UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE ELETRÔNICA

CURSO DE ENGENHARIA ELETRÔNICA

GUILHERME BORGES DOS SANTOS

RAPHAEL LUCAS ELBL SILVA

AUTOMATIZAÇÃO DE MICROAMBIENTE PARA DESENVOLVIMENTO DE CULTURAS BIOLÓGICAS

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA

2015

GUILHERME BORGES DOS SANTOS

RAPHAEL LUCAS ELBL SILVA

AUTOMATIZAÇÃO DE MICROAMBIENTE PARA DESENVOLVIMENTO DE CULTURAS BIOLÓGICAS

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia Eletrônica do Departamento de Eletrônica da Universidade Tecnológica Federal do Paraná, Campus Ponta Grossa.

Orientador: Prof. Dr. Sergio Luiz Stevan Jr.

PONTA GROSSA

2015

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Ponta Grossa

Diretoria de Graduação e Educação Profissional

TERMO DE APROVAÇÃO

AUTOMATIZAÇÃO DE MICROAMBIENTE PARA DESENVOLVIMENTO DE CULTURAS BIOLÓGICAS

por

GUILHERME BORGES DOS SANTOS
RAPHAEL LUCAS ELBL SILVA

Este Trabalho de Conclusão de Curso (TCC) foi apresentado em 25 de junho de 2015 como requisito parcial para a obtenção do título de Bacharel em Engenharia Eletrônica. Os candidatos foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr. Sergio Luiz Stevan Júnior
Prof. Orientador

Prof. Dr. Elói Agostini Júnior
Membro titular

Prof. Ms. Murilo de Oliveira Leme
Membro titular

Prof. Ms. Jéferson José Gomes
Coordenador do Curso
UTFPR - Campus Ponta Grossa

- O TERMO DE APROVAÇÃO ASSINADO ENCONTRA-SE ARQUIVADO NA SECRETARIA ACADÊMICA -

AGRADECIMENTOS

Nesse grande passo em nossas vidas, aqui apenas, não poderemos agradecer a todos que de alguma forma colaboraram para o desenvolvimento desse projeto. Se não lembrados em palavras, certamente em nossos pensamentos, com imensa gratidão a cada um.

Primeiramente a Deus, nos dando sentido a vida.

Agradecemos as nossas famílias, sem as quais não seria possível vencer essa etapa.

Agradecemos a todos os amigos que de alguma forma fazem parte dessa história.

Agradecemos em especial ao nosso professor orientador Prof. Dr. Sergio Luiz Stevan Jr, pela paciência e sabedoria para nos guiar.

RESUMO

DOS SANTOS, Guilherme Borges; SILVA, Raphael Lucas Elbl. **Automatização de microambiente para desenvolvimento de culturas biológicas.** 2015. 107. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Eletrônica) – Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2015.

O presente trabalho apresenta o desenvolvimento do projeto de automação de um microambiente utilizado para o desenvolvimento de culturas biológicas, para o qual o controle de variáveis como temperatura, umidade e luminosidade são importantes para o propósito geral do mesmo. Por meio de sensores fez-se a aquisição dos valores correntes das variáveis, onde posteriormente tratados esses valores por uma placa microcontrolada, o controle foi realizado mediante o programa desenvolvido em linguagem C. O controle mostrou-se efetivo através de atuadores alocados no microambiente, tais como resistência de aquecimento, umidificador, LEDs e um ventilador de convecção. O sistema central baseia-se na plataforma Arduino Mega 2560, a qual possui como microcontrolador o ATmega2560, responsável por interpretar os dados e agir mediante a *firmware* desenvolvida para o propósito. Para validar o controle do projeto de automação proposto são apresentados resultados, simulações e testes físicos os quais são condizentes ao controle concebido.

Palavras-chave: Controle de variáveis. Microambiente automatizado. Microcontroladores. Arduino Mega 2560.

ABSTRACT

DOS SANTOS, Guilherme Borges; SILVA, Raphael Lucas Elbl. **Automation of a microenvironment for the development of bio cultures**. 2015. 107. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Eletrônica) – Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2015.

The automation of a microenvironment used for the development of bio cultures by controlling variables such as temperature, humidity and light in view of the general purpose of the project. Through sensors were done acquisitions of the current values of variables, where these values subsequently processed by a microcontrolled board, the control was carried out through the program developed in C language. The control proved to be effective through actuators placed in the microenvironment, such as heating resistor, humidifier, LEDs, and a convection fan. The central system is based on Arduino Mega platform 2560, which has as the microcontroller ATmega2560, responsible for interpreting the data and acting upon the firmware developed for the purpose. In order to validate the proposed control automation project, results are presented, simulations and physical tests which are consistent with designed control.

Keywords: Variable control. Automated microenvironment. Microcontrollers. Arduino Mega 2560.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS

IHM	Interface Homem Máquina
LDR	Sigla em inglês para *Light Dependent Resistor* (Resistor Dependente de Luz).
Vdc	Sigla em inglês para *Direct Current* (Corrente Direta).
Vcc	Sigla em inglês para *Continuous Current* (Corrente Contínua).
Vca	Sigla em inglês para *Alternating Current* (Corrente Alternada).
UR	Umidade Relativa
OTP	Sigla em inglês para *One Time Programmable* (Programável uma vez).
RAM	Sigla em inglês para *Randon Access Memory* (Memória de Acesso Aleatório).
EEPROM	Sigla em inglês para *Electrically Erasable Programmable Read Only Memory* (Memória Somente de Leitura Eletronicamente Apagável e Programável).
SDA	Sigla em inglês para *Serial Data* (Dados Seriais).
GND	Sigla em inglês para *Ground* (Terra).
LED	Sigla em inglês para *Light Emitting Diode* (Diodo Emissor de Luz).
CPU	Sigla em inglês para *Central Processing Unit* (Unidade Central de Processamento).
ULA	Unidade Lógica Aritmética
USB	Sigla em inglês para *Universal Serial Bus* (Barramento Serial Universal).
PWM	Sigla em inglês para *Pulse Width Modulation* (Modulação por Largura de Pulso).
ADC	Sigla em inglês para *Analog Digital Converter* (Conversor Analógico Digital).
IDE	Sigla em inglês para *Integrated Development Environment* (Ambiente de Desenvolvimento Integrado).
PCI	Placa de circuito impresso
RPM	Rotações por minuto

LISTA DE SÍMBOLOS

A	*Ampere*; unidade de medida de corrente elétrica.
V	*Volts*; unidade de medida de tensão elétrica.
W	*Watt*; unidade de medida de potência elétrica.
F	*Farad*; unidade de medida de capacitância.
Ω	*Ohm*; unidade de medida de resistência elétrica.
°C	*Celsius*; unidade de medida de temperatura.
Hz	*Hertz*; unidade de medida de frequência.

SUMÁRIO

1 INTRODUÇÃO

A ideia de utilizar ambientes fechados para o controle de fatores climáticos no cultivo de plantas data da antiguidade. No século XVI na Itália, a construção de estufas e estruturas mais aprimoradas começava a ser disseminada, e mais tarde, países como Inglaterra e Holanda também conheceram essa engenhosidade. No Brasil, no ano de 1970 é que a ideia de utilizar estufas veio à tona, graças a iniciativas de empresas privadas e de órgãos ligados à pesquisa do ramo agrícola (PINHEIRO, 2011, p.40).

O Brasil, em todo seu território, apresenta um clima favorável para o cultivo e desenvolvimento de varias espécies, tanto na fauna como na flora, no entanto com os decorrentes problemas quanto às variações climáticas e o fato de algumas espécies não se adaptarem bem ao nosso clima tropical, é exigido cada vez mais do homem providências e soluções para monitorar e controlar essas variáveis climáticas, no intuito de encontrar uma aproximação fiel e favorável das condições para o desenvolvimento destas espécies. Citam-se como variáveis do processo: temperatura, umidade e luminosidade.

Particularizando o termo “controle de variáveis” para um microambiente, pode-se admitir uma grande quantidade de aplicações que vão desde cultivo de orquídeas, até o desenvolvimento e criação de outras culturas biológicas, passando por estufas dedicadas à secagem de madeira, onde o controle simultâneo ou isolado de algumas variáveis é realizado pelas mesmas.

Proporcionando um ambiente controlado em relação a fatores climáticos, tais como temperatura, umidade, luminosidade, visando o crescimento em um ambiente ideal as plantas, o cultivo das mesmas tem a possibilidade de ser realizado em estufas de diferentes tipos, tamanhos e materiais. Muitas dessas estufas contam com a automatização de alguns processos, assegurando uma climatização otimizada ao desenvolvimento dos vegetais produzidos (SILVA, 1986).

A utilização de microambientes automatizados possibilita o estudo do comportamento metabólico de certas culturas biológicas, podendo, por exemplo, aperfeiçoar a climatização de um ambiente de modo a acelerar ou minimizar o seu crescimento.

Com este propósito foi criado e desenvolvido um módulo que permita a automatização de um microambiente, proporcionando condições as quais satisfaçam as necessidades de desenvolvimento de várias espécies e culturas, com as variáveis alteradas e manipuladas eletronicamente.

1.1 DELIMITAÇÃO DO TEMA

Desenvolvimento de dispositivo eletrônico microcontrolador para controle de três variáveis de ambiente, sendo temperatura, umidade e luminosidade (fotociclos) em um microambiente.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Construir um módulo que permita realizar o controle de algumas variáveis tais como: temperatura, luminosidade, umidade e ventilação, para ser utilizado em um microambiente.

1.2.2 Objetivos Específicos

- Realizar o estudo teórico referente aos dispositivos eletrônicos envolvidos;
- Realizar ensaios experimentais dos sensores e atuadores para compreensão do seu funcionamento;
- Desenvolvimento de um *firmware* de controle para o projeto;
- Realizar simulações computacionais dos circuitos eletrônicos;
- Desenvolver uma interface de utilização para controle do processo;
- Realizar a análise dos resultados obtidos.

1.3 JUSTIFICATIVA

Com o desenvolvimento deste controlador, podem-se prover meios de monitoramento e ajuste das principais grandezas ambientais e assim prover o estudo metabólico de diferentes culturas confinadas neste ambiente. O microambiente automatizado proposto permite o controle no processo metabólico de culturas biológicas, baseado na sua interação com as grandezas ambientais.

2 REFERENCIAL TEÓRICO

2.1 MICROAMBIENTES

O uso de ambientes protegidos e controlados é uma das soluções adotada para o sucesso no cultivo de plantas que não pertencem ao seu *habitat* natural. Destinadas ao controle de condições edafoclimáticas, essas estufas são construídas para um sistema de produção agrícola específico (FIGUEIREDO, 2011, p.10).

Referindo-se as variáveis envolvidas em microambiente temos como as principais: temperatura, umidade, luminosidade, pressão, entre outras.

Aplicações de microambientes controlados surgem, por exemplo, em incubadoras de ovos, onde o controle da temperatura, umidade e a equalização de temperatura através de ventiladores são efetuados através do sensoriamento e atuação do sistema. Através de sensores é feita a aquisição de dados do microambiente, como por exemplo, valores temporais de temperatura, umidade relativa do ar e luminosidade, e por meio de atuadores é possível modificá-las conforme aos parâmetros desejados.

O metabolismo dos seres vivos, mesmo suportando mudanças bruscas, é influenciado diretamente por condições climáticas e do ambiente, onde o mesmo pode ser otimizado caso haja o controle dessas condições. As orquídeas, como exemplo, dependem de condições apropriadas para seu desenvolvimento, e fatores como luminosidade, água, temperatura, umidade e ventilação incidem diretamente no sucesso de seu crescimento (PLOUGHMAN, 2007, p. 11).

Na área medicinal, imunobiológicos e vacinas que se enquadram na PORTARIA Nº 2661 da Agência Nacional de Vigilância Sanitária (ANVISA), necessitam da preservação de suas propriedades, exigindo câmaras de conservação específicas para o controle de temperatura interna.

2.2 SENSORES

O elemento sensor não é propriamente um instrumento, porém é parte integrante da maioria dos instrumentos. Sua função é converter a variável física de entrada para outra forma mais usável. Em relação à grandeza física de entrada geralmente é diferente da grandeza de saída (RIBEIRO, 2002, p.67).

Quando o elemento sensor converte a grandeza de entrada para uma grandeza de saída elétrica, como tensão, resistência ou corrente, este recebe a denominação de transdutor (FUENTES, 2005, p. 03).

2.2.1 Sensor de Luminosidade

Os sensores de luminosidade convertem a variável de processo medida em uma variação de tensão elétrica devido à variação da quantidade de luz incidente em sua junção formada por algum material semicondutor.

Os LDRs (*Light Dependent Resistors*), mostrado na figura 1, são compostos basicamente por uma junção de sulfeto de cádmio (CdS), são sensores que quando estimulados por uma energia luminosa, apresentam em seus terminais uma variação de resistência.

Figura 1 - Sensor LDR modelo NSL19–M51
Fonte: RS Components (2013).

Quando expostos à ambiente com a ausência de luz, apresentam valores de resistência máxima entre 1MΩ e 10MΩ, e quando expostos à iluminação ambiente podem apresentar valores de resistência tipicamente entre 75Ω e 500Ω (THOMAZINI, 2005, p. 64), conforme a figura 2 onde r1 e r0 representa os valores máximos e mínimos de resistência respectivamente e s0 e s1 os valores referentes quando se tem menor e maior incidência de luminosidade sobre o sensor.

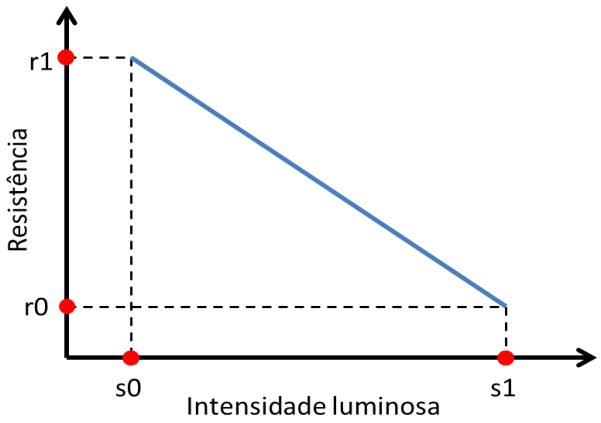


Figura 2 – Curva característica de um LDR comum
Fonte: Adaptado de Thomazini (2005, p. 63).

O LDR tem um “tempo de recuperação” muito longo, o tempo de recuperação de um LDR refere-se à resposta da variação de sua resistência no tempo, no instante em que é exposto a um estímulo luminoso.

O tempo de recuperação apresenta um tempo menor quando se passa para condição de pouca para uma de maior luminosidade, por exemplo, um LDR leva menos tempo para ir do valor máximo de resistência (r1) ao valor correspondente a resistência mínima (r0) quando submetido à passagem de uma condição de escuro para uma de iluminado, conforme a curva em azul na figura 3. Quando iluminado e cortando-se a luz incidente sobre o LDR, ele leva um intervalo de tempo maior para que sua resistência, em valor mínimo (r0), volte para o valor máximo (r1), conforme ilustrado na curva em vermelho na figura 3, onde (t0) e (t1) correspondem ao intervalo de observação. (THOMAZINI, 2007, p. 64).

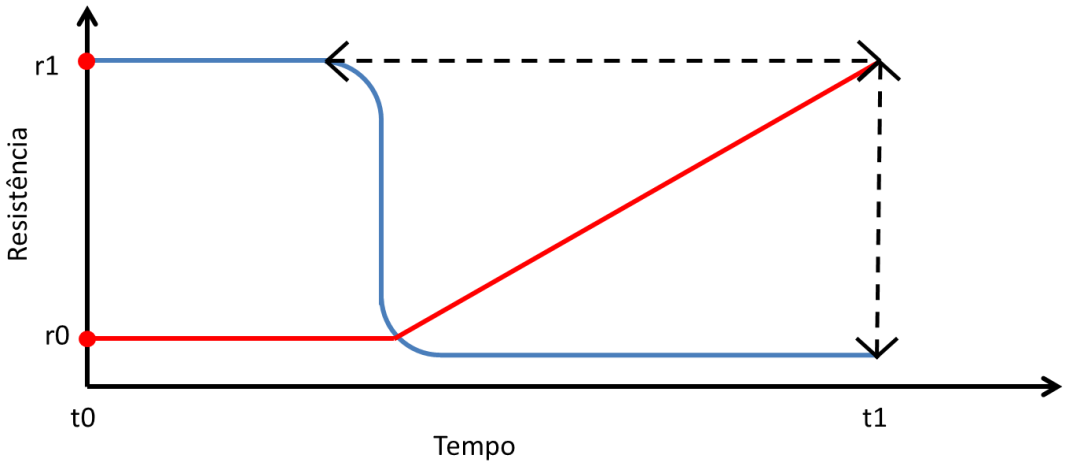


Figura 3 – Tempo de recuperação de um LDR
Fonte: Thomazini (2005, p. 65).

O LDR apresenta a desvantagem em sua velocidade de resposta, é um dispositivo lento quando comparado a outros tipos de sensores sensíveis à luz, como os fotodiodos e os fototransistores. Estes são extremamente rápidos podendo detectar pulsos de luz em taxas que chegam a dezenas ou mesmo centenas de megahertz, enquanto o LDR acaba não operando em velocidades maiores do que algumas dezenas de quilohertz.

2.2.2 Sensores de Temperatura

A energia cinética molecular média de um corpo é interpretada como temperatura. No entanto, só a energia cinética de translação influencia na grandeza temperatura, aspecto por vezes não citado (TIPLER; MOSCA, 2010, p. 571).

De um modo geral, os sensores de temperatura são classificados em mecânicos e elétricos. Os sensores mecânicos mais utilizados são o bimetal, enchimento termal, haste de vidro. Já os elétricos mais usados são os termopares, os de resistência metálica, os termistores e os de resistência a semicondutor (RIBEIRO, 2002, p. 258). O quadro 1, apresenta as características de alguns sensores de temperatura existentes no mercado, com a tecnologia empregada a cada um deles.

Parâmetro	**Modelo**			
	LM35	**Termopar J**	**PT100**	**AM2302**
Classificação	Elétrico	Elétrico	Elétrico	Elétrico
Tecnologia	Semicondutor	Resistivo	Resistivo	Resistivo
Faixa de medição	-55°C a 150°C	0°C a 800°C	-259°C a 960°C	-40°C a 80°C
Precisão	± 0,5°C	±2,2°C	±0,5°C	±0,5°C
Vantagem	Baixo custo, dimensão reduzida, fácil acessibilidade.	Tempo de resposta, baixo custo.	Grande precisão, estabilidade.	Grande precisão, custo médio.
Desvantagem	Apresenta falhas após ser usado por muito tempo em ambientes úmidos.	A característica temperatura X militensão não é totalmente linear.	Custo elevado, difícil acesso, tempo de resposta.	Intervalo mínimo de resposta entre medições de 2s

Quadro 1 - Comparativo entre sensores de temperatura
Fonte: Adaptado de Fialho (2002).

2.2.3 Sensores de Umidade

A umidade relativa do ar é definida como a relação entre a quantidade de água existente no ar (umidade absoluta) e a quantidade máxima que poderia haver na mesma temperatura (ponto de saturação). Ela representa o quão próximo o ar está da saturação, logo não indicando a real quantidade de vapor d'água no ar. Indicações de níveis 100% de umidade relativa nos instrumentos, apontam para a situação em que o ar está totalmente saturado com vapor d'água (SILVA, 2003).

Santos e Kawakita (1992) destacam alguns métodos nos quais se baseiam o funcionamento de alguns sensores de umidade do ar conhecidos: Medidores por impedância elétrica, por sensores capacitivos, por sensores resistivos, por higrômetros de cloreto de lítio saturado, por filme de pentóxido fosforoso, e por cristal de quartzo (piezoelétrico).

Segundo Santos e Kawakita (1992), de todos esses citados o mais comum é o do tipo capacitivo, onde a mudança de umidade é medida em função das alterações na capacitância do elemento sensor, sendo composto comumente de duas placas de metal com uma película de polímero não condutor entre elas.

O quadro 2 apresenta as características de alguns sensores de umidade presentes no mercado.

Parâmetro	**Modelo**			
	HS1101	**HIH-3610**	**HR202**	**AM2302**
Classificação	Elétrico	Elétrico	Elétrico	Elétrico
Tecnologia	Capacitivo	Semicondutor	Capacitivo	Capacitivo
Faixa de medição	0% a 100% UR	0% a 100% UR	20% a 90% UR	0% a 100% UR
Precisão	±5 %	±2 %	±5 %	±2 %
Vantagem	Tempo de resposta, confiabilidade	Saída de tensão linear, baixo tempo de resposta	Baixo custo	Grande precisão, custo médio
Desvantagem	Difícil acesso, custo elevado	Custo elevado	Baixa confiabilidade, elevado tempo de resposta	Intervalo mínimo de resposta entre medições de 2s

Quadro 2 - Comparativo entre sensores capacitivos de umidade
Fonte: Adaptado de Fialho (2002).

2.2.4 Sensor de Temperatura e Umidade AM2302

Existem no mercado, alguns sensores que em um único encapsulamento, são capazes de realizar as medições de temperatura e de umidade relativa. O AM2302 é um caso destes, como apresentado nos quadros 1 e 2, fazendo-se a comparação do mesmo com sensores de temperatura e umidade relativa separadamente.

O sensor de temperatura e umidade relativa do ar AM2302 (figura 4) é um sensor eletrônico de alta precisão sendo capaz de transmitir os dados de forma digital de umidade e temperatura, por meio de um sensor capacitivo e um termistor, respectivamente (AOSONG ELETRONICS, 2014, p. 02).

Figura 4 – Sensor de Umidade AM2302
Fonte: Aosong Eletronics (2014, p. 01).

O Quadro 3 apresenta as especificações técnicas desse sensor.

Parâmetro	Descrição
Faixa de medição – Umidade	0 a 100% UR
Faixa de medição – Temperatura	-40° a +80°C
Corrente	2,5mA máxima
Precisão de medição – Umidade	±2,0% UR
Precisão de medição - Temperatura	± 0,5°C
Intervalo mínimo entre medições	2s

Quadro 3 – Especificações Técnicas do AM2302
Fonte: Adaptado de Aosong Eletronics (2014, p. 03).

Seu módulo possui um sensor capacitivo de alta precisão para umidade, e um sensor resistivo para medição de temperatura, ambos ligados a uma memória OTP (*One Time Programmable*) onde na fabricação são gravados os coeficientes de calibração, e a um pequeno microcontrolador para o envio dos dados de forma serial (AOSONG ELECTRONICS, 2014).

O AM2302 apresenta 4 terminais, sendo um deles sem finalidade eletrônica, ou seja, não conectado (NC). Os demais são VCC, GND e DAS. O terminal VCC corresponde a sua entrada para alimentação que pode ser na faixa de 3.3V até 5.5V, recomendando-se 5V, o terminal SDA (*Serial Data*) tem como função ser o canal para a transmissão de dados do sensor para o meio externo, e o terminal GND (*ground*) por sua vez é dedicado para conexão ao terra da fonte de alimentação usada. A figura 5 apresenta um esquema típico para sua conexão a um microcontrolador externo.

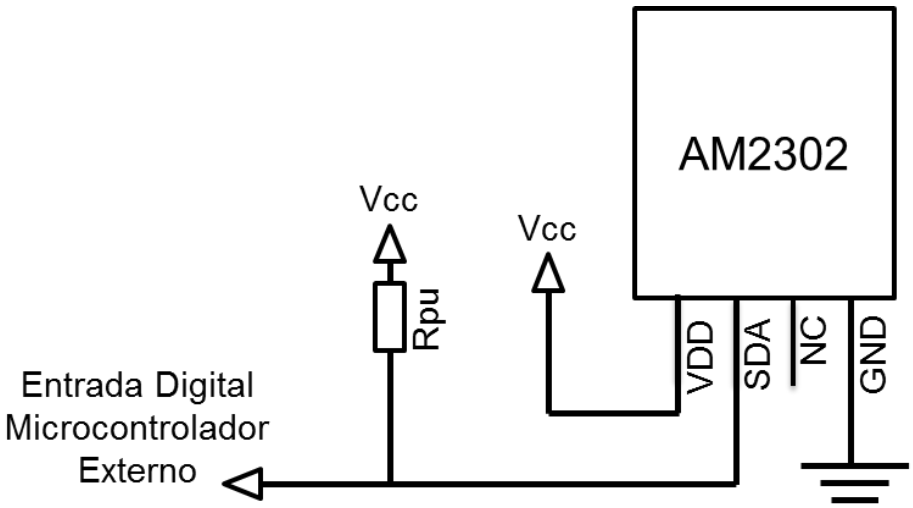


Figura 5 – Conexão típica do AM2302 com um microcontrolador
Fonte: Adaptado de Aosong Eletronics (2014, p. 05).

A tabela 1 apresenta as características de funcionamento do AM3202.

Tabela 1 – Características de funcionamento do AM2302

Parâmetro	Condição	Min.	Típico	Max.
Alimentação	Normal	3,3V	5V	5,5V
Consumo	Dormência	10µA	15µA	
	Mensurando		500µA	
	Média		500µA	
Tensão elétrica de saída em nível alto	Rpu < 25kΩ	90% de Vcc		100% de Vcc
Tensão elétrica de entrada em nível baixo	Descida	0% de Vcc		30% de Vcc
Tensão elétrica de saída em nível alto	Subida	70% de Vcc		100% de Vcc
Corrente de saída	Ligado		8mA	
	Desligado	10	20µA	

Fonte: Adaptado de Aosong Eletronics (2014)

A comunicação do sensor com o microcontrolador externo é realizada serialmente através de um protocolo específico. O pacote de dados possui 5 *bytes* de dados onde os dois primeiros ($Bmsb_{umi}$ e $Blsb_{umi}$) referem-se à umidade, o terceiro ($Bmsb_{temp}$) e o quarto ($Blsb_{temp}$) referem-se à temperatura e o último ($B_{paridade}$) relaciona-se ao *byte* de paridade que é utilizado para detectar algum erro durante a transmissão. A figura 6 ilustra essa disposição dos *bytes* em um pacote de dados do sensor AM2302.

