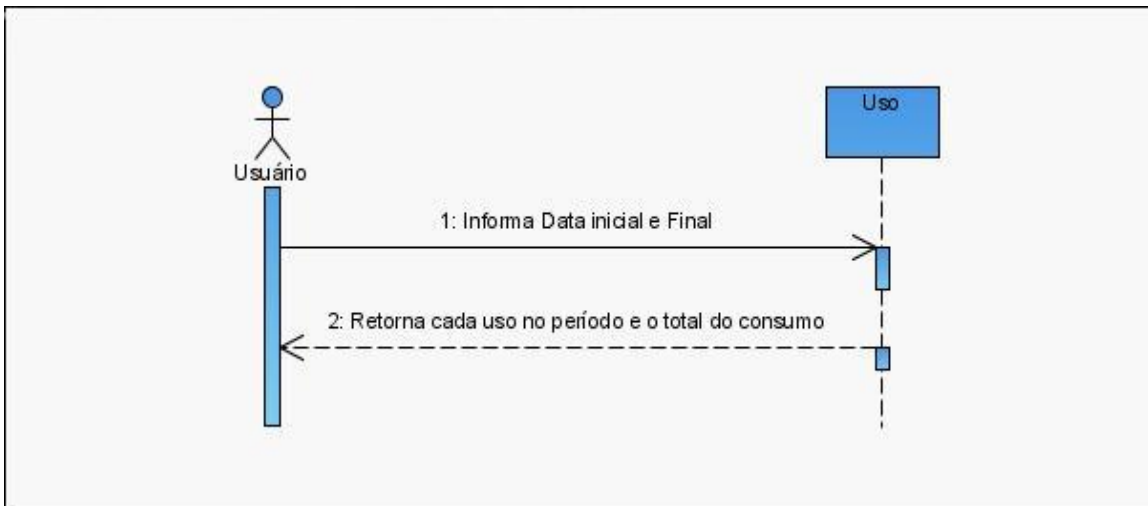


Figura 3.12 - Caso de uso da requisição do relatório de consumo

Os componentes mais relevantes ao projeto (arquivos, classes, banco de dados,...) que estarão no servidor web e os componentes que estarão demultiplexador. Os dois estão ligados pela porta paralela.

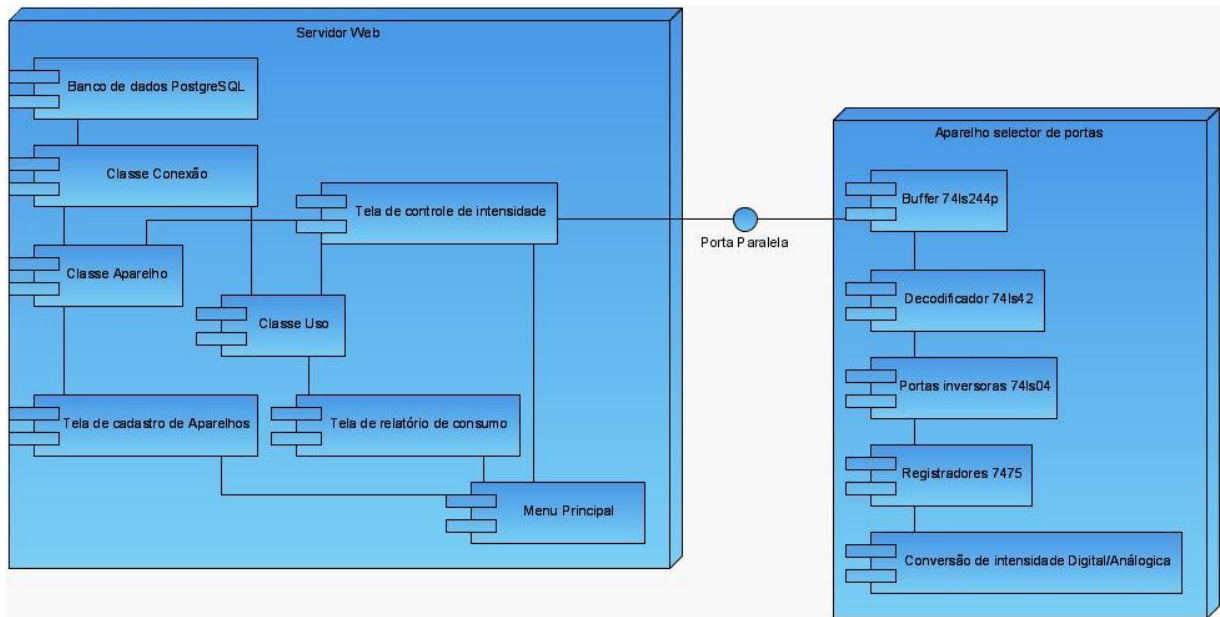


Figura 3.13 - Digrama de implantação do sistema

### 3.3 Linguagem Java

É utilizada a linguagem Java por ser Open Source (não precisando pagar pela licença, assim diminuindo o custo do projeto), pode ser usada para fazer aplicativos para a internet e por ser multiplataforma é possível usar como sistema operacional tanto Windows como Linux ou outro sistema operacional que possua a Máquina Virtual Java instalada. Outras opções poderiam ser o PHP (bastante usado para internet, mas, com menos recursos que o Java) e o ASP.NET (parecido com o Java mas exige pagamento de Licença)

A linguagem de programação Java foi desenvolvida pela empresa Sun Microsystems. É uma linguagem multiplataforma, necessita para seu funcionamento a "Máquina Virtual Java", JVM, feita específica para cada sistema

operacional sendo responsável por intermediar o aplicativo desenvolvido em Java com a respectiva plataforma.

Pode ser utilizada para internet em forma JSP (Java Server Pages) gerando páginas dinamicamente.

Através da implementação de código nativo, interligação entre java e linguagens de baixo nível como C e assembler, a linguagem Java é capaz de controlar dispositivos do computador como a Porta Paralela (usada nesse projeto).

### **3.4 Banco de dados PostgreSQL**

O PostgreSQL é um SGBD (Sistema Gerenciador de Banco de Dados) objeto-relacional de código aberto.

É utilizado no projeto por não precisar pagar licença por seu uso. Existem outras opções que poderiam ser usadas como o também Open Source (Código livre para distribuição e modificação) MySQL e o Access que embora seja pago só precisaríamos de um arquivo para utilizá-lo como banco de dados da aplicação.

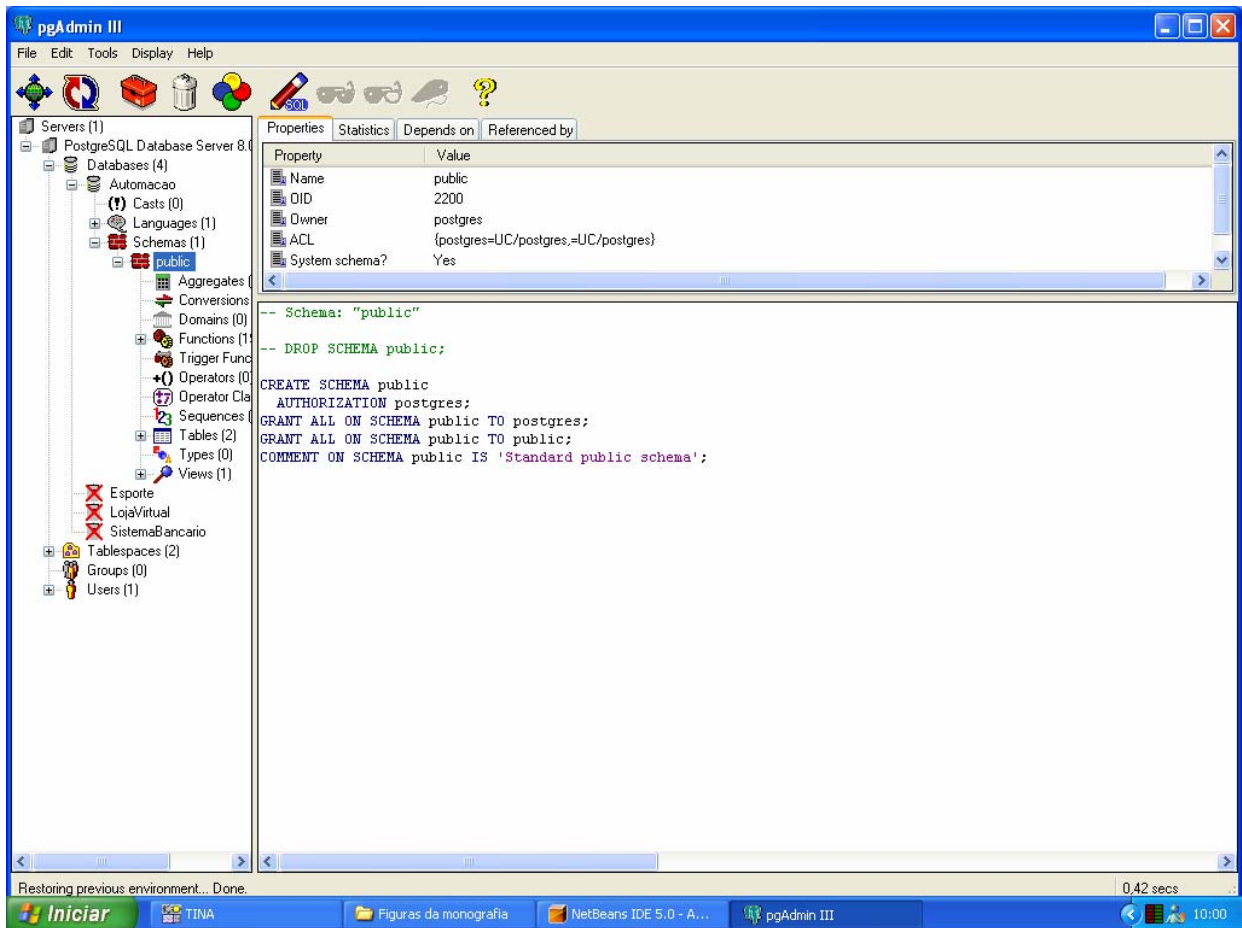


Figura 3.14 - Tela de administração do PostgreSQL (pgAdmin III)

### 3.5 IDE NetBeans

O NetBeans é uma Interface de desenvolvimento de aplicativos da linguagem Java desenvolvido pela empresa Sun Microsystems. Foi utilizado no projeto por já possuir vários componentes, que facilitam o uso, já instalados (diferente da outra opção que seria o Eclipse onde precisamos instalar vários utilitários importantes separadamente) diminuindo assim o tempo de projeto. As ferramentas para desenvolver páginas JSPs, que é a base do aplicativo que está sendo descrito nessa seção, já são instaladas junto com o NetBeans.



