WebHome – Automação residencial utilizando *Raspberry PI*

Emerson Fausto Lisboa[1] e Ariadne Arrais Cruz[2].

[1]Graduado em Engenharia de Automação e Controle, emerson.lisboa@foxconn.com,
[2]Mestre em Engenharia da Computação, UNISAL, ariadne.cruz@sj.unisal.br

Resumo *– Este artigo propõe o desenvolvimento de um sistema de controle automatizado para residências, também conhecido como domótica. Tais controles englobam o acionamento de luzes, persianas, monitoramento de estado portas e janelas, como também controle e monitoramento de temperatura dos ambientes. Todos estes recursos podem ser observados e controlados através de algum dispositivo conectado a Internet ou em um ambiente wi-fi. A princípio, foi desenvolvido um sistema capaz de ser executado por qualquer navegador, inclusive por dispositivos móveis como tablets e celulares que utilizam o sistema operacional Android. O principal ganho com a proposta deste projeto é a mobilidade, comodidade, segurança, bem estar e a utilização de um recurso extremamente eficaz com baixo consumo de energia para gerenciamento de toda a aplicação.*

Palavras-chave*: Automação residencial, Internet, mobilidade, Raspberry PI.*

Abstract *– This paper presents an automated control system for homes, also known as home automation. These controls include turning lights on and off, closing and opening blinds, monitoring doors and windows states, as well as temperature control of air-cooling devices. All these features can be handled through a device connected to the Internet over wi-fi. At first, we developed a system that can be executed by any browser, including mobile devices like tablets and smartphones based on Android operating system. The main advantage provided by this proposal is the mobility, convenience, safety and welfare of using a highly effective home automation system with very low power consumption.*

Keywords: *home automation, Internet, mobility, Raspberry PI.*

I. INTRODUÇÃO

A domótica é definida como implementação de tecnologias capazes de prover o gerenciamento dos diversos dispositivos presentes em um ambiente residencial [1].

O conceito de automação residencial vem há décadas deixando de ser apenas de um conceito e tornando-se realidade. O protocolo X10 foi um dos padrões com maior consistência e relevância concebido em 1975 [2]. Tal tecnologia foi projetada para permitir o controle de eletrodomésticos através de comandos transmitidos entre emissores e receptores, onde sinais eram trafegados pela rede elétrica. Estes sinais eram formados por mensagens como "ligar" e "desligar" via rajadas de frequência de rádio [2]. Ao longo dos anos, este sistema foi suportado pelos sistemas operacionais Windows e OS/2 tornando-se ainda mais interessante e atrativo para o desenvolvimento. Porém o revolucionário X10 possui grandes falhas como: interferências, perdas de comandos transmitidos, baixa velocidade de transmissão, poucos comandos entre outros que tornaram inviável a aplicação [2].

Com o avanço da tecnologia e o surgimento da Web nos anos 90 surgiram também tecnologias de computação doméstica a baixo custo que inspiraram os amantes da domótica a criarem ferramentas, sistemas, eletrodomésticos inteligentes e aparelhos eletrônicos capazes de interagirem uns com os outros através de computadores. Com normalização dos protocolos FTP e HTTP, os desenvolvedores de *hardware* viram a oportunidade de impulsionar ainda mais o avanço na comunicação entre dispositivos domésticos via *web*. Assim, nos dias atuais, nota-se uma grande gama de dispositivos e tecnologias que interagem de forma objetiva e eficiente [3], [4].

O projeto desenvolvido é uma implementação de um sistema, capaz de gerenciar este ambiente remotamente através da Internet utilizando recursos com baixo consumo de energia elétrica.

O controle destes dispositivos ocorre através de comandos vindos do usuário nos quais o sistema interpreta e executa uma ação. O maior benefício obtido é a mobilidade, bem estar e redução no consumo de energia, um assunto bem difundido nos dias atuais.

No âmbito deste contexto, é proposto um modelo de automação residencial que utiliza um dispositivo Android, placa de prototipagem Arduino e *Raspberry PI*.

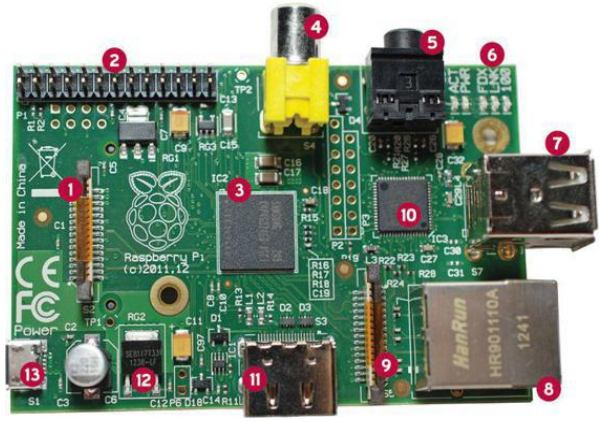
Nesta proposta, a interface é capaz de monitorar a temperatura, gerenciamento do estado de portas e acionamento de lâmpadas. A aplicação apresentada opera mediante comandos de entrada e saída, o que torna possível o controle ou monitoramento de qualquer dispositivo. Logo, a aplicação pode ser aperfeiçoada para incluir o monitoramento por imagens dos ambientes, medição do consumo de energia elétrica e envio de SMS com o *status* geral do sistema.