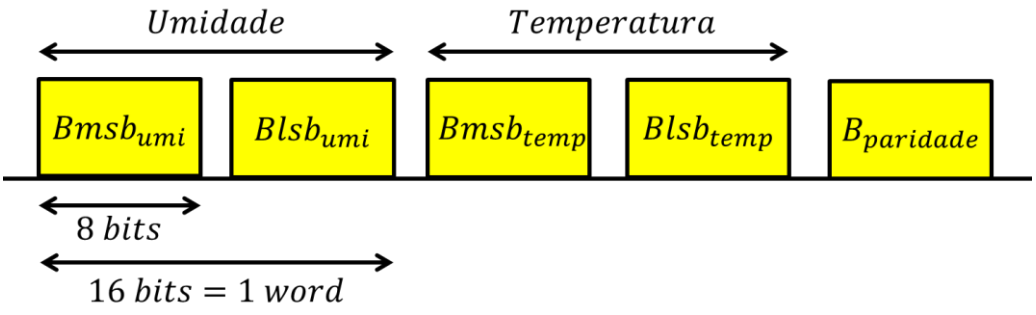


Figura 6 - *Bytes* em um pacote de dados do AM2302
Fonte: Adaptado de Aosong Eletronics (2014, p. 05)

O *byte* de paridade ($B_{paridade}$) é corresponde à soma binária dos outros 4 bytes referentes aos valores de umidade e temperatura. Por exemplo, uma informação 46,5% de umidade relativa e de 55°C de temperatura apresentará o byte de paridade como resultado a soma dos 4 bytes que formam essa informação conforme a equação (1).

$$B_{paridade} = Bmsb_{umi} + Blsb_{umi} + Bmsb_{temp} + Blsb_{temp}$$

$$B_{paridade} = Bmsb_{umi} + Blsb_{umi} + Bmsb_{temp} + Blsb_{temp} \quad (1)$$

Logo para o exemplo citado anteriormente teríamos o byte de paridade:

- Umidade: = 46,5*10 = 465 = 0000 0001 0000 1101
- Temperatura: = 55*10 = 555 = 0000 0010 0010 1011
- $B_{paridade}$ = 0000 0001 + 0000 1101 + 0000 0010 + 0010 1011 = 0011 1011

Caso o microcontrolador verificar que há diferença entre o *byte* de paridade recebido ($B_{paridade}$) com o *byte* de paridade proveniente da soma dos 4 *bytes* de temperatura e umidade que recebeu ($B'_{paridade}$), o mesmo constatará a perda de algum dado durante a transmissão e aguardará 2 segundos para um novo recebimento conforme descrito no fluxograma da figura 7.

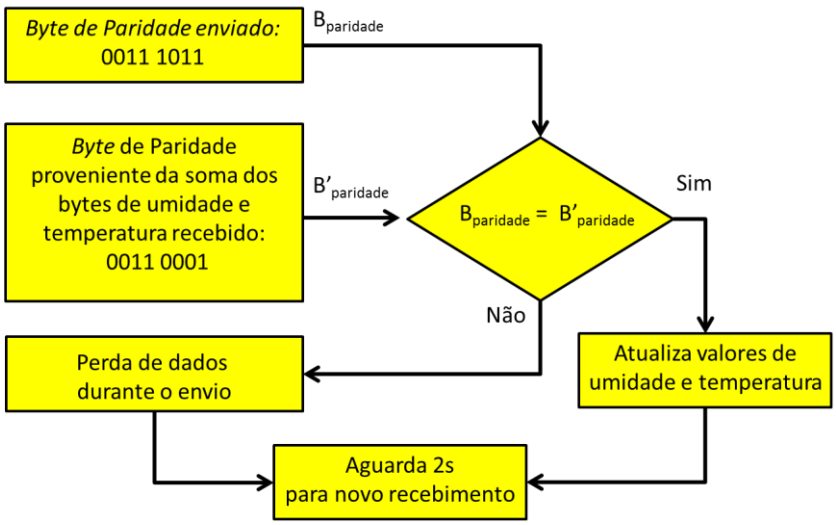


Figura 7 - Fluxograma de validação de dados do AM2302
Fonte: Autoria Própria.

O bit da extrema esquerda, que compõe a informação de temperatura, serve para representar temperaturas abaixo de zero, onde quando em nível lógico alto 1, representa uma temperatura negativa, quando em nível lógico baixo 0, uma temperatura positiva. Logo para representação de uma temperatura -39°C têm-se o seguinte*:*

Temperatura: 39*10 = 390 = 00001111 01100011

Mudando o bit da extrema esquerda para representar a temperatura negativa de -39°C obtém-se: 10001111 01100011

A comunicação com o microcontrolador externo inicia-se após ele enviar um sinal em nível baixo de no mínimo 800µs para o AM2302, posteriormente o AM2302 envia um sinal de em nível baixo de 80µs de duração, seguido de outro sinal em nível alto de mesma duração informando que o mesmo encontra-se preparado para dar início à transmissão de dados. A análise temporal da inicialização da comunicação está ilustrada na figura 8.

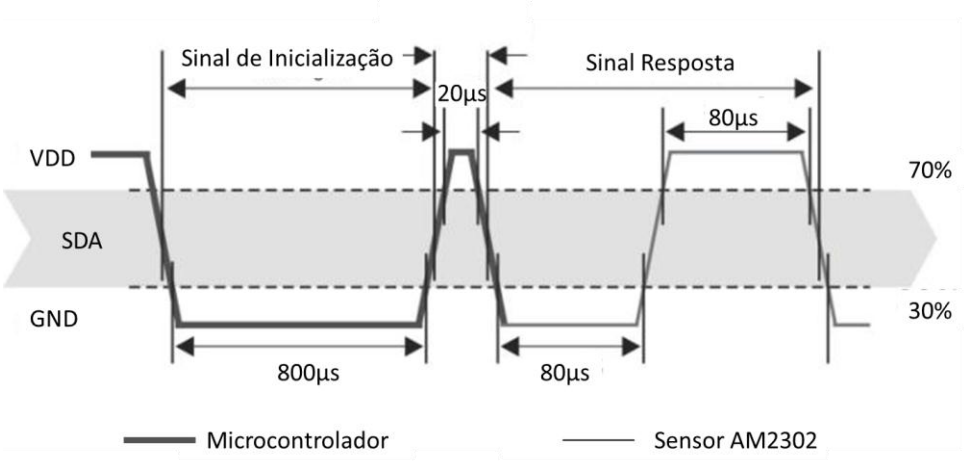


Figura 8 – Inicialização temporal da comunicação de dados
Fonte: Adaptado de Aosong Eletronics (2014, p. 08).

Uma vez realizada a inicialização da comunicação, o AM2302 prepara os 5 frames de informação para enviar ao microcontrolador externo, antes disso a informação é coletada.

2.3 ATUADORES

2.3.1 Diodos Emissores de Luz de Alto Brilho

Os diodos são componentes eletrônicos formados por semicondutores de junção tipo p e tipo n. Estas duas junções quando em colocadas em contato resultam em um equilíbrio entre os elétrons da junção n e as lacunas da junção p, formando uma diferença de potencial entre as duas junções (TIPLER; MOSCA, 2006, p. 146). Analogamente um diodo ideal comporta-se como um curto-circuito quando polarizado diretamente, e como um circuito aberto quando polarizado inversamente. A figura 9 ilustra o processo dessas duas formas de polarização.

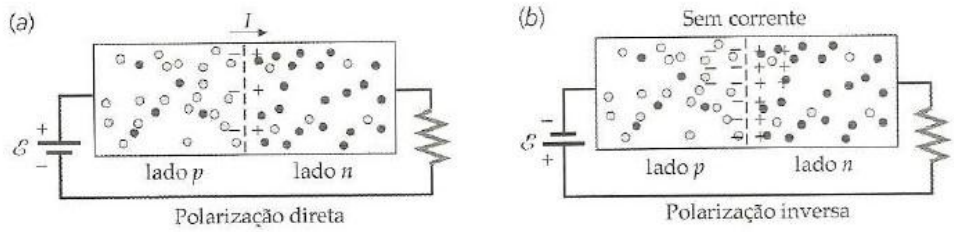


Figura 9 – (a) Diodo polarizado diretamente (b) Diodo polarizado inversamente
Fonte: Tipler; Mosca (2006, p. 147).

Porém um diodo real, representado por uma força eletromotriz $V_{(TO)}$ em série com uma resistência r_T, (figura 10) apresenta algumas características que distorcem essa curva de tensão pela corrente quando polarizados direta e inversamente. A figura 11 apresenta a curva de tensão-corrente de um diodo real, onde V_{RRM} é a tensão reversa máxima que o diodo pode suportar, I_{FSM} é a corrente máxima de surto e I_R a corrente reversa.

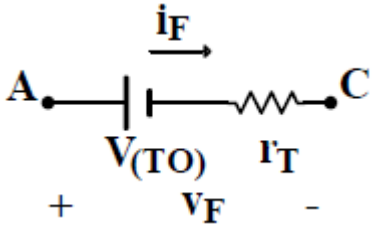


Figura 10 – Circuito equivalente de um diodo
Fonte: Barbi (2000, p. 6).

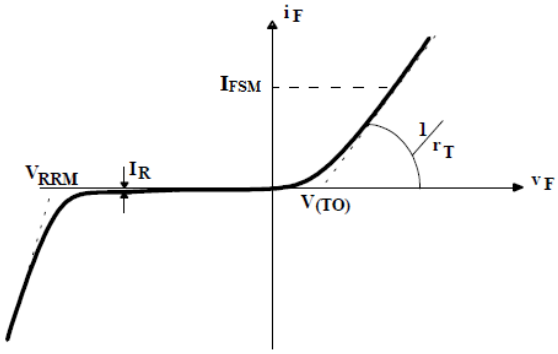


Figura 11 – Curva de tensão-corrente de um diodo real
Fonte: Barbi (2006, p. 6).

Os LEDs (*Light Emitting Diode*) são diodos que quando polarizados diretamente liberam energia na forma de luz. Atualmente vêm sendo empregados na construção de lâmpadas mais eficientes e com maior vida útil.

Cassarly (2008) divide os LEDs em três categorias: indicativos, alto brilho e de alta potência, sendo os dois últimos ilustrados na figura 12. Os LEDs indicativos são frequentemente usados para sinalização do estado de ligado ou desligado em máquinas, aparelhos eletroeletrônicos e entre outros. Através de um invólucro de epóxi colorido, funcionando como filtro óptico, eles são capazes de emitir luz na cor correspondente a da cor do invólucro. O LED de alto brilho é classificado como aquele que produz um brilho de forma a causar desconforto no observador caso olhe diretamente para ele.

Tipicamente os LEDs de potência possuem uma corrente nominal de 350mA a 1000mA enquanto que os LEDs de alto brilho trabalham com correntes nominais em torno de 20mA (RODRIGUES *et al.*, 2011).

Figura 12 – (a) LED de alto brilho (b) LED de alta potência
Fonte: Adaptado de Dias (2012, p. 10).

2.3.2 Resistência de Aquecimento

A resistência eletrônica é definida como a dificuldade encontrada à passagem de corrente através de um condutor submetido à determinada tensão, a resistência elétrica tem como sua unidade no SI o Ohm (Ω). Na situação de aplicar-se uma tensão V, em um condutor qualquer, gerando nele uma corrente elétrica de intensidade I, o que se consta é que para a maioria dos condutores as duas grandezas em questão são diretamente proporcionais (JOHNSON; HILBURN, 1994, p. 15).

Quando uma corrente elétrica percorre um condutor é gerado calor, sendo ele proporcional a resistência do condutor multiplicado pelo quadrado da tensão aplicada. Por meio deste efeito, descrito por James Joule em 1841, as resistências de aquecimento convertem a energia elétrica em calor (GUEDES, 2015, p. 02).

As resistências de aquecimento são distinguidas por algumas caracterísiticas intrínsecas do material qual as compõe, dentre as quais se destacam: coeficiente de temperatura, resistência elétrica, condutividade térmica. O material condutor mais adequado para a fabricação de resistências de aquecimento nem sempre é escolhido somente pelas suas boas características elétricas, pois algumas ligas apesar de eletricamente menos vantajosas possuem condições que satisfazem o seu propósito de utilização, que é gerar calor. Como a liga de níquel-cromo, formada por 80% de níquel 20% de cromo, o que garante uma elevada resistividade, baixo coeficiente de temperatura e grande capacidade de suportar altas temperaturas, sendo assim largamente utilizada para a fabricação de resistências de aquecimento e aparelhos de calefação (BRANCO, 2015, p. 02).

2.3.3 Ventilador Para Desumidificação

Segundo o Centro de Gerenciamento de Emergências (2012), a umidade relativa do ar, expressa em percentual (%), varia com a temperatura e representa a quantidade de água em forma de vapor presente na atmosfera em relação à quantidade máxima que poderia existir nas mesmas condições de temperatura (ponto de saturação).

Frade (2006, p. 11) exibe alguns processos para a retirada de umidade de um ambiente, qual se destaca o de Desumidificação com Arrefecimento. Nesse método o decréscimo da umidade relativa é obtido por meio da condensação sofrida pelo ar através de ventiladores de convecção. Isto ocorre devido ao ar entrar em contato com uma superfície que apresente uma temperatura inferior da temperatura de ponto de orvalho, parte do vapor d'água presente no ar condensa-se, provocando assim uma subtração da umidade relativa.

A convecção forçada é caracterizada pelo escoamento ser proveniente de meios externos, como ventiladores. Os ventiladores de convecção através de um motor, comumente um motor de indução, e uma hélice convertem energia elétrica em energia mecânica, forçando assim a circulação do ar onde ele está instalado. (INCROPERA; DEWITT, 1990, p. 05).

2.3.4 Umidificador de Ambiente

O aumento da umidade relativa é frequentemente alcançado por meio de métodos que empregam umidificadores. Frade (2006, p. 11) exibe alguns desses métodos, qual destaca o de Umidificação com Vapor e o de Arrefecimento com Umidificação.

Na umidificação com vapor o vapor d'água é inserido propriamente sobre o fluxo de ar, este vapor pode ser oriundo de fontes de vapores ou ainda por mecanismos que fazem o aquecimento de água por resistências elétricas submergidas em água. Nesta condição o vapor d'água agirá diretamente no acréscimo da umidade relativa do ambiente.

Na técnica de Arrefecimento com Umidificação, saturadores adiabáticos são empregados com o intuito de se introduzir gotículas de água no ar corrente, estas pequenas gotículas passarão por um processo de evaporação diminuindo a temperatura do ar.

Os umidificadores são aparelhos cuja função básica é gerar vapor de água, o funcionamento de alguns deles consiste na captação de água fria e através de uma parede porosa e um ventilador, introduzir esse vapor em um ambiente (BRAIN, 2010).

Há também os umidificadores ultrassônicos, que comumente são empregados para uso doméstico com o intuito de manter a umidade relativa do ar dentro dos níveis recomendados para o conforto humano. Seu funcionamento é baseado na utilização de um cristal piezoelétrico imerso em um reservatório de água, vibrando em uma frequência por volta de 1,6MHz, gerando uma fina e homogênea névoa de vapor de água frio e inodoro (BRAIN, 2010). A fotografia 1 apresenta um umidificador doméstico onde seu funcionamento baseia-se na utilização de cristais piezoelétricos vibrando em frequência ultrassônica.

Fotografia 1 – Umidificador ultrassônico doméstico
Fonte: Autoria Própria.

2.4 MICROCONTROLADORES

Um microcontrolador (µC) é um componente eletrônico que pode ser programado de acordo com a sua utilidade e com a necessidade do programador, e é utilizado no controle de processos lógicos (SOUZA, 2005, p. 03).

Os microcontroladores foram projetados com o intuito de permitir a comunicação de seu microprocessador com o meio externo, pois apesar da grande capacidade de processamento de um microprocessador o mesmo é incapaz de exercer essa função. Para isto, o microcontrolador é dotado de circuitos internos como conversores A/D, temporizadores, memórias em um único circuito integrado. Na figura 13, são ilustrados de uma forma geral os elementos internos de um microcontrolador genérico.

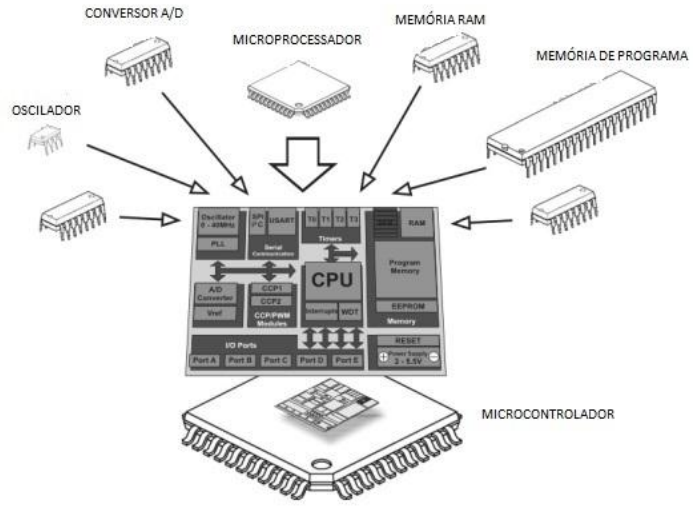


Figura 13 – Elementos construtivos de um microcontrolador genérico
Fonte: Adaptado de Araujo (2015, p. 2).

O microcontrolador é programável, pois toda a lógica de operação é estruturada na forma de um programa e gravada dentro de um componente. Depois disso, toda vez que o microcontrolador for alimentado, o programa interno será executado. Quanto à “inteligência” do componente, pode-se associá-la à Unidade Lógica Aritmética (ULA), pois é nessa unidade que todas as operações lógicas e aritméticas são executadas. Quanto mais poderosa a ULA do componente, maior sua capacidade de processar informações (SOUZA, 2005, p. 04).

Atentando à unidade central de processamento, comumente chamada de CPU, essa consiste em uma unidade lógica aritmética (ULA), uma unidade de controle e também unidades de memória especiais, chamados de registradores. A unidade lógica aritmética é a região onde são realizadas as operações lógicas aritméticas.

Todas as funções ou resultados dessas operações realizadas na ULA podem ser transmitidos para uma memória de dados tanto como para uma unidade de saída. Anteriormente foi citada a unidade de controle, a qual é responsável por comandar as operações da ULA e de todas as outras unidades que virão a estar conectadas à CPU, fornecendo sinais de controle e temporização.

Entre os microcontroladores mais habituais destacam-se os da família ATmega, 8051 e PIC. O PIC desenvolvido pela Microchip, destaca-se por ser precursor na utilização da arquitetura do tipo RISC (*Reduced Instruction Set Computer*), onde diferentemente da arquitetura Von Neumann, esta possui

barramentos distintos para programas e para dados. O quadro 4 apresenta alguns dos principais microcontroladores pertencentes a esta família.

Microcrontrolador	Memória de Programa	Memória RAM	Memória EEPROM	Frequência Máxima
PIC16F83	51kB Flash	36kB	64kB	10MHz
PIC16F84	1kB Flash	68kB	64kB	10MHz
PIC16CR83	51kB ROM	36kB	64kB	10MHz
PIC16CR84	1kB ROM	68kB	64kB	10MHz

Quadro 4 – Principais microcontroladores da família PIC
Fonte: Adaptado de Júnior e Avegliano (2005, p. 8).

2.4.1 Plataforma Arduino

O Arduino é uma plataforma eletrônica, com controle de entradas e saídas de dados, que através desses sistemas ligados a sensores e atuadores, consegue responder uma ação física. É uma plataforma padronizada aberta (*open source*), baseando-se em microcontroladores da família ATmega da Atmel (ARDUINO, 2015). A figura 14 apresenta os principais blocos e funcionalidades comuns da arquitetura de um Arduino.

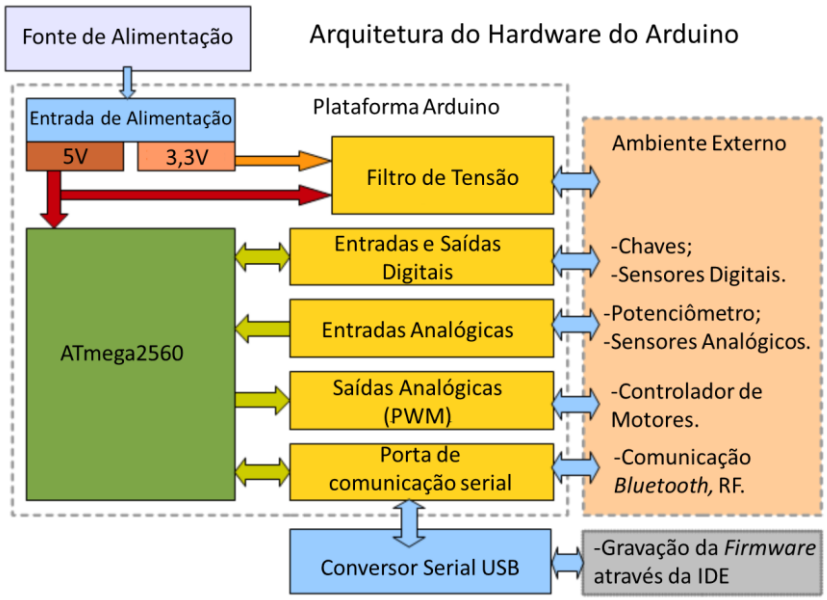


Figura 14 – Arquitetura interna Hardware Arduino
Fonte: Basconcello (2012).

Atualmente existem vários modelos da plataforma Arduino, onde variações na quantidade de entradas e saídas digitais e analógicas, memória, saídas PWM, são determinantes para a escolha do utilizado em projeto. O quadro 5 apresenta as principais características de alguns modelos comerciais da plataforma Arduino.

Parâmetro/Modelo	Arduino Uno	Arduino Mega	Arduino Leonardo	Arduino Due	Arduino Nano
Microcontrolador	ATmega328	ATmega2560	ATmega32u4	AT915AM3X8E	ATmega168 ou ATmega328
Entradas e Saídas digitais	14	54	20	54	14
Entradas analógicas	6	16	12	12	8
Saídas PWM	6	15	7	12	6
Memória	32kB	256kB	32kB	512kB	16kB ou 32kB
Clock	16MHz	16MHz	16MHz	84MHz	16MHz
Corrente máxima por entrada e saída digital	40mA	40mA	40mA	130mA	40mA

Quadro 5: Comparativo entre plataformas Arduino
Fonte: Adaptado de Monk (2013).

2.4.1.1Conversor analógico-digital (ADC)

Os sinais oriundos de sensores que captam variações no meio externo são na sua maior parte sinais analógicos. O processamento de sinais no domínio analógico é uma tarefa complexa e pouco flexível devido ao uso de elementos de circuitos como capacitores, indutores e resistores, diodos e amplificadores. Por outro lado o processamento de sinais na forma digital recorre a três elementos básicos da eletrônica digital: multiplicadores, somadores e memórias (HAYKIN, 2001, p. 33).

Na abordagem analógica tem-se o conveniente da capacidade de resolução de equações diferenciais que descrevem modelos físicos sem ter que utilizar soluções aproximadas para elas, em contrapartida no processamento digital tem-se que recorrer a computações numéricas para suas operações. Computações estas que levam um tempo para ser realizadas, trazendo a desvantagem de que nem sempre é possível dizer que o processamento digital seja capaz de operar em tempo real.

No entanto a abordagem digital apresenta algumas vantagens importantes sobre o processamento na forma analógica. Podemos destacar a sua flexibilidade, onde um mesmo *hardware* é capaz de processar diferentes tipos de sinais somente fazendo-se modificações em seu *software*, destaca-se também a sua capacidade de repetitividade, na qual se refere à capacidade de uma operação apresentar os mesmos resultados de maneira exata depois de repetidas vezes, enquanto que na abordagem analógica mudanças como temperatura ou da tensão da fonte de alimentação já são suficientes para surgir mudanças.

A conversão de sinais analógicos para digitais se faz necessário quando se quer representar grandezas físicas na forma continua em forma discreta, usando para isto as técnicas de amostragem, quantização e codificação. A figura 15 ilustra o processo de conversão de um sinal analógico para digital, o qual é amostrado, quantizado e posteriormente codificado.

Figura 15 – Processo de conversão analógica – digital
Fonte: Adaptado de Aragão Filho (2013).

O desempenho de um conversor A/D comumente está atrelado a sua qualidade de filtragem, velocidade e resolução de seus circuitos internos envolvidos na aquisição e conversão do sinal. De modo geral, um sinal analógico é convertido para digital em três etapas: filtragem *anti-aliasing*, amostragem (S/H) e amostragem e quantização do sinal amostrado, conforme ilustrado na figura 16.

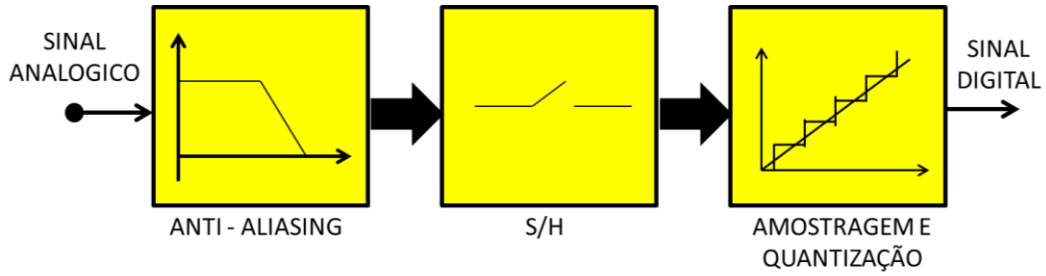


Figura 16 – Generalização dos circuitos internos de um conversor A/D
Fonte: Adaptado de Allen e Holberg (2002, p. 931).

O primeiro bloco posterior ao sinal analógico de entrada, de denominação filtro *anti-aliasing*, tem como finalidade evitar o ruído gerado fora da banda útil do sinal e assim não degradar o sinal devido ao efeito de *aliasing,* que corresponde a um erro de atribuição inerente às análises espectrais de dados discretos onde há uma consequente sobreposição dos espectros do sinal, isto ocorre devido aos limites impostos pelo intervalo de amostragem onde para evitar este efeito deve-se obedecer ao critério de Nyquist.

> Se um sinal for amostrado em uma taxa de amostragem menor que duas vezes a frequência de Nyquist, uma ou mais componentes de frequência mais baixa aparecerão nos dados amostrados. Esse fenômeno é denominado *aliasing* (NATIONAL INSTRUMENTS, 2014).

O circuito S/H (*Sample and Hold*) tem como função amostrar o sinal analógico de entrada, ou seja, discretizar através de amostras (*sample*) esse sinal contínuo no tempo, utilizando um período definido para as amostragens. Já o segurador (*hold*) tem como finalidade reter o sinal amostrado. A figura 17 ilustra um sinal amostrado a partir de um sinal analógico de entrada.

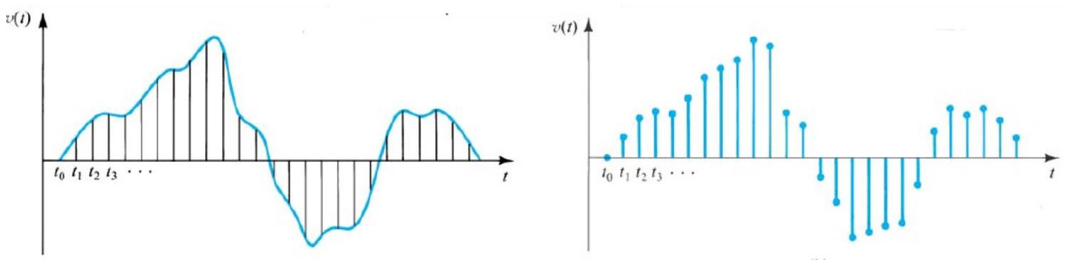


Figura 17 – Sinal analógico e Sinal amostrado
Fonte: Klimach e Fabris (2014).