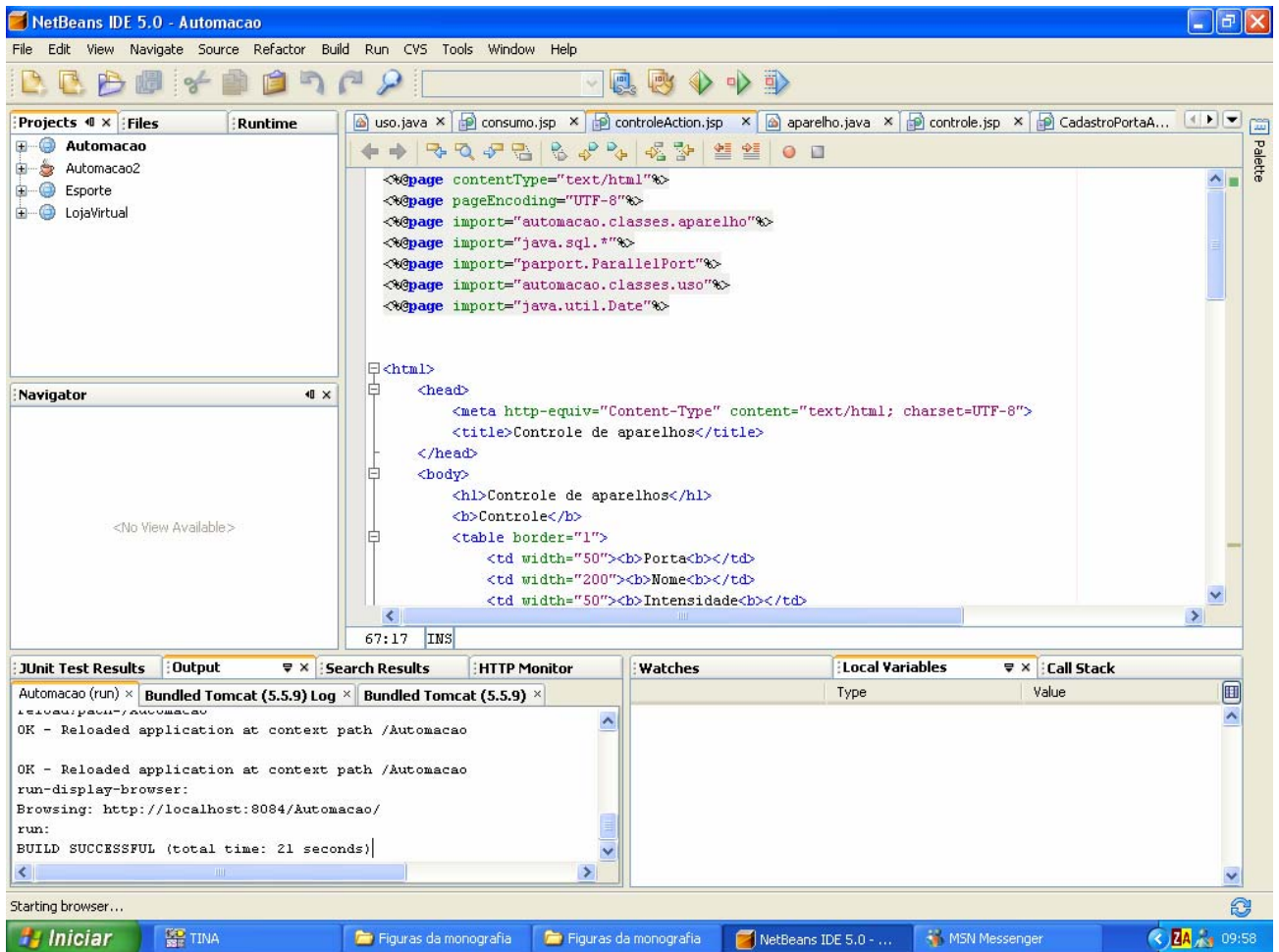


Figura 3.15 - Tela da IDE NetBeans

## 3.6 Problemas encontrados

Durante o desenvolvimento desse aplicativo foram encontrados desafios que foram superados conforme descrito nos próximo tópico.

### 3.6.1 Conseguir fazer a linguagem Java controlar a porta paralela

Durante a implementação do projeto existiu o problema de conseguir mandar os bits de controle para a porta paralela através do aplicativo desenvolvido em Java.

Foi testada a API javax.comm responsável pela comunicação serial e paralela mas não foi obtido nenhum resultado.

A solução foi uma biblioteca (arquivo “.dll”) denominada parport e chamada por método nativo (utilização de código de baixo nível como C e assembler) no Java. Antes de se conseguir o resultado esperado, a solução não funcionou porque alguns arquivos não estavam no lugar certo (a pasta parport tem que ficar no mesmo diretório do projeto e na pasta c:\jdk1.5.1\_06\lib, o arquivo parport.dll tem que ficar localizado nos diretórios c:\window\system32 e c:\jdk1.5.1\_06\bin além do arquivo UserPort.sys ficar localizado na pasta c:\windows\system32\drivers).

### **3.6.2 Utilizar o endereço 0000 do demultiplexador**

Ao utilizar o demultiplexador foi observado que ao selecionar uma porta e voltar ao modo normal do demultiplexador (sem nenhum comando a enviar) a primeira porta (de endereço 0000) era selecionada e mantinha os valores da outra porta selecionada anteriormente.

Para resolver isso após enviar o comando de seleção de porta, é enviado outro comando (1110) selecionando uma porta que não existe no demultiplexador. Assim caso queira acionar a primeira porta ela manterá os valores para ela enviados obtendo o funcionamento pretendido do sistema.

## **3.7 Montagem**

Descreve as estruturas montadas para o software como tabelas de banco de dados (onde ficarão os registros utilizados pelo software), classes básicas do software (como aparelho e uso) e as telas e suas funções.

### **3.7.1 Banco de dados**

O banco de dados é composto pelas seguintes tabelas e seus respectivos campos:

Tabela 3.1 - Tabela aparelho do banco de dados

<b>Aparelho</b>	
Nome	Por exemplo, lâmpada da sala.
Descrição	Permite escrever informações adicionais.
Lugar	Descreve o local em que está o aparelho a ser acionado, por exemplo, garagem, sala...
Porta	Número que identifica a porta no demultiplexador
Portabinario	Valor binário que identifica o equipamento e é enviado pela porta paralela para o demultiplexador indicando o endereço da porta a ser acionada.
Consumo	Valor do consumo médio indicado pelo fabricante do aparelho a ser acionado.
Intensidade	Valor atual da intensidade que está em cada porta, a cada mudança de intensidade esse campo é automaticamente atualizado.
Intensidadebinário	Valor binário do campo intensidade
Uso	Indica o número do registro de uso relativo a última mudança de intensidade da porta responsável por esse registro

Tabela 3.2 - Tabela uso do banco de dados

<b>Uso</b>	
Id	Campo de identificação do registro de uso
Porta	Indica a porta responsável por esse uso
Horario_inicial	Registra o horário do começo desse uso
Horário_final	Registra o horário do fim desse uso
Data_inicial	Registra a data do início desse uso
Data_final	Registra a data do fim desse uso
Intensidade	Registra a intensidade da porta para esse uso

### 3.7.2 Classes Java

Tabela 3.3 - Tabela de classes java do aplicativo

<b>Classes</b>	
Aparelho	Controla a manipulação de dados entre o aplicativo e as informações do banco de dados relativas às portas do demultiplexador.
Uso	Controla a manipulação de dados entre o aplicativo e as informações do banco de dados relativas aos usos das portas.
Conexão	É responsável pelos dados básicos de conexão com o banco de dados como String e funções de conexão.
ParallelPort	É a classe responsável pela manipulação da porta paralela

### 3.7.3 Telas JSP

Tabela 3.4 - Telas do aplicativo

<b>Telas</b>	
Menu principal	É o menu principal do sistema, possuindo a opção de escolher a porta e um botão que levará a outra tela para poder modificar seu cadastro, existe outro botão para entrar na tela de controle de aparelhos além de campos para inserção de data inicial e data final com um botão que mostrará o relatório de estimativa de consumo no período estipulado.
Cadastro de portas	Essa tela é responsável por inserir informações sobre os aparelhos conectados em cada porta.
Controle de intensidade	Mostra os aparelhos cadastrados, possui a opção de mudar a intensidade e um botão para enviar a nova intensidade para a porta paralela.
Consumo	Mostra o relatório de estimativa de consumo no período estipulado na tela principal.

### 3.8 Estimativa de consumo

Para o relatório de estimativa de consumo, o cálculo é feito multiplicando o tempo em que um aparelho elétrico ficou ligado pelo consumo médio cadastrado no sistema.

O relatório mostrará hora inicial e final com a intensidade de cada mudança de intensidade da tela de controle.

### 3.9 Telas

O menu principal do sistema é a primeira tela mostrada pelo aplicativo ao ser iniciado, possuindo a opção de escolher a porta e um botão que levará a outra tela para poder modificar seu cadastro, existe outro botão para entrar na tela de controle de aparelhos além de campos para inserção de data inicial e data final com um botão que mostrará o relatório de estimativa de consumo no período estipulado.

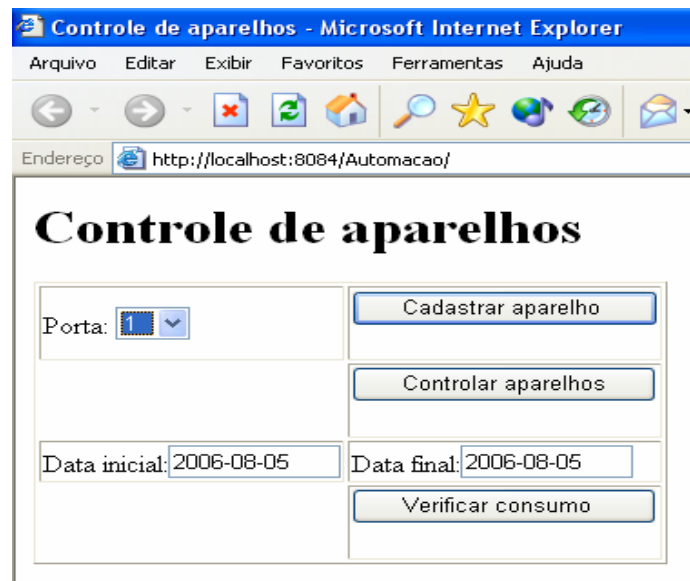
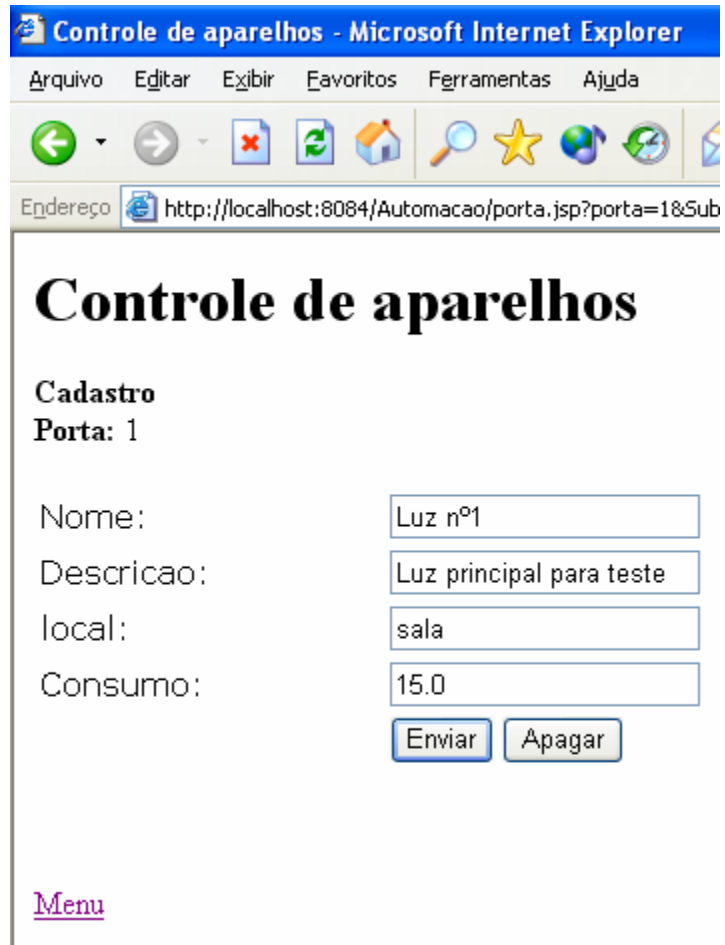


Figura 3.16 - Tela do menu principal

A tela responsável por inserir informações sobre os aparelhos conectados em cada porta:



The image shows a screenshot of a Microsoft Internet Explorer browser window. The title bar reads "Controle de aparelhos - Microsoft Internet Explorer". The address bar shows the URL "http://localhost:8084/Automacao/porta.jsp?porta=1&Sub". The main content area displays the title "Controle de aparelhos" in a large, bold, black font. Below the title, the text "Cadastro" and "Porta: 1" is shown. The form contains four input fields: "Nome:" with the value "Luz nº1", "Descricao:" with "Luz principal para teste", "local:" with "sala", and "Consumo:" with "15.0". At the bottom of the form are two buttons: "Enviar" and "Apagar". A link labeled "Menu" is located at the bottom left of the page.

