IPG Politécnico da Guarda
Polytechnic of Guarda

RELATÓRIO DE PROJETO

Licenciatura em Engenharia Informática

Ricardo António Correia Ferreira

setembro | 2016

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

RELATÓRIO DE PROJETO
GESTÃO DE UMA ESTUFA

RICARDO ANTÓNIO CORREIA FERREIRA

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO
EM ENGENHARIA INFORMÁTICA

SETEMBRO /2016

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

PROJETO DE INFORMÁTICA

2015/2016

GESTÃO DE UMA ESTUFA

Orientador: Prof. Luís Figueiredo

Realizado por: Ricardo António Correia Ferreira

Numero de aluno:

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

RELATÓRIO DE PROJETO
GESTÃO DE UMA ESTUFA

RICARDO ANTÓNIO CORREIA FERREIRA

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO
EM ENGENHARIA INFORMÁTICA

SETEMBRO /2016

ORIENTADOR: PROFESSOR LUÍS FIGUEIREDO

AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer ao Instituto Politécnico da Guarda pela formação dada ao longo dos anos, aos docentes e não docentes que participaram na minha educação direta ou indiretamente.

Agradeço ao meu orientador, o professor Luís Figueiredo pelo tempo que me disponibilizou durante a elaboração do meu projeto. Foi um privilégio ser seu orientado.

Gostaria de agradecer à minha família pelo apoio e dado durante todo o meu percurso académico, nunca deixando de me apoiar e estando lá sempre para me ajudar no que fosse preciso.

Também aos amigos, que fiz nesta Instituição ao longo do meu percurso, que me ajudaram e que me acompanharam neste caminho desde o inicio até ao presente dia.

Um agradecimento muito especial à pessoa que sempre esteve do meu lado mesmo quando as coisas não corriam bem, apoiando e criticando, fazendo me sentir sempre bem.

Obrigado a todos os que me ajudaram.

RESUMO

O tema deste projeto é a gestão de uma estufa através da criação de um sistema capaz de gerir uma estufa e obter maior qualidade nas plantações, para isto é preciso ter em conta que uma estufa tem fatores de risco que podem por em causa toda a plantação. Os fatores tidos em conta são a velocidade do vento, a temperatura e a humidade do solo. Através deste sistema, a gestão será feita remotamente de forma a evitar deslocações e reduzindo o tempo de reação. Por outro lado, este sistema permitirá racionalizar o consumo de água tornando assim mais eficiente a gestão da estufa por parte dos agricultores.

O método utilizado para a resolução deste problema, é o uso de um microcontrolador apoiado por um modulo *Wi-fi(ESP8266)* capaz de receber comandos para ligar ou desligar atuadores. Com este também existe a possibilidade de enviar para um *web server* os dados da temperatura de dentro e fora, a velocidade do vento e da humidade do solo, dando assim ao utilizador uma perspetiva de como a estufa se encontra naquele momento.

Com este projeto pretende-se criar melhores condições no interior da estufa de forma a obter uma qualidade maior da plantação. O sistema está preparado para responder às alterações que possam prejudicar toda a produção, reduzindo assim o custo e tempo de resposta.

ABSTRACT

The theme of this project is the management of a greenhouse by creating a system capable of managing a greenhouse and get higher quality in plantations, for this we must take into account that a greenhouse has risk factors that can jeopardize the whole plantation. The factors taken into account are the wind speed, temperature, and soil moisture. Through this system, management will be done remotely to avoid dislocations and reducing the reaction time. Moreover, this system allows to rationalize the consumption of water thus making it more efficient management of the greenhouse by farmers.

The method used to solve this problem is the use of a microcontroller supported by a Wi-Fi module (ESP8266) capable of receiving commands to turn on or off actuators. With this it is also possible to send to a web server on the temperature data from outside and inside, the wind speed and soil moisture, thereby giving the user a perspective of how the heater is at that moment.