Os conversores A/D apresentam algumas características intrínsecas que qualificam seu desenvolvimento como resolução, faixa dinâmica (*span*), tempo de conversão e erros quantização. A faixa dinâmica ou (*span*) refere-se à diferença entre o valor do fundo de escala e o valor mínimo no qual o conversor é capaz de medir. A resolução (Res) é a menor variação percebível pelo conversor A/D dentro de sua faixa dinâmica (Fd). Ela é dada pela relação entre a faixa dinâmica e o número de bits do conversor de acordo com a equação 2.

$$Res = \frac{\mathrm{Fd}}{2^{n}-1} \qquad (2)$$

Conforme a equação 2, o microcontrolador ATmega2560 apresenta então um conversor A/D com resolução na casa de aproximadamente 4,887mV, levando em consideração possuir um conversor de 10 bits com uma faixa dinâmica de 0V a 5V, e a figura 18 apresenta o sinal analógico quantizado.

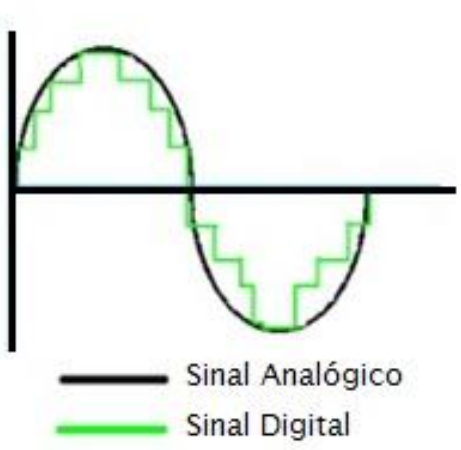


Figura 18 – Sinal analógico quantizado
Fonte: Adaptado de Klimach e Fabris (2014).

O tempo de conversão é o período mínimo de tempo o qual o conversor assegura em que se tenha o valor digital válido na sua saída, a partir do instante que foi aplicado um sinal analógico de entrada e iniciado o processo de conversão. Este tempo depende da topologia e da resolução do conversor A/D, comumente quanto maior a resolução maior o intervalo de conversão (ALLEN; HOLBERG, 2002, p. 998).

Para a conversão em 10 bits do conversor A/D do ATmega2560, tem se um tempo de conversão entre 13µs e 260µs, onde para isto deve ser alimentado com um *clock* com frequência entre 50kHz e 200kHz (ATMEL, 2014, p. 268).

Seu conversor A/D possui um total de 16 canais que podem ser utilizados com apenas um conversor A/D, onde através da habilitação por modo diferencial, quatro pares dos mesmos podem ser lidos um em relação ao outro, não necessitando ter como base a comparação com o GND. Nesse caso se dois canais medidos tiverem tensões iguais, a saída lida será zero.

Os pares de canais do conversor que carregam esse recurso são os canais ADC1 com ADC0, ADC com ADC2, ADC9 com ADC8 e ADC11 com ADC10, esse

recurso permite um estágio de ganho programável, proporcionando passos de amplificação de 0dB, 20dB ou 46dB sobre a tensão de entrada diferencial, antes da conversão do A/D (ATMEL, 2014, p. 268).

Entre as diferentes topologias de conversores A/D, os microcontroladores que contém este recurso normalmente utilizam a de aproximações sucessivas os quais apresentam um tempo de conversão igual para qualquer valor analógico de entrada, dentro de sua gama de valores projetada, além de apresentarem um custo de implementação reduzido quando comparado com as demais topologias (TOCCI, 2007). A figura 19 apresenta o diagrama de blocos do conversor A/D por aproximações sucessivas.

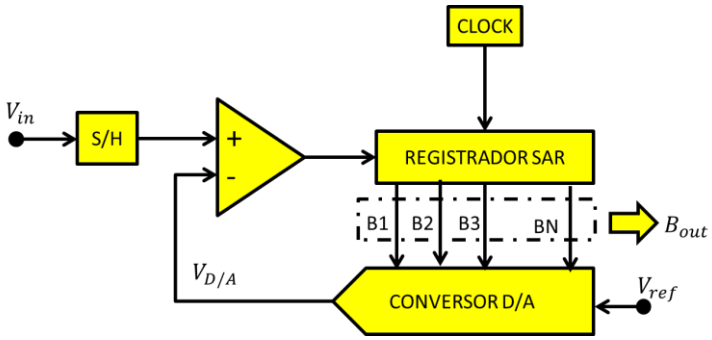


Figura 19 – Diagrama de blocos de um ADC por aproximações sucessivas
Fonte: Adaptado de Baker (2015, p. 1003).

O conversor analógico digital acima representado é composto basicamente por um registrador, um conversor D/A e um comparador. O bloco do registrador de aproximações sucessivas (SAR) tem como função obter em sua saída a palavra digital que corresponde ao valor de tensão próximo do sinal analógico de entrada V_{in}, ele é composto por duas entradas, uma para o sinal de entrada proveniente do comparador, e a outra do sinal do *clock* (BAKER, 2010, p. 1004).

O circuito comparador possui duas entradas, uma proveniente do conversor D/A no qual tem como saída um valor de tensão ($V_{D/A}$), a outra correspondente ao sinal de entrada (V_{in}) que será convertido e uma saída digital que vai para o registrador SAR. O sinal que será convertido inicialmente será comparado com o valor inicial proveniente do conversor D/A, a partir disso a saída do comparador admitirá nível lógico alto ou nível lógico baixo, onde a mesma será enviada para o registrador SAR fornecer uma nova palavra digital, que será correspondente a

metade do valor de fundo escala (V_{ref}), devido o numero da primeira iteração (n) corresponder a um, conforme a equação 3.

$$V_{D/A_{(n=1)}} = \frac{V_{ref}}{2^n} \qquad (3)$$

A partir disso, o comparador realizará sucessivas análises entre os dois sinais distintos para determinar qual dos dois possui o maior valor e, assim, incrementando o número da iteração. Caso sua saída admita o valor de nível lógico 1, ou seja, se V_{in} for maior que $V_{D/A,}$ o registrador SAR apresentará em sua saída B_{out} a palavra digital correspondente a V_{ref} divido por 2^n , onde é somada ao valor de tensão anterior do conversor D/A ($V_{D/A(n-1)}$), o mesmo ocorre de forma oposta caso a saída do comparador admita um valor em nível logico 0, V_{in} menor que $V_{D/A,}$ o registrador SAR subtrai V_{ref} divido por 2^n do valor de tensão anterior do conversor D/A , conforme a equação 4 e equação 5.

$$V_{D/A_{(n)}} = V_{D/A_{(n-1)}} + \frac{V_{ref}}{2^n} \quad \text{,para } V_{in} > V_{D/A} \qquad (4)$$

$$V_{D/A_{(n)}} = V_{D/A_{(n-1)}} - \frac{V_{ref}}{2^n} \quad \text{,para } V_{in} < V_{D/A} \qquad (5)$$

Este processo é repetido n vezes e a cada nova iteração se acrescenta ou subtrai o valor de $V_{ref}/2^{n-1}$. Ao final de n iterações tem-se em B_{out}, a palavra digital correspondente ao valor analógico de V_{in}.

A tabela 2 ilustra uma conversão A/D por aproximações sucessivas do conversor A/D de 10 bits, V_{ref} igual a 5V. Para o exemplo utiliza-se um sinal de entrada V_{in} de 4,37V.

Tabela 2 – Conversão A/D, 10 bits, por aproximações sucessivas

n	$V_{ref}/2^{n-1}$	$Vd/a_{(n-1)}$	Vd/a_n	$V_{in} > V_{d/a}$
1	2,50000V	0,00000V	2,50000V	Sim
2	1,25000V	2,50000V	3,75000V	Sim
3	0,62500V	3,75000V	4,37500V	Não
4	0,31250V	4,37500V	4,06250V	Sim
5	0,15625V	4,06250V	4,21875V	Sim
6	0,07812V	4,21875V	4,29687V	Sim
7	0,03906V	4,29687V	4,33594V	Sim
8	0,01953V	4,33594V	4,37500V	Não
9	0,00977V	4,37500V	4,359375V	Sim
10	0,00391V	4,35937V	4,362381V	Sim

Fonte: Autoria Própria.

Levando em consideração que há um o erro inerente aos truncamentos e arredondamentos no processo de quantização, definido por 2^n dos níveis da escala do quantizador (ALLEN; HOLBERG, 2012, p. 1005). O conversor do ATmega2560 possui um erro de 0,5 LSB (*Least Significant Bit*), um LSB corresponde a distância entre dois níveis na escala de quantização. Conforme a figura 20 desta maneira todo ponto que se encontrar no intervalo do segmento em verde terá o nível da escala de quantização correspondente ao representado pela bola em amarelo.

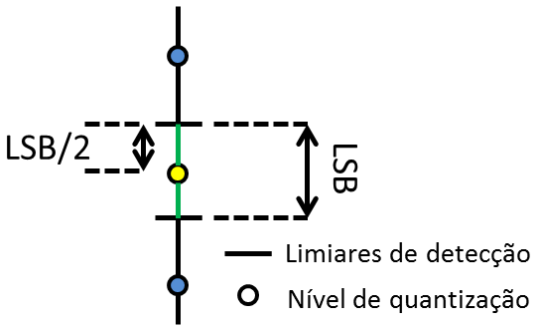


Figura 20 – Níveis de quantização e limiares de detecção
Fonte: Autoria Própria.

2.4.1.2 *Timers* e contadores

Todo e qualquer microcontrolador deve possuir um recurso de contagem de tempo, no qual o comando é feito através de um sinal de frequência geralmente proveniente de um cristal oscilador, interno ou externo, que sincronizará todo o

funcionamento do microprocessador, monitorando o tempo de cada um dos eventos relacionados aos periféricos e dispositivos integrados a ele (PEREIRA, 2009, p. 256).

Os *timers* são na sua maior parte contadores digitais assíncronos, ou seja, seu estado varia conforme um sinal de entrada denominado *clock*, onde a cada pulso de *clock* faz o *timer* incrementar o seu valor de contagem. O ATmega2560 possui em seu interior 6 *timers* distintos, Timer0, Timer1, Timer2, Timer3, Timer4 e Timer5, sendo três deles de 8bits e os outros três de 16bits. O quadro 6 apresenta o número de bits de cada *timer*:

Timer	Número de bits	Número máximo de contagens
Timer0	8	256
Timer1	8	256
Timer2	8	256
Timer3	16	65536
Timer4	16	65536
Timer5	16	65536

Quadro 6 - Número de bits por timer
Fonte: Adaptado de Atmel (2014, p.167).

Cada *timer* carrega consigo registradores, formado cada um por 8 bits, que permitem a configuração do modo de operação, fonte do pulso de *clock* e outros ajustes de cada timer correspondente. Os registradores mais importantes dos *timers* do ATmega2560 são:

-TCCRx (*Timer/Counter Control Register*) – Registrador de configuração da fonte de pulso *clock*;

-OCRx (*Output Compare Register*) - Registrador de comparação;

-ICRx (*Input Capture Register*) - Registrador de captura, disponível somente nos *timers* de 16bit;

-TIFRx (*Timer/Counter Interrupt Flag Register*) – Registrador de sinalização de interrupção.

Uma vez que o conteúdo do registrador estiver completamente preenchido, um evento de interrupção pode ser gerado, a fim de executar uma determinada ação.

A fonte de *clock* dos *timers* do ATmega2560 pode ser proveniente de três fontes diferentes, através do cristal oscilador interno, por um cristal oscilador externo ou pelo seu *prescaler*, o qual divide frequência do *clock* interno por um fator pré-determinado.

Quanto ao modo de funcionamento têm-se a de modo normal, onde o *timer* conta até seu valor máximo e volta para zero, a de modo CTC *(Count to Compare*) onde o *timer* conta de 0 até um valor definido e compara com o valor no registrador OCRnA, e o modo para configuração para Modulação por Largura de Pulso.

Havendo uma grande quantidade de periféricos que necessitam de um timer, para que suas rotinas sejam executadas em intervalos periódicos, temporizadores são essenciais. Vale ressaltar que o código contido nas interrupções deve ser o mais minimizado possível, com a intenção de evitar que o tempo para execução desse código seja superior ao tempo da próxima interrupção.

2.4.1.3 Modulação PWM

Em ocasiões onde há a necessidade do controle da potência a ser entregue à uma carga pode-se empregar transistores, usados como chave, para abrir ou fechar um circuito. Um sinal gerado por um circuito integrado específico ou por um microcontrolador dá o comando de quando este componente eletrônico deve conduzir ou deixar de conduzir no circuito empregado. Variando-se a largura do pulso e também o intervalo, de modo a termos ciclos ativos diferentes nesse sinal, pode-se controlar a potência média aplicada a uma carga. Assim, quando a largura do pulso varia de zero até o máximo, a potência também varia na mesma proporção. Este princípio é usado no controle por Modulação por Largura de Pulso (PWM) onde a variação da largura do pulso controla o ciclo ativo (*Duty Cycle*) do sinal aplicado a uma carga e, com isso, a potência aplicada a ela (ARAUJO, 2010, p. 73).

A figura 21 exemplifica um sinal PWM com *Duty Cycle* variando de 0 a 100%, em intervalos periódicos. Para um ciclo ativo de 0% tem-se 0V na saída e no outro extremo ciclo ativo em 100% tem-se 5V. Valores intermediários de *Duty Cycle* de 25%, 50% e 75% representariam a transferência média entregue de ¼ , ½ e de ¾

da potência máxima, respectivamente; e analogamente desta forma para outro valor intermediário.

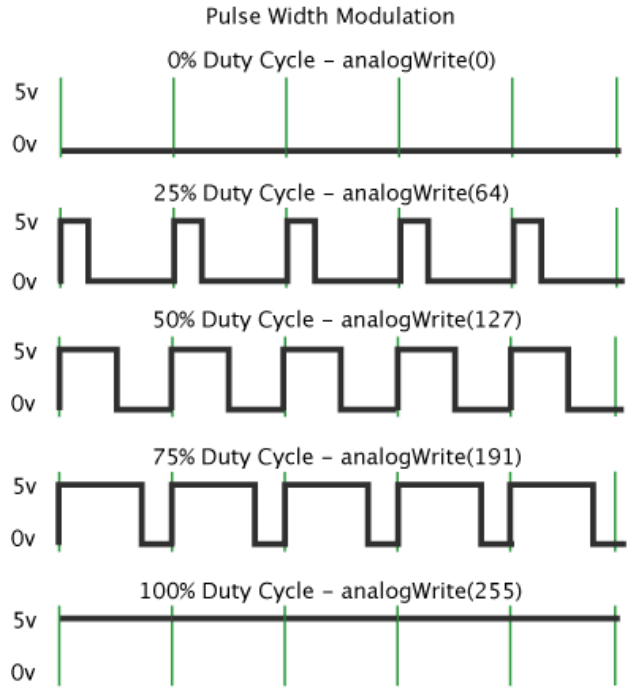


Figura 21 – Modulação PWM com ciclo ativo de variável
Fonte: Hirzel (2015).

Em relação às saídas PWM no Arduino, elas podem apresentar uma resolução de saída de 8 ou 16 bits, recurso este proveniente do microcontrolador ATmega2560 (ATMEL, 2014, p.7). Os pinos 2, 3, 5, 6, 7, 8, 11, 12, 44, 45 e 46 possuem uma saída PWM com resolução de 16 bits, enquanto que os pinos 4, 9, 10 e 13 possuem uma resolução de 8 bits.

Como exemplos de utilização deste controle de potência através do PWM, tem-se o controle da iluminação de LEDs de alto brilho e do aquecimento (potência térmica) de resistências para este fim.

2.4.2 Software Para Programação em C/C++

A plataforma para compilação C/C++, onde são gerados os códigos e gravados no microprocessador, é uma IDE (*Integrated Development Environment* – Ambiente de Desenvolvimento Integrado) própria do Arduino.

A linguagem C é uma linguagem de alto nível e como as outras linguagens desse nível ela está dividida em módulos ou estruturas independentes. Esses

módulos e estruturas são chamados de funções na linguagem C, onde visam facilitar o desenvolvimento de códigos (PEREIRA, 2009, p. 247).

Envolvidos pela filosofia básica que é adotada, o C é uma linguagem de programação de finalidade geral, onde muitas aplicações em sistemas operacionais, sistemas de comunicação, programas para solução de problemas a executam.

Durante o processo de compilação, arquivos fonte são compilados separadamente, gerando um arquivo chamado código-objeto onde contêm instruções em linguagem de máquina. Na parte referente à passagem do programa fonte, todas as bibliotecas declaradas nos códigos e todos os arquivos objeto são processados em conjunto (AHO; ULLMAN, 2008, p. 04).

A figura 22 ilustra as etapas de processo de compilação para a geração de um código executável.

Uma característica interessante da linguagem C é a sua alta portabilidade, tornando possível tomar um código fonte escrito para uma máquina, compilá-lo e rodá-lo em outra máquina com pouca ou nenhuma alteração.

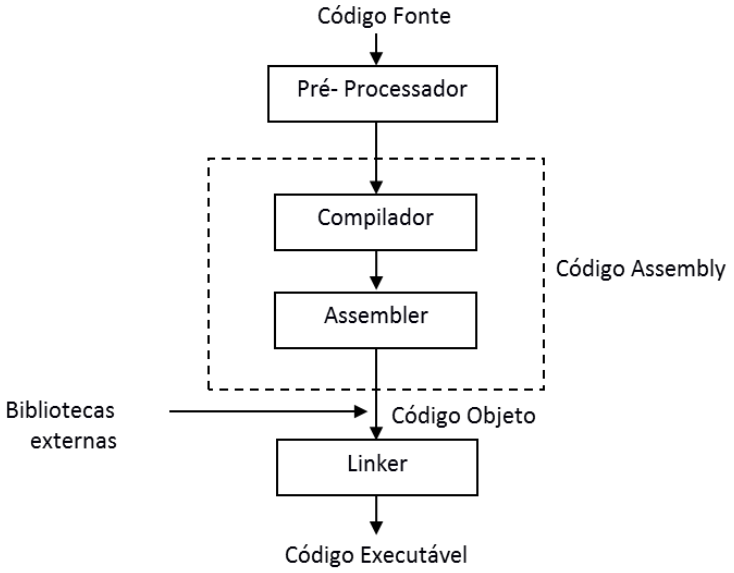


Figura 22 – Etapas do processo de compilação/passagem de programa
Fonte: Adaptado de Aho e Ullman (2008, p. 3).

Toda a parte de programação em linguagem C desenvolvida para o projeto encontra-se em anexo no Apêndice A.

2.4.3 Módulo Rele

O rele é um interruptor eletromecânico, onde a ação física deste interruptor ocorre quando a corrente elétrica percorre as espiras da bobina do relé, criando um campo magnético que atrai a chave responsável pela mudança de estado dos contatos. Normalmente o rele é formado por dois circuitos elétricos independentes, sendo o primeiro circuito composto por uma bobina responsável por gerar o campo magnético e o segundo circuito é composto uma chave de contato, utilizada para fechar e abrir o circuito o qual está operando, conforme a figura 23.

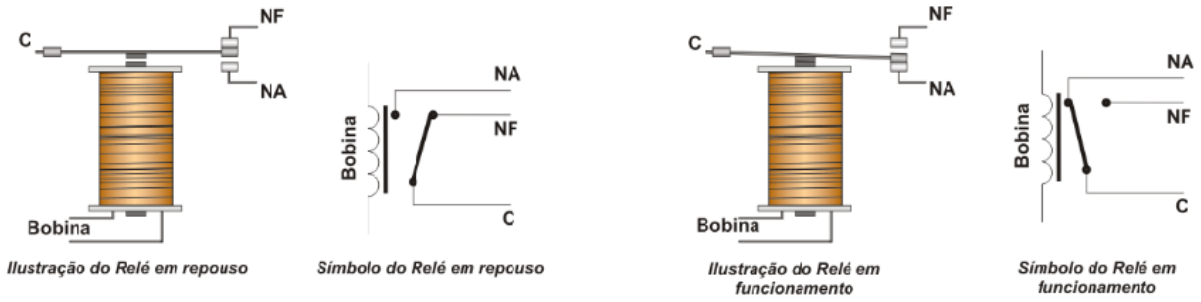


Figura 23 – Rele em estado aberto e fechado
Fonte: Saavedra *et al*. (2013, p. 38).

A comutação do relé ocorre quando uma corrente elétrica percorre as espiras da bobina, gerando um campo eletromagnético, que por sua vez aciona o contato do segundo circuito.

Geralmente os reles apresentam dois tipos de contato, normalmente aberto (NA) e normalmente fechado (NF). No momento que é acionado, ou seja, quando o rele recebe o pulso elétrico, ocorre a comutação da chave eletromecânica, fazendo com que o contato normalmente aberto seja fechado e o contato normalmente fechado seja aberto (SAAVREDA *et al.*, 2013, p. 38).

Geralmente são utilizados em ocasiões onde cargas exijam um nível maior de corrente ou tensão no qual o circuito de comando pode oferecer. Sendo essa característica de grande utilidade, pois a partir de correntes e tensões muito pequenas, em relação ao circuito que se almeja controlar, o mesmo consegue controlar circuitos de altas correntes e tensões, utilizando para isso transistores, os quais fornecem correntes baixas de saída e não conseguiriam por si só suprir as

necessidades da carga nos terminais do rele. A figura 24 apresenta uma aplicabilidade de um rele, em um circuito de controle por transistores.

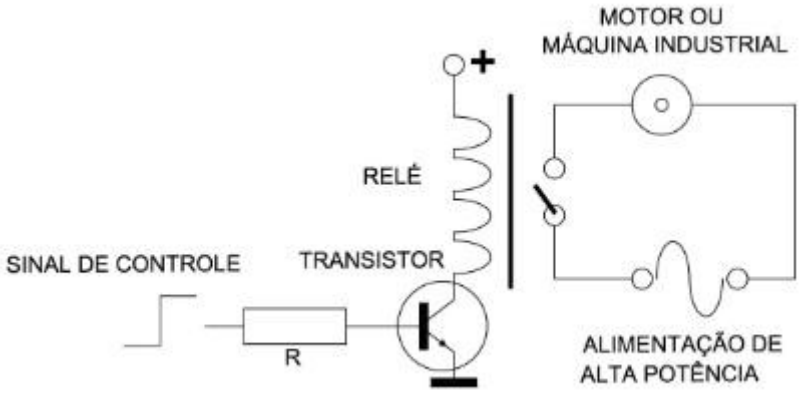
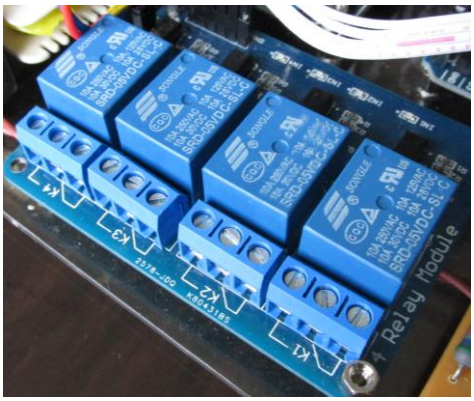


Figura 24 – Rele com circuito acionador através de transistor
Fonte: Adaptado de Braga (2012).

Algumas placas possuem conjunto de reles para utilização, como o módulo de reles SRD-05VDC-SL-C presente na fotografia 2, permitindo controlar 4 reles, saídas K1 a K4, com cargas de 110Vac a 225Vac e de 28Vdc a 30Vdc com correntes de até 10A, a partir de um sinal de 5Vdc.



Fotografia 2 – Módulo rele SRD-05VDC-SL-C
Fonte: Autoria Própria

2.4.4 Interface Homem Máquina – IHM

Em um contexto geral, a interação humana com uma máquina ou equipamento é feita através de teclas e *displays* para assim definição dos *set-points,* parâmetros e navegação do operador. Esse conjunto com esses objetivos recebe a

de denominação de Interface Homem Máquina (IHM), que é basicamente todo e qualquer sistema utilizado como sinalizador de eventos ou *status* de uma máquina, de forma amigável, eficiente e ergonômica entre sistemas de automação complexos e seu operador. (MARTINS, 2012, p. 129).

As IHMs (exemplificadas na figura 25) permitiram que os sistemas de controle de processos se tornassem muito mais interativos, possibilitando o operador usar "*displays*" simples para acompanhar as condições de uma máquina e através de um teclado realizar configurações (MARTINS, 2012, p.130).

Figura 25 – Exemplo de IHMs
Fonte: Martins (2012, p. 131).

3 DESENVOLVIMENTO

3.1 VISÃO GERAL DO PROJETO

Para a automatização do microambiente propiciando o controle e o monitoramento de temperatura, umidade e luminosidade em seu interior, utilizaram-se sensores, atuadores, circuitos eletrônicos específicos e um sistema central, conforme ilustrado no diagrama de blocos da figura 26. As linhas em vermelho indicam a comunicação com o Arduino.

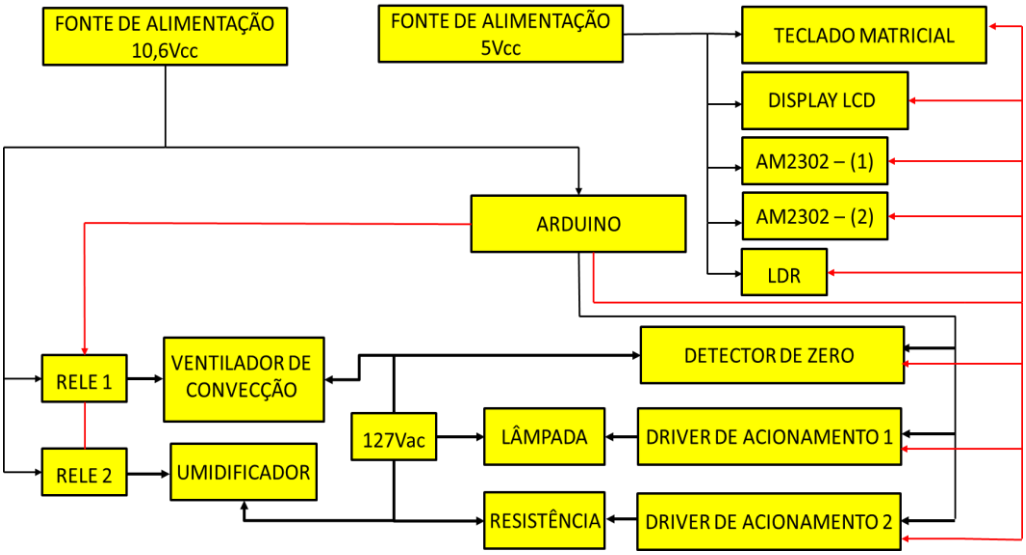


Figura 26 – Diagrama de Blocos Geral do projeto
Fonte: Autoria Própria

A estrutura utilizada como microambiente, trata-se de uma estufa laboratorial mod. 347 da fabricante Fanem, apresentada na fotografia 3.