II. CONCEITOS

A. *Raspberry PI*

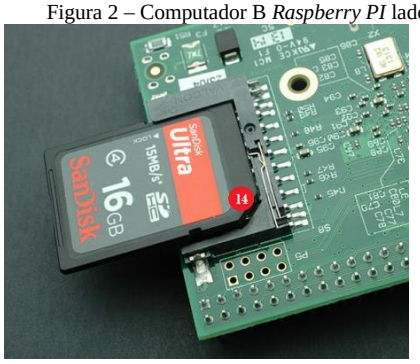
O *Raspberry PI* é um computador pessoal de baixo custo. Em 2006 no Reino Unido, pais de origem, está placa foi lançada a $35 dólares [5]. A principal ideia dos criadores foi desenvolver um produto com preço acessível, tamanho reduzido e com diversas funcionalidades capazes de integrar facilmente o desenvolvimento de projetos eletrônicos com *software*. H dois modelos desta placa, sendo o "A", o primeiro modelo lançado com apenas uma entrada USB, sem entrada RJ45 e 256 MB de memória RAM. Já o modelo "B", revisão dois, é o *update* do "A", pois conta com as mesmas funcionalidades mais o acréscimo de uma entrada USB, uma interface RJ45 e uma memória de 512 MB. O *Raspberry* pode ser utilizado com um computador pessoal e não apenas como uma plataforma de microcontrolador, pois ele trabalha com processamento sobre um sistema operacional que executa ações de entrada, saída e armazenamento. Diferentemente dos microcotroladores que trabalham com ciclos de *clock* sem o controle interno de um sistema operacional [6]. Nas Figuras 1 e 2, estão dispostas todas as interfaces e recursos disponíveis no modelo "B" e cujas especificações são apresentadas a seguir.

Figura 1 – Computador B *Raspberry PI* lado A.



Fonte: Retirado de [6].

Figura 2 – Computador B *Raspberry PI* lado B



Fonte: Retirado de [7].

1. *DSI* vídeo: Possui uma saída de vídeo com interface serial, neste conector é possível instalar um monitor.

2. *GPIO* - portas *input/output*: é um complemento muito interessante e fundamental para o *Raspberry*, elas proporcionam uma maneira fácil de conectar aos *hardwares* desenvolvidos. Nesta interface pode ser conectado qualquer dispositivo externo para expansão, sensores, atuadores, permite a criação de uma interface de comunicação serial. A GPIO é composta de 26 pinos distribuídos entre *Groud*, *in/out*, sendo todas digitais, tensão de 3.3 Volts, tensão de 5 Volts, PWM, RX, TX, MOSI, MISO, SCK, SDA e SCL. A distribuição destes pinos está descrita a seguir na Figura 3.

Figura 3 – Distribuição de pinos GPIO.

Fonte: Retirado de [10].

3. *CPU*, *GPU* e *RAM*: O *Raspberry* opera com processador ARM11 e velocidade base de 700 MHz. Também conta com um processador que decodifica e reproduz vídeo até 1080p em alta definição e uma memória RAM de 512 MB.

4. *RCA* vídeo: Saída de vídeo analógica.

5. Audio estéreo: Permite a montagem de uma central multimídia, neste conector pode efetuar a ligação de caixa de som.

6. *LED status*: Indica o status de energia.

7. Dois USBs: Possui duas entradas USB que pode ser utilizada para a integração com diversos dispositivos. Os mais usuais são o mouse, teclado ou um USB *Switch*.

8. RJ45- Ethernet: Dispõe de uma interface onboard para ligação da placa em rede. Basta ligar um cabo de rede convencional a um roteador, se o mesmo tiver sinal para acesso a Internet, o *Raspberry* poderá facilmente acessá-la.

9. Entrada para câmera: Possui uma interface serial de câmera, neste conector é possível instalar uma *webcam* portátil.

10. Controle USB e Internet: Microchip responsável em executar as funções de controle de acesso à Web e conexões USB.

11. HDMI: Ao utilizar a saída HDMI há a capacidade de enviar 1080p de alta definição para monitor ou televisão HD

12. Regulador de tensão: Devido às saídas de 3,3 Volts foi implementado o regulador de tensão, que reduz os 5 Volts de tensão de entrada.

13. Alimentação elétrica micro USB: A entrada deve ser de 5 Volts (VDC) ± 5% de tolerância. Para o modelo "B" espera-se no mínimo um corrente de 700 mA.