**Controle de aparelhos**

**Cadastro**  
**Porta: 1**

Nome:

Descricao:

local:

Consumo:

[Menu](#)

Figura 3.17 - Tela do cadastro de aparelhos

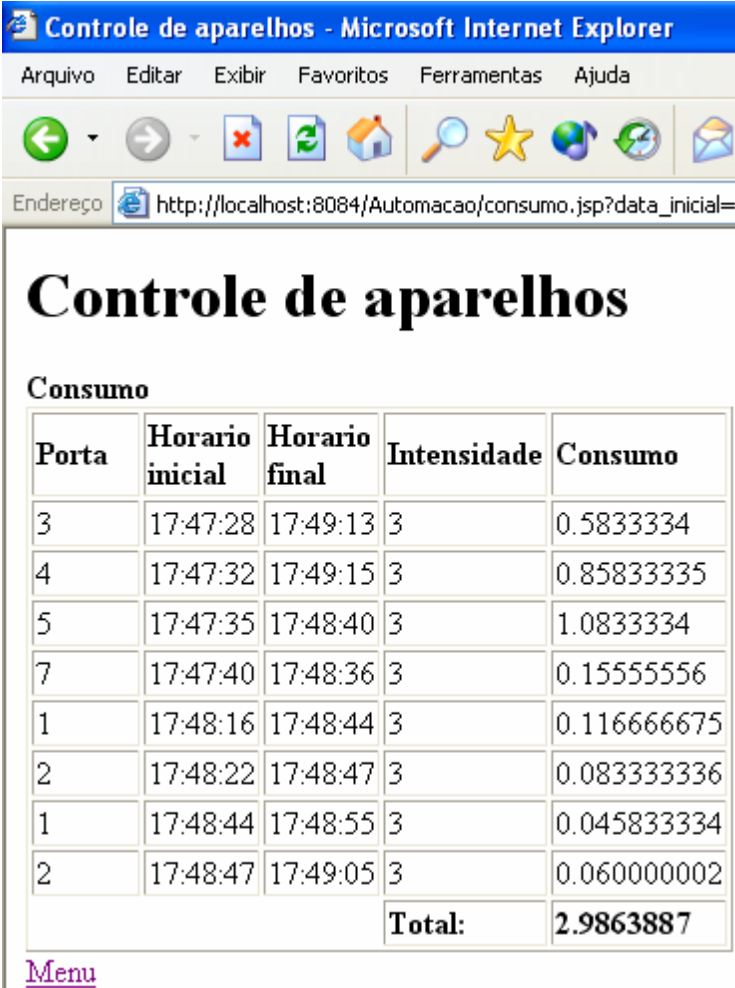
Os aparelhos cadastrados, possui a opção de mudar a intensidade e um botão para enviar a nova intensidade para a porta paralela:

Porta	Nome	Intensidade	
1	Lâmpada principal	0 ▾	Enviar
2	luminaria	0 ▾	Enviar
3	fluorescente	0 ▾	Enviar
4	abajur	3 ▾	Enviar
5	Luminaria	3 ▾	Enviar
7	luz externa	3 ▾	Enviar

[Menu](#)

Figura 3.18 - Tela do controle (envio) de intensidade

O relatório de estimativa de consumo no período estipulado na tela principal:



The screenshot shows a Microsoft Internet Explorer browser window with the title 'Controle de aparelhos - Microsoft Internet Explorer'. The address bar displays 'http://localhost:8084/Automacao/consumo.jsp?data\_inicial=:'.

## Controle de aparelhos

**Consumo**

Porta	Horario inicial	Horario final	Intensidade	Consumo
3	17:47:28	17:49:13	3	0.5833334
4	17:47:32	17:49:15	3	0.85833335
5	17:47:35	17:48:40	3	1.0833334
7	17:47:40	17:48:36	3	0.15555556
1	17:48:16	17:48:44	3	0.116666675
2	17:48:22	17:48:47	3	0.083333336
1	17:48:44	17:48:55	3	0.045833334
2	17:48:47	17:49:05	3	0.060000002
			<b>Total:</b>	<b>2.9863887</b>

[Menu](#)

Figura 3.19 - Tela do relatório de consumo

No próximo capítulo é explicado como foi feito o teste de todo o sistema com os resultados obtidos. É mostrado um exemplo da utilização além das fotos do protótipo.



## 4 Implementação

O objetivo da implementação é cadastrar as lâmpadas no aplicativo, visualizá-las enviando a sua intensidade conseguindo assim observar seu acionamento com a intensidade mandada podendo enviar outros valores de intensidade. Visualizamos um relatório com a estimativa de consumo dos usos citados acima.

São cadastradas lâmpadas que tem no campo “lugar” o valor “auditório”, pois estarão localizadas no auditório, o valor “servirá para a implementação do projeto final” no campo descrição e 60 W e 100 W (Consumo médio) de consumo respectivamente a primeira e a segunda lâmpada.

Visualizamos elas na segunda opção do menu principal onde serão listadas todas as portas cadastradas e podemos mudar a intensidade enviada para cada uma. Aqui se manda intensidade máxima para as lâmpadas e depois enviamos outras intensidades para comprovar o seu funcionamento.

Como ficarão cadastrados os horários em que estiveram ligadas as lâmpadas em cada intensidade emitimos um relatório com o consumo realizado acima (para o protótipo só podemos utilizar como limite dias, não horas), para isso coloca-se os dias limites do período e selecionaremos a última opção (Consumo). Assim visualizam-se quantos Watts foram gastos no período determinado.

Para o controle de intensidade é medida a tensão na lâmpada com intensidade baixa, média e máxima.

### 4.1 Funcionamento completo do sistema

As lâmpadas são conectadas ao hardware e cadastradas no software, para que o sistema saiba da existência delas e em qual porta estão conectadas, em uma tela seguinte as lâmpadas são listadas. Ao modificar a intensidade de alguma porta, o número decimal relativo à intensidade é convertido (pelo software) para binário junto com o número da porta, em um total de 6 bits, esses

seis bits serão enviados para a porta paralela, em seguida é enviado a mesma intensidade em bits junto com um endereço que não existe no hardware (1110, pois o último endereço é o 1001 correspondente à 10ª porta) para fixar o valor da intensidade nos registradores com flip-flops.

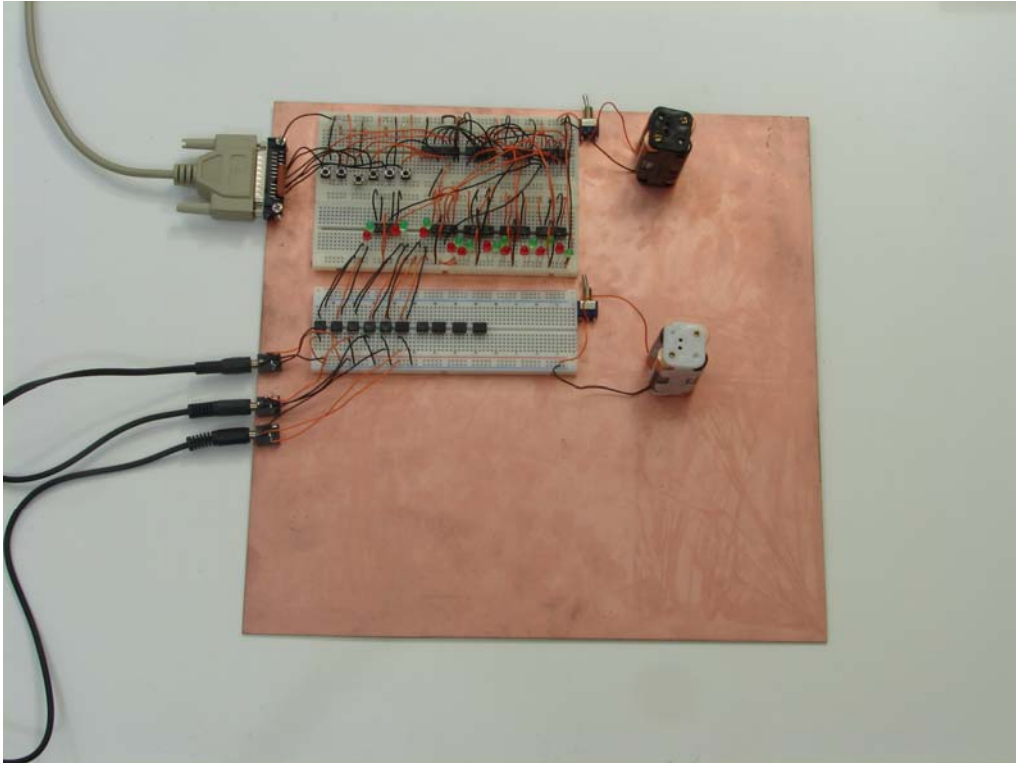


Figura 4.1 - Demultiplexador

Os bits serão recebidos no circuito demultiplexador por um buffer, que tem a função de proteger a porta paralela de variações de corrente que possam danificar o computador, em seguida os bits relativos ao endereço da porta vão para um circuito integrado decodificador binário/decimal acionando o par de flip-flops correspondente ao endereço. Em seguida será recebido o endereço não existente para deixar o par de flip-flops não mais selecionados fixando o valor da intensidade (em bits) e não influenciando em qualquer outra porta, pois se o flip-flop receber nível alto (estiver selecionado) seus bits podem ser modificados, se receber nível baixo (bit = 0), os bits não podem ser modificados armazenando-os até que receba nível alto (bit = 1) de novo.

Os bits relativos à intensidade sairão direto do buffer para todos os flip-flops onde apenas os que tiverem selecionados modificarão para a intensidade atual.

Cada bit em nível alto gerará 1,8 V na saída dos registradores, sendo suficiente para acionar cada acoplador óptico, mas insuficiente para acionar diretamente os relés. Quando acionados, os acopladores ópticos fecham um circuito de tensão contínua com 4 pilhas de 1,5 V acionando o respectivo relé (em um total de 6 V).