Fotografia 3 - Incubadora MOD. 347 Fanem
Fonte: Autoria Própria.

Um microambiente para esse fim deve possuir em sua estrutura materiais que o isolem do ambiente externo, facilitando o controle sobre as variáveis em estudo.

A figura 27 ilustra o fluxo de dados no sistema, onde os sensores, AM2302 e LDR, fazem a leitura das variáveis dentro do microambiente, repassando essas informações à placa Arduino, que por sua vez, através de seu microcontrolador ATmega2560 analisa essa informação e age conforme a *firmware* programada sobre os atuadores, os quais são resistência de aquecimento, LEDs de alto brilho, ventilador para convecção e um umidificador.

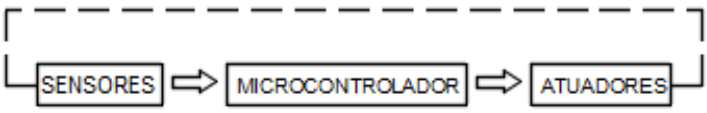


Figura 27 – Esquemático do fluxo de dados no microambiente
Fonte: Autoria Própria.

Pela necessidade de um grande número de portas digitais, totalizando 23 (8 para o teclado matricial, 5 para o display LCD, 4 para comando dos relés, 2 para os sensores AM2302, 1 utilizada como interrupção para o detector de zero, 4 saídas PWM para os LEDs) e somente uma entrada analógica para o sensor LDR, optou-se pela escolha do Arduino Mega 2560, visto que outras opções que contemplavam os

requisitos como o Arduino Due e o Leonardo, ou tinham preço mais elevado ou aprsentaram difículdade de aquisição.

A alimentação do Arduino Mega 2560 pode ser feita através de conexão USB ou por meio de uma fonte externa, sendo recomendável nível de tensão de 6 a 20Vcc. O modelo Arduino Mega 2560 é ilustrado na figura 28.

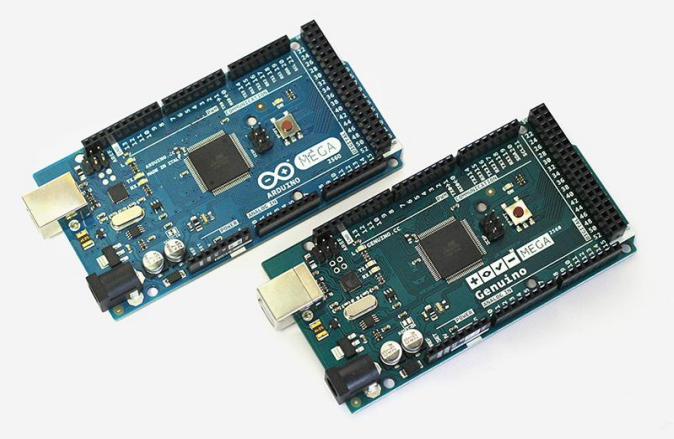


Figura 28 – Placas Arduino Mega 2560
Fonte: Arduino (2015).

A retificação da tensão em 12Vac é obtida através de uma ponte de diodos 1N4004, a tensão alternada em 12Vac é proveniente de um transformador 127/12Vac. Em relação à alimentação do módulo de controle, ela será proveniente de uma fonte de alimentação com saídas de 5Vdc e 10,6Vdc, fornecidas pelos reguladores de tensão LM7805 e LM317, respectivamente, representado na figura 30. A figura 29 apresenta a fonte de alimentação com seus principais componentes, com exceção do transformador.

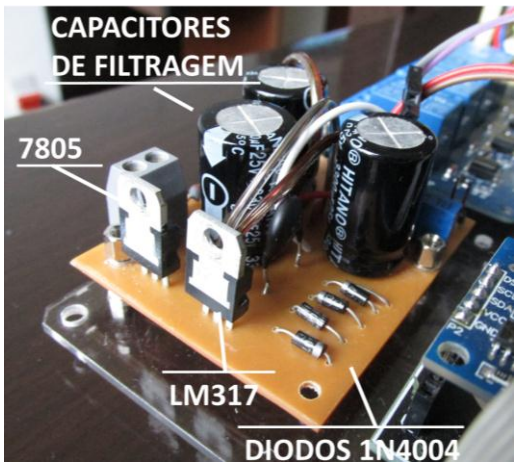


Figura 29 – Fonte de alimentação
Fonte: Autoria Própria.

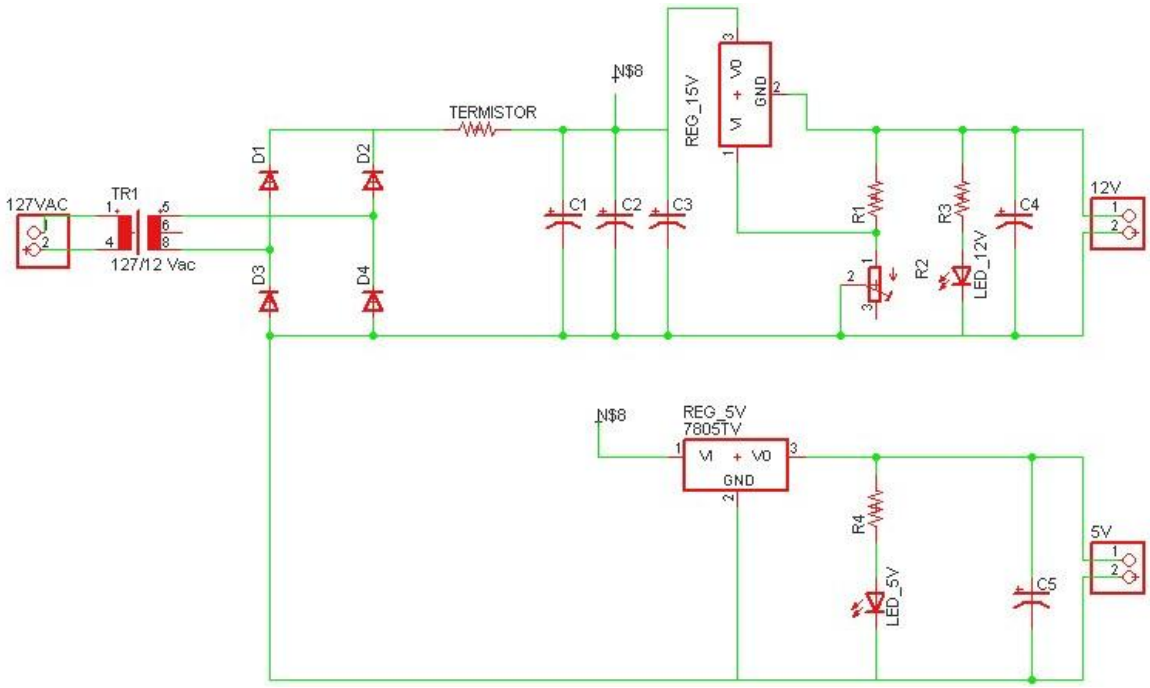


Figura 30 – Circuito esquemático fonte de alimentação
Fonte: Autoria Própria.

3.2 SISTEMA DE LUMINOSIDADE

Em um microambiente automatizado, os principais elementos que alteram a intensidade luminosa são o sombreamento e as fontes luminosas. Neste projeto deve-se ter o cuidado em relação à fonte luminosa utilizada, por exemplo, uma lâmpada incandescente pode interferir não só na luminosidade como também na temperatura, sendo assim a utilização de LEDs de alto brilho adequado.

Necessitando apenas acompanhar os valores correntes de luminosidade no microambiente, já que para a variável luminosidade são aplicados fotociclos pré-estabelecidos, optou-se pela utilização do LDR à base de sulfeto de cádmio, NSL19–M51, conforme a figura 3, devido sua fácil acessibilidade, preço, dimensões compactas e compatibilidade com os parâmetros de luminosidade trabalhados.

Para o monitoramento analógico da resposta do LDR foi utilizado um divisor de tensão formado por um resistor no valor de 10kΩ para que as tensões de saída estivessem de acordo com as especificações desejadas na entrada analógica A0 do microcontrolador, conforme ilustrado na figura 31.

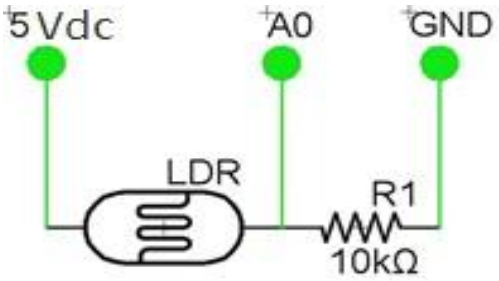


Figura 31 – Circuito LDR com divisor de tensão
Fonte: Autoria Própria.

Logo o valor de tensão a ser lido pelo conversor A/D do Arduino Mega2560 ($V_{(A0)}$) é dado pela relação entre a tensão de alimentação 5Vdc com o valor de resistência do LDR (R_{LDR}), conforme a equação (6).

$$V_{(A0)} = \frac{10*10^3}{10*10^3+R_{LDR}} \quad (6)$$

Em um microambiente, os principais elementos que alteram a intensidade luminosa são o sombreamento e as fontes luminosas. Neste projeto deve-se ter o cuidado em relação à fonte luminosa utilizada, por exemplo, uma lâmpada incandescente pode interferir não só na luminosidade como também na temperatura, sendo assim a utilização de LEDs alto brilho adequada. Para o chaveamento dos LEDs através da saída PWM Pin 9 do Arduino, necessitou-se utilizar um transistor NPN qual suportasse uma corrente no coletor (I_c) de no mínimo 120mA, pois cada um suporta uma corrente máxima de 30mA. Logo usaram-se resistores no valor de 220Ω para limitar a corrente dos LEDs e um transistor BC337 o qual possui uma corrente de coletor máxima de até 800mA. A figura 32 ilustra o circuito.

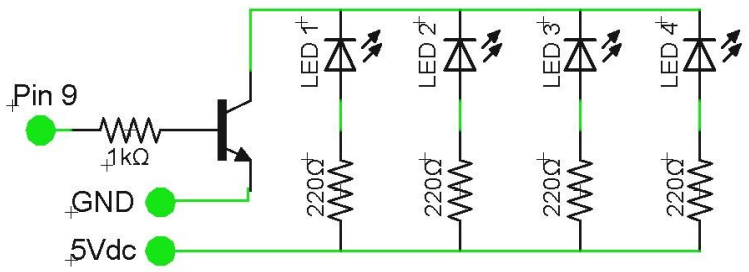


Figura 32 – Circuito LEDs com transistor BC337.
Fonte: Autoria Própria.

O sistema de luminosidade deverá reproduzir os fotociclos de luminosidade simulando os ciclos diários de dia e noite. A estratégia no controle de luminosidade contempla três tipos distintos de fotociclos, onde primeiro fotociclo (Fotociclo 1) simula condições de dia/noite de um dia completo de 24 horas. O segundo fotociclo (Fotociclo 2), tem-se dois ciclos dia/noite no período de 24 horas, ou seja, a cada 12 horas o ciclo se repete. No terceiro (Fotociclo 3) tem-se três ciclos no período de 24 horas, onde a cada 8 horas o fotociclo inicia novamente. Em resumo o período do primeiro fotociclo é de 24 horas, do segundo de 12 horas e do terceiro de 8 horas. As figuras 33, 34 e 35 apresentam o acontecimento desses fotociclos em função do tempo.

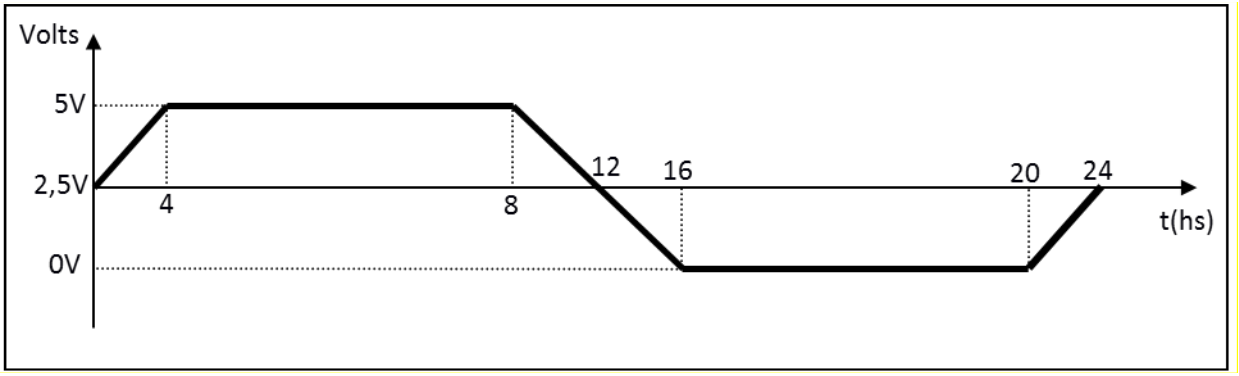


Figura 33 – Fotociclo 1
Fonte: Autoria Própria.

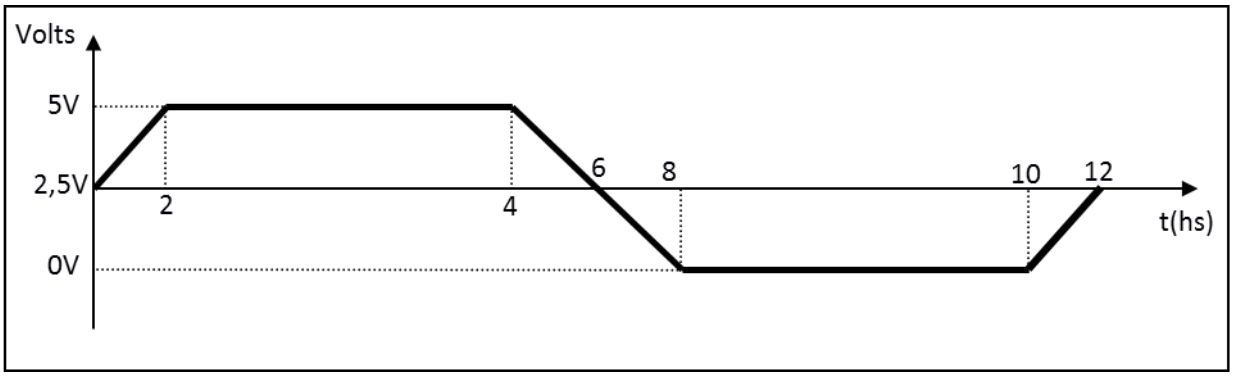


Figura 34 – Fotociclo 2
Fonte: Autoria Própria.

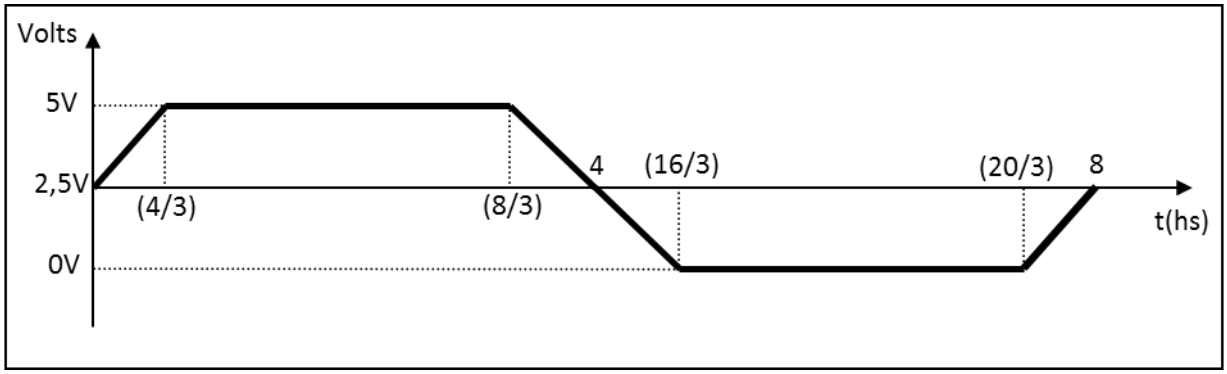


Figura 35 – Fotociclo 3
Fonte: Autoria Própria.

Busca-se simular um sombreamento próximo do natural para diminuir a intensidade luminosa do microambiente de forma gradual, chegando assim o mais próximo de uma condição natural de luminosidade ao decorrer do dia. Visando a não mudança de estado claro e escuro de forma abrupta, foi programado o controle visando o incremento e decremento de tensão nos LEDs em 0,625Vdc, utilizando para isso as saídas PWM do Arduino Mega2560, em função do tempo total do fotociclo. As equações de (7) até (11) relacionam o valor de tensão ($V_{(fotociclo\ 1)}$) sobre o led em função do tempo (t) para o fotociclo 1.

$$V_{(fotociclo\ 1)} = 0{,}625 * t + 2{,}5 \qquad \text{para } 0 < t \leq 4 \qquad (7)$$

$$V_{(fotociclo\ 1)} = 5 \qquad \text{para } 4 < t \leq 8 \qquad (8)$$

$$V_{(fotociclo\ 1)} = -0{,}625 * t + 10 \qquad \text{para } 8 < t \leq 16 \qquad (9)$$

$$V_{(fotociclo\ 1)} = 0 \qquad \text{para } 16 < t \leq 20 \qquad (10)$$

$$V_{(fotociclo\ 1)} = 0{,}625 * t - 12{,}5 \qquad \text{para } 20 < t \leq 24 \qquad (11)$$

As equações de (12) até (16) relacionam o valor de tensão ($V_{(fotociclo\ 2)}$) sobre os LEDs de alto brilho em função do tempo (t) para o fotociclo 2.

$$V_{(fotociclo\ 2)} = 1{,}25 * t + 2{,}5 \qquad para\ \ 0 < t \leq 2 \qquad (12)$$

$$V_{(fotociclo\ 2)} = 5 \qquad para\ \ 2 < t \leq 4 \qquad (13)$$

$$V_{(fotociclo\ 2)} = -1{,}25 * t + 10 \qquad para\ \ 4 < t \leq 8 \qquad (14)$$

$$V_{(fotociclo\ 2)} = 0 \qquad para\ \ 8 < t \leq 10 \qquad (15)$$

$$V_{(fotociclo\ 2)} = 1{,}25 * t - 12{,}5 \qquad para\ \ 10 < t \leq 12 \qquad (16)$$

As equações de 16 até 20 relacionam o valor de tensão ($V_{(fotociclo\ 3)}$) sobre os LEDs de alto brilho em função do tempo (t) para o fotociclo 3.

$$V_{(fotociclo\ 3)} = 1{,}875 * t + 2{,}5 \qquad para\ \ 0 < t \leq 1{,}33 \qquad (17)$$

$$V_{(fotociclo\ 3)} = 5 \qquad para\ \ 1{,}33 < t \leq 2{,}66 \qquad (18)$$

$$V_{(fotociclo\ 3)} = -1{,}875 * t + 10 \qquad para\ \ 2{,}66 < t \leq 5{,}33 \qquad (19)$$

$$V_{(fotociclo\ 3)} = 0 \qquad para\ \ 5{,}33 < t \leq 6{,}66 \qquad (20)$$

$$V_{(fotociclo\ 3)} = 1{,}875 * t - 12{,}5 \qquad para\ \ 6{,}66 < t \leq 8 \qquad (21)$$

3.3 SISTEMA DE UMIDADE

Outro aspecto importante a ser analisado e detalhado dentro do microambiente é o controle sobre a umidade do ar. Para medições de umidade relativa do ar atestou-se que o HR202 não era suficiente para as medições aferidas no projeto, pois apresentava uma baixa precisão, além do mesmo apresentar uma capacidade de reter umidade em seu elemento sensor, colocando assim em questionamento a confiabilidade das medições.

O HIH-3610 por sua vez apresenta uma ótima precisão e um tempo de resposta condizente com os intervalos de medições do projeto, porém o mesmo apresenta um custo elevado e difícil acessibilidade. Sendo assim a escolha do sensor confluiu para o AM2302, dentre os sensores analisados o AM2302 possui uma maior facilidade de operação, além da capacidade de medições de temperatura com um baixo custo.

Para garantir uma tensão de nível lógico estável durante a comunicação do AM2302 com o Arduino, será necessário colocar um resistor de *pull-up* no valor de 10kΩ ligado entre a entrada de alimentação 5Vdc e a saída de comunicação SDA do sensor, na qual é conectada a entrada digital Pin 8 do Arduino, conforme a figura 36.

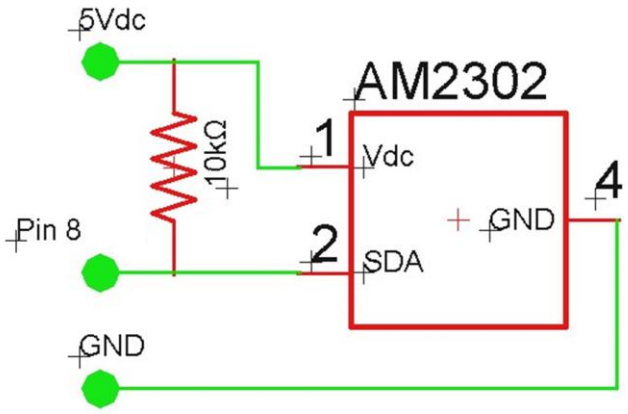


Figura 36 – AM2302 com resistor de *pull-up*
Fonte: Autoria Própria.

No microambiente são utilizados os processos de Umidificação com Vapor e Arrefecimento com Desumidificação, citados na seção 2.3.4, para umidificação e desumidificação, respectivamente. Para o método de Arrefecimento com Desumidificação é utilizado um ventilador de convecção, fotografia 4, baseado em

um motor de indução CA, que pode ser alimentado tanto em 110Vac quanto em 220Vac, sendo sua velocidade de rotação de 1500RPM.

Fotografia 4 – Ventilador de convecção
Fonte: Autoria Própria

Em relação a seu controle, ele é realizado pelo relé (K2), representado na Fotografia 2, do módulo SRD-05VDC-SL-C, descrito na seção 2.4.3.

Com o propósito de elevar a umidade relativa do ar é utilizado um umidificador ultrassônico de uso doméstico, o mesmo alocado no microambiente é representado na figura 37, sendo seu controle realizado pelo rele (K4), o qual também é apresentado na Fotografia 2, do módulo SRD-05VDC-SL-C, descrito na seção 2.4.3.

Figura 37 – Umidificador alocado no microambiente
Fonte: Autoria Própria.

Baseado na atuação de um umidificador e de um ventilador de convecção para o parâmetro de umidade relativa do ar deve manter sua estabilidade e controle seguindo a leitura do sensor AM2302 e a resposta do sistema conforme a figura 38.

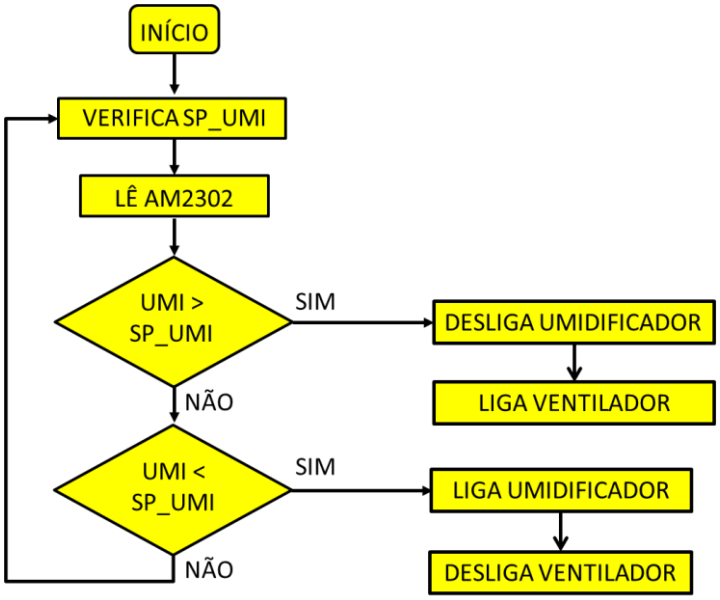


Figura 38 – Fluxograma do sistema de umidade
Fonte: Autoria Própria

Os intervalos entre as medições são de 2 minutos, pois antes disso não há nenhuma mudança significativa de umidade no microambiente, sendo assim desnecessário programar limites de histerese para o controle.

3.4 SISTEMA DE TEMPERATURA

A estratégia do controle de temperatura baseia-se na comparação entre o valor de temperatura corrente com seu o valor de *set-point* (SP_TEMP), conforme a figura 39, onde caso a temperatura estiver maior que o set-point ajustado o *Driver* de acionamento comandado por um sinal do Arduino desligará a resistência, caso contrário, a resistência permacecerá ligada em 50% do valor da tensão eficaz .

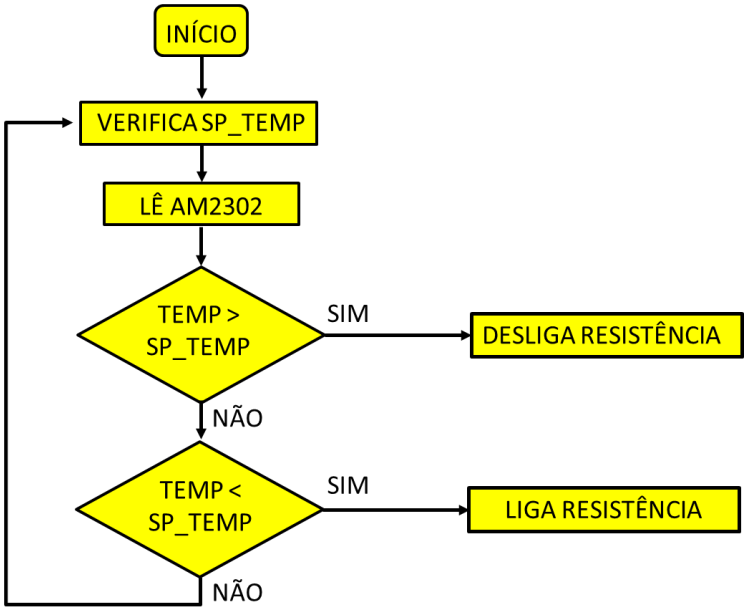
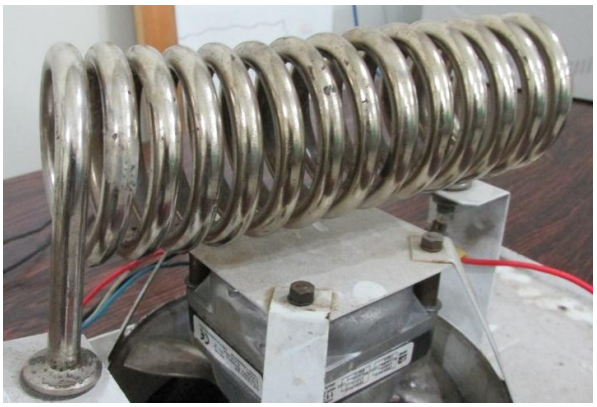


Figura 39 – Fluxograma do sistema de temperatura
Fonte: Autoria Própria

Igualmente no sistema de umidade, os intervalos entre as medições são de 2 minutos, onde antes disso não há nenhuma mudança significativa de temperatura no microambiente, sendo assim desnecessário programar limites de histerese para o controle.