14. Leitor SD: É necessário um dispositivo de armazenamento removível, cartão SD, pois o *Raspberry* não possui em sua estrutura a capacidade de armazenamento permanente. É importante ressaltar que a escolha do cartão SD têm influências no desempenho geral do sistema. A velocidade de leitura e gravação será proporcionalmente a classe de classificação. Desta, forma o ideal é escolher os modelos de classes de especificações mais elevadas. A imagem do sistema operacional está armazenada no cartão SD.

B. Linux – Raspbian

A fundação *Raspberry* desenvolveu uma distribuição otimizada e *open-source* baseada em *Linux* embarcado chamada *Raspbian*. É um sistema operacional composto de centenas de pacotes de *software* com desenvolvimento ativo e constante de atualizações com ênfase na melhoria de estabilidade e desempenho [8].

Há outras distribuições alternativas lançadas por outras organizações que são otimizadas para serem utilizadas para outros fins.

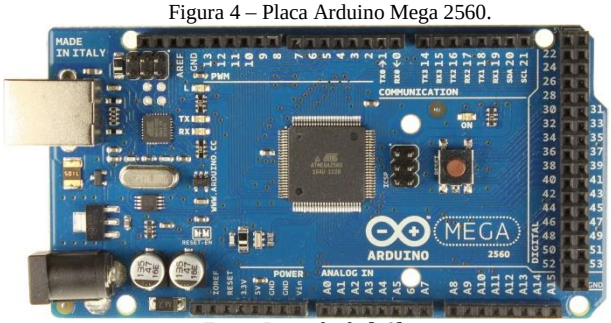
C. Arduino

O Arduino é uma plataforma de desenvolvimento de s*oftware* e *hardware open source*. O ambiente foi projetado para ser de fácil compreensão e utilização. Com o Arduino é possível construir projetos que podem responder e controlar dispositivos, circuitos eletrônicos, sensores e atuadores. Estão disponíveis diversos modelos de placa. A escolha fica condicionada à aplicação na qual será utilizada. O Arduino é utilizado em muitos casos na construção de protótipos de projetos e desenvolvimento educacional em diversas instituições [9].

O Arduino *software* é composto de uma IDE (ambiente de desenvolvimento integrado), um ambiente de desenvolvimento que permite a edição e criação de códigos que são convertidos em instruções de *hardware*. Estes códigos podem ser transferidos facilmente do computador para a placa, através de uma função *upload* presente na IDE. Caso não haja falha, o *hardware* passa a executar as funções criadas e carregadas. A linguagem de programação é proprietária, mas são baseadas em C e C++[9].

O Arduino *hardware* é composto de uma placa eletrônica com diversos pinos analógicos e digitais de entrada e saída, de acordo com a modelo, entrada USB e conector de alimentação elétrica. O Arduino Mega 2560 é apresentado na Figura 4. Esta plataforma conta com um microcontrolador da Atmel ATMEGA2560, que possui 54 pinos digitais, 16 pinos de entrada analógicas, três pinos *ground*, 8 pinos para comunicação serial e um cristal oscilador que opera num *clock* de 16 MHz. Também dispõe de uma memória *flash* de 256 KB para gravação do código, gravação do *bootloader* e memória EPROM. O grande benefício deste modelo é a quantidade de pinos que podem ser utilizados como entrada ou saída e, com isto, é possível controlar / monitorar uma grande quantidade de dispositivos [9].

Figura 4 – Placa Arduino Mega 2560.



Fonte: Retirado de [10].

D. Python

O Python é uma linguagem de programação *open source* que conta com uma máquina virtual, denominada "interpretador de código". Com esta tecnologia é possível escrever desde pequenos *scripts* aritméticos até sistemas *Web* completos. É uma linguagem de alto nível composta de uma vasta gama de bibliotecas capazes de executar diversas funções [11].

As distribuições mais comuns *Raspberry PI* já vêm uma IDE do Python como uma ferramenta de desenvolvimento. Esta linguagem tem uma forte ligação com esta plataforma *Raspberry*, tanto que já inclui "PI" na denominação, referindo-se à linguagem Python. Todavia, é válido salientar que o *Raspberry PI* aceita e compila outras linguagens.

Em Python, para criar e executar um programa basta criar um *script* em qualquer editor de texto ou mesmo no sistema operacional e executar o comando

```
$ sudo python meuPrograma.py
```

E. Servidor web

Toda aplicação *web* deve ser hospedada em um computador que pode ser acessado por todos os clientes. Para que todo este processo ocorra de forma estável e com segurança, faz-se necessária uma configuração do computador capaz de interpretar e responder todas as solicitações vindas dos clientes. Este computador é conhecido normalmente como *servidor web* e deve ser tal que a estrutura possua containers capazes de hospedar e executar diversos *scripts* e estruturas completas de uma aplicação que possa contar com acessos a um gerenciador de banco de dados[12].