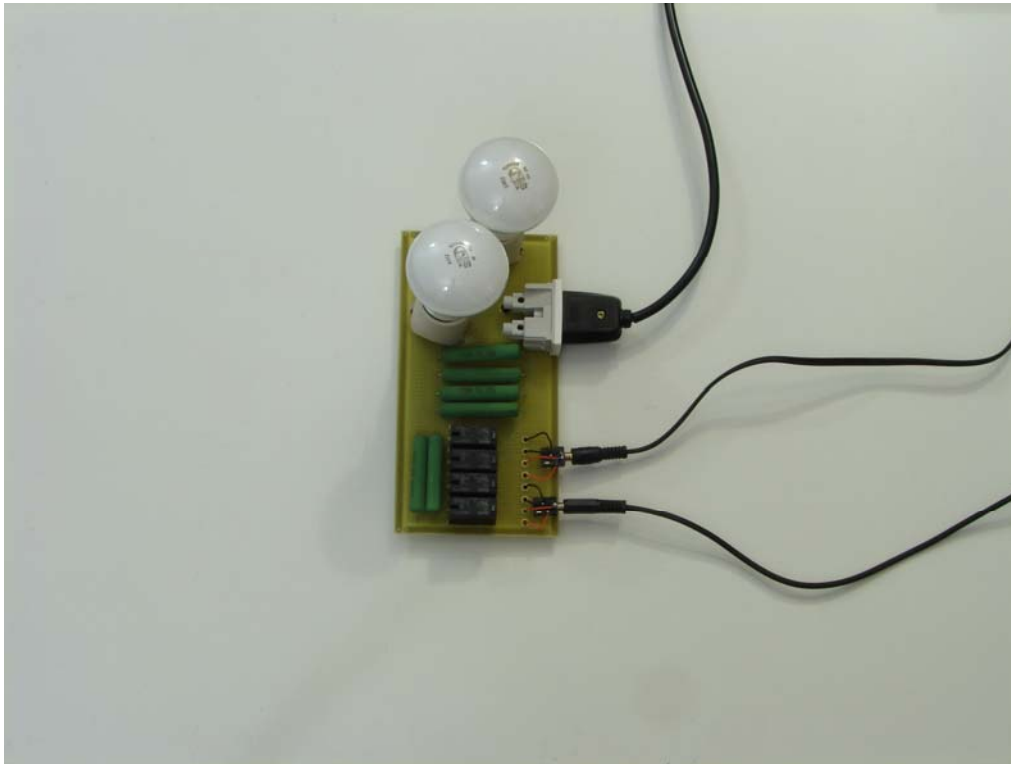


Figura 4.2 – Controlador de intensidade (Lâmpada)

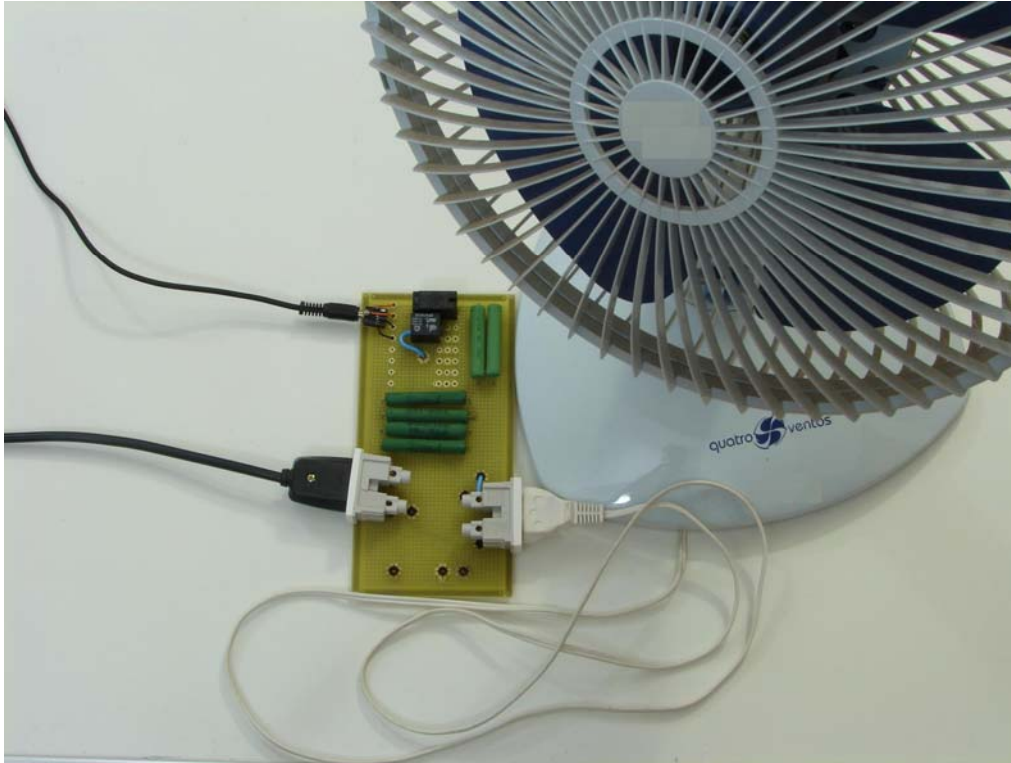


Figura 4.3 – Controlador de intensidade (Outros dispositivos)

A intensidade de uma lâmpada é controlada por um par de relés. Se o primeiro for acionado será ligado um circuito com lâmpada de 60 W em série com um resistor de  $470 \Omega$  e 15 W, se o segundo for acionado é ligado um outro circuito com a lâmpada de 60 W em série com 2 resistores (em paralelo) de  $330 \Omega$  e 15 W cada, por último se acionado os dois relés, é ligado um circuito sem resistência diretamente na lâmpada (a corrente passará pelo caminho de menor resistência desprezando os outros circuitos).

O hardware demultiplexador possui três pilhas de 1,5 V e recebe 4,5 V (Medidos com um multímetro) de cada saída da porta paralela.

## 4.2 Um exemplo de utilização

Nesta seção será mostrado um exemplo de uso do aplicativo para ter uma idéia melhor do seu funcionamento.

1º passo - Cadastrar os aparelhos conforme as figuras abaixo:

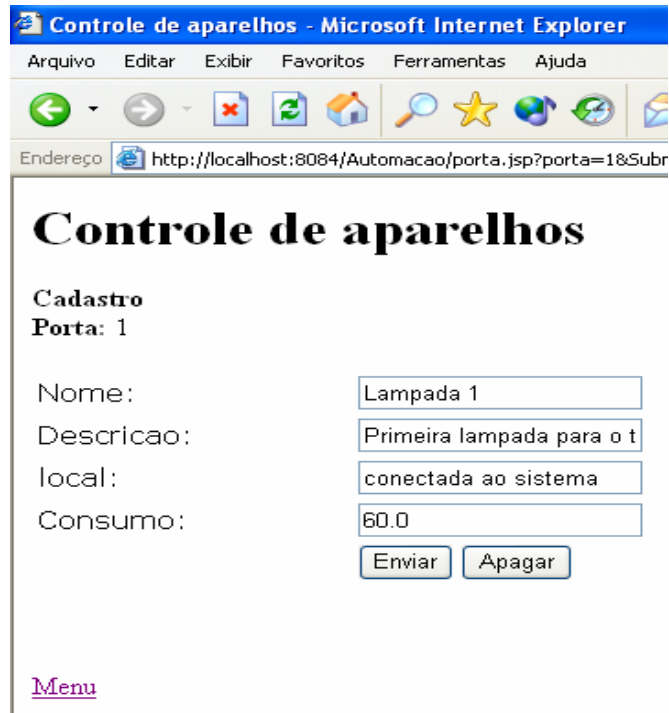


Figura 4.4 – Cadastro da primeira lâmpada

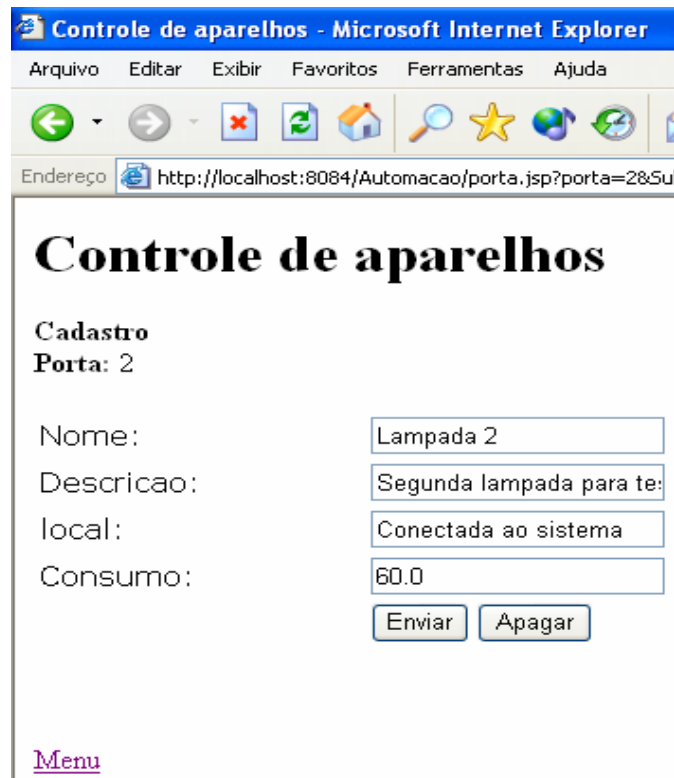


Figura 4.5 – Cadastro da segunda lâmpada

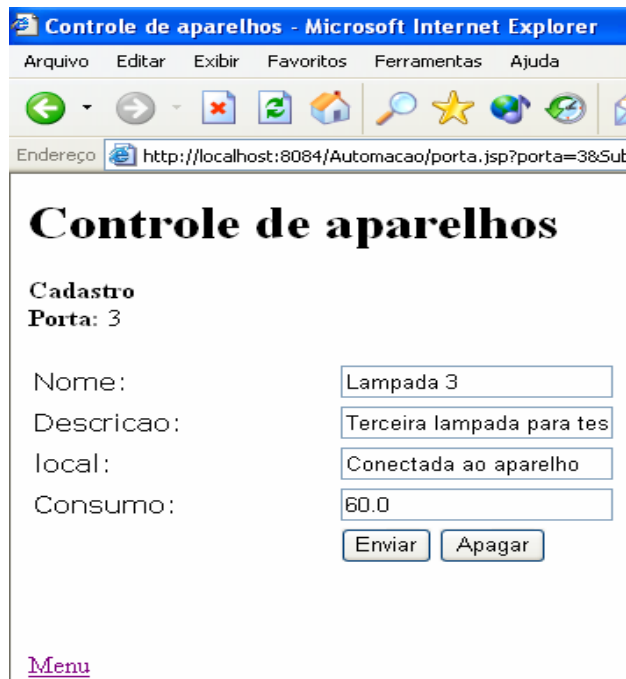


Figura 4.6 – Cadastro da terceira lâmpada

2º passo – Visualizar os aparelhos cadastrados para modificar sua intensidade.

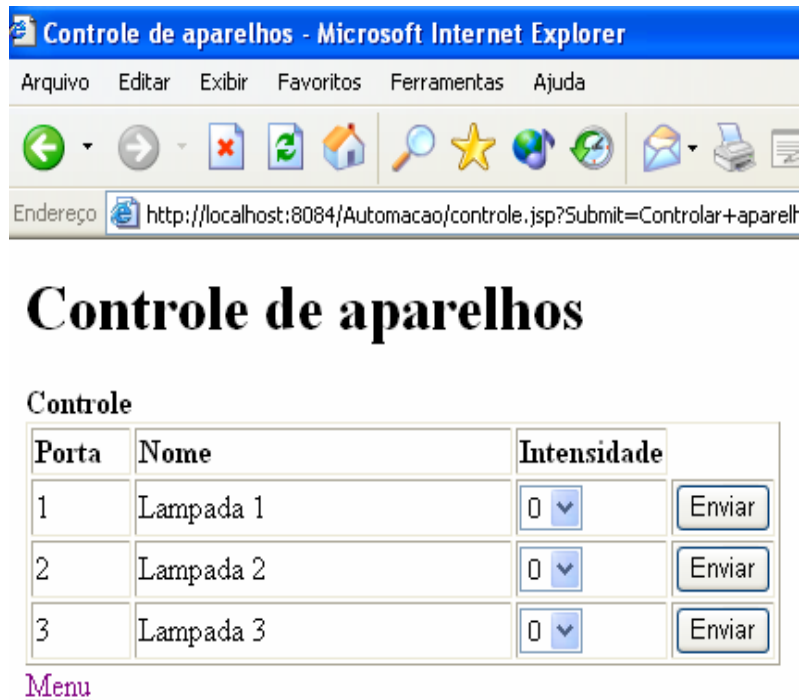


Figura 4.7 – Visualizar os aparelhos cadastrados

3º passo – Modificar a intensidade do primeiro aparelho para 1, do segundo para 2 e do terceiro para 3 (Esses valores não seguem nenhuma regra podendo ser alterados para qualquer outro).