Para as medições de temperatura o LM35 mostrou-se eficiente, porém apresentava falhas em ambientes com umidade, já o PT100 agregava um alto custo, e dentre o termopar e o AM2302, o último tem uma maior facilidade de operação e possui a capacidade de realizar medições de umidade como descrito na seção anterior.

Na fotografia 5 pode-se ver o modelo de resistência de aquecimento a ser utilizada. Esta possui potência nominal de 240W e deve ser alimentada com uma tensão alternada de 110Vac, suportando uma corrente elétrica máxima de 1,09A.



Fotografia 5 - Resistência de Aquecimento
Fonte: Autoria Própria.

Para mantê-la em 50% da tensão eficaz da rede, é necessário modular essa tensão por meio de um circuito eletrônico o qual foi denominado de Driver de Acionamento. Neste projeto essa tensão é modulada por meio do disparo de um triac BTA12, onde a referência para o disparo é dada através de um circuito denominado Detector de Zero, que por meio de uma interrupção externa informa ao ATmega2560 o ponto da rede referente a 1,3V. Com essa informação o microcontrolador através de uma parte do código no *firmware,* calcula qual instante será o próximo ponto zero e assim é possível efetuar o disparo para o valor de tensão correspondente. O trecho de programa na figura 40 corresponde a inicialização da interrupção externa do ATmega2560.

```
 8  void setup()
 9  {
10    pinMode(loadR, OUTPUT);
11    attachInterrupt(0, zero_crosss_int, RISING);
12  }
```

Figura 40 – Inicialização da interrupção e função zero_crosss_int
Fonte: Autoria Própria.

A variável loadR corresponde a saída digital 12 para o disparo do triac, zero_crosss_int, mostrado na figura 41, é a função a ser chamada todo momento que a entrada digital 2 (interrupção 0) passar o estado 0 para 1 (*RISING*).

```
17  void zero_crosss_int()
18  {
19    // Cálculo do ângulo de disparo: 60Hz-> 8.33ms (1/2 ciclo)
20    int powertime = (32*(256-power));
21    delayMicroseconds(powertime);
22    digitalWrite(loadR, HIGH);
23    delayMicroseconds(8.33);
24    digitalWrite(loadR, LOW);
25  }
```

Figura 41 – Função zero_cross_int
Fonte: Autoria Própria.

Através do parâmetro *power*, que corresponde ao nível de tensão eficaz qual se almeja atingir é calculado *powertime* que é o tempo no qual deve-se aguardar para que seja dado o disparo, passado o intervalo *powertime*, que esta em torno de poucos microssegundos, é dado o comando HIGH para o disparo do triac através da saída *loadR*.

O circuito Detector de Zero, figura 42, é formado basicamente por uma ponte de diodos 4N4004, os quais suportam uma tensão de até 400V, ligados à um transformador com saída de 12Vac e um foto acoplador 4N25.

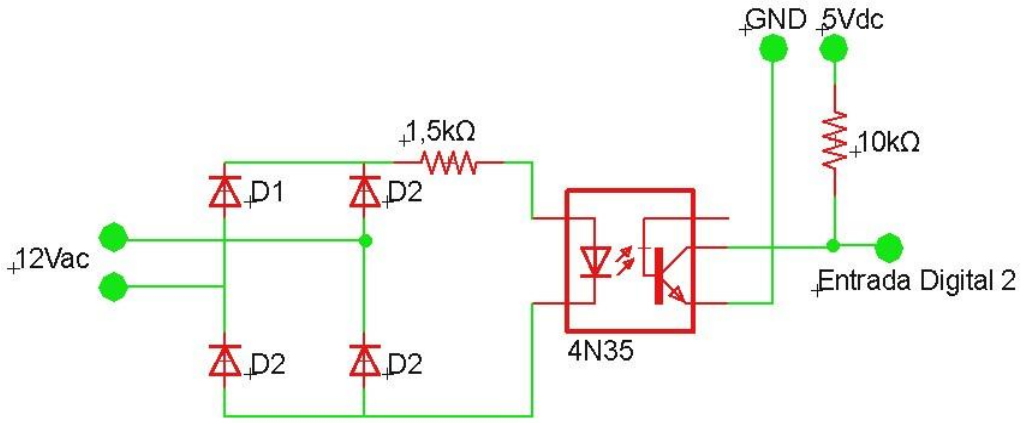


Figura 12 – Circuito Detector de Zero
Fonte: Autoria Própria.

Sabendo que a corrente máxima suportada pelo LED interno do 4N25 é de 60mA, é utilizado um resistor de 1,5kΩ para limitar sua corrente, sendo também necessário um resistor de *pull-up* no valor de 10kΩ no coletor do fototransistor interno do 4N25.

O Detector de Zero funciona basicamente da seguinte forma: Toda vez que a tensão entre os terminais do LED interno do 4N25 chega a aproximadamente 1,3Vdc, o LED passa a conduzir e assim emitir luz. Incidindo essa luz sobre o fototransistor, o mesmo passa a conduzir entre os terminais do coletor e emissor e assim envia um sinal de 5Vdc a entrada digital 2 do Arduino.

O circuito *Driver* de Acionamento, responsável por manter a tensão eficaz da rede em 50% do seu valor sobre a Resistência de Aquecimento, é composto basicamente de um fotoaclopador MOC3021 e um triac BTA12, conforme o circuito apresentado na figura 43.

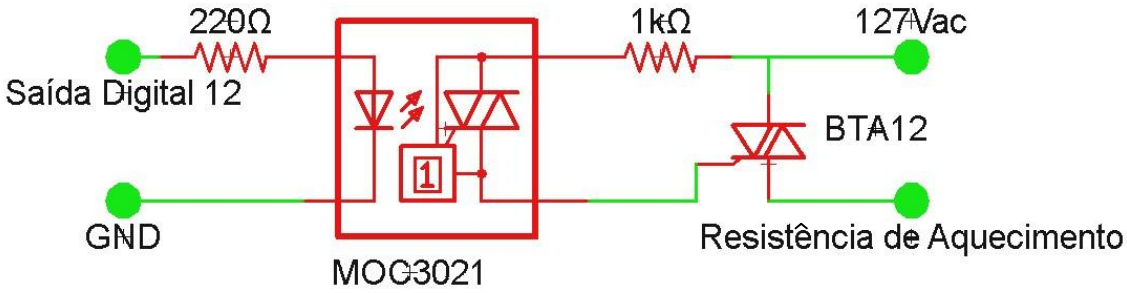


Figura 43 – Circuito *Driver* de Acionamento
Fonte: Autoria Própria.

O LED interno do MOC3021 suporta uma corrente de até 50mA, logo torna-se necessário de um resistor de no mínimo 100Ω para limitar a corrente, foi usado um resistor de 220Ω o qual limita a corrente a 22mA. Quanto ao fototriac, o mesmo suporta uma corrente de até 1,2A e tensão de 400V, o resistor de 1kΩ limita essa corrente para 179mA.

A figura 44, obtida através de simulação computacional, apresenta as formas de onda do pulso de disparo do triac BTA12 em azul, da tensão de entrada em amarelo, juntamente com a tensão retificada em 85%, representada na cor rosa.

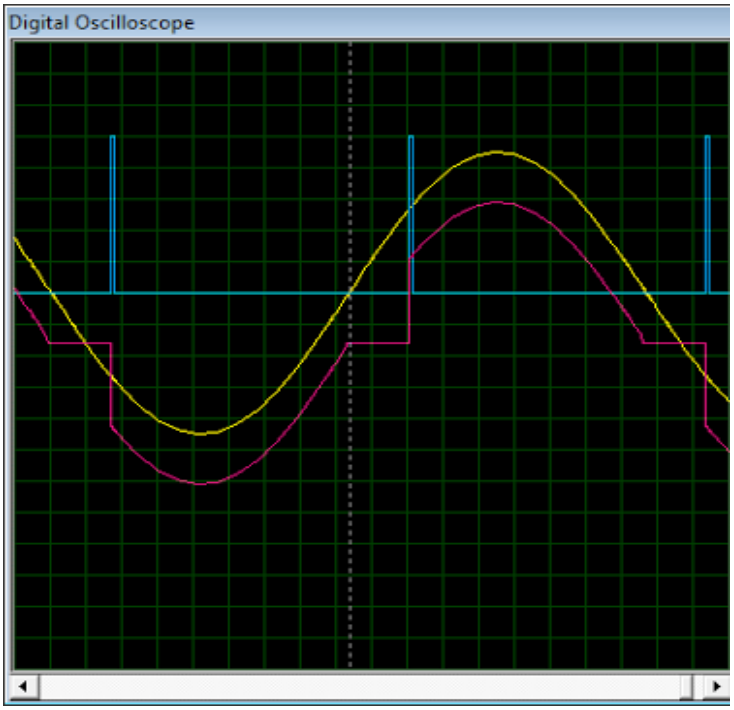


Figura 44 – Simulação formas de onda do circuito Driver de Acionamento
Fonte: Autoria Própria.

4 METODOLOGIA

Avaliando-se a forma de abordagem deste trabalho, afirma-se que este é quantitativo, já que muitas mensurações foram feitas auxiliando no acerto de decisões. Observando os objetivos, classifica-se a pesquisa como descritiva, visto que a mesma se caracteriza pela descrição das características de determinados contextos, bem como o estabelecimento de relações entre as variáveis. Quanto aos procedimentos técnicos, embasa-se em uma pesquisa bibliográfica devido a ser elaborada a partir de materiais já publicados, e experimental, pois se busca controlar uma grandeza de estudo, manipulando/ajustando as variáveis que são capazes de influenciá-la.

A primeira etapa de desenvolvimento tratou da análise do microambiente, estudando o comportamento das variáveis que interferem no sistema. O segundo passo foi o projeto da automatização do microambiente juntamente com o método de controle a ser implementado, entrando aqui o estudo da melhor alocação de sensores e atuadores dentro do microambiente, e também o aprofundamento nos estudos relativos à adequação dos componentes eletrônicos e a *firmware* a serem empregados. Posteriormente, realizou-se a construção prática do sistema, incluindo aqui a confecção da placa eletrônica para controle, bem como a colocação de atuadores e sensores dentro do microambiente para validar a eficácia do projeto, sendo o passo final a análise dos resultados obtidos.

5 RESULTADOS

5.1 CARACTERIZAÇÃO DO SENSOR DE LUMINOSIDADE

O método para levantamento da resistência do LDR pela luminosidade foi obtido através do auxílo de um aplicativo para Android, figura 45, visto a indisponibilidade de um luxímetro profissional apropriado para os testes, salientando que o mesmo não há opção de ser calibrado, e que para esse caso o foco principal é a correlação de amanhecer, dia, crepúsculo e noite apenas.

Figura 45 - Luximetro Ouro Lux para Android
Fonte: Autoria Própria.

A caracterização consistiu basicamente em fazer medições de luminosidade utilizando o sensor LDR posicionado a uma distância perpendicular de 1 metro da fonte luminosa. Utilizando um Arduino para caputura dos dados, realizaram-se 20 medições consecutivas em um intervalo de 1 minuto cada. A figura 46 e a figura 47 ilustram o caracterização em questão.

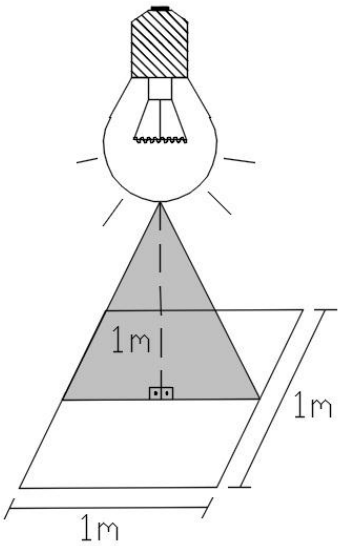


Figura 46 – Esquema para medição de luminosidade
Fonte: Autoria Própria.

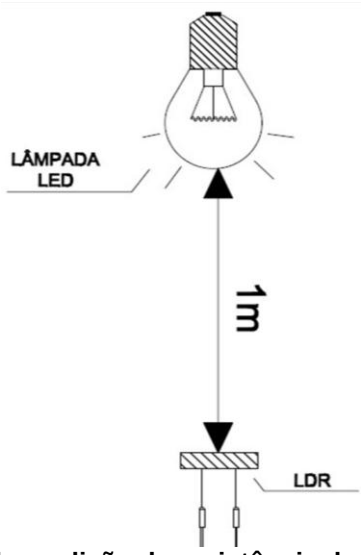


Figura 47 – Esquema de medição de resistência do LDR por luminosidade
Fonte: Autoria Própria.

Os resultados práticos apontaram uma resistência nos terminais do LDR de 232Ω para um valor médio de 1127lux para a uma lâmpada LED com valor nominal de no máximo 1650lux, e um valor de aproximadamente 1MΩ quando a lâmpada estava desligada. A figura 48 apresenta o gráfico que relaciona a resistência do LDR com o nível de luminosidade. Para comparação, valores indicados no *datasheet* do componente são representados pela reta na cor preta, e valores experimentais relacionados com a reta em cor vermelha.

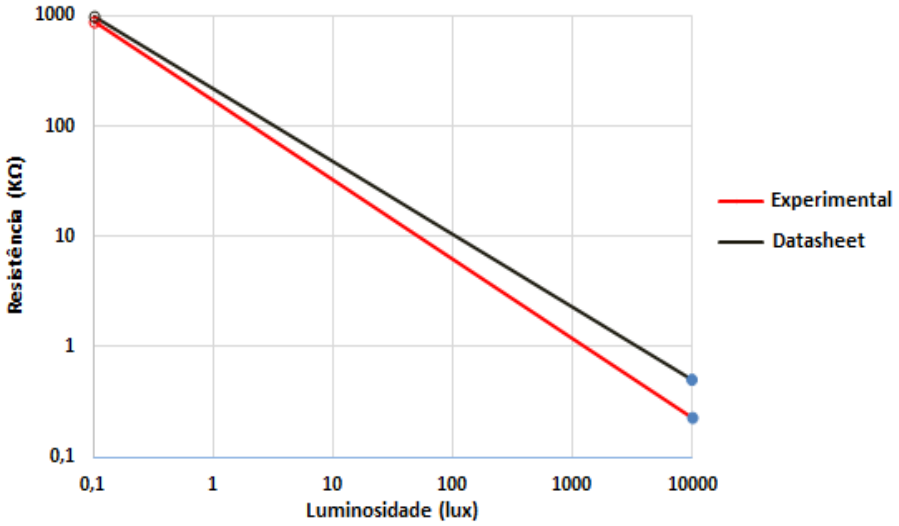


Figura 48 – Relação Resistência do LDR X Luminosidade
Fonte: Autoria Própria.

5.2 CARACTERIZAÇÃO DO SISTEMA DE TEMPERATURA

Testes realizados comprovaram a homogeneidade da temperatura em diferentes pontos do microambiente em um mesmo instante de tempo. A figura 49 apresenta a disposição dos sensores no microambiente.

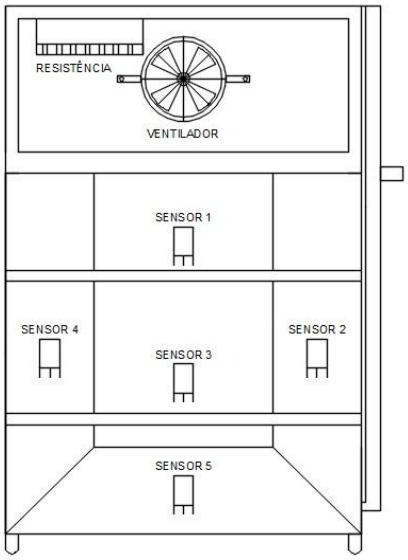


Figura 49 – Posicionamento de sensores de temperatura
Fonte: Autoria Própria.

Buscou uma distribuição de forma estratégica, almejando fazer aferições em todos os pontos chave do microambiente. A figura 50 elaborada a partir dos valores coletados dos mesmos cinco sensores em diferentes pontos corrobora a homogeneidade da temperatura no microambiente com controle aplicado.

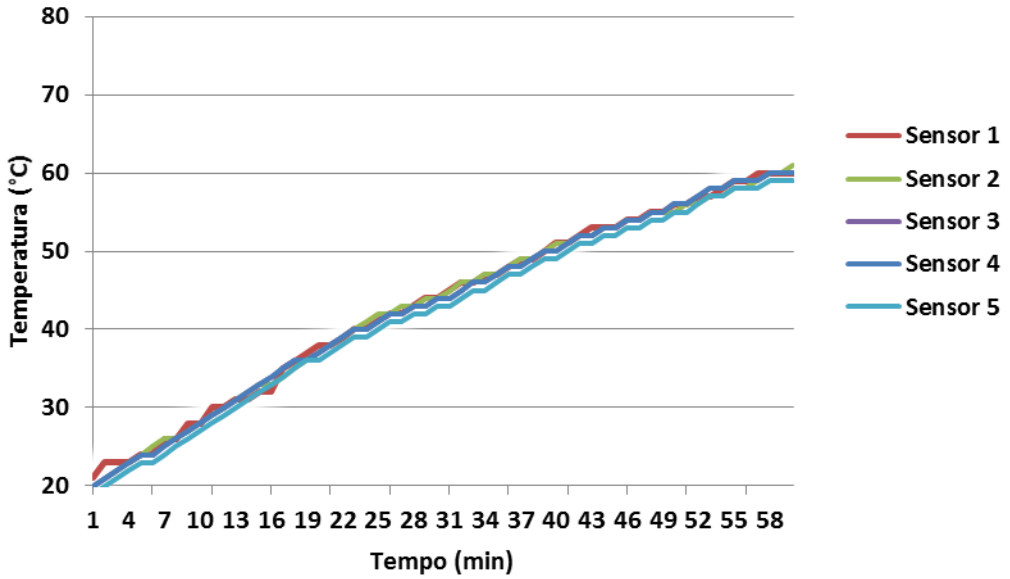


Figura 50 - Teste de homogeneidade de temperatura
Fonte: Autoria Própria.

Posteriormente, realizou-se também a avaliação do resfriamento não forçado do microambiente, onde aos 60 minutos do teste desligou-se a resistência de aquecimento conforme a figura 51.

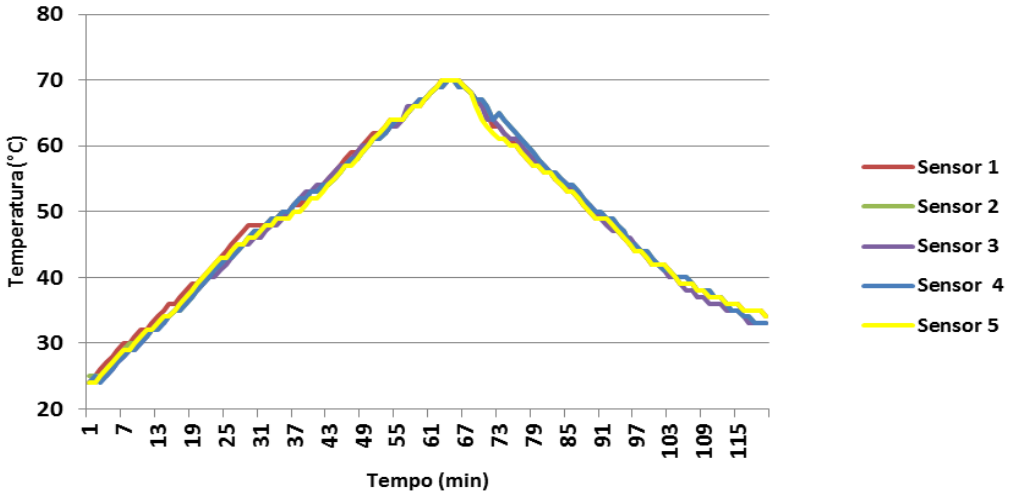


Figura 51 – Gráfico resfriamento e aquecimento em relação ao Tempo
Fonte: Autoria Própria.

Ao que se pode notar, tanto para aumento quanto para decremento da temperatura interna do microambiente, trabalha-se com duas funções praticamente lineares em relação ao tempo conforme o gráfico na figura 51.

A validação nos testes de temperatura foi dada com referência de um *set-point* de 30°C, onde os testes realizados apresentam o gráfico da figura 52.

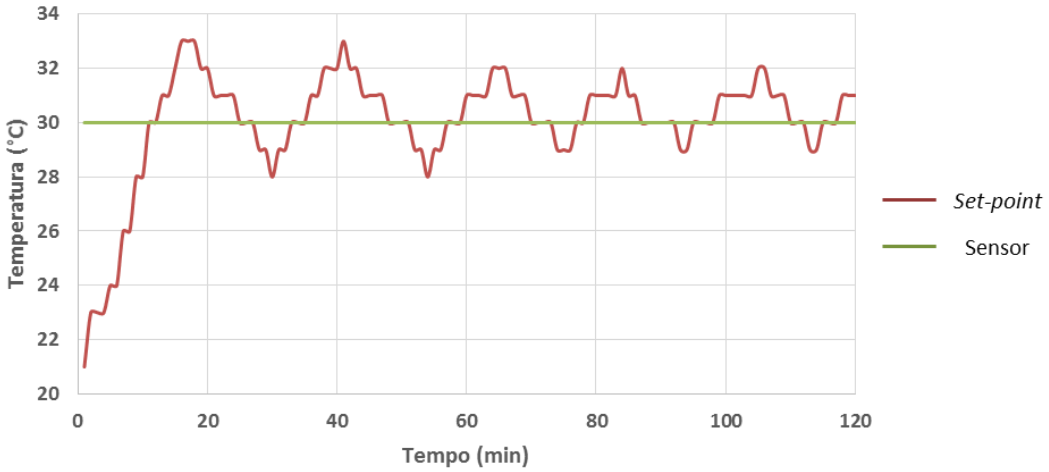


Figura 52 – Validação do sistema de temperatura
Fonte: Autoria Própria.

5.3 CARACTERIZAÇÃO DO SISTEMA DE UMIDADE RELATIVA DO AR

Testes realizados comprovaram a homogeneidade da umidade relativa do ar em diferentes pontos do microambiente, sendo a alocação dos sensores a mesma utilizada nos testes de temperatura. O teste consistiu na coleta de dados durante o tempo de 30 minutos e a figura 53 apresenta o gráfico correspondente aos valores coletados dos cinco sensores em relação ao tempo.

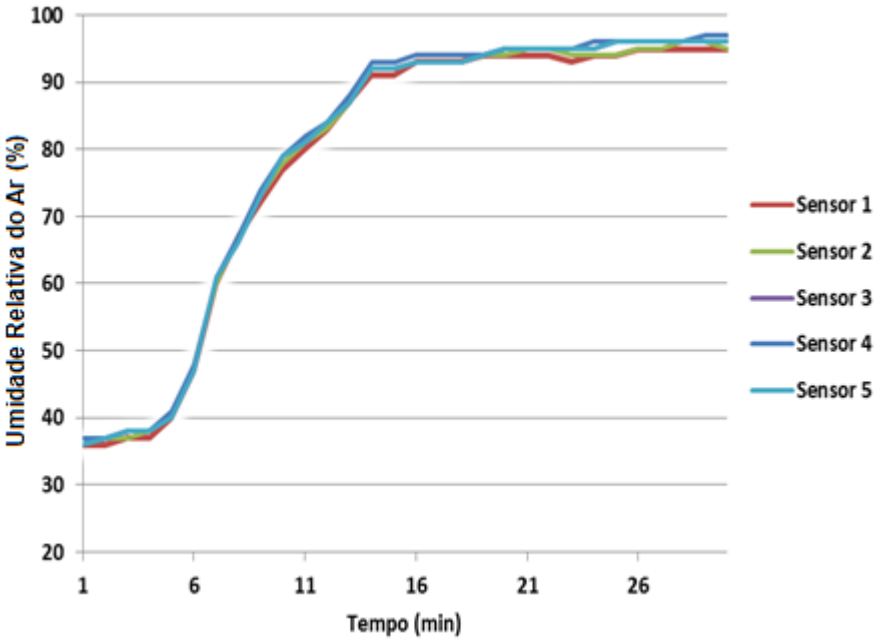


Figura 53 – Gráfico de Homogeneidade de umidade
Fonte: Autoria Própria.

Posteriormente, realizou-se também a avaliação da desumidificação forçada pelo ventilador de convecção do microambiente, sendo o mesmo ligado aos 25 minutos do teste, com o simultâneo desligamento do umidificador. A figura 54 apresenta o gráfico da situação exposta acima.

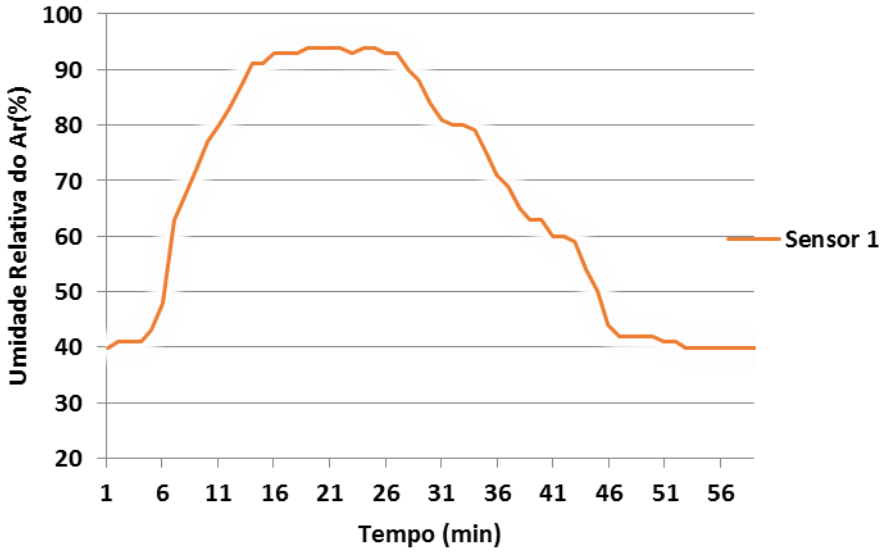


Figura 54 - Desumidificação em relação ao Tempo
Fonte: Autoria Própria.

Realizado um teste para verificar a resposta do sistema quanto ao *set-point* e a histerese, configurou-se o mesmo para operar na faixa de 85% de umidade relativa do ar, sendo que o controle on/off estimado para o controle do umidificador funcionou de forma coerente, e pode ser visto graficamente na figura 55.