Atualmente há diversas aplicações disponíveis que transformam simples computadores em servidores Web. A escolha de qual aplicação utilizar está ligada normalmente a linguagem padrão na qual o sistema foi desenvolvido e recursos de *hardware* disponíveis. Dentre as possibilidades, destaca-se o *Apache Tomcat* que é uma implementação de *software* de código aberto de tecnologia que executa códigos escritos na linguagem Java [13].

Quando há recursos limitados de *hardware*, faz-se necessária a utilização de implementações "leves", capazes de executar o sistema de maneira confiável.

Devido às limitações de memória, processamento e armazenamento, no caso do *Raspberry* foi utilizada uma distribuição compacta conhecida com *Flask*. Este *software* é capaz de executar códigos escritos em Python e tornar o *Raspberry* um servidor *web* dinâmico, apto a executar e implementar diversas aplicações.

F. Android

O Android é um sistema operacional desenvolvido empresa Google. Sua primeira versão comercial foi lançada no ano de 2008, e, ao longo dos anos, foi aprimorado e passou a ser empregado em diversas funcionalidades.

O Android ganhou muito espaço com o surgimento dos *smartphones* e *tablets*. Diversas das grandes montadoras de dispositivos móveis optaram em lançar seus aparelhos com o sistema operacional Android, tornando-o sistema operacional móvel mais difundido e utilizado no mundo.

Este sistema operacional trabalha com uma máquina virtual chamada *Dalvik Virtual Machine*, [14], que compila os programas desenvolvidos em Java para dispositivos móveis. Todas as aplicações podem ser executadas independentemente do modelo e fabricante do aparelho [15], exceto os casos onde a limitação é de *hardware.*

Para o desenvolvimento de aplicativos, faz-se necessária a instalação de uma IDE. Logo, é necessário instalar e configurar um *plugin* chamado de ADT (*Android Development Tools*), que foi desenvolvido para proporcionar um ambiente integrado para o desenvolvimento de aplicativos para Android. É uma ferramenta completa, que amplia os recursos da IDE Eclipse, na qual pode-se criar rapidamente as aplicações com recursos de interface, realizar a depuração em um simulador que pode ser selecionado de acordo com o ambiente do dispositivo móvel, e, por fim, possibilitar a criação do arquivo final para a distribuição [15].

A Figura 5 mostra a arquitetura de toda a plataforma Android

Figura 5 – Estrutura geral do sistema operacional Android.

Fonte: Retirado de [15].

G. Java

Java é uma linguagem de programação que destaca-se principalmente na portabilidade e no conceito de programação orientada a objetos, que possibilita o reaproveitamento de código. Devido a estas características, a linguagem possibilita desenvolver aplicações multiplataforma, ou seja, que podem ser executadas em qualquer dispositivo independente do sistema operacional [16].

Nesta linguagem, utiliza-se o mecanismo de máquina virtual, que trata-se de um tradutor que executa a conversão da informação tornando-a legível ao sistema operacional na qual aplicação está sendo executada.

Esta linguagem, criada na década dos anos 90, pela *Sun*, mais tarde a *Oracle* adquiriu seus diretos e deu continuidade no seu desenvolvimento [16]. Ela conta com uma ampla biblioteca que torna sua estrutura robusta possibilitando uma redução no tempo de desenvolvimento das aplicações.

III. DESENVOLVIMENTO

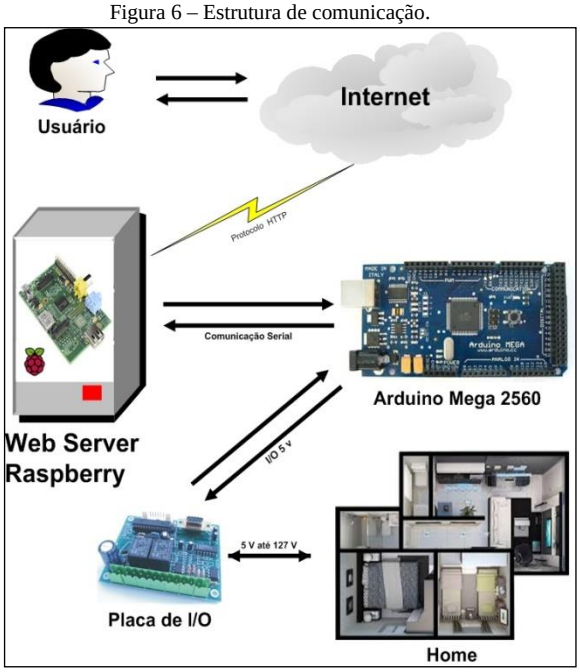
A proposta deste trabalho consiste da criação de um ambiente físico que simula a automação de uma residência. Tem-se como principal característica a criação de uma aplicação que pode ser gerenciada remotamente utilizando um servidor *web* com baixo consumo de energia elétrica.

O desenvolvimento inclui um projeto com base eletrônica e um conjunto de códigos computacionais que permite o monitoramento de temperatura, estado de portas e acionamento de dispositivos.