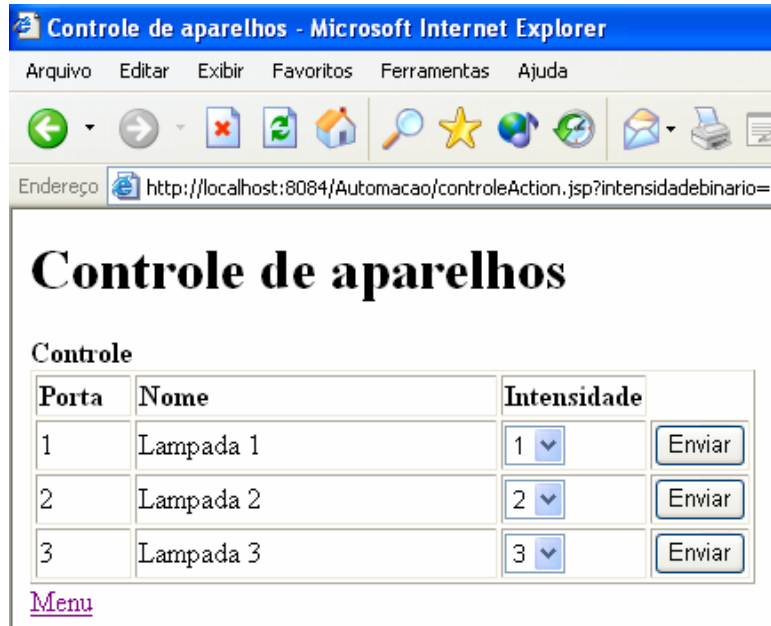


Figura 4.8 – Modificar as intensidades das lâmpadas

4º passo – Visualizar o efeito obtido na lâmpada

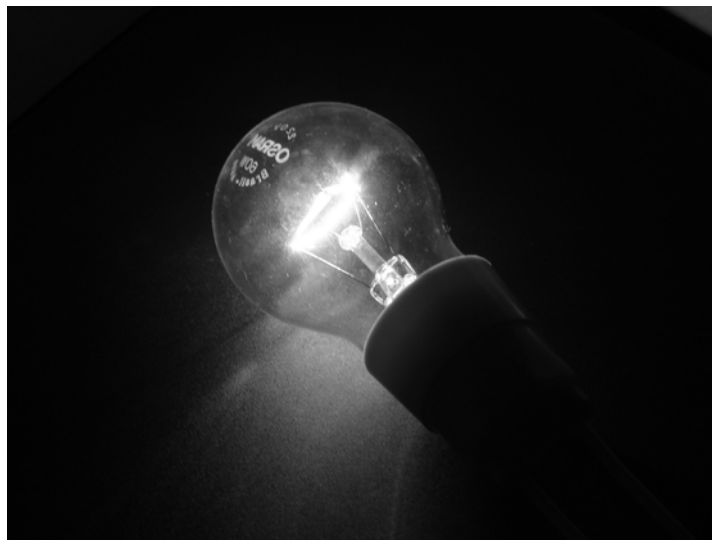


Figura 4.9 – Intensidade na primeira lâmpada



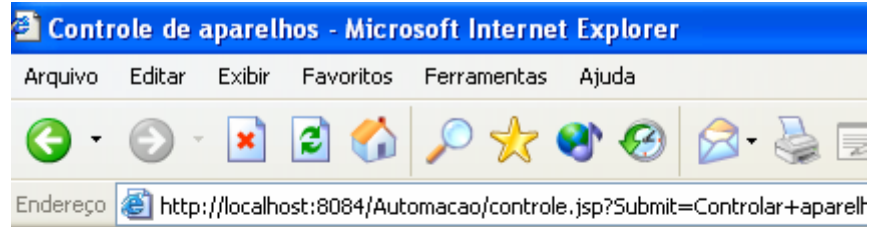
Figura 4.10 – Intensidade na segunda lâmpada



Figura 4.11 – Intensidade na terceira lâmpada



5º passo – modificar a intensidade das três lâmpadas para 0 (desligar).



## Controle de aparelhos

### Controle

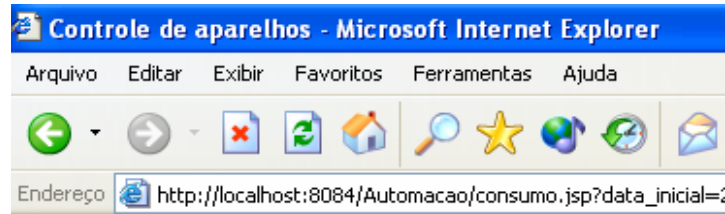
Porta	Nome	Intensidade	
1	Lampada 1	0 ▾	Enviar
2	Lampada 2	0 ▾	Enviar
3	Lampada 3	0 ▾	Enviar

[Menu](#)

Figura 4.12 – Desligar os aparelhos

No decorrer do teste não foi observado nenhum problema de desempenho ocorrendo o funcionamento conforme esperado. Por usar resistores de alta potência (15 W) foi observado uma elevação da temperatura do hardware responsável por controlar a intensidade.

6º passo – Visualizar o relatório de estimativa de consumo de toda a utilização descrita acima:



## Controle de aparelhos

### Consumo

Porta	Horario inicial	Horario final	Intensidade	Consumo
1	14:52:18	14:53:54	1	0.53333336
2	14:52:33	14:53:56	2	0.46111116
3	14:52:37	14:53:59	3	0.4555556
			<b>Total:</b>	<b>1.45</b>

[Menu](#)

Figura 4.13 – Visualizar o relatório com o total de consumo em W

### 4.3 Resultados encontrados

Ao utilizar um voltímetro, para medir tensão de corrente alternada, foram obtidos os seguintes resultados:

Tabela 4.1 - Tensão medida no teste prático do sistema

Valor da resistência	Tensão medida		Tensão Calculada	
	Resistência	Lâmpada de 60W	Resistência	Lâmpada de 60W
2 resistências de 330 $\Omega$ em paralelo	41 V	179 V	37,36 V	182,64 V
1 resistência de 470 $\Omega$	93 V	127 V	80,99 V	139,01 V
Sem resistência	Zero	220 V	Zero	220 V

No próximo capítulo é mostrado o os resultados obtidos com o trabalho mostrando sugestões de projetos futuros utilizando outras soluções. É mostrado desafios encontrados e suas soluções.

## 5 Conclusão

Objetivou-se neste trabalho o desenvolvimento de um software e um hardware de automação residencial. As ferramentas utilizadas foram eletrônica digital, eletrônica analógica, linguagem Java Server Pages, banco de dados PostgreSQL e comunicação utilizando a porta paralela.

Foram encontrados diversos desafios como aumentar o número de saídas da porta paralela de 8 para 10, sendo cada uma com 2 bits para o controle de intensidade (em um total de 20 saídas). Outro desafio foi a comunicação entre o software e a porta paralela, mas, utilizando um dispositivo desenvolvido por terceiros, esse problema foi resolvido.

A primeira parte foi a pesquisa dos componentes utilizados, a segunda foi o desenvolvimento do software, depois tivemos a construção do demultiplexador e por último a construção do controlador de intensidade.

Como resultado obteve-se um sistema que pode ser acionado em rede, controla 10 dispositivos enviando 3 níveis de intensidade mais o estado desligado.

### 5.1 Sugestões para projetos futuros

Para projetos finais de engenharia que queiram continuar a proposta dessa monografia é sugerido:

- Aumentar a segurança do sistema utilizando autenticação por login e senha.
- Aumentar o número de portas controladas utilizando mais demultiplexadores.
- Aumentar a quantidade de níveis de intensidade de lâmpadas utilizando tiristores ou PWM.
- Controlar outros dispositivos elétricos

- Utilizar a porta USB ao invés da porta paralela para comunicação entre o servidor e o hardware.

## Referências

All Datasheets. [www.alldatasheets.com](http://www.alldatasheets.com) Acessado em 10/08/2006

BOLEYSTAD, Robert L., “Introdução a Análise de Circuitos,” Rio de Janeiro, LTC Editora, 1998

FREGNI, Saraiva, “Engenharia do projeto lógico digital: Conceitos e Prática”, São Paulo, Editora Edgard Blucher, 1995

Grupo de usuários java, controle da porta paralela.  
<http://www.guj.com.br/posts/list/40242.java> Acessado em 07/09/2006

IDOETA, Ivan Valeije, CAPUANO, Francisco Gabriel, “Elementos de eletrônica digital”, São Paulo, Editora Érica, 1998

MESSIAS, Antônio Rogério. Porta paralela.  
[www.rogercom.com/pparalela/introducao.htm](http://www.rogercom.com/pparalela/introducao.htm) Acessado em 25/08/2006

SILVA, Vagner Santos. Acesso a Dispositivos externos com interface java  
<http://br.geocities.com/vagnersantosdasilva/artigos/artigojava.htm>  
Acessado em 24/11/06

SPINOLA, Eduardo Oliveira. Conversões – Parte I. Disponível em  
<http://www.devmedia.com.br/articles/viewcomp.asp?comp=2695>  
Acessado em 02/09/2006

## Apêndice A - Código do aplicativo

### A.1 Classe aparelho.java

```
1. /**
2.  *
3.  * @author Glauber
4.  */
5.
6. /*Classe que controla os registros de aparelhos (lâmpadas por exemplo)
7.  *conectados no demultiplexador (hardware que envia a intensidade) e
8.  *cadastrados no banco de dados
9.  */
10.
11. public class aparelho {
12.     //Atributos da classe aparelho
13.
14.     //Número e índice principal que identifica o aparelho no demultiplexador
15.     private String porta;
16.     //Nome que identifica o aparelho
17.     private String nome;
18.     //Informações adicionais sobre o aparelho
19.     private String descricao;
20.     //Local do ambiente onde se localiza o aparelho
21.     private String lugar;
22.     //Valor do consumo por hora passado pelo fabricante do aparelho
23.     private String consumo;
24.     //Intensidade atual que está na porta
25.     private String intensidade;
26.     //Valor binário do atributo porta
```

```
27. private String portabinario;
28. //Valor binário do atributo intensidade
29. private String intensidadebinario;
30.
31. private conexao conec = null;
32. private String sql = "";
33.
34. /** Cria uma nova instância de aparelho */
35. public aparelho() throws Exception{
36.     this.conec = new conexao();
37.
38. }
39.
40.
41. //Funções de encapsulamento dos atributos da classe
42. public String getPorta() {
43.     return porta;
44. }
45.
46. public void setPorta(String porta) {
47.     this.porta = porta;
48. }
49.
50. public String getNome() {
51.     return nome;
52. }
53.
54. public void setNome(String nome) {
55.     this.nome = nome;
56. }
57.
```



```
58. public String getDescricao() {
59.     return descricao;
60. }
61.
62. public void setDescricao(String descricao) {
63.     this.descricao = descricao;
64. }
65.
66. public String getLugar() {
67.     return lugar;
68. }
69.
70. public void setLugar(String lugar) {
71.     this.lugar = lugar;
72. }
73.
74. public String getConsumo() {
75.     return consumo;
76. }
77.
78. public void setConsumo(String consumo) {
79.     this.consumo = consumo;
80. }
81.
82. public String getIntensidade() {
83.     return intensidade;
84. }
85.
86. public void setIntensidade(String intensidade) {
87.     this.intensidade = intensidade;
88. }
```

```

89.
90. public String getPortabinario() {
91.     return portabinario;
92. }
93.
94. public void setPortabinario(String portabinario) {
95.     this.portabinario = portabinario;
96. }
97.
98. public String getIntensidadebinario() {
99.     return intensidadebinario;
100. }
101.
102. public void setIntensidadebinario(String intensidadebinario) {
103.     this.intensidadebinario = intensidadebinario;
104. }
105.
106. /*Insere um novo registro deaparelho no banco de dados
107. *Essa função é usada na JSP "CadastroPortaAction.jsp"
108. */
109. public void inserir(){
110.     try{
111.         sql= "insert into aparelho (porta, nome, descricao, lugar, " +
112.             "consumo, portabinario) values (" +
113.             porta + ", " + nome + ", " + descricao + ", " +
114.             lugar + ", " + consumo + ", " + portabinario + ")";
115.         conec.getSt().executeQuery(sql);
116.
117.     } catch(Exception ex){
118.
119.     }