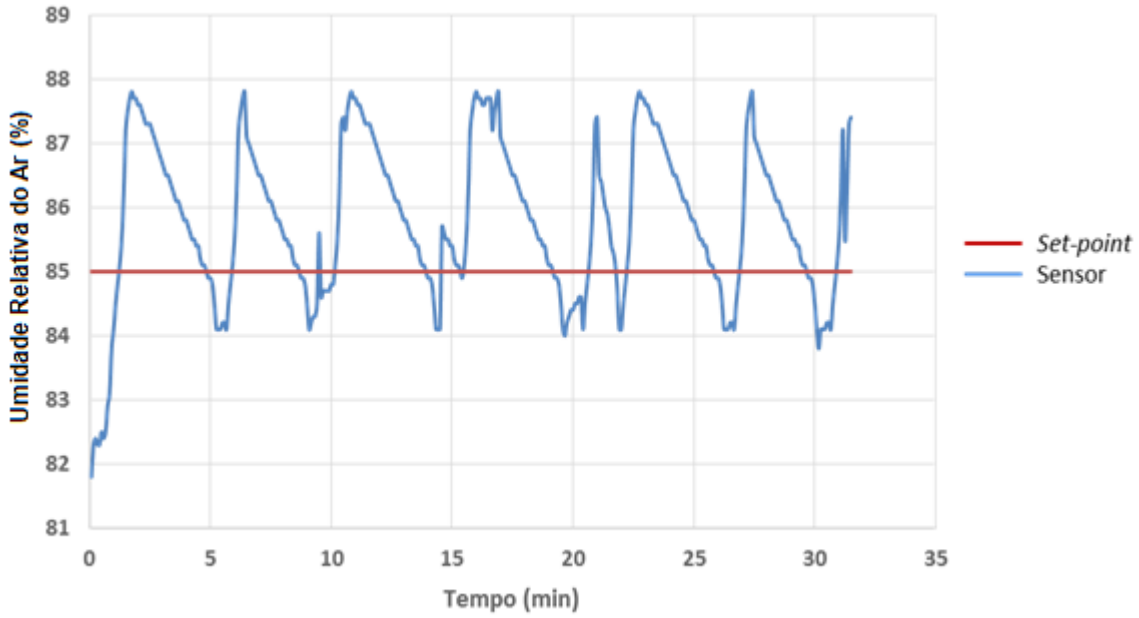


Figura 55 – Validação do sistema de umidade
Fonte: Autoria Própria.

5.4 CONFECÇÃO DAS PCI'S

Uma vez projetada a caixa de acomodação das placas eletrônicas, houve-se a necessidade da criação de alguns circuitos para o interfaceamento entre Arduino, sensores, saídas para atuadores e fonte de alimentação. Diante disso, prepararam-se placas em circuito impresso para acomodar os componentes eletrônicos. Primeiramente se iria provar os subsistemas de uma maneira discreta em uma protoboard, porém durante os trabalhos de pesquisa descobriram-se vários obstáculos e intempéries que existem ao utilizar uma protoboard, podendo-se destacar os ruídos, confusão que vários fios poderiam causar e o fato de que ao ser instalada na estrutura do microambiente surgiriam problemas. Com base nesse fato, elaborou-se em um *software* de projetos para circuito impresso, placas eletrônicas para acomodar os componentes.

5.4.1 Placa de Interfaceamento

A figura 56 apresenta o projeto do circuito responsável pelo interfaceamento entre sensores, teclado matricial, display LCD e Arduino, juntamente com a fotografia 6, a qual apresenta sua forma física.

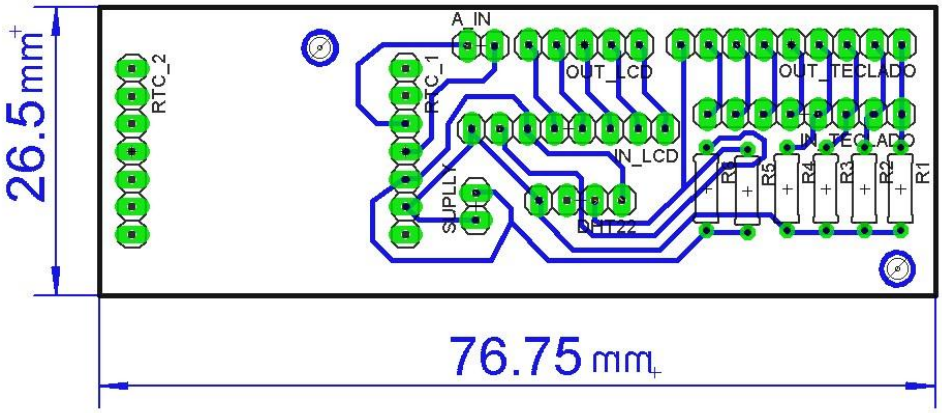
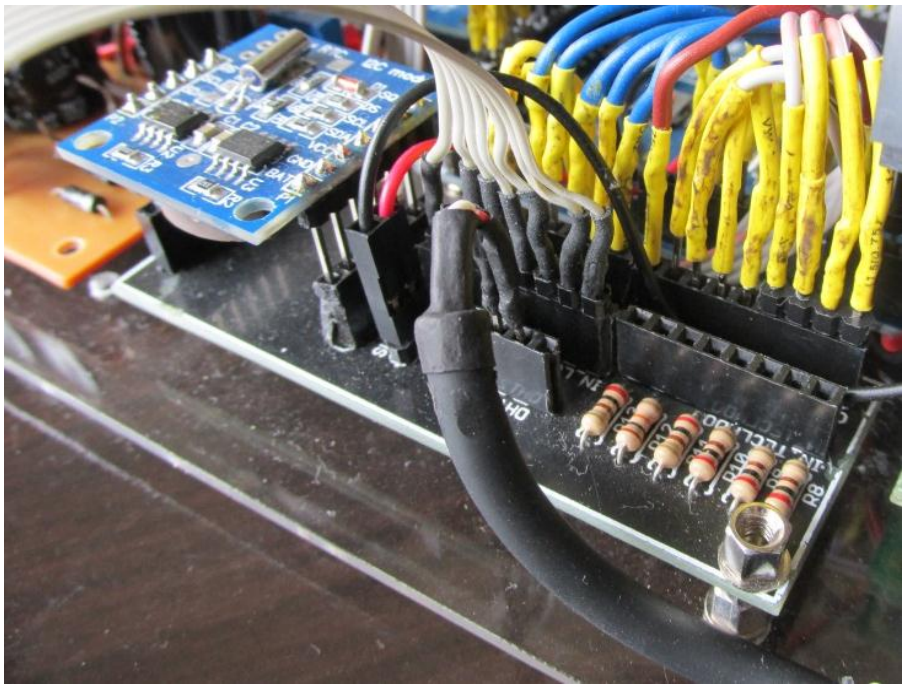


Figura 56 – Projeto PCI para Placa de Interfaceamento
Fonte: Autoria Própria.



Fotografia 6 – Placa de Interfaceamento
Fonte: Autoria Própria.

5.4.2 Fonte de alimentação externa

A figura 57 apresenta o projeto do circuito da fonte de alimentação que é responsável em fornecer tensão em nível de corrente contínua para sensores, teclado matricial, display LCD, Arduino e outros periféricos, e a fotografia 7 apresenta a forma final dessa fonte.

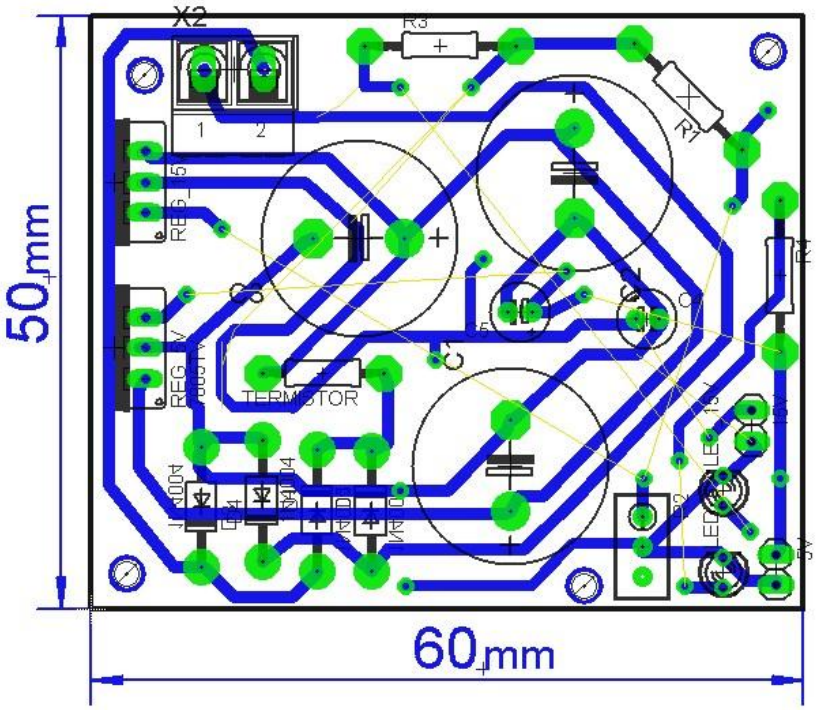
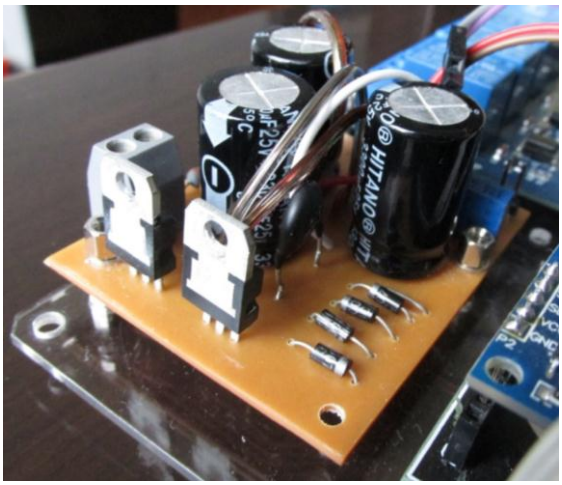


Figura 57 – Projeto PCI para Fonte de Alimentação
Fonte: Autoria Própria.



Fotografia 7 – Fonte de Alimentação 5Vdc e 10,6Vdc
Fonte: Autoria Própria.

5.4.3 Circuito de detecção de zero

A figura 58 abaixo apresenta o *layout* da placa de circuito impresso para o Detector de Zero, e a fotografia 8 seu layout físico.

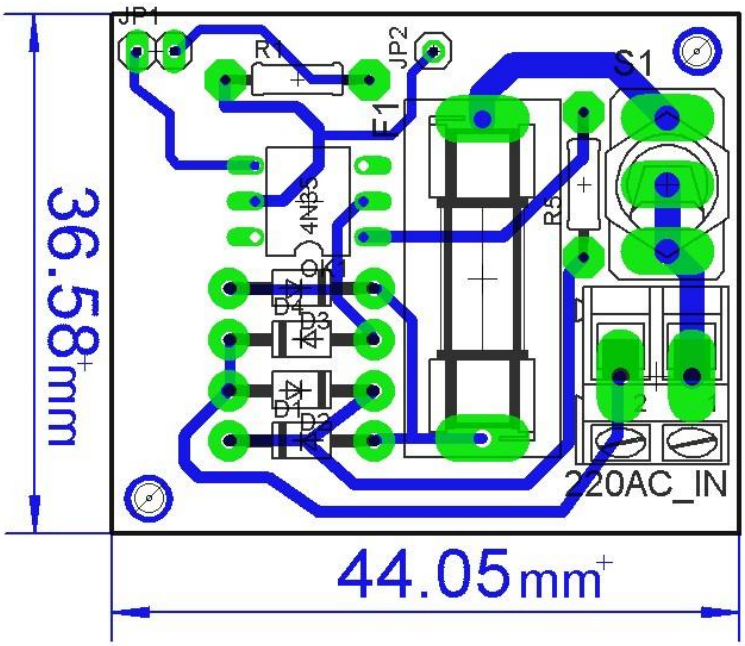
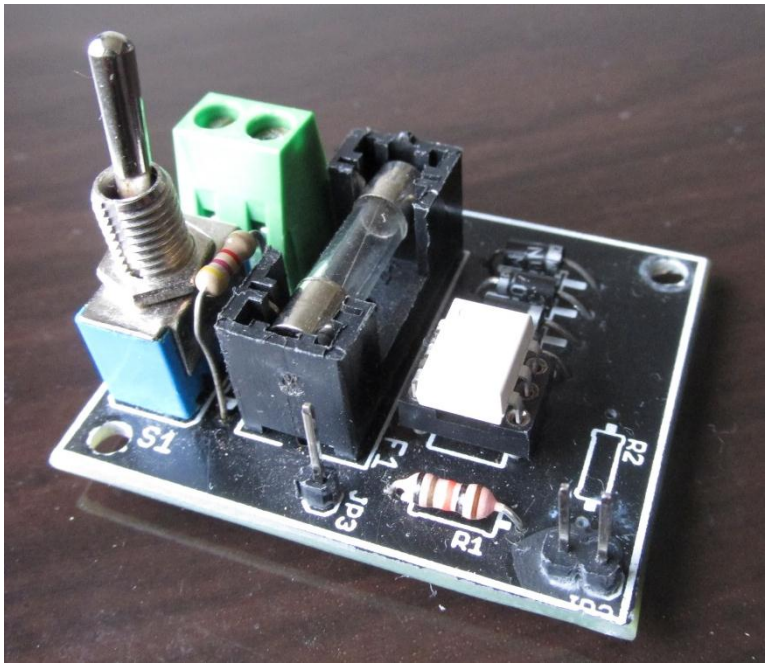


Figura 58 – Projeto PCI do Detector de Zero
Fonte: Autoria Própria.



Fotografia 8 – Circuito Detector de Zero
Fonte: Autoria Própria.

5.4.4 Circuito driver de acionamento

A Figura 59 abaixo apresenta o *layout* do projeto do circuito de Driver de acionamento, o qual é responsável por modular a potência entregue ao resistor de aquecimento e os LED alto brilho, e posteriormente, a fotografia 9 apresenta o driver na prática.

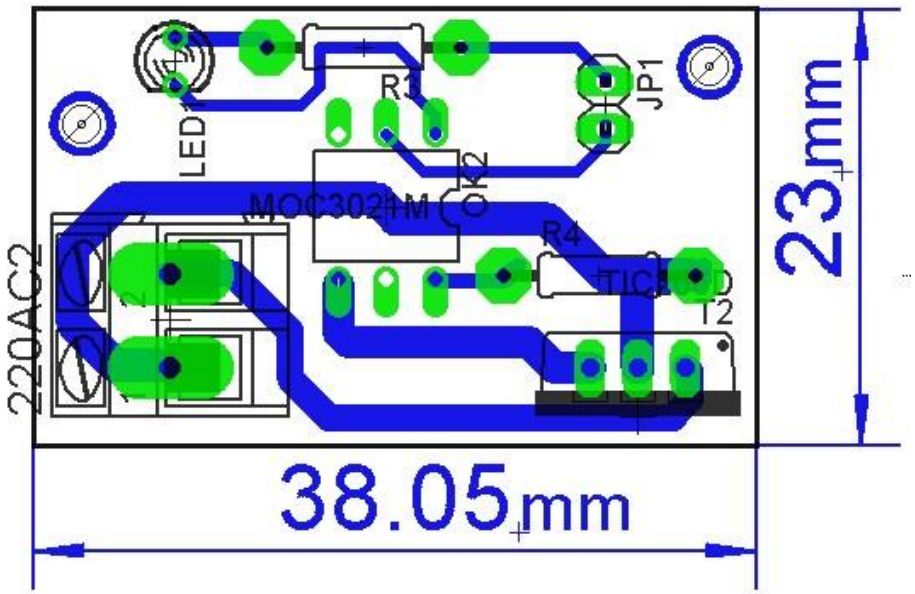
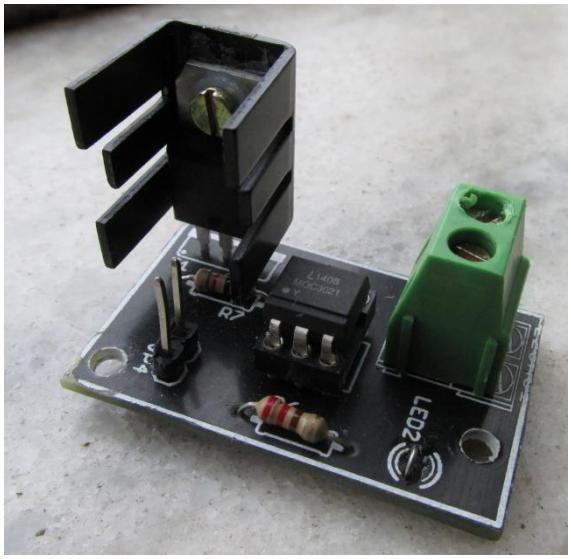


Figura 59 – Projeto circuito Driver de Acionamento
Fonte: Autoria Própria.



Fotografia 9 – Circuito Driver de Acionamento
Fonte: Autoria Própria.

As formas de onda sobre a resistência de aquecimento, em 25% e 50% de tensão aplicada, seguem respectivamente nas figuras 60 e 61.

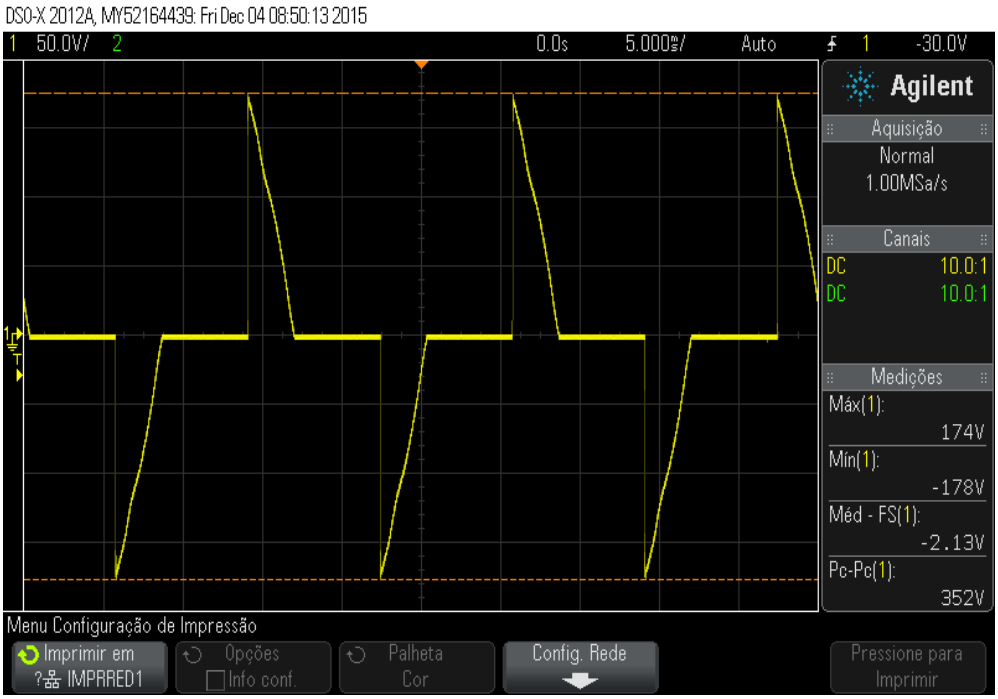


Figura 60 – Forma de onda na saída do circuito Driver de Acionamento – 25%
Fonte: Autoria Própria.

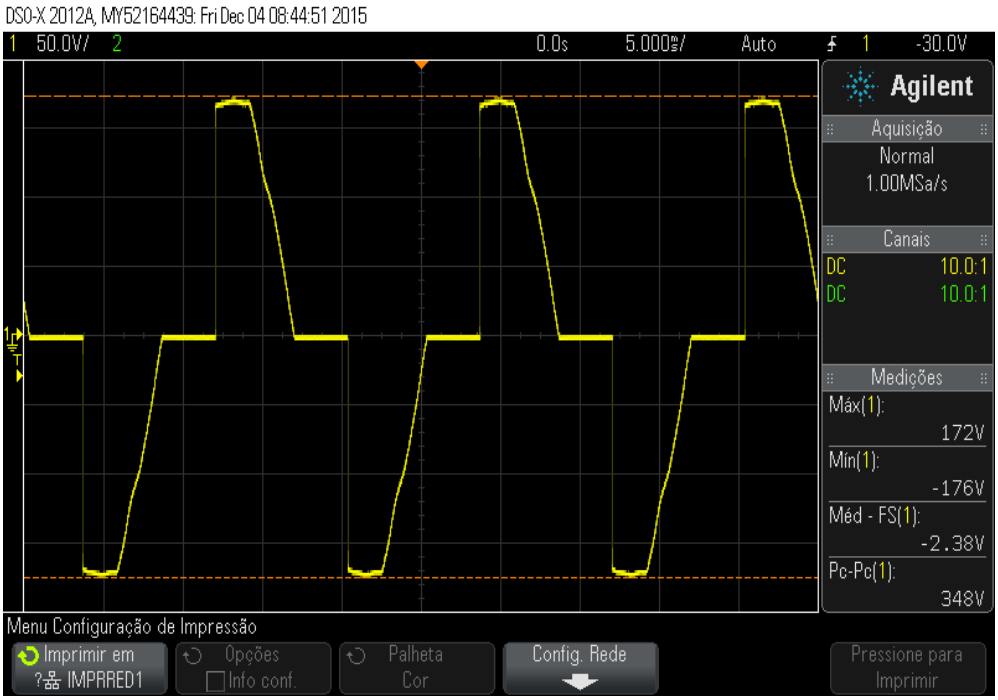
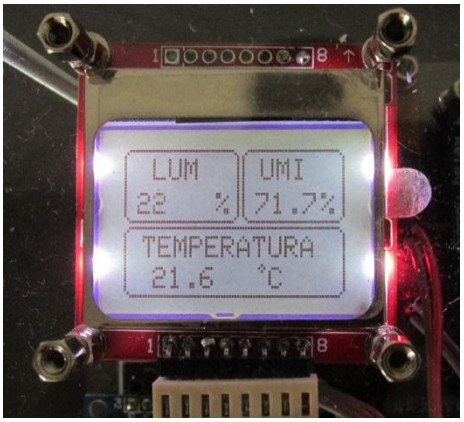


Figura 61 – Forma de onda na saída do circuito Driver de Acionamento – 50%
Fonte: Autoria Própria.

5.5 IHM

Para facilitar a utilização da placa pelo usuário é apresentada a seguir a configuração quanto às telas de navegação, set-points, forma de entrada de dados referentes ao módulo de controle desenvolvido.

Ao ser conectado na rede elétrica 127Vac, a primeira tela apresenta a leitura das variáveis (temperatura, luminosidade, umidade) onde pode-se monitorar a condição das mesmas em tempo real, observando-se possíveis variações. Conforme fotografia 10 apresenta-se a tela de monitoramento em questão.

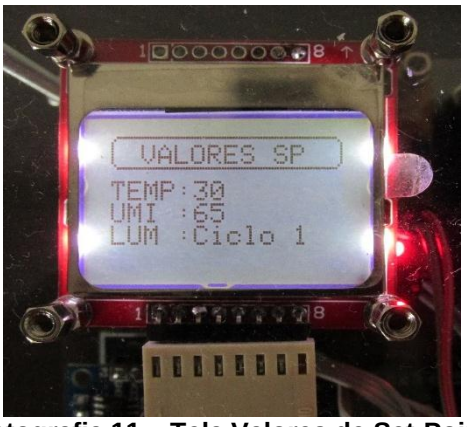


Fotografia 10 – Tela de monitoramento das variáveis
Fonte: Autoria Própria.

O valor de luminosidade é apresentado em quantidade de lux, umidade relativa do ar em porcentagem e temperatura em graus Celsius.

Alterações nos valores de alguma das três variáveis, em muitos dos casos, serão justificadas pela referência que as conduz, ou seja, o valor de set-point que estará operando-as.

A tela citada acima permanecerá visível até o operador pressionar a tecla “#”, onde uma sub-rotina detectará essa ação e far-se-á o programa migrar para a seguinte tela, onde os valores de *set-points* poderão ser visualizados, ilustrativamente temos a situação descrita na fotografia 11.



Fotografia 11 – Tela Valores de Set-Point
Fonte: Autoria Própria.

O sistema de luminosidade, como mencionado no escopo do trabalho, será operado em três ciclos distintos, sendo que o último selecionado é apresentado nessa mesma tela de *set-point.*

Ressaltando que o sistema tem uma determinada flexibilidade e que a faixa de operação para temperatura, umidade e luminosidade está dentro de limites que podem ser suportados pelo mesmo. O Quadro 7 apresentada os valores limites.

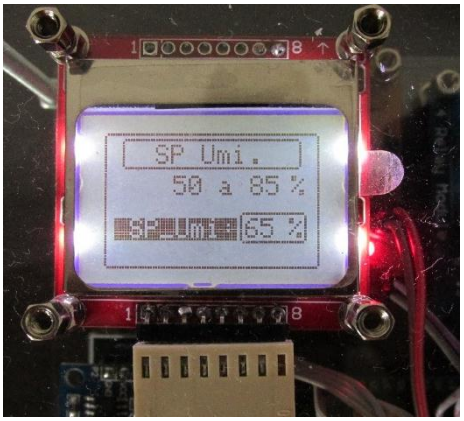
Variável	Mínimo	Máximo	Unidade
Temperatura	20	40	°C
Umidade	50	85	%
Luminosidade	0	100	%

Quadro 7 – Valores máximos e mínimos das variáveis
Fonte: Autoria Própria

Os valores de luminosidade variam nessa faixa atendendo aos fotociclos estabelecidos.

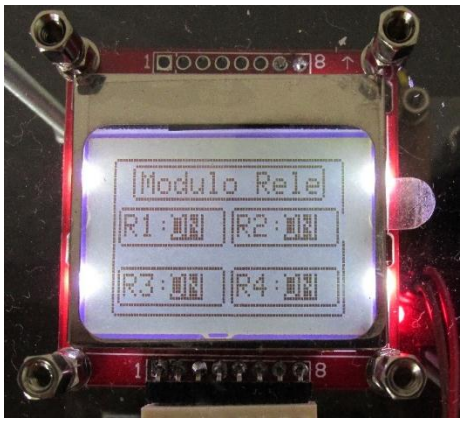
Novamente a tela citada acima permanecerá visível até o operador pressionar a tecla “#”, que encaminha o usuário até a tela de definição dos valores de *set-points*. Para definição desses valores deverão ser utilizadas as teclas especiais “A” e “B” do teclado matricial, sendo a primeira para incremento e segunda para decremento no valor do *set-point* o qual deseja-se alteração. Para confirmação da entrada de dados, a tecla “#” deve ser pressionada, encaminhando o operador para definição da variável seguinte. Caso queira redefinir alguma variável já

atribuída, a tecla “*” pressionada retornará a definição da variável anterior. A fotografia 12 ilustra a tela para alteração do *set-point* relativo a variável de umidade.