A Figura 6 mostra de maneira simplificada a estrutura de comunicação. O comando pode ser enviado pelo usuário e executado em um dispositivo de saída. É possível também monitorar o *status* de um sensor, tarefa que é inicia pela leitura na residência, seguida do envio da informação para o servidor *web* que então irá disponibilizar a informação através de um navegador ou aplicativo em Android do dispositivo móvel do usuário.

A integração da parte eletrônica com o *software* desenvolvido possibilita a execução do sistema no método de I/O (*input/output*) ou entrada e saída.

Figura 6 – Estrutura de comunicação.



Fonte: Elaborada pelos autores.

Conforme apresentado na Figura 6, o projeto está subdividido nas seguintes etapas: o usuário, que conta com uma aplicação Android ou acesso através de um *web navegador*; a *Internet* e o servidor *web*; a placa Arduino; uma placa de I/O composta por três circuitos : sensor de monitoramento de estado de portas ou janelas, acionamento de ar condicionado e acionamento de lâmpadas; e uma maquete com sensores e *led*s que simula um ambiente residencial.

Descreve-se a seguir detalhes da implementação de cada uma das etapas mencionadas.

1) Aplicação Android

O sistema desenvolvido pode ser gerenciado através de dispositivos móveis que utilizam o Android como sistema operacional. Para isso, foi desenvolvida uma aplicação que executa os acionamentos do tipo liga ou desliga a iluminação, verifica o estado de portas e apresenta o gráfico do histórico de temperatura em um determinado período. A aplicação pode ser executada diretamente no dispositivo ou através de um emulador. A linguagem de programação utilizada é Java. A Figura 7 mostra a aplicação em execução através do emulador.

2) Servidor web

O servidor *web* é formado apenas pela placa *Raspberry* que trabalha com sistema operacional *Linux* chamado *Raspbian*. Devido à estrutura enxuta desta plataforma, foi configurada uma micro *framework* de hospedagem de aplicações *web* chamada *Flask*, que é um *software* leve com diversos recursos que enquadraram-se muito bem à necessidade da aplicação desenvolvida.

Figura 8 – Tela de simulação de um dispositivo móvel Android.

Fonte: Acervo dos autores.

A linguagem de programação utilizada no servidor *web* é o Python 2. A estrutura do Python suporta diversas bibliotecas e protocolos. O principal protocolo empregado neste projeto foi a de comunicação com o Arduino, chamada de *Firmata*, que é considerada um protocolo genérico para comunicação do microcontrolador do Arduino com o *host Raspberry*. O principal objetivo é controlar e monitorar as portas do Arduino através dos comandos vindos da aplicação, processados no *Raspbian*.

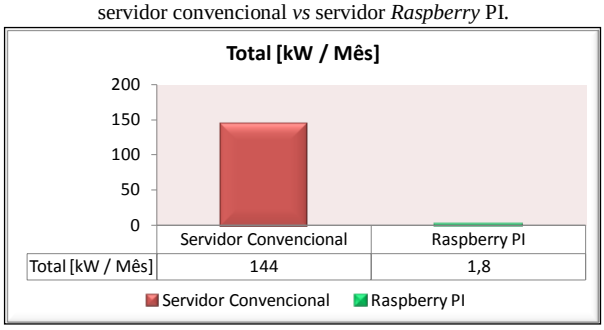
Os comandos estão definidos por uma lógica simples. Foi desenvolvido um *script* no servidor que executa e aguarda a chegada de um vetor com duas posições. Neste vetor tem-se o número de identificação do dispositivo e um indicador de *status* que pode ser *on* ou *off*, utilizados para ligar ou desligar, respectivamente. O *script* também faz uma varredura do estado dos outros dispositivos e retorna um vetor com a informação de todos os dispositivos e estados atuais. Tais informações são fornecidas como resposta para os clientes que podem fazer a leitura e a apresentação através de um navegador *web* ou da aplicação Android.

Um diferencial da utilização da *Raspberry* como servidor é o baixo consumo de energia elétrica. A Tabela 1 e a Figura 8 mostram um comparativo entre um servidor *web* convencional e a utilização da *Raspberry*. Esta característica torna a utilização desta tecnologia atrativa.

Tabela 1 – Comparativo de consumo de energia elétrica, [7].

Consumo				
Equipamento	Potência[W]	Dias Uso	Tempo Uso [h]	Total [kW / Mês]
Servidor Convencional	200	30	24	144
Raspberry PI	2,5	30	24	1,8

Figura 8 – Comparação de consumo mensal de energia elétrica de um servidor convencional *vs* servidor *Raspberry* PI.



Fonte: Retirado de [7].

3) Placa Arduino

Na placa Arduino foi instalado o *Firmware Stardand* do protocolo *Firmata* e configurado todas as saídas para a placa de I/O.