```

```

120.     }
121.
122.     /*Lista os aparelhos cadastrados no banco de dados
123.     *Essa função é chamada na JSP "controle.jsp" para listar as
124.     *portas que receber intensidade
125.     */
126.
127.     public ResultSet mostraPortas(){
128.         try{
129.             sql="select * from aparelho order by porta";
130.             return conec.getSt().executeQuery(sql);
131.         }catch(Exception ex){
132.             return null;
133.         }
134.     }
135.
136.     /*Seleciona um registro de aparelho específico
137.     *Essa função é chamada na JSP "porta.jsp" para mostrar os
138.     *dados do aparelho já cadastrado para serem atualizados
139.     */
140.
141.     public ResultSet selecionaPortas(){
142.         try{
143.             sql="select * from aparelho where porta=" + porta +
144.             " order by porta";
145.             return conec.getSt().executeQuery(sql);
146.         }catch(Exception ex){
147.             return null;
148.         }
149.     }
150.

```

```

151.
152.     /*Modifica os dados de um registro especifico de aparelho
153.     *Essa Função é chamada na JSP "AtualizaPortaAction.jsp"
154.     */
155.     public void Atualizar(){
156.         try{
157.             sql= "update aparelho set nome=" + nome + ", descricao=" +
158.                 descricao + ", lugar=" + lugar + ", consumo=" + consumo +
159.                 ", portabinario=" + portabinario + " where porta=" + porta;
159.             conec.getSt().executeQuery(sql);
160.
161.         } catch(Exception ex){
162.
163.         }
164.     }
165.
166.     /*Modifica os campos intensidade e seu valor no formato binário
167.     *Essa função será chamada na JSP "controleAction.jsp" para
168.     *atualizar os valores de intensidade modificados no envio de
169.     *intensidade para a porta selecionada
170.     */
171.     public void AtualizarAcionamento(){
172.         try{
173.             sql= "update aparelho set intensidade=" +
174.                 String.valueOf(Integer.parseInt(intensidadebinario, 2)) +
175.                 ", intensidadebinario=" + intensidadebinario +
176.                 "where porta=" + porta;
177.             conec.getSt().executeQuery(sql);
178.
179.         } catch(Exception ex){
180.

```

```
181.     }
182.     }
183.
184.     }
185.
```

## A.2 Conexao.java

```
1. /*
2.  * Conexao.java
3.  *
4.  * Created on 3 de Março de 2006, 15:57
5.  *
6.  * To change this template, choose Tools | Template Manager
7.  * and open the template in the editor.
8.  */
9.
10. package automacao.classes;
11. import java.sql.*;
12.
13. /**
14.  *
15.  * @author Glauber
16.  */
17. public class conexao {
18.
19.     //Atributos da classe
20.     private Connection conn;
21.     private Statement st;
22.     private String sql;
23.
```

```

24.
25.
26.  /** Cria uma nova instância de Conexao */
27.  public conexao() throws Exception {
28.      Class.forName("org.postgresql.Driver");
29.      //Contem a string de conexão com o banco de dados PostgreSQL
30.      this.conn =
31.      DriverManager.getConnection("jdbc:postgresql://localhost:" +
32.          "5432/Automacao?user=postgres&password=rocha");
33.      this.st = conn.createStatement();
34.  }
35.
36.  //Funções de encapsulamento dos atributos da classe
37.  public Connection getConn() {
38.      return conn;
39.  }
40.
41.  public void setConn(Connection conn) {
42.      this.conn = conn;
43.  }
44.
45.  public Statement getSt() {
46.      return st;
47.  }
48.
49.  public void setSt(Statement st) {
50.      this.st = st;
51.  }
52.
53.  public String getSql() {
54.      return sql;

```

```
55. }
56.
57. public void setSql(String sql) {
58.     this.sql = sql;
59. }
60.
61.}
```

### A.3 uso.java

```
1. /*
2.  * uso.java
3.  *
4.  * Created on 15 de Agosto de 2006, 08:28
5.  *
6.  * To change this template, choose Tools | Template Manager
7.  * and open the template in the editor.
8.  */
9.
10. package automacao.classes;
11. import java.sql.*;
12. /**
13.  *
14.  * @author Glauber
15.  */
16. public class uso {
17.     //Atributos da classe
18.
19.     //Indice do registro no banco de dados
20.     private String id;
21.     //Campo de ligação entre os registros de uso e aparelho
```

```

22. private String porta;
23. //Registra a hora inicial do uso
24. private String horario_inicial;
25. //Registra a hora final do uso
26. private String horario_final;
27. //Registra a data inicial do uso
28. private String data_inicial;
29. //Registra a data final do uso
30. private String data_final;
31. //Registra a intensidade enviada
32. private String intensidade;
33. //Registra o consumo cadastrado no aparelho
34. private String consumo;
35.
36. private conexao conec = null;
37. private String sql = "";
38.
39. /** Cria uma nova instância de uso */
40. public uso() throws Exception{
41.     this.conec = new conexao();
42. }
43.
44. //Encapsulamento dos atributos da classe
45. public String getId() {
46.     return id;
47. }
48.
49. public void setId(String id) {
50.     this.id = id;
51. }
52.

```



```
53. public String getPorta() {
54.     return porta;
55. }
56.
57. public void setPorta(String porta) {
58.     this.porta = porta;
59. }
60.
61. public String getHorario_inicial() {
62.     return horario_inicial;
63. }
64.
65. public void setHorario_inicial(String horario_inicial) {
66.     this.horario_inicial = horario_inicial;
67. }
68.
69. public String getHorario_final() {
70.     return horario_final;
71. }
72.
73. public void setHorario_final(String horario_final) {
74.     this.horario_final = horario_final;
75. }
76.
77. public String getData_inicial() {
78.     return data_inicial;
79. }
80.
81. public void setData_inicial(String data_inicial) {
82.     this.data_inicial = data_inicial;
83. }
```

```
84.
85. public String getData_final() {
86.     return data_final;
87. }
88.
89. public void setData_final(String data_final) {
90.     this.data_final = data_final;
91. }
92.
93. public String getIntensidade() {
94.     return intensidade;
95. }
96.
97. public void setIntensidade(String intensidade) {
98.     this.intensidade = intensidade;
99. }
100.
101.     public String getConsumo() {
102.         return consumo;
103.     }
104.
105.     public void setConsumo(String consumo) {
106.         this.consumo = consumo;
107.     }
108.
109.     /*Registra um novo uso no banco de dados atualizando o
110.     *registro de uso antecessor relativo ao mesmo aparelho
111.     * inserindo a data e horário finais calculando o intervalo de
112.     *horário e data
113.     *Essa função é chamada na JSP "controle.jsp" quando é
114.     *enviada uma
```

```

115.         *nova intensidade à porta
116.         */
117.
118.     public void inserir(){
119.         try{
120.             sql= "update uso set horario_final=" + horario_final +
121.                 ", data_final=" + data_final +
122.                 " where id=(select uso from aparelho where porta=" +
123.                 porta + ");" +
124.
125.                 "update uso set diferenca_data=(select " +
126.                 "(data_final-data_inicial) from uso where id=" +
127.                 "(select uso from aparelho where porta=" + porta + ")), " +
128.                 "diferenca_horario=(select (horario_final - " +
129.                 "horario_inicial) from uso where id=(select uso from " +
130.                 "aparelho where porta=" + porta + ")), consumo=" +
131.                 "(select consumo from aparelho where porta=" + porta +
132.                 ") where id=(select uso from aparelho where porta=" +
133.                 porta + ");" +
134.
135.                 "insert into uso (id, porta, horario_inicial, " +
136.                 "data_inicial, intensidade) values " +
137.                 "((select max(id) + 1 from uso), " + porta + ", " +
138.                 "horario_inicial + " + " + data_inicial + " + " +
139.                 "intensidade + ");" +
140.                 "update aparelho set uso=(select max(id) from uso) " +
141.                 "where porta=" + porta;
142.
143.             conec.getSt().executeQuery(sql);
144.
145.         } catch(Exception ex){

```

```

146.
147.     }
148.   }
149.
150.
151.   /*Lista os usos limitados por um período estipulado na tela inicial do
152.   *aplicativo(menu principal) calculando a estimativa de consumo por
153.   *registro e estimativa de consumo total.
154.   *Essa função é chamada na JSP "consumo.jsp"
155.   */
156.   public ResultSet MostrarConsumo(){
157.       try{
158.           sql= "select * from uso where data_inicial >=" + data_inicial +
159.               "" and data_final <=" + data_final + "" and not " +
160.               "diferenca_horario is null order by id";
161.           return conec.getSt().executeQuery(sql);
162.       } catch(Exception ex){
163.           return null;
164.       }
165.   }

```

#### A.4 index.jsp

```

1. <%@page contentType="text/html"%>
2. <%@page pageEncoding="UTF-8"%>
3.
4. <html>
5.   <head>
6.     <meta http-equiv="Content-Type" content="text/html; charset=UTF
7.     -8">

```

```

8.     <title>Controle de aparelhos</title>
9.  </head>
10. <body>
11.     <h1>Controle de aparelhos</h1>
12.     <table border="1">
13.         <tr>
14.             <form name="form1" method="get" action="porta.jsp">
15.                 <td>Porta:
16.                     <select name="porta">
17.                         <%
18.                             /*Faz opções de 10 portas, quantidade existente
19.                             *no demultiplexador
20.                             */
21.                             for (int i=1; i<11; i++){
22.                                 %>
23.                                 <option value="<%=i%>"><%=i%></option>
24.                                 <%
25.                                     }
26.                                 %>
27.                             </select>
28.                         </td>
29.                         <td height="50">
30.                             <input type="submit" name="Submit" value="Cadastrar
31. aparelho ">
32.                             </form>
33.                         </td>
34.                     </tr>
35.                     <tr>
36.                         <form name="form2" method="get" action="controle.jsp">
37.                             <td>
38.                             </td>

```

```

39.         <td height="50" valign="top">
40.             <input type="submit" name="Submit" value="Controlar
41. aparelhos">
42.         </td>
43.     </form>
44. </tr>
45. <tr>
46.
47.     <form name="form3" method="get" action="consumo.jsp">
48.     <td>Data inicial:<input type="text" size="10" name="data_inicial"
49. value="yyyy-mm-dd" >
50.     </td>
51.     <td>Data final:<input type="text" size="10" name="data_final"
52. value="yyyy-mm-dd" >
53.     </td>
54. </tr>
55.     <tr>
56.     <td>
57.     </td>
58.
59.     <td height="50" valign="top">
60.         <input type="submit" name="Submit" value="Verificar
61. consumo ">
62.     </td>
63. </form>
64. </tr>
65.
66. </table>
67.
68. </body>
69. </html>