Fotografia 12 – Tela para alteração do set-point de umidade
Fonte: Autoria Própria.

Definidos todos os valores de *set-points* a tecla “#” deverá ser pressionada, e uma seguinte tela apresenta ao usuário o estado atual da saída dos relés, sendo saída ativa em *ON* e quando desabilitada em *OFF*. A Fotografia 13 expõe o descrito acima.



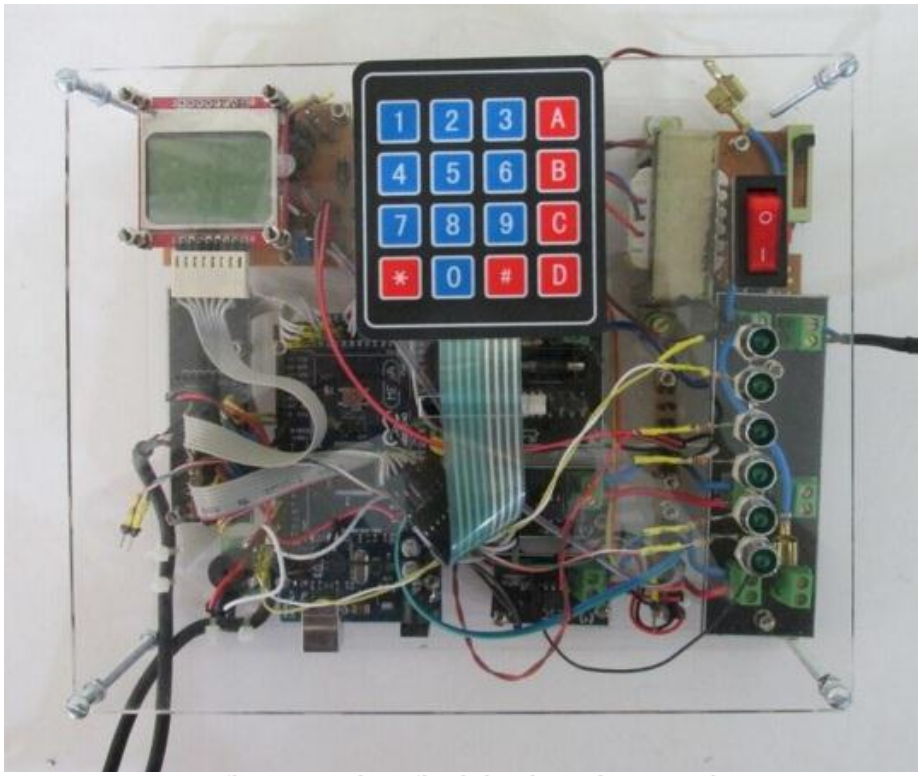
Fotografia 13 - Tela de acompanhamento das saídas dos reles
Fonte: Autoria Própria

Pressionando-se novamente a tecla “#” o usuário será encaminhado novamente à tela inicial, onde aparecem os valores correntes de temperatura, umidade e luminosidade. As teclas C e D não tem funcionalidade no projeto.

5.6 PROJETO FINAL

O aspecto físico final do projeto de controle desenvolvido é ilustrado na fotografia 14, onde todas as placas desenvolvidas, bem como fonte e transformador, interface homem máquina, os LEDs de indicação de estados e o microcontrolador são agrupados, já na fotografia 15 é ilustrada a forma final da placa de controle desenvolvida.

Fotografia 14 – Projeto final
Fonte: Autoria Própria.



Fotografia 15– Projeto final da placa de controle
Fonte: Autoria Própria.

5.7 MELHORIAS FUTURAS

Alguns pontos podem ser listados para que este projeto seja melhorado e expandido:

- Pesquisar métodos para minimizar o consumo energético do projeto e aperfeiçoar seu rendimento;
- Substituir o Arduino por outro microcontrolador que tenha uma relação custo benefício superior a este projeto em especial;
- Melhorias no código do programa com o intuito de torná-lo mais fluido e flexível a mudanças;
- Permitir ao usuário acesso aos dados do microambiente de maneira remota;
- Reparar compressor de resfriamento para aproveitamento no controle de temperatura;
- Reparar amassados na estrutura do microambiente garantindo assim uma maior isolação térmica;
- Devido aos diversos problemas que surgiram em decorrência da fonte Vcc, fazer levantamento e projeto de uma fonte que melhor se adeque ao projeto no sentido de supri-lo em tensão e corrente;
- Realizar estudos de viabilidade econômica do projeto para casos de produção em maiores quantidades.
- Desenvolvimento de placa dedicada sem o uso do Arduino;
- Corrigir problema de leitura do teclado matricial.

6 CONCLUSÃO

De um princípio o qual tinha como pilar motivador o desenvolvimento de um microambiente destinado à criação culturas biológicas, muito se estudou acerca das condições que reunissem em um ponto quiescente o ambiente ideal para criação e desenvolvimento das mesmas. Referindo-se a essa idealidade, três variáveis necessitavam determinada sintonia, sendo temperatura, umidade relativa do ar e luminosidade dentro do microambiente. Não se atendo apenas a essa idealidade, um adendo do projeto permitiria ao operador atribuir os valores dessas variáveis da forma que assim desejasse, aumentando o range de aplicabilidades do microambiente, sendo essa uma questão que englobaria o escopo do projeto.

Objetivando o controle das variáveis citadas acima, através de sensores, foram-se feitas as aquisições de valores correntes das mesmas, as quais eram interpretadas pelo programa e confrontadas com o valor que se estava determinado no *set point*, ou seja, a forma como o sistema estava pré-definido a funcionar. Em consequência dessas confrontações, o sistema poderia acionar ou não atuadores, os quais seriam agentes modificadores diretos das condições que se esperavam trabalhar. Um exemplo aleatório se dá quando necessitávamos aumentar a temperatura interna do microambiente, onde a resistência de aquecimento era acionada, e permanecia ligada em determinada potência até atingirmos o ponto onde queríamos chegar. Vale ressaltar que os testes em relação ao comportamento dessas variáveis foram efetuados, visando observar como elas se comportariam ao longo do tempo.

Foi desenvolvida uma interface do sistema (IHM) utilizando um teclado matricial e um visor LCD, utilizado para reconfiguração ou simplesmente consulta dos sensores.

Os resultados finais, advindos de testes, simulações e aferições, se apresentaram de forma satisfatória, onde o objetivo geral almejado foi alcançado, e o controle das variáveis se tornou viabilizado. Problemas surgiram em decorrência. Desta maneira, o vigente projeto apresentou-se de notável valor para o enriquecimento curricular dos envolvidos, os quais julgam ponderado frisar a possibilidade de aprimoramento do trabalho, partindo para o alojamento físico de

sensores, atuados, módulo de controle e outros no microambiente como também um melhor estudo sobre eles na aplicação presente.

Por fim, ressalta-se a necessidade de evolução do projeto, melhorando o menu da interface IHM, bem como uma avaliação contínua do controle em função das características de utilização do microambiente.

REFERÊNCIAS

AHO, Alfred V.; ULLMAN, Jeffrey D.; COMPILADORES. **Compiladores:** Princípios, Técnicas e Ferramentas. 2. ed. São Paulo: Pearson, 2008.

ALLEN, Phil E.; HOLBERG, Doug R.. **CMOS Analog Circuit Design.** New York: Oxford University Press, 2012.
ALMEIDA, José Luiz Antunes de. **Dispositivos semicondutores**: Tiristores. 11º ed. São Paulo: Editora Érica, 2007.

ARAGÃO FILHO, Luiz Augusto C. Moniz de. **http://aquarius.ime.eb.br/~moniz/instrumentacao/instrument_02b.pdf.** Rio de Janeiro: Video, 2013. 7 slides, color. Disponível em: <http://aquarius.ime.eb.br/~moniz/instrumentacao/instrument_02b.pdf>. Acesso em: 18 nov. 2015.

ARAUJO, Fagner. **Microcontroladores PIC e Linguagem C:** O mundo dos Microcontroladores. Disponível em: <http://pt.slideshare.net/chelltonalmeida/capitulo-1-33711773>. Acesso em: 18 nov. 2015.

ARDUINO. **What is Arduino?** Disponível em: <https://www.arduino.cc/en/Guide/Introduction>. Acesso em: 20 nov. 2015.

ATMEL. **Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V.** Disponível em: <http://www.atmel.com/pt/br/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf>. Acesso em: 27 nov. 2015.

AUSONG ELECTRONICS. **Digital-output relative humidity & temperature sensor/module AM2303:** Description. Disponível em: <https://www.adafruit.com/datasheets/DHT22.pdf>. Acesso em: 20 nov. 2015.

BAKER, Jacob. **CMOS:** Circuit Design, Layout, and Simulation. 3. ed. New Jersey: IEE Press, 2010.

BALBINOT, A.; BRUSAMARELLO, V. J. **Instrumentação e Fundamentos de Medidas** Volume 1, 1ª ed, LTC, Rio de. Janeiro, 477p, 2006.

BARBI, I.; **Eletrônica de Potência**. 3. ed. Florianópolis: Ed. do Autor, 2000.

BASCONCELLO FILHO, Daniel Oliveira. **Curso de Arduino:** Hardware. Disponível em: <http://www.robotizando.com.br/curso_arduino_hardware_pg1.php>. Acesso em: 18 nov. 2015.

BRAGA, Newton. **Comando de Reles:** Rele. 2012. Disponível em: <http://www.newtoncbraga.com.br/index.php/artigos/54-dicas/5086-art673.html>. Acesso em: 28 nov. 2015.

BRAIN, Marshall; NICE, Karim. **O interior de um umidificador.** 2010. Disponível em: <casa.hsw.uol.com.br/umidificadores3.htm>. Acesso em: 26 nov. 2015.

BRANCO, Andre; **Ligas de Altas Resistivida.** Disponível em: <http://www.eletrica.ufpr.br/piazza/materiais/AndreBranco.pdf>. Acesso em: 26 nov. 2015.

CARTOLANO JÚNIOR, Etienne Américo; AVEGLIANO, Priscila Barreira. **Microcontroladores PIC:** Família 16F84A. Disponível em: <http://www.pcs.usp.br/~jkinoshi/2005/PCS2031_projeto01_Introducao_PIC_v4.doc> . Acesso em: 26 nov. 2015.

CASSARLY, W. J. **High-Brightness LEDs**. Optics and Photonics News, v. 19, ed. 1, p. 18-23, jan. 2008.

CGE: Centro de Gerenciamento de Emergências. **Umidade Relativa do Ar**, 2012. Disponível em: <http://www.cgesp.org/v3/umidade-relativa-do-ar.jsp> Acesso em 12 de agosto de 2012.

COSTA, Ennio C. **Física Aplicada à Construção.** 4. ed. São Paulo: Edgard Blucher, 1991.

DEWITT, D. P. ; INCROPERA, F. P., **Fundamentos de Transferência de Calor e de Massa**, LTC, 5ª edição, 2003.

DIAS, Marcelo Paschoal. **Avaliação do emprego de um pré-regulador boost de baixa frequência no acionamento de leds de iluminação.** 2012. 80 f. Dissertação (Mestrado) - Curso de Mestrado em Engenharia Elétrica, Universidade Federal de Juiz de Fora, Juiz de Fora, 2012. Disponível em: <http://www.ufjf.br/ppee/files/2012/02/Dissertação_Marcelo-Paschoal-Dias.pdf>. Acesso em: 27 nov. 2015.

FIALHO, Arivelto Bustamente. **Instrumentação Industrial - Conceitos, Aplicações e Análises.** Brasil: Editora Ética, 2002.

FIGUEIREDO, Gilberto. **CASA DA AGRICULTURA: Produção em ambientes protegidos.** São Paulo: CECOR/CATI, v. 2, 2011. Trimestral. Disponível em: <http://www.asbraer.org.br/arquivos/bibl/56-ca-producao.pdf>. Acesso em: 18 nov. 2015.

FRADE, Joao, **Climatização Geral**. Disponível em < http://www.dem.isel.ipl.pt/seccoes/sfrio/VFrade/ClimaGeral.pdf > Acessado em 11 de julho de 2015.

FUENTES, Rodrigo Cardozo. **Apostila de Automação Industrial:** Sensores. 2005. Disponível em: <http://w3.ufsm.br/fuentes/index_arquivos/CA03.pdf>. Acesso em: 26 nov. 2015.

GUEDES, Manuel Vaz. **A lei de Joule.** Disponível em: <http://paginas.fe.up.pt/histel/Lei_Joule.pdf>. Acesso em: 26 nov. 2015.

HAYKIN, Simon. **Sinais e Sistemas**. 3. ed. Porto Alegre: Bookman, 2001.

HIRZEL, Timothy. **Arduino:** PWM. Disponível em: <https://www.arduino.cc/en/Tutorial/PWM>. Acesso em: 27 nov. 2015.

IDOETA, Ivan Valeije; CAPUANO, Francisco Gabriel. **Elementos de eletrônica digital.** 40. ed. São Paulo: Érica, 2008.

JOHNSON, David E.; HILBURN, John L.. **Fundamentos de analise de Circuitos Elétricos.** Rio de Janeiro: Prentice Hall do Brasil, 1994.

KLIMACH, Hamilton; FABRIS, Eric. **Conversores de Sinais Analógicos e Digitais CMOS:** Porto Alegre: Imagem, 2014. Color. Disponível em: <https://chasqueweb.ufrgs.br/~hklimach/CursoCI_Brasil/BA02/MIC78_Intro.pdf>. Acesso em: 27 nov. 2015.

OUROLUX. **Luxímetro Ourolux.** 2015. Disponível em: <http://www.ourolux.com.br/blog/luximetro-ourolux/>. Acesso em: 30 nov. 2015.

PEREIRA, Fábio; **Microcontroladores PIC – Programação em C.** 7ª Ed. São Paulo: Editora Érica Ltda., 2009.

PERTENCE JÚNIOR, Antonio. **Eletrônica analógica:** amplificadores operacionais e filtros ativos: teoria, projetos, aplicações e laboratório. 6. ed. Porto Alegre: Bookman, 2003.

PETRUZELLA, Frank D. **Controladores Lógicos Programáveis.** 4. ed. Porto Alegre: Bookman, 2013.

PINHEIRO, Cleusa. **CASA DA AGRICULTURA: Produção em ambientes protegidos.** São Paulo: CECOR/CATI, v. 2, 2011. Trimestral. Disponível em: <http://www.asbraer.org.br/arquivos/bibl/56-ca-producao.pdf>. Acesso em: 18 nov. 2015.

PLOUGHMAN, Terry. **Understanding Orchids**, 2007. Disponível em: <http://www.the-wow-collection.com/software/orchids.pdf> Acesso em: 27 out. 2015.

MALVINO, Albert. **Eletrônica**. Volume 2. 7ª ed. São Paulo: McGraw-Hill, 2007.

MARTINS, Geomar Machado. **Princípios de Automação Industrial.** 2012. Disponível em: <http://coral.ufsm.br/desp/geomar/automacao/Apostil a_032012.pdf>. Acesso em: 27 nov. 2015.

MELLO, Hilton.A. **Dispositivos semicondutores**. 3º ed. Rio de Janeiro: Livros Técnicos e Científicos, 1978.

MONK, Simon. **Programação Com Arduino:** Começando Com Sketches. Porto Alegre: Bookman Companhia Editora Ltda, 2013. 160 p.

NATIONAL INSTRUMENTS. **Conceitos básicos da amostragem analógica:** Aliasing e filtros antialiasing. Disponível em: <http://www.ni.com/white-paper/3016/pt/>. Acesso em: 27 nov. 2015.

RIBEIRO, Marco Antônio. **Instrumentação Industrial**. 9. ed. Salvador: Tek Treinamento & Consultoria Ltda, 2002.

RODRIGUES, C. R. B. S.; ALMEIDA, P. S.; SOARES, G. M.; JORGE, J. M.; PINTO, D. P.; BRAGA, H. A. C. **Experimental characterization regarding two types of phosphor-converted white high-brightness LEDs: Low power and high power devices**. Brazilian Power Electronics Conference (COBEP), p. 734-740, 2011.

RS COMPONENTS. **Datasheet - Light dependent resistors: NSL19 - M51**. Disponível em: <http://www.biltek.tubitak.gov.tr/gelisim/elektronik/dosyalar/40/LDR_NSL19_M51.pdf >. Acesso em: 26 nov. 2015.

SAAVEDRA, Alexis Barbieri; TOLEDO, Leonardo Felipe de; VITIELLO, Vinícius Rafael. **Projeto de um sistema de automação e monitoramento residencial via web.** 2013. 62 f. Monografia (Especialização) - Curso de Engenharia Elétrica, Universidade São Francisco, Itatiba, 2013.

SANTOS, Cláudia dos; KAWAKITA, Kazuto. **Métodos de medição de umidade em gases**. In: FEIRA E CONGRESSO DE AR CONDICIONADO, REFRIGERAÇÃO, AQUECIMENTO E VENTILAÇÃO INDUSTRIAL, 98., 1992, São Paulo. **Anais... .** São Paulo: Instituto de Pesquisas Tecnológicas, 1998. p. 56 - 67.

SILVA, Jesue Graciliano da. **Introdução à tecnologia da refrigeração e da Climatização.** 1. ed. São Paulo: Artliber Editora, 2003.

SILVA, Waldemar. **Cultivo de orquídeas no Brasil**. 6. ed. São Paulo: Nobel, 1986.

SOARES, Antônio Wallace Antunes; ARAUJO, Leonardo Vale de. **Conversor Analógico-Digital por aproximações sucessivas.** Disponível em: <http://www.dee.ufrn.br/~luciano/arquivos/ins_ele/Apresenta%E7%E3o%202010.1/Leonardo_Antonio/Relat_instr_elet.pdf >. Acesso em: 19 nov. 2015.

SOLER & PALAU, S.A. **A climatização de estufas**, 2006. Disponível em:<http://www.solerpalau.pt/formacion_01_39.html> Acesso em: 17 out. 2015.
SOUZA, David José de; **Desbravando o PIC.** 8ª ed. São Paulo: Editora Érica Ltda., 2005.

THOMAZINI, Daniel; ALBUQUERQUE, Pedro Urbano Braga; **Sensores Industriais-Fundamentos e Aplicações**. 4ª ed. São Paulo: Editora Érica, 2007.

TIPLER, Paul A.; MOSCA, Gene. **Física para cientistas e engenheiros:** Mecânica, Oscilações e Ondas, Termodinâmica. 6. ed. v. 1. Rio de Janeiro: LTC, 2010.

TIPLER, Paul A.; MOSCA, Gene. **Física para cientistas e engenheiros:** Mecânica quântica, relatividade e a estrutura da matéria. 5. ed. v. 3. Rio de Janeiro: LTC, 2006.

TOCCI, Ronald J.; WIDMER, Neal S.; MOSS, Gregory L..**Sistemas Digitais:** Princípios e Aplicações. 10ª ed. São Paulo: Pearson, 2007. 830 p.