Com este *firmware* a comunicação com o *hardware* final torna-se direta, possibilitando o total controle das saídas e entradas do Arduino. Assim, todo o controle das portas fica sob o controle dos comandos recebidos pela comunicação serial vindos do servidor *Raspberry*.

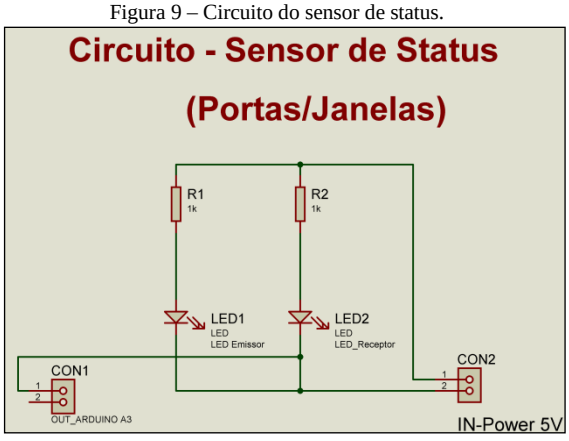
A principal atribuição da placa Arduino é aumentar as possibilidades de inserção de dispositivos para serem controlados.

4) Sensor de status para portas e janelas

A função do circuito de sensor de *status* é monitorar as condições atuais de portas e janelas. Há basicamente duas condições: aberto ou fechado. O circuito é composto por um conector de interface com a fonte de alimentação de 5V, um conector de interface que disponibiliza o sinal analógico que varia de 0 até 5 Volts, dois resistores 1KΩ para proteção dos *leds* e dois *leds* infravermelhos, sendo um emissor e outro receptor.

A Figura 9 mostra o esquema elétrico da placa de monitoramento de status de portas e janelas.

Figura 9 – Circuito do sensor de status.



Fonte: Elaborada pelos autores.

O funcionamento do circuito ocorre da seguinte forma: os *leds* devem estar alinhados um de frente para o outro. Enquanto não há interrupção do feixe, o circuito envia uma tensão de, no máximo, 5V. Caso o feixe seja obstruído, a tensão cai para 0V. Este sinal é enviado para uma porta de entrada analógica do Arduino que, por sua vez, interpreta como nível lógico alto ou baixo e, por fim, disponibiliza a informação ao servidor *web*.

Este circuito conta também com um dispositivo mecânico que ao simular abertura ou fechamento da porta, o sinal alterna o valor entre nível alto ou baixo. A Figura 10 mostra este circuito nos dois estados possíveis (fechado/aberto).

Figura 11 – Foto do circuito de estado de portas e janelas.

Fonte: Acervo dos autores.

5) Controle de temperatura e ar condicionado

A função do circuito de controle de temperatura é monitorar a temperatura e realizar o acionamento do ar condicionado.

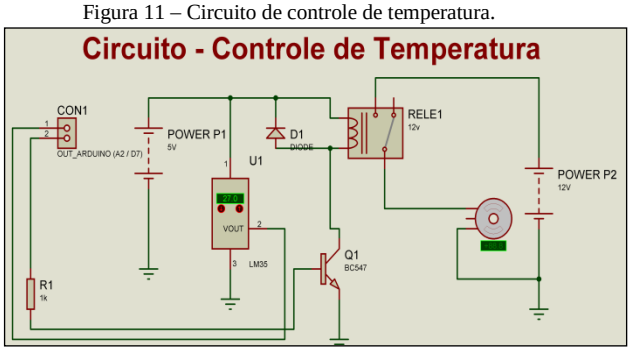
O circuito é composto por um LM35 (sensor de temperatura com tensão de saída linear), um conector com duas saídas, ambas ligadas ao Arduino, um diodo IN4004, um relé que opera entre 5V e 127V 10A, um resistor de 1KΩ, uma ventoinha que simula o ar condicionado e um transistor NPN BC547.

O princípio de funcionamento é formado pela leitura do LM35 que envia níveis de tensão à porta analógica A2 do Arduino. Esta, por sua vez, executa a conversão para uma unidade de medida de temperatura. A conversão para graus Celsius é dados pela fórmula:

$$tempCelcius = (5 * tensaoOutLM *100) /1024$$

Após a leitura, compara-se o valor da temperatura atual do ambiente com a definida previamente pelo usuário. Se a temperatura estiver acima, o Arduino envia um sinal para a porta D7 que alimenta o coletor do transistor BC574 e emite um sinal ao relé de acionamento do ar condicionado. Caso a temperatura estiver abaixo, nenhum sinal é emitido e o relé retorna ao estado de interrupção da tensão da fonte de 12V. Em uma implementação futura, pode-se acrescentar a função que, a cada intervalo de tempo, um valor de temperatura é armazenado em um banco de dados no servidor, que pode então disponibilizar o histórico de temperatura ao usuário final. A Figura 11 descreve o circuito que executa todo o controle de temperatura do ambiente.

Figura 11 – Circuito de controle de temperatura.