```

## A.5 porta.jsp

```
1. <%@page contentType="text/html"%>
2. <%@page pageEncoding="UTF-8"%>
3. <%@page import="automacao.classes.aparelho"%>
4. <%@page import="java.sql.*"%>
5. <html>
6.   <head>
7.     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
8.     <title>Controle de aparelhos</title>
9.   </head>
10.  <body>
11.    <h1>Controle de aparelhos</h1>
12.    <b>Cadastro</b><br>
13.    <%
14.      aparelho aparelho = new aparelho();
15.      aparelho.setPorta(request.getParameter("porta"));
16.      /*Verifica se existe porta cadastrada e se existe mostra seus
17. dados
18.      *para modificação
19.      */
20.      ResultSet rs = aparelho.selecionaPortas();
21.
22.      if(rs.next() == true) {
23.    %>
24.
25.    <b>Porta:</b> <%=request.getParameter("porta")%>
26.
27.
28.    <form name="form1" method="get" action="AtualizaPortaAction.jsp">
```

```

29.     <table width="50%">
30.
31.         <tr>
32.             <input name="porta" type="hidden"
33. value="<%=request.getParameter("porta")%>">
34.             <td><font face="Verdana, Arial, Helvetica, sans-
35. serif">Nome:</font></td>
36.             <td><input name="nome" type="text" width="200"
37. value="<%=rs.getString("nome")%>"></td>
38.         </tr>
39.         <tr>
40.             <td><font face="Verdana, Arial, Helvetica, sans-
41. serif">Descricao:</font></td>
42.             <td><input name="descricao" type="text" width="200"
43. value="<%=rs.getString("descricao")%>"></td>
44.         </tr>
45.         <tr>
46.             <td><font face="Verdana, Arial, Helvetica, sans-
47. serif">local:</font></td>
48.             <td><input name="lugar" type="text" width="200"
49. value="<%=rs.getString("lugar")%>"></td>
50.         </tr>
51.         <tr>
52.             <td><font face="Verdana, Arial, Helvetica, sans-
53. serif">Consumo:</font></td>
54.             <td><input name="consumo" type="text" width="200"
55. value="<%=rs.getFloat("consumo")%>"></td>
56.         </tr>
57.         <tr>
58.             <td>&nbsp;</td>
59.             <td><input type="submit" name="Submit" value="Enviar">

```



```

60.         <input type="reset" name="Reset" value="Apagar"></td>
61.     </tr>
62. </table>
63. <p>&nbsp;</p></form></td>
64. </tr>
65. </table>
66. <%}
67.
68.     else {
69. %>
70.
71.
72.     <b>Porta:</b><%=request.getParameter("porta")%>
73.
74.
75.     <form name="form1" method="get"
76. action="CadastroPortaAction.jsp">
77.     <table width="50%">
78.
79.         <tr>
80.             <input name="porta" type="hidden"
81. value="<%=request.getParameter("porta")%>">
82.             <td><font face="Verdana, Arial, Helvetica, sans-
83. serif">Nome:</font></td>
84.             <td><input name="nome" type="text" width="200" ></td>
85.         </tr>
86.         <tr>
87.             <td><font face="Verdana, Arial, Helvetica, sans-
88. serif">Descricao:</font></td>
89.             <td><input name="descricao" type="text" width="200"></td>
90.         </tr>

```

```

91.      <tr>
92.          <td><font face="Verdana, Arial, Helvetica, sans-
93. serif">local:</font></td>
94.          <td><input name="lugar" type="text" width="200"></td>
95.      </tr>
96.      <tr>
97.          <td><font face="Verdana, Arial, Helvetica, sans-
98. serif">Consumo:</font></td>
99.          <td><input name="consumo" type="text" width="200"></td>
100.      </tr>
101.      <tr>
102.          <td>&nbsp;</td>
103.          <td><input type="submit" name="Submit" value="Enviar">
104.          <input type="reset" name="Reset" value="Apagar"></td>
105.      </tr>
106.  </table>
107.  <p>&nbsp;</p></form></td>
108. </tr>
109. </table>
110. <%}%>
111. <a href="index.jsp">Menu</a>
112. </body>
113. </html>

```

## A.6 CadastroPortaAction.jsp

1. <%@page contentType="text/html"%>
2. <%@page pageEncoding="UTF-8"%>
3. <%@page import="automacao.classes.aparelho"%>
4. <%@page import="java.sql.\*"%>

```

5. <%--
6. The taglib directive below imports the JSTL library. If you uncomment it,
7. you must also add the JSTL library to the project. The Add Library... action
8. on Libraries node in Projects view can be used to add the JSTL 1.1 library.
9. --%>
10.<%--
11.<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
12.--%>
13.
14.<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
15. "http://www.w3.org/TR/html4/loose.dtd">
16.
17.<html>
18. <head>
19. <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
20. <title>Controle de aparelhos</title>
21. </head>
22. <body>
23. <h1>Controle de aparelhos</h1>
24. <%
25.     aparelho aparelho = new aparelho();
26.
27. //Atribui o respectivo valor binário ao atributo "portabinario"
28.     int x = Integer.parseInt(request.getParameter("porta"));
29.     switch(x) {
30.         case 1: aparelho.setPortabinario("0000");break;
31.         case 2: aparelho.setPortabinario("0001");break;
32.         case 3: aparelho.setPortabinario("0010");break;
33.         case 4: aparelho.setPortabinario("0011");break;
34.         case 5: aparelho.setPortabinario("0100");break;
35.         case 6: aparelho.setPortabinario("0101");break;

```

```

36.         case 7: aparelho.setPortabinario("0110");break;
37.         case 8: aparelho.setPortabinario("0111");break;
38.         case 9: aparelho.setPortabinario("1000");break;
39.         case 10: aparelho.setPortabinario("1001");break;
40.     }
41.     /*Atualiza os atributos com os valores vindos da página anterior
42.     **"porta.jsp"
43.     */
44.     aparelho.setPorta(request.getParameter("porta"));
45.     aparelho.setNome(request.getParameter("nome"));
46.     aparelho.setDescricao(request.getParameter("descricao"));
47.     aparelho.setLugar(request.getParameter("lugar"));
48.     aparelho.setConsumo(request.getParameter("consumo"));
49.     //Chama a função para inserir os dados no banco de dados
50.     aparelho.inserir();
51.
52.
53.
54.     %>
55.     Aparelho atualizado <br>
56.     <a href="index.jsp">Menu</a>
57.
58. </body>
59. </html>

```

### A.7 AtualizaPortaAction.jsp

```

1. <%@page contentType="text/html"%>
2. <%@page pageEncoding="UTF-8"%>
3. <%@page import="automacao.classes.aparelho"%>
4. <%@page import="java.sql.*"%>

```

```
5.
6. <html>
7.   <head>
8.     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9.     <title>Controle de aparelhos</title>
10.  </head>
11.  <body>
12.    <h1>Controle de aparelhos</h1>
13.    <%
14.      aparelho aparelho = new aparelho();
15.
16.      //Atribui o respectivo valor binário ao atributo "portabinario"
17.      int x = Integer.parseInt(request.getParameter("porta"));
18.      switch(x) {
19.        case 1: aparelho.setPortabinario("0000");break;
20.        case 2: aparelho.setPortabinario("0001");break;
21.        case 3: aparelho.setPortabinario("0010");break;
22.        case 4: aparelho.setPortabinario("0011");break;
23.        case 5: aparelho.setPortabinario("0100");break;
24.        case 6: aparelho.setPortabinario("0101");break;
25.        case 7: aparelho.setPortabinario("0110");break;
26.        case 8: aparelho.setPortabinario("0111");break;
27.        case 9: aparelho.setPortabinario("1000");break;
28.        case 10: aparelho.setPortabinario("1001");break;
29.      }
30.      /*Atualiza os atributos com os valores vindos da página anterior
31.      *"porta.jsp"
32.      */
33.      aparelho.setPorta(request.getParameter("porta"));
34.      aparelho.setNome(request.getParameter("nome"));
35.      aparelho.setDescricao(request.getParameter("descricao"));
```

```

36.     aparelho.setLugar(request.getParameter("lugar"));
37.     aparelho.setConsumo(request.getParameter("consumo"));
38.     //Chama a função para atualizar os dados no banco de dados
39.     aparelho.Atualizar();
40.     %>
41.     Aparelho atualizado <br>
42.
43.     <a href="index.jsp">Menu</a>
44. </body>
45.</html>

```

## A.8 controle.jsp

```

1. <%@page contentType="text/html"%>
2. <%@page pageEncoding="UTF-8"%>
3. <%@page import="automacao.classes.aparelho"%>
4. <%@page import="java.sql.*"%>
5.
6.
7. <html>
8.   <head>
9.     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
10.    <title>Controle de aparelhos</title>
11.  </head>
12.  <body>
13.    <h1>Controle de aparelhos</h1>
14.    <b>Controle</b>
15.    <table border="1">
16.      <td width="50"><b>Porta<b></td>

```

```

17.      <td width="200"><b>Nome<b></td>
18.      <td width="50"><b>Intensidade<b></td>
19.      <td width="50"><b><b></td>
20.      <%  aparelho aparelho = new aparelho();
21.
22.          //Lista os aparelhos cadastrados
23.          ResultSet rs = aparelho.mostraPortas();
24.          while(rs.next()){
25.      %>
26.
27.      <tr>
28.          <td width="50"><%=rs.getInt("porta")%></td>
29.          <td width="200"><%=rs.getString("nome")%></td>
30.          <form name="form1" method="get" action="controleAction.jsp">
31.              <td width="50">
32.                  <select name="intensidadebinario">
33.                      <option selected
34.                          value="<%=rs.getString("intensidadebinario")%>"><%=
35.                          rs.getInt("intensidade")%>
36.                  </option>
37.                      <option value="00">0</option>
38.                      <option value="01">1</option>
39.                      <option value="10">2</option>
40.                      <option value="11">3</option>
41.                  </select>
42.              </td>
43.              <input name="porta" type="hidden" width="200"
44. value="<%=rs.getString("porta")%>">
45.                  <input name="portabinario" type="hidden" width="200"
46. value="<%=rs.getString("portabinario")%>">
47.              <td width="50"><input type="submit" name="Submit"

```

```

48. value="Enviar"></td>
49.     </form>
50. </tr>
51. <%}%>
52. </table>
53. <a href="index.jsp">Menu</a>
54. </body>
55. </html>

```

### A.9 ControleAction.jsp

```

1. <%@page contentType="text/html"%>
2. <%@page pageEncoding="UTF-8"%>
3. <%@page import="automacao.classes.aparelho"%>
4. <%@page import="java.sql.*"%>
5. <%@page import="parport.ParallelPort"%>
6. <%@page import="automacao.classes.uso"%>
7. <%@page import="java.util.Date"%>
8.
9.
10. <html>
11. <head>
12. <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13. <title>Controle de aparelhos</title>
14. </head>
15. <body>
16. <h1>Controle de aparelhos</h1>
17. <b>Controle</b>
18. <table border="1">
19. <td width="50"><b>Porta<b></td>
20. <td width="200"><b>Nome<b></td>