APÊNDICE A - Código *Firmware*

```
// Inicialização bibliotecas utilizada
#include <SPI.h>
#include <Adafruit_GFX.h> //Biblioteca do display 5110
#include <Adafruit_PCD8544.h> //Biblioteca do display 5110
#include <TimerOne.h>  //Biblioteca com as funçoes do timer 1
#include <DHT.h>    //Biblioteca AM2302
#include "Dimmer.h"

#define DHTPIN_1 8 //Entrada digital 8 p/ AM2302 1
#define DHTPIN_2 13 //Entrada digital 13 p/ AM2302 2
#define DHTTYPE DHT22  //Sensor AM2302(DHT22)

char tecla; //variavel que guarda o caracter da o teclado matricial
float valorSPTemp = 30 , valorSPTemp_f = 30; //Set-point p/ temperatura
float valorSPUmi = 85 , valorSPUmi_f = 85; //Set-point p/ umidade


unsigned int i_minutes = 0, i_hours = 0;  //variavel de contagem de tempo (min)
float i_seconds = 0; //variavel de contagem de tempo (s)

int cont, x, y; //variveis de relaçao linhaxcoluna do teclado matricial
int cont_cd = 1; //variavel do tipo de fotociclo
int cont_cdf = 1; //variavel do tipo de fotociclo
int cont_ciclos = 0; //variavel contadora de fotociclos


String stringState; //string do estado de saida dos reles

String title, range, unit, change; //String de valores de temp, umi e lum

 int porta_rele1 = 40; // saida rele 1
 int porta_rele2 = 41; // saida rele 2


/* Abaixo segue as variaveis referentes a leitura dos sensores AM2302
  Salienta-se que há uma para sensor, e as variaveis tipo boll
  que auxiliam no calculo de média  */
float valorSensorUmi_1, valorSensorTemp_1, valorSensorUmi_2,  valorSensorTemp_2 ;
float valorSensorUmi, valorSensorTemp;
bool  med_1, med_2;

int valorSensorLum;  //variavel de leitura do sensor  LDR
bool on_maq_estados = true; //variavel de controle da maquina de estados

//Cria objeto Adafruit_PCD8544(CLK, DIN, DC, CE, RST);
```

```
Adafruit_PCD8544 display = Adafruit_PCD8544(3, 4, 5, 6, 7);
DHT dht_1(DHTPIN_1, DHTTYPE); //Cria objeto dht_1 (AM2302 1)
DHT dht_2(DHTPIN_2, DHTTYPE); //Cria objeto dht_1 (AM2302 2)
Dimmer res(11);

////////////////////////////
//      FUNCOES           //
////////////////////////////

/*  A funçao abaixo corresponde a funçao
de leitura do teclado matricial onde
atraves d varredura nas entradas
digitais é possivel saber a tecla
pressionada  */
void KeyboardScan(){
   tecla = ' ';
   for (int ti = 30; ti<34; ti++)
   {
   //Alterna o estado dos pinos das linhas
   digitalWrite(30, LOW);
   digitalWrite(31, LOW);
   digitalWrite(32, LOW);
   digitalWrite(33, LOW);
   digitalWrite(ti, HIGH);

   //Verifica se alguma tecla da coluna 1 foi pressionada
   if (digitalRead(34) == HIGH)
   {
     imprime_linha_coluna(ti-29, 1);
     while(digitalRead(34) == HIGH){}
   }

   //Verifica se alguma tecla da coluna 2 foi pressionada
   if (digitalRead(35) == HIGH)
   {
     imprime_linha_coluna(ti-29, 2);
     while(digitalRead(35) == HIGH){};
   }

   //Verifica se alguma tecla da coluna 3 foi pressionada
   if (digitalRead(36) == HIGH)
   {
     imprime_linha_coluna(ti-29, 3);
     while(digitalRead(36) == HIGH){}
   }

   //Verifica se alguma tecla da coluna 4 foi pressionada
```

```
   if (digitalRead(37) == HIGH)
   {
    imprime_linha_coluna(ti-29, 4);
    while(digitalRead(37) == HIGH){}
   }
  }}

/* Função complementar da KeyboardScan(),
essa função é responsavel por verificar o
valor do caracter correspondente ao botão
pressionado */

void imprime_linha_coluna(int x, int y)
{
   if(x == 1 && y ==1){tecla = '1'; }
   if(x == 2 && y ==1){tecla = '4'; }
   if(x == 3 && y ==1){tecla = '7'; }
   if(x == 4 && y ==1){tecla = '*'; }

   if(x == 1 && y ==2){tecla = '2'; }
   if(x == 2 && y ==2){tecla = '5'; }
   if(x == 3 && y ==2){tecla = '8'; }
   if(x == 4 && y ==2){tecla = '0'; }

   if(x == 1 && y ==3){tecla = '3'; }
   if(x == 2 && y ==3){tecla = '6'; }
   if(x == 3 && y ==3){tecla = '9'; }
   if(x == 4 && y ==3){tecla = '#'; }

   if(x == 1 && y ==4){tecla = 'A'; }
   if(x == 2 && y ==4){tecla = 'B'; }
   if(x == 3 && y ==4){tecla = 'C'; }
   if(x == 4 && y ==4){tecla = 'D'; }
}


/* Função clear para o display*/
void clear_display(){

  display.clearDisplay();  //Apaga o buffer e display
  display.display();

}


/* Função de leitura das saídas reles
```

```
através da varivel cont */
void read_relay(){

    display.fillRect(x,y, x, 8, 0);
    display.setCursor(x+1,y);
    if(digitalRead(cont)==HIGH){ stringState = "OFF";}
    if(digitalRead(cont)==LOW){ stringState = "ON";}
    display.setTextColor(WHITE,BLACK);
    display.print(stringState);

}


/* Função de escrita no display a tela p/
alteração dos Set-points */
void SP_Temp_display(int x) {


  display.drawRoundRect(0,0, 84,48, 1, 1);
  display.drawRoundRect(7,2, 70,11, 1, 1);
  display.setCursor(20,4);
  display.println(title);

  display.setCursor(27,15);
  display.println(range);
  if(unit == "C"){display.drawCircle(71, 15, 1,1);}
  display.setCursor(73,15);
  display.print(unit);

  display.setTextColor(WHITE,BLACK);
  display.setCursor(4,30);
  display.println(change);
  display.setTextColor(BLACK);


  display.drawRoundRect(53,28, 26,11, 1, 1);
  display.fillRect(55,29, 11, 8, 0);
  display.setCursor(55,30);
  display.println(x);

  if(unit == "C"){display.drawCircle(70, 30, 1,1);}
  display.setCursor(72,30);
  display.print(unit);
  display.display();

}
```

```
/* enumerate dos estados da
maquina de estado */
enum nomeestados {

  EST_telainicial, // estado inicial
  EST_tela_mostra_parametros, //estado onde mostra os SP e qual fotociclo
  EST_change_SP_TEMP, //estado p/ mudança do sp de temperatura
  EST_change_SP_UMI, //estado p/ mudança do sp de umidade
  EST_change_Cardiano, //estado p/ mudança de fotociclo
  EST_tela_sup_relay //estado de leitura dos reles

};
nomeestados estado;


void maq_estados(){

switch(estado){

case EST_telainicial: //mostrar valores atuais das variaveis

        display.clearDisplay();   //Apaga o buffer e o display
        display.drawRoundRect(0,0, 44,24, 3, 2); //Desenha Retangulo Lum
        display.setCursor(11,3);  //Seta a posição do cursor (CxL)
        display.print("LUM");    //Texto inicial Luminosidade
        display.setCursor(30,14); //Seta a posição do cursor (CxL)
        display.println("lx");

        display.drawRoundRect(45,0, 39 ,24, 3, 2); //Desenha Retangulo Umi
        display.setCursor(52,3); //Seta cursor (CxL)
        display.print("UMI");    //Texto inicial Umidade
        display.setCursor(75,14); //seta cursor (CxL)
        display.print("%");

        display.drawRoundRect(0,25, 84 ,23, 3, 2); //Desenha Retangulo Temp
        display.setCursor(7,28); //seta cursor (CxL)
        display.print("TEMPERATURA"); //Texto inicial Temperatura
        display.setCursor(55,38); //seta cursor (CxL)
        display.drawCircle(52, 38, 1,1); //desenha pequeno circulo
        display.print("C");

       /* Abaixo a rotina em laço a qual escreve os valores
     de cada varivel em questao*/
```

```
      for(int cont_fill=3; cont_fill>0; cont_fill--){
        if(cont_fill == 3){display.fillRect(4,13, 25 , 10, 0); display.setCursor(5,14);
display.println(valorSensorLum,1);}
        if(cont_fill == 2){display.fillRect(50,13, 23 , 10, 0); display.setCursor(50,14);
display.println(valorSensorUmi,1);}
        if(cont_fill == 1){display.fillRect(4, 37, 46 , 10, 0); display.setCursor(11,38);
display.println(valorSensorTemp,1);}
      }
        display.display();
        if (tecla == '#'){estado = EST_tela_mostra_parametros;}


break;

case EST_tela_mostra_parametros: //mostra SP setados

        display.clearDisplay();   //Apaga o buffer e o display
        display.drawRoundRect(0,0, 84,12, 3, 2);

        /*Abaixo o laço for escreve os valores atuais dos SPs*/
        for(int cont_par = 3; cont_par>-1; cont_par--){
        if(cont_par == 3){display.setCursor(11,3); display.println("VALORES SP");}
        if(cont_par == 2){display.setCursor(0,16); display.println("TEMP:");
display.setCursor(30,16); display.println(valorSPTemp);}
        if(cont_par == 1){display.setCursor(0,24); display.println("UMI :");
display.setCursor(30,24); display.println(valorSPUmi);}
        if(cont_par == 0){display.setCursor(0,32); display.println("LUM :");
display.setCursor(30,32);
                    if(cont_cdf==1){display.println("Ciclo 1");}
                    if(cont_cdf==2){display.println("Ciclo 2");}
                    if(cont_cdf==3){display.println("Ciclo 3");}
                  }
        }
        display.display();

        if (tecla=='#'){  estado = EST_change_SP_TEMP;}
        if (tecla=='*'){  estado = EST_telainicial;}

break;

case EST_change_SP_TEMP: //estado para mudança de SP de Temperatura

  clear_display(); //limpa display
  title = "SP Temp.";
  range = "20 a 40";
  unit = "C";
  change = "SP_Temp:";
```

```
  SP_Temp_display(valorSPTemp); //chama funçao para escrita no display


    if (tecla=='#'){ valorSPTemp_f = valorSPTemp; estado = EST_change_SP_UMI; }
    if (tecla=='A'){if(valorSPTemp < 40){valorSPTemp = valorSPTemp+2;}}
    if (tecla=='B'){if(valorSPTemp > 20){valorSPTemp = valorSPTemp-2;}}
break;


case EST_change_SP_UMI: //estado p/ mudança de SP de Umidade

  title = "SP Umi.";
  range = "50 a 85";
  unit = "%";
  change = " SP_Umi:";
  clear_display();
  SP_Temp_display(valorSPUmi); //chama funçao para escrita no display

    if (tecla=='#'){valorSPUmi_f = valorSPUmi; estado = EST_change_Cardiano;}
    if (tecla=='A'){if(valorSPUmi < 85){valorSPUmi = valorSPUmi+5;}}
    if (tecla=='B'){if(valorSPUmi > 50){valorSPUmi = valorSPUmi-5;}}

break;


case EST_change_Cardiano: //estado para mudança de fotociclo

  clear_display(); //limpa display
  display.drawRoundRect(0,0, 84,48, 1, 1); //desenha retangulo
  display.drawRoundRect(2,2, 80,11, 1, 1); //desenha retangulo
  display.setCursor(4,4); //seta cursor
  display.println("Ciclo Cicard.");

  if(cont_cd==1 ){
  display.setTextColor(WHITE,BLACK); //texto com fundo preto
  display.drawRoundRect(2,15, 39,12, 1, 1); display.setCursor(8,18); display.println("CC_1");
  display.setTextColor(BLACK); //texto normal
  display.drawRoundRect(43,15,      39,12,      1,      1);      display.setCursor(49,18);
display.println("CC_2");
  display.drawRoundRect(2,33, 39,12, 1, 1); display.setCursor(8,36); display.println("CC_3");
  }
   if(cont_cd==2 ){
  display.drawRoundRect(2,15, 39,12, 1, 1); display.setCursor(8,18); display.println("CC_1");
  display.setTextColor(WHITE,BLACK); //texto com fundo preto
```

```
  display.drawRoundRect(43,15,      39,12,      1,      1);      display.setCursor(49,18);
display.println("CC_2");
  display.setTextColor(BLACK); //texto normal
  display.drawRoundRect(2,33, 39,12, 1, 1); display.setCursor(8,36); display.println("CC_3");
  }
   if(cont_cd==3 ){
  display.drawRoundRect(2,15, 39,12, 1, 1); display.setCursor(8,18); display.println("CC_1");
  display.drawRoundRect(43,15,      39,12,      1,      1);      display.setCursor(49,18);
display.println("CC_2");
  display.setTextColor(WHITE,BLACK); //texto com fundo preto
  display.drawRoundRect(2,33, 39,12, 1, 1); display.setCursor(8,36); display.println("CC_3");
  display.setTextColor(BLACK); //texto normal
  }
  display.display(); //funçao responvel por enviar as informaçoes pro display

    if (tecla=='A'){ cont_cd = cont_cd+1; if(cont_cd == 4){ cont_cd = 3;}}
    if (tecla=='B'){ cont_cd = cont_cd-1; if(cont_cd == 0){ cont_cd = 1;}}
    if (tecla=='#'){ cont_cdf = cont_cd; estado = EST_tela_sup_relay; }

 break;

  case EST_tela_sup_relay:

    clear_display();  //limpa display
    display.drawRoundRect(0,0, 84,48, 1, 1); //desenha retangulo

    /*Abaixo, laço for para supervisao
   dos reles*/

    for (cont=39; cont<45; cont++){
    if(cont   ==   39){display.drawRoundRect(7,2,   70,11,   1,   1);   display.setCursor(10,4);
display.println("Modulo Rele");
    }
    if(cont == 40){x = 20; y = 17; read_relay(); display.setTextColor(BLACK);
                display.drawRoundRect(2,15,     39,12,     1,     1);     display.setCursor(4,17);
display.println("R1:");}
    if(cont == 41){x = 61; y = 17; read_relay(); display.setTextColor(BLACK);
                display.drawRoundRect(43,15,    39,12,     1,     1);     display.setCursor(45,17);
display.println("R2:");}
    if(cont == 42){x = 20; y = 35; read_relay(); display.setTextColor(BLACK);
                display.drawRoundRect(2,33,     39,12,     1,     1);     display.setCursor(4,35);
display.println("R3:");}
    if(cont == 43){x = 61; y = 35; read_relay(); display.setTextColor(BLACK);
                display.drawRoundRect(43,33,    39,12,     1,     1);     display.setCursor(45,35);
display.print("R4:");
                display.display();}
     }
```

```
    if (tecla=='#'){estado = EST_telainicial; }

    break;


}}


void setup() {

  Serial.begin(9600); //inicializa serial monitor

  dht_2.begin(); //inicializa AM2302 1
  dht_1.begin();  //inicializa AM2302 2

  //abaixo segue inicialização do display
  display.begin();
  display.setContrast(50); //Ajusta o contraste do display
  display.clearDisplay();   //Apaga o buffer e o display
  display.setTextSize(1);  //Seta o tamanho do texto
  display.setTextColor(BLACK);
  delay(1000); //aguarda um segundo
  display.display();

  /*Abaixo segue laço for para definiçao
 das entradas e saídas do teclado*/
  for(int cont_tec = 37; cont_tec>29; cont_tec--){
    if(cont_tec >= 34 ) pinMode(cont_tec, INPUT);
    if(cont_tec <= 33 ) pinMode(cont_tec, OUTPUT);
  }

  pinMode(40, OUTPUT); //saida rele 1
  pinMode(41, OUTPUT); //saida rele 2


  /* abaixo segue definiçao de saídas
 dos leds de sinalizaçao */
  for(int cont_led = 51; cont_led>45; cont_led--){
    pinMode(cont_led, OUTPUT);
  }

  estado = EST_telainicial; //
  Timer1.initialize(83000000); // 2S set timer duration in microseconds
  // Abaixo, Configura a função time_cont como
  //a função para ser chamada a cada interrupção do Timer1
```

```
  Timer1.attachInterrupt(time_cont);
}

void loop() {


  if(on_maq_estados  == true){

    read_sensors(); //chama funçao leitura do sensor
    maq_estados(); //chama funçao maquina de estados
    on_maq_estados  = false;
    }

  //if(cont_cdf == 1){ ciclo_cicardiano_1(); }
  //if(cont_cdf == 2){ ciclo_cicardiano_2(); }
  //if(cont_cdf == 3){ ciclo_cicardiano_3(); }


    analogWrite(10, 255);
    //digitalWrite(50, HIGH);


  /*Se # * A ou B for pressionado
 o programa chama a funçao maquina
 de estados novamente*/
  KeyboardScan();
  if (tecla=='#' ||
      tecla=='*' ||
      tecla=='A' ||
      tecla=='B'
      ) {

          maq_estados();
      }
 }

//Abaixo, funçao leitura de sensor
void read_sensors(){

valorSensorLum = analogRead(0); //Leitura do LDR
//Valores limites de luminosidade (0 - 1127)
valorSensorLum = map(valorSensorLum, 0, 1023, 1127, 0);

  Serial.print("Tempo: ");
  Serial.print(i_minutes); //transmite atraves da seria os minutos
  Serial.print(" min");
```

```
Serial.print(" ---> ");

if(cont_cd==1){Serial.print("Ciclo Circadiano: 1");}
if(cont_cd==2){Serial.print("Ciclo Circadiano: 2");}
if(cont_cd==3){Serial.print("Ciclo Circadiano: 3");}

Serial.print(" - Numero:");
Serial.print(cont_ciclos);

Serial.print(" - Lum: ");
Serial.print(valorSensorLum);
Serial.println(" lux");

Serial.print("SPTemp: ");
Serial.print(valorSPTemp_f);
Serial.print(" - SP_Umi: ");
Serial.println(valorSPUmi_f);


res.begin(false); //desliga driver de acionamento

valorSensorUmi_1 = dht_1.readHumidity();
valorSensorTemp_1 = dht_1.readTemperature();
valorSensorUmi_2 = dht_2.readHumidity();
valorSensorTemp_2 = dht_2.readTemperature();

res.begin(true); //liga driver de acionamento

if (isnan(valorSensorUmi_1) || isnan(valorSensorTemp_1)) {
Serial.println("Falha AM2302_1 !!!");
med_1 = false;

}

else {
med_1 = true;
Serial.print("Temp 1: ");
Serial.print(valorSensorTemp_1);
Serial.print(" *C  ");
Serial.print("Umi 1: ");
Serial.print(valorSensorUmi_1);
Serial.println(" %");}


if (isnan(valorSensorUmi_2) || isnan(valorSensorTemp_2)) {
Serial.println("Falha AM2302_2 !!!");
```

```
med_2 = false;}


else{
med_2 = true;
Serial.print("Temp 2: ");
Serial.print(valorSensorTemp_2);
Serial.print(" *C  ");
Serial.print("Umi 2: ");
Serial.print(valorSensorUmi_2);
Serial.println(" %");}



if (med_1 == false && med_2 == false) {
Serial.println("SENSORES COM FALHA - VERIFICAR!!!");
Serial.println(" ");}

if (med_1 == false && med_2 == true) {
valorSensorUmi = valorSensorUmi_2;
valorSensorTemp = valorSensorTemp_2;
media_AM2302();}

if (med_1 == true && med_2 == false) {
valorSensorUmi = valorSensorUmi_1;
valorSensorTemp = valorSensorTemp_1;
media_AM2302();}

if (med_1 == true && med_2 == true){
valorSensorUmi = (valorSensorUmi_1 + valorSensorUmi_2)/2;
valorSensorTemp = (valorSensorTemp_1 + valorSensorTemp_2)/2;
media_AM2302();}

if(valorSensorTemp > valorSPTemp_f  ){ res.set(100, OFF); digitalWrite(51, LOW); }
if(valorSensorTemp < valorSPTemp_f  ){ res.set(100, ON); digitalWrite(51, HIGH); }

if(valorSensorUmi > valorSPUmi_f  ){ digitalWrite(porta_rele1, HIGH);
                                     digitalWrite(porta_rele2, LOW);}

 if(valorSensorUmi < valorSPUmi_f  ){ digitalWrite(porta_rele1, LOW);
                                     digitalWrite(porta_rele2, HIGH);}


if(digitalRead(porta_rele1)==HIGH){  Serial.print("Umidificador:  OFF  |");  digitalWrite(46,
LOW);}
```

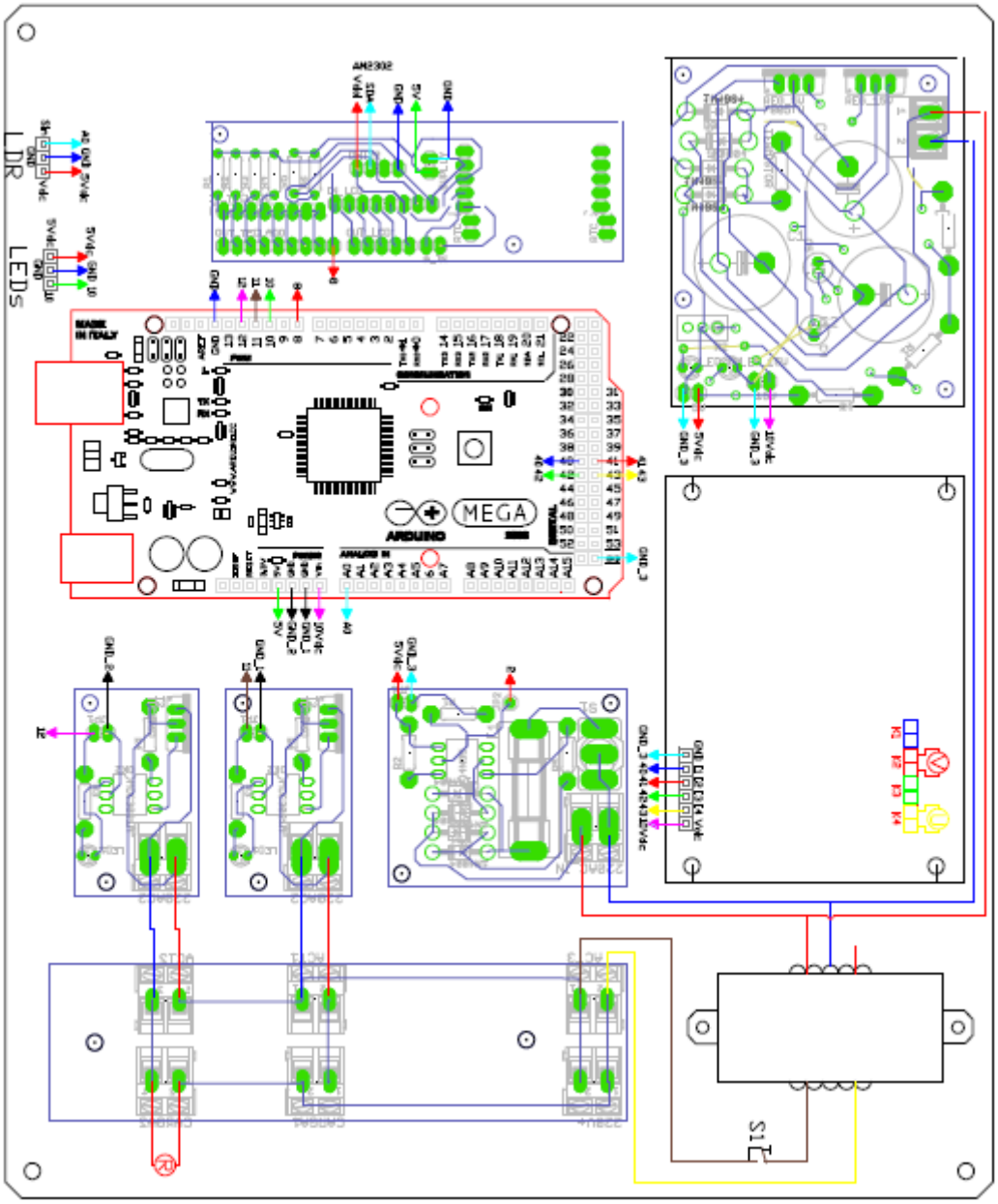
```
  if(digitalRead(porta_rele1)==LOW){  Serial.print("Umidificador:  ON  |");  digitalWrite(46,
HIGH);}

  if(digitalRead(porta_rele2)==HIGH){  Serial.println("Ventilador  :  OFF  |");  digitalWrite(47,
LOW);}
  if(digitalRead(porta_rele2)==LOW){  Serial.println("Ventilador  :  ON  |");  digitalWrite(47,
HIGH);}


  Serial.print("/----------------------------------------------------------------/");
  Serial.println(" ");




}



void time_cont(){


i_seconds = i_seconds + 1;
if(i_seconds * 8.3 >= 60){
   i_seconds = 0;
   i_minutes = i_minutes + 1;

    read_sensors();
    //control();
    maq_estados();

    if(i_minutes == 60 ){
      i_seconds = 0;
      i_minutes = 0;
      i_hours = i_hours + 1;

          if(i_hours == 24){
             i_seconds = 0;
             i_minutes = 0;
             i_hours = 0;
            }
}}}

void media_AM2302(){

  Serial.print("Temperatura Média: ");
```

```
  Serial.print(valorSensorTemp);
  Serial.print(" *C  ");
  Serial.print("Umidade Média : ");
  Serial.print(valorSensorUmi);
  Serial.println(" %");

}


void ciclo_cicardiano_1() {
  if(i_hours == 0){ analogWrite(13, 128); }
  if(i_hours == 1){ analogWrite(13, 160);}
  if(i_hours == 2){ analogWrite(13, 192);}
  if(i_hours == 3){ analogWrite(13, 224);}

  if(4 <= i_hours < 9 ){ analogWrite(13, 255);}

  if(i_hours == 9)  { analogWrite(13, 224);}
  if(i_hours == 10 ){ analogWrite(13, 192);}
  if(i_hours == 11 ){ analogWrite(13, 160);}
  if(i_hours == 12 ){ analogWrite(13, 128);}


  if(i_hours == 13 ){ analogWrite(13, 96);}
  if(i_hours == 14 ){ analogWrite(13, 64);}
  if(i_hours == 15 ){ analogWrite(13, 32);}
  if(16 <= i_hours < 21 ){ analogWrite(13, 0);}

  if(i_hours == 21) { analogWrite(13, 32);}
  if(i_hours == 22 ){ analogWrite(13, 64);}
  if(i_hours == 23 ){ analogWrite(13, 96);}
  if(i_hours == 24 ){ analogWrite(13, 128);
    cont_ciclos = cont_ciclos + 1;
    i_seconds = 0;
    i_minutes = 0;
    i_hours = 0;}
  }


  void ciclo_cicardiano_2() {

  if(0 >= (i_hours*100 + i_minutes) <31 ){ analogWrite(13, 128);}
  if(31 >= (i_hours*100 + i_minutes) <101 ){ analogWrite(13, 160);}
  if(101 >= (i_hours*100 + i_minutes) <131){ analogWrite(13, 192);}
  if(131 >= (i_hours*100 + i_minutes) <201){analogWrite(13, 224);}
```

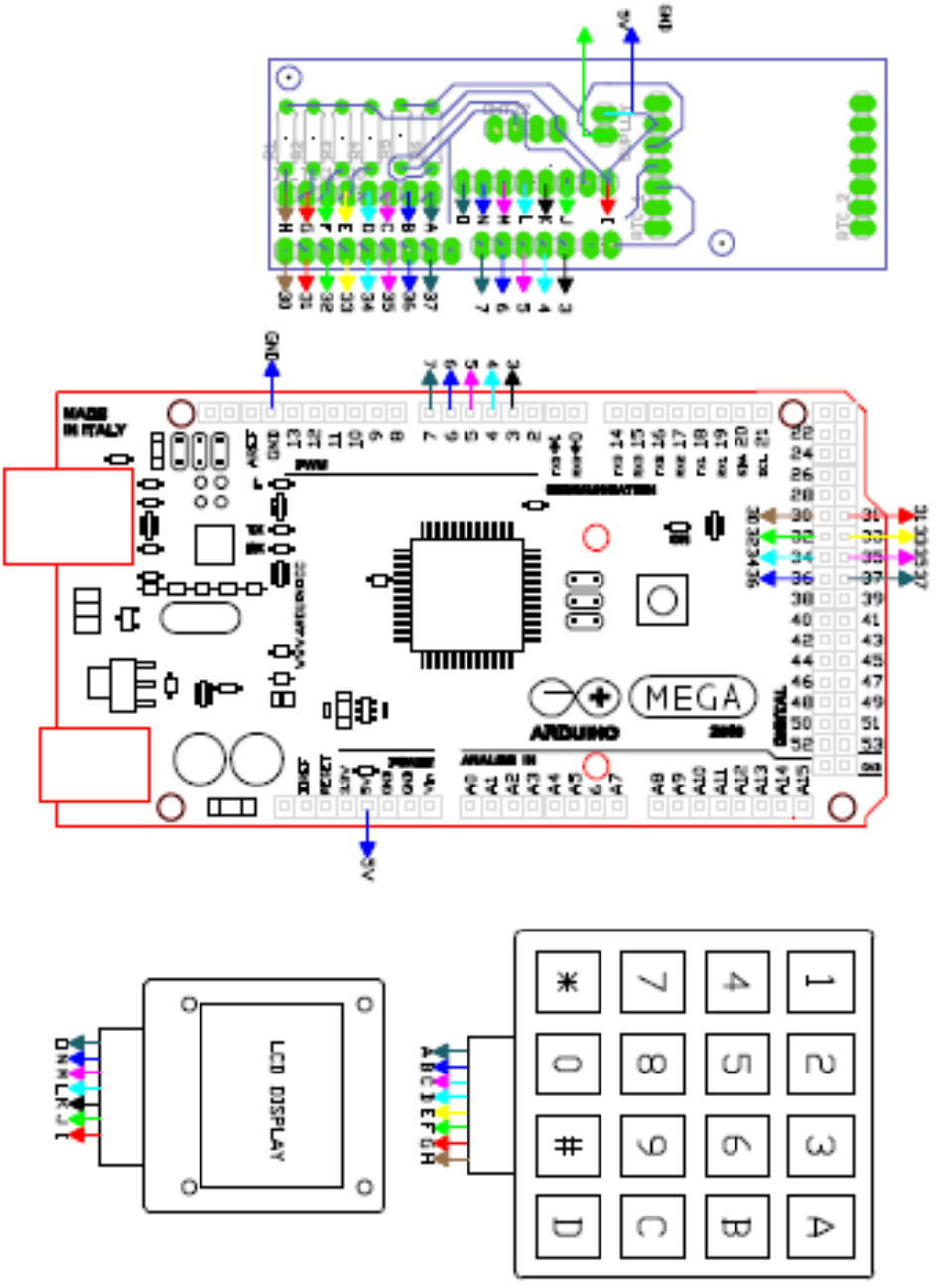
```
if(201 >= (i_hours*100 + i_minutes) < 401 ){ analogWrite(13, 255);}

if(401 >= (i_hours*100 + i_minutes) < 431){ analogWrite(13, 224);}
if(431 >= (i_hours*100 + i_minutes) < 501){ analogWrite(13, 192);}
if(501 >= (i_hours*100 + i_minutes) < 531){ analogWrite(13, 160);}
if(531 >= (i_hours*100 + i_minutes) < 601 ){analogWrite(13, 128);}


if(601 >= (i_hours*100 + i_minutes) < 631 ){ analogWrite(13, 96);}
if(631 >= (i_hours*100 + i_minutes) < 701){ analogWrite(13, 64);}
if(701 >= (i_hours*100 + i_minutes) < 731 ){ analogWrite(13, 32);}
if(731 >= (i_hours*100 + i_minutes) < 801){ analogWrite(13, 0);}

if(801 >= (i_hours*100 + i_minutes) < 1001){ analogWrite(13, 0);}


if(1001 >= (i_hours*100 + i_minutes) < 1031) { analogWrite(13, 32);}
if(1031 >= (i_hours*100 + i_minutes) < 1101){ analogWrite(13, 64);}
if(1001 >= (i_hours*100 + i_minutes) < 1131){ analogWrite(13, 96);}
if(1131 >= (i_hours*100 + i_minutes) < 1201){ analogWrite(13, 128);

  cont_ciclos = cont_ciclos + 1;
  i_seconds = 0;
  i_minutes = 0;
  i_hours = 0; }
}

void ciclo_cicardiano_3() {

if(i_hours == 0){ analogWrite(13, 128);}
if(i_minutes == 20){ analogWrite(13, 160);}
if(i_minutes == 40){ analogWrite(13, 192);}
if(i_minutes == 60){ analogWrite(13, 224);}



if( 120 <= (i_hours*100 + i_minutes) < 241 ){analogWrite(13, 255);}


if(241 <= (i_hours*100 + i_minutes) < 301  ){analogWrite(13, 224);}
if(301 <= (i_hours*100 + i_minutes) < 321  ){analogWrite(13, 192);}
if(321 <= (i_hours*100 + i_minutes) < 341  ){analogWrite(13, 160);}
if(341 <= (i_hours*100 + i_minutes) < 401  ){ analogWrite(13, 128);}
```

```
if(401 <= (i_hours*100 + i_minutes) < 421   ){ analogWrite(13, 128);}
if(421 <= (i_hours*100 + i_minutes) < 441  ){analogWrite(13, 96);}
if(441 <= (i_hours*100 + i_minutes) < 501   ){ analogWrite(13, 64);}
if(501 <= (i_hours*100 + i_minutes) < 521 ){ analogWrite(13, 32);}

if(521 <= (i_hours*100 + i_minutes) < 641) { analogWrite(13, 0);}


if(641 <= (i_hours*100 + i_minutes) < 701 ){ analogWrite(13, 32);}
if(701 <= (i_hours*100 + i_minutes) < 721){ analogWrite(13, 64);}
if(721 <= (i_hours*100 + i_minutes) < 741){ analogWrite(13, 96);}
if(741 <= (i_hours*100 + i_minutes) < 801){ analogWrite(13, 128);

  cont_ciclos = cont_ciclos + 1;

  i_seconds = 0;
  i_minutes = 0;
  i_hours = 0;
  }


}
```

APÊNDICE B – Esquemático de ligação parte inferior


LDR
LEDS
MEGA
ARDUINO
S1

APÊNDICE C – Esquemático de ligação parte superior


MADE IN ITALY
ARDUINO
MEGA
LCD DISPLAY
1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D