Fonte: Elaborado pelos autores.

6) Placa de I/O

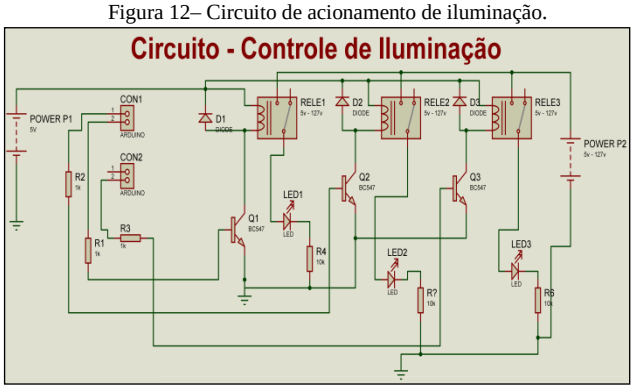
A placa de I/O ou controle de entradas e saídas é responsável em executar todo o acionamento das lâmpadas, seja este acionamento realizado através do navegador, aplicativo do dispositivo móvel ou interruptores físicos.

O circuito foi desenvolvido para trabalhar com chaveamento de tensão de até 127 Volts, sendo utilizados também dois *leds* de 5V para executar a simulação de acionamento.

A placa é formada por relés, resistores de 1KΩ e 10KΩ, transistores BC547 e diodo IN4007. A placa tem interação constante com a placa Arduino, recebe realimentação dos acionamentos físicos e também dos comandos dos aplicativos para ligar ou desligar um determinado *led* que simula uma lâmpada.

A Figura 12 descreve o circuito de acionamento em questão.

Figura 12– Circuito de acionamento de iluminação.

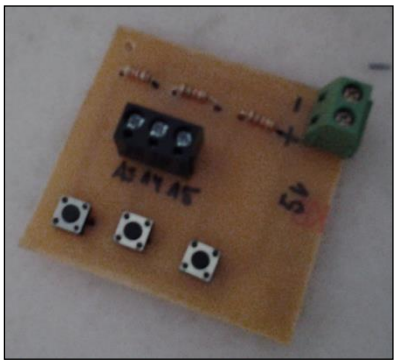


Fonte: Elaborada pelos autores.

7) Placa de Interface – Acionamento convencional das lâmpadas

A função desta placa é a disposição de uma interface física de acionamento das lâmpadas ao usuário. Ela é formada por botões tipo *push-bottom.* Quando acionados, o Arduino reconhece como um comando e inverte o nível lógico e estado atual da respectiva lâmpada (Figura 13).

Figura 13 – Interface de acionamento convencional das lâmpadas.



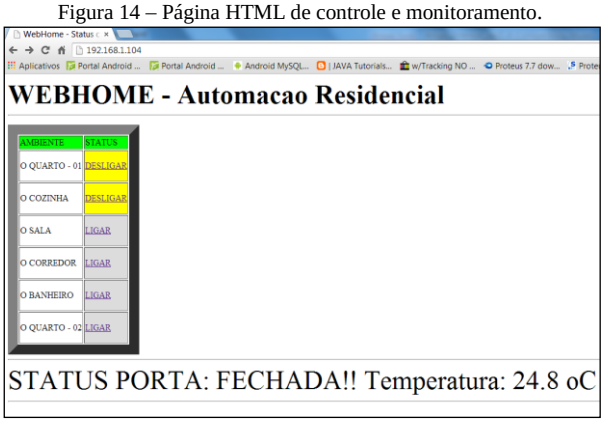
Fonte: Acervo dos autores.

8) Página HTML

A página HTML desenvolvida traz os estados de cada dispositivo que está sendo monitorado ou controlado.

Esta página está hospedada no servidor *web Raspberry* PI, e através dela o cliente executa as requisições e o servidor responde com o estado atual do sensor da porta, de temperatura e lâmpadas. Para visualizar as páginas *web* é necessário um navegador web instalado no computador. Navegadores bastante convencionais e bastante utilizados são o *Chrome* da Google e o *Internet Explorer* da Microsoft. A Figura 14 mostra a página HTML desenvolvida para o projeto.

Figura 14 – Página HTML de controle e monitoramento.



Fonte: Elaborada pelos autores.

9) Maquete

Desenvolveu-se uma maquete para simular um ambiente doméstico composto por: uma porta com sensor que identifica o estado atual, um sensor de temperatura LM35, uma ventoinha de computador, que simula um ar-condicionado, cinco *leds* que representam as lâmpadas e um painel de interface para acionamento de três *LEDs* (Figura 15).

Figura 15 – Maquete para simulação do ambiente

Fonte: Acervo dos autores.

IV. RESULTADOS

Após a implementação das diversas tecnologias apresentadas na seção II, o funcionamento do projeto proposto ocorreu de maneira satisfatória. A Tabela 2 resume os testes realizados e os resultados obtidos.

Tabela 2 – Demonstração de resultados.