```



```

21.      <td width="50"><b>Intensidade<b></td>
22.      <td width="50"><b><b></td>
23.      <% aparelho aparelho = new aparelho();
24.          aparelho.setPorta(request.getParameter("porta"));
25.          aparelho.setIntensidadebinario(request.getParameter("intensida
26. debinario"));
27.
28.          /*Atualiza os campos intensidade e intensidadebinario do
29.             *registro de aparelho
30.             */
31.          aparelho.AtualizarAcionamento();
32.
33.          /*Lista os aparelhos cadastrados no banco de dados como na
34.             *página anterior "controle.jsp"
35.             */
36.          ResultSet rs = aparelho.mostraPortas();
37.          ParallelPort lpt1 = new ParallelPort(0x378);
38.
39.          //Envia a intensidade para o aparelho selecionado
40.          lpt1.write(Integer.parseInt(request.getParameter("intensidadebin
41. ario") + request.getParameter("portabinario"), 2));
42.          lpt1.write(Integer.parseInt(request.getParameter("intensidadebin
43. ario") + request.getParameter("portabinario"), 2));
44.          lpt1.write(Integer.parseInt(request.getParameter("intensidadebin
45. ario") + request.getParameter("portabinario"), 2));
46.          uso uso = new uso();
47.          Date data = new Date();
48.          uso.setData_final((data.getYear() - 100 + 2000) + "-" +
49. data.getMonth() + "-" + data.getDay());
50.          uso.setData_inicial((data.getYear() - 100 + 2000) + "-" +
51. data.getMonth() + "-" + data.getDay());

```

```

52.         uso.setHorario_final(data.getHours() + ":" + data.getMinutes() +
53. ":" + data.getSeconds());
54.         uso.setHorario_inicial(data.getHours() + ":" + data.getMinutes()
55. + ":" + data.getSeconds());
56.         uso.setIntensidade(String.valueOf(Integer.parseInt(request.getP
57. arameter("intensidadebinario"), 2)));
58.         uso.setPorta(request.getParameter("porta"));
59.
60.         /*Cadastra um novo uso no banco de dados para futuro calculo
61.         *de estimativa de consumo
62.         */
63.         uso.inserir();
64.
65.         while(rs.next()){
66.             %>
67.
68.             <tr>
69.                 <td width="50"><%=rs.getInt("porta")%></td>
70.                 <td width="200"><%=rs.getString("nome")%></td>
71.                 <form name="form1" method="get" action="controleAction.jsp">
72.                     <td width="50">
73.                         <select name="intensidadebinario">
74.                             <option selected
75. value="<%=rs.getString("intensidadebinario")%>"><%=rs.getInt("intensid
76. ade")%></option>
77.                             <option value="00">0</option>
78.                             <option value="01">1</option>
79.                             <option value="10">2</option>
80.                             <option value="11">3</option>
81.                         </select>
82.                     </td>

```

```

83.         <input name="porta" type="hidden" width="200"
84. value="<%=rs.getString("porta")%>">
85.         <input name="portabinario" type="hidden" width="200"
86. value="<%=rs.getString("portabinario")%>">
87.         <td width="50"><input type="submit" name="Submit"
88. value="Enviar"></td>
89.     </form>
90. </tr>
91. <%}
92.
93.     /*Envia a intensidade e um endereço de porta que não existe no
94.     *demultiplexador para fixar a intensidade nos flip-flops
95.     */
96.     lpt1.write(Integer.parseInt(request.getParameter("intensidadebin
97. ario") + "1110", 2));
98.     lpt1.write(Integer.parseInt(request.getParameter("intensidadebin
99. ario") + "1110", 2));
100.     /*Envia apenas um endereço que não existe no demultiplexador
101.     *(para não interferir nas outras portas)
102.     *finalizando o procedimento
103.     */
104.     lpt1.write(Integer.parseInt("00001110", 2));
105.     %>
106. </table>
107.     <a href="index.jsp">Menu</a>
108. </body>
109. </html>

```

## A.10 consumo.jsp

```

1. <%@page contentType="text/html"%>

```

2. `<%@page pageEncoding="UTF-8"%>`
3. `<%@page import="gnu.io.*"%>`
4. `<%@page import="java.io.*"%>`
5. `<%@page import="automacao.classes.uso"%>`
6. `<%@page import="java.sql.*"%>`
7. `<%--`
8. The taglib directive below imports the JSTL library. If you uncomment it,
9. you must also add the JSTL library to the project. The Add Library... action
10. on Libraries node in Projects view can be used to add the JSTL 1.1 library.
11. `--%>`
12. `<%--`
13. `<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>`
14. `--%>`
- 15.
16. `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"`
17. `"http://www.w3.org/TR/html4/loose.dtd">`
- 18.
19. `<html>`
20. `<head>`
21. `<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">`
22. `<title>Controle de aparelhos</title>`
23. `</head>`
24. `<body>`
25. `<h1>Controle de aparelhos</h1>`
26. `<b>Consumo</b>`
- 27.
28. `<table border="1">`
29. `<td width="10"><b>Porta<b></td>`
30. `<td width="50"><b>Horario inicial<b></td>`
31. `<td width="50"><b>Horario final<b></td>`
32. `<td width="50"><b>Intensidade<b></td>`

```

33. <td width="50"><b>Consumo<b></td>
34. <% float total = 0;
35.     uso uso = new uso();
36.
37.     //Busca o período determinado na página anterior "index.jsp"
38.     uso.setData_inicial(request.getParameter("data_inicial"));
39.     uso.setData_final(request.getParameter("data_final"));
40.
41.     /*Lista os registros de uso no período determinado na página
42.     *anterior "index.jsp"
43.     */
44.     ResultSet rs = uso.MostrarConsumo();
45.     while(rs.next()){
46. %>
47.
48. <tr>
49.     <td width="50"><%=rs.getInt("porta")%></td>
50.     <td width="50"><%=rs.getTime("horario_inicial")%></td>
51.     <td width="50"><%=rs.getTime("horario_final")%></td>
52.     <td width="50"><%=rs.getInt("intensidade")%></td>
53.     <%//Calcula o consumo por registro
54.
55.         //Variável para diferença entre horario final e inicial
56.         String dh;
57.         //Variáveis para diferença de datas, hora, minuto e segundo
58.         int dd, h, m, s, intensidade;
59.         //Variáveis para consumo e consumo total
60.         float consumo, ct;
61.
62.         /*Conversão dos valores para os tipos necessários para os
63.         *cálculos de consumo

```

```

64.         */
65.         consumo = rs.getFloat("consumo");
66.         intensidade = rs.getInt("intensidade");
67.         dh= rs.getString("diferenca_horario");
68.         dh=String.valueOf(dh);
69.         h= Integer.parseInt(dh.substring(0,2));
70.         m= Integer.parseInt(dh.substring(3,5));
71.         s= Integer.parseInt(dh.substring(6,8));
72.         dd= rs.getInt("diferenca_data");
73.
74.         /*Expressão para calcular o consumo feito relativo a cada
75.         *registro
76.         */
77.         ct = (((dd*24*3600 + s + m*60 + h*3600) * (consumo/3600)) /
78. 3);
79.         /*Expressão que acumula e calcula o valor total do período
80.         *estipulado
81.         */
82.         total= total + ct;
83.         %>
84.         <td width="50"><%=ct%></td>
85.     </tr>
86.     <%
87.         }
88.     %>
89.     <tr>
90. <td></td><td></td><td></td><td><b>Total:<b></td><td><b><%=total%>
91. <b></td></tr>
92. </table>
93. <a href="index.jsp">Menu</a>
94.

```

95. </body>

96.</html>

## Apêndice B - Circuitos integrados utilizados

Serão explicados os circuitos integrados que foram utilizados nesse projeto. Esses circuitos foram escolhidos por serem de fácil obtenção no mercado.

### B.1 Família de circuitos TTL – Transistor-Transistor Logic

A família TTL possui como entrada transistores multi-emissor. O estágio de saída é feito por um circuito com dois transistores: um que “puxa” (pull) corrente para o terra e outro que, alternativamente, “empurra” (push) corrente de  $V_{cc}$ .

Os circuitos da família TTL são hoje produzidos em um vasto repertório de módulos funcionais, catalogados em famílias de prefixos:

74 – Versão comercial (a que está sendo usada nesse projeto).

54 – Versão militar.

Versão militar  $-55 < T < +125^{\circ} C$

$$V_{cc} = 5V + \text{ou} - 10\%$$

Versão comercial  $0 < T < +70^{\circ} C$

$$V_{cc} = 5V + \text{ou} - 5\%$$

### B.2 Família TTL LOW POWER SCHOTTKY (54LS/74LS)

Nesta família conseguiu-se atingir reduções em potências, em cerca de 5 vezes relativas aos circuitos da família TTL padrão. A utilização dos diodos

SCHOTTKY reduz a perda de tempo provocada pelos transistores saturados. O desempenho desta família excede o da família TTL padrão, consumindo muito menos potência. Uma porta típica consome 2mW de potência e apresenta um tempo de retardo de 10 ns.

### B.2.1 O decodificador 74LS42

Esse componente foi escolhido pela facilidade de encontrá-lo no mercado e por diminuir bastante o tamanho e tempo do projeto em relação a usar circuitos integrados compostos apenas por portas lógicas montando cada parte do demultiplexador (com circuitos integrados compostos apenas por portas lógicas, usei nos testes 5 peças para fazer a função somente de uma peça 74ls42).

O integrado 74LS42 é conhecido como decodificador decimal. Ele é um decodificador binário incompleto. Isto é, decodifica apenas as combinações decimais das entradas. Ele é utilizado neste projeto pela facilidade de encontrá-lo no mercado. Suas saídas acionarão os flip-flops dos circuitos 7475 permitindo que estes possam armazenar os bits relativos à intensidade do acionamento da lâmpada. Segue abaixo a tabela verdade (truth table) e o circuito interno de portas lógicas do 74LS42. Os inversores 74ls04.



Tabela B.1 - Tabela verdade do 74LS42

Entradas				Saídas									
E3	E2	E1	E0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0

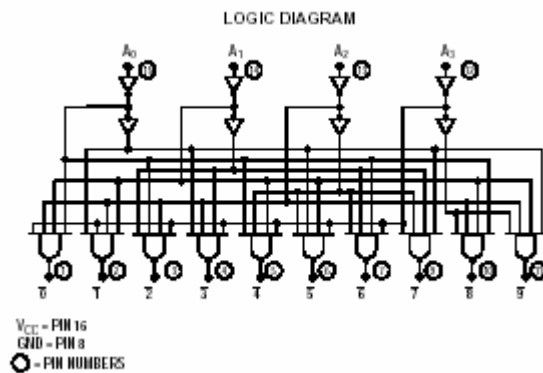


Figura B.1 - Portas lógicas do 74LS42

Fonte: [www.alldatasheets.com](http://www.alldatasheets.com)

## B.2.2 Os inversores 74LS04

O integrado 74LS04 possui seis inversores conforme mostrado abaixo. Esses inversores são usados no projeto para inverter a saída do decodificador

porque o 74LS42 coloca nível baixo na saída selecionada e o registrador 7475 (que será descrito mais a frente) precisa de nível alto para mudar o conteúdo dos seus flip-flops.

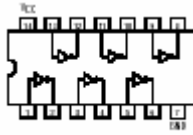


Figura B.2 - Portas inversoras no 74LS04

Fonte: [www.alldatasheets.com](http://www.alldatasheets.com)

### B.2.3 O registrador 7475

Optei por esse circuito integrado pela facilidade de consegui-lo no mercado. Foi o primeiro componente para essa função que foi testado e conseguido os resultados esperados. Uma vantagem desse componente é a facilidade de possuir 4 flip-flops podendo controlar (permitir gravar ou manter os bits) cada par separadamente.

O circuito integrado 7475 possui 4 flip-flops. Esse circuito irá armazenar os bits relativos à intensidade da lâmpada. Se o nível que vier do decodificador for 1 (alto), o registrador modificará os bits armazenados para os atuais que estão sendo enviados pelo sistema, caso o nível proveniente do decodificador for 0 (baixo), o registrador manterá os bits armazenados anteriormente sem mudar seu atual estado. Segue abaixo a tabela verdade do flip-flop tipo D (utilizado no 7475):

Tabela B.2 - Tabela verdade do registrador 7475

Entradas		Saídas	
D	C	Q	Q'
0	1	0	1
1	1	1	0
x	0		