This project aims to create better conditions inside the greenhouse in order to obtain a higher quality of planting. The system is prepared to respond to changes that might harm the entire production, thereby reducing the cost and response time.

Índice

Índice de Figuras

Índice de tabela

INTRODUÇÃO

Com a evolução da Tecnologia, hoje em dia, é possível usar e desenvolver um conjunto de sistemas incríveis, inimagináveis há uma década, possibilitando fazer um pouco de tudo. Nem sempre é necessário um grande esforço para ser construído um sistema com grande fiabilidade e boa qualidade.

O problema proposto foi a gestão de uma estufa onde existem vários fatores a ter em conta, como a humidade do solo, o vento e a temperatura. A necessidade de uma deslocação por parte do responsável à estufa e o consumo do recurso natural que é a água são também aspetos que se pretende melhorar com este trabalho.

A solução que é sugerida neste trabalho passa pela automatização, onde existe a possibilidade de mais controlo sobre os recursos, assim facilitando o utilizador e ajudando a poupança dos mesmos. Deste modo a utilização de tecnologia é um benefício para ajudar a alcançar os objetivos pretendidos.

O projeto utiliza como tecnologia, um microcontrolador, sensores, atuadores e módulo Wi-fi, onde o objetivo principal vai ser a monitorização e automatização da estufa fazendo assim com que o produtor não se desloque tantas vezes ao local onde esta se encontra.

Com este trabalho existe uma gestão mais rigorosa da água, não a utilizando em demasia, nem deixando as plantas sujeitas ao stresse da falta de água, ajudando o produtor a não ter prejuízos. Pretende-se assim que o lucro dos produtos passe a ser maior e as despesas menores.

Este projeto possibilita fazer a manutenção da estufa por baixo custos. As outras possibilidades que existem para fazer este projeto são mais dispendiosas para as duas partes, quem o faz, devido ao elevado custo e para o utilizador final que acaba por ter de pagar muito mais pelo sistema.

A grande inovação deste projeto é o facto de que o utilizador poder ver o que está a acontecer naquele exato momento na sua estufa sem sair de casa o que lhe permite intervir, se o desejar, em tempo real.

Para uma gestão à distância é utilizado conjuntamente com um microcontrolador, um modulo que permite a este ter internet e aceder a um site onde serão disponibilizados todos os dados sobre a estufa, também podendo ligar qualquer um dos motores usados para controlar as janelas ou abrir uma torneira e ligar as luzes.

Todas as plantas têm vários fatores de risco, como por exemplo, a temperatura muito baixa. Basta algumas horas ou até minutos para a plantação morrer e gerar elevados prejuízo

para o agricultor. Estes fatores podem ser evitados com este sistema porque o mesmo consegue adaptar-se a estas situações.

Em zonas ventosas, zonas muito quentes, zonas muito húmidas é preciso ter cuidado com os fatores que condicionam a plantação inteira causando prejuízos enormes para os donos das mesmas. O sistema permite não só a redução dos prejuízos como o aumento do lucro tornando assim um sistema que pode revolucionar a área da agricultura.

Nos capítulos que se seguem serão desenvolvidos os pontos fulcrais deste projeto, mostrando assim a descrição do problema, uma solução encontrada para o mesmo e tudo o que foi usado para a realização deste projeto.

CAPITULO 1 -ENQUADRAMENTO TEÓRICO

1.1 O que é uma estufa?

Uma estufa é uma estrutura com o objetivo que acumular calor e manter uma temperatura maior que do lado de fora. Esta estrutura é muito utilizada para o cultivo de legumes, plantas, árvores e flores. Esta é feita de materiais semitransparentes, diminuindo as trocas de ar entre o interior e o exterior fazendo assim com que o calor se mantenha não sendo disperso pelas correntes de ar.

1.2 O Problema

Este tema surgiu no âmbito da realização do Projeto final de curso e tem em vista a implementação de um sistema de gestão de uma estufa.