Item	*Descrição*	*Circuito executou o processo?*	*Atualizado status da página web?*
1	Acionamento de todas as lâmpadas através de um navegador *web* ou aplicação Android	Sim	Sim
2	Acionamento de uma lâmpada aleatória através de um navegador *web*	Sim	Sim
3	Acionamento de uma lâmpada aleatória através do interruptor físico tipo *push-bottom*	Sim	Sim
4	Acionamento de uma lâmpada aleatória através da aplicação Android	Sim	Sim
5	Mudança de estado de porta. Mantido em estado aberto	Sim	Sim
6	Mudança de estado de porta. Mantido em estado fechado	Sim	Sim
7	Acionamento (liga) da ventoinha para temperatura acima do valor estipulado	Sim	Atualizado o valor da temperatura no navegador
8	Acionamento (desliga) da ventoinha para temperatura abaixo do valor estipulado	Sim	Atualizado o valor da temperatura no navegador

V. CONCLUSÕES

A automação residencial é uma área com grandes possibilidades a serem exploradas. Podem-se integrar facilmente os avanços tecnológicos da eletrônica e computação. Conforme descrito neste artigo, há diversas tecnologias e recursos de custo reduzido que possibilitam a execução de projetos que trazem consigo maior comodidade e segurança aos usuários.

Neste projeto, a integração dos recursos tecnológicos, principalmente eletrônica, *Linux* embarcado, comunicação e desenvolvimento de *software* possibilita que um usuário em um lugar remoto com acesso a Internet ou em um ambiente *wi-fi* execute o monitoramento e controle de todos os dispositivos de uma residência, desde que estejam integrados ao sistema.

Uma das principais características observada foi o funcionamento da placa *Raspberry* PI como servidor *web* e o baixo consumo de energia elétrica, o que faz a implementação ser relevante e economicamente viável.

Também é importante salientar que o avanço e o crescimento significativo da demanda em utilização dos dispositivos móveis como *tablets* e *smartphones* é um ponto positivo que fornece impulsos ao desenvolvimento de aplicações capazes de impactarem situações do cotidiano dos usuários.

Conclui-se, portanto, que a domótica é uma realidade presente e que pode ser vastamente explorada utilizando recursos acessíveis e com resultados satisfatórios.

REFERÊNCIAS

[1] SGARBI, Julio A., TONIDANDEL, Flavio. Domótica Inteligente: Automação Residencial baseada em Comportamento. In: *Workshop de Teses e Dissertações em Inteligência Artificial, Ribeirão Preto, São Paulo,* 2006.

[2] **Protocolo X10.** Disponível em: http://www.casadigital.telecom.pt/Tecnologia/domotica/Pages/X10.aspx. Acesso em: 20/05/2014.

[3] KUROSE, James F.; ROSS, Keith, W. **Redes de Computadores e a Internet: uma nova abordagem**. Addison Wesley, 2003.

[4] GRAHAM, Ian S. **The HTML sourcebook**. John Wiley & Sons, Inc., 1995

[5] Raspberry PI: The Credit Card Sized 35 Dollar Computer with Amazing Possibilities. Disponível em <http://davidpapp.com/2013/04/17/Raspberry-pi-the-credit-card-sized-35-dollar-computer-with-amazing-possibilities> Acesso em: 20/05/2014.

[6] WARNER, *T.* **Hacking Raspberry PI** . Indianapolis: Que Publishing, 2014.

[7] Raspberry PI SD Card Slot. Disponível em: <http://www.adafruit.com/blog/2013/07/19/how-to-repair-a-broken-Raspberry-pi-sd-card-slot-Raspberry_pi-piday-Raspberrypi/>. Acesso em 16/06/2014.

[8] Raspberry PI. Disponível em: <http://www.advamation.com/knowhow/Raspbrrypi>. Acesso em: 20/05/2014.

[9] Welcome to Raspbian. Disponível em: <http://www.raspbian.org/>. Acesso em 20/05/2014.

[10] SCHMIDT, Mark. **Arduino – A Quick Start Guide.** Dallas: The Pragmatic Programmers, LLC, 2014.

[11] Arduino Mega 2560**.** Disponível em: <http://www.projetoarduino.com.br/arduino-mega-2560-p19>. Acesso em: 20/05/14.

[12] PILGRIM, Mark. **Mergulhando no Python**. Rio de Janeiro: Alta Books, 2004.

[13] MEMBREY, Peter and HOWS, David. **Learn Raspberry PI with Linux**. Apress, 2012.

[14] Apache Tomcat. Disponível em: <http://tomcat.apache.org/>. Acesso em 15/04/2014.

[15] Android Developer Tools**.** Disponível em: <http://developer.Android.com/tools/index.html> Acesso em: 25/04/2014.

[16] DEITEL, Harvey M. e DEITEL, Paul J. **Java-como programar**. 6ª. ed. São Paulo: Prentice Hall, 2005.