As novas tecnologias vieram impulsionar e dar a descobrir novos métodos que podem enriquecer a produção de plantas (árvores de frutos, árvores decorativas, plantações de legumes, etc.) em estufa.

Para uma estufa para ter uma boa produção deve ter uma temperatura ideal. Por vezes existem desperdícios de água ou até mesmo descuidos como por exemplo, deixar uma janela aberta durante uma noite fria e no dia a seguir a plantação estar queimada. O problema a resolver é mesmo este, fazer com que não haja espaço para se perder uma plantação inteira à custa de um descuido. Usar apenas a água que a planta precisa e tentando sempre manter uma temperatura ideal para as plantas crescerem.

1.3 Solução

A solução encontrada foi a utilização de um microcontrolador para fazer o controlo da estufa. O uso do Arduino (microcontrolador) vai informar o produtor do estado físico em que se encontram os seus produtos, controlando a humidade do solo, a temperatura e sabendo a velocidade que o vento sopra.

Usando os sensores para obter as informações sobre o estado da estufa, o sistema permite saber quando é preciso efetuar uma ação. Este sistema tem o fim de ajudar o utilizador a alcançar o máximo de potencial da sua plantação que seja possível. Procurará manter, a temperatura sempre ideal para a plantação, não a deixando descer em demasia. A rega não será feita de uma forma em que haja desperdício e quando o vento começar a ficar mais perigoso vai fechar imediatamente a janelas. Para fazer isto vão ser usados um conjunto de atuadores que vão realizar este trabalho conjuntamente com as informações recebidas pelos sensores. O utilizador tanto poderá ativar ou desativar estes atuadores, enviando o comando para o Arduíno, como poderá deixar o próprio microcontrolador decidir por ele o que deve fazer.

Esta utilização implica o máximo de rigor e profissionalismo, pois trará benefícios quer para o produtor quer para o meio ambiente.

1.4 Vantagens deste Projeto

A criação deste projeto tem vantagens que proporcionam ao utilizador final uma boa gestão e um melhor controlo, em tempo real, da sua plantação. Terá acesso de uma forma eficaz e eficiente às informações dos seus produtos, podendo assim intervir quando quiser.

Este método irá facilitar o produtor na ótica do consumo energético pois visa, diminuir as despesas relacionadas com a eletricidade, o que levará a uma poupança. O consumo da água irá também ser diminuído, o que pode possibilitar uma poupança significativa em termos monetários. Por fim há ainda a poupança de tempo do utilizador.

Uma das grandes vantagens deste projeto é a adaptabilidade a qualquer tipo de estufa e os atuadores e sensores a esta associados, sendo que a variação das necessidades de cada projeto influencia o tipo dos seus atuadores e sensores e dessa forma o orçamento dos projetos.

1.5 Análise do Mercado

As soluções que existem no mercado, permitem aos utilizadores fazer o controlo da estufa a preços mais elevados, devido ao facto de que cada atuador utilizado ser especifico e estar programado para realizar uma tarefa específica.

Uma das soluções encontradas foi da empresa Estufas Minho (ESTUFASMINHO, 2016), que tem um conjunto variado de soluções, para vários tipos plantações. Os sensores são o equivalente aos usados neste projeto. Os atuadores permitem controlar as janelas e a rega da estufa.

Outras empresas como HortaNova (HortaNova, 2016) tem soluções para a qualidade das plantas, nomeadamente, ***NFT*** (***Nutrient Film Technique***), um tipo elevado de plantação que é isenta de bicharada, controlando também os nutrientes da água (Figura 1).

Figura 1 - Solução HortaNova

A PRILUX (Prilux, 2016) comercializa vários tipos de estufas, cada uma delas para um certo tipo de plantação (Figura 2), podendo o utilizador escolher os atuadores que pretende ter na estufa, como janelas, rega gota a gota, entre outros (Figura 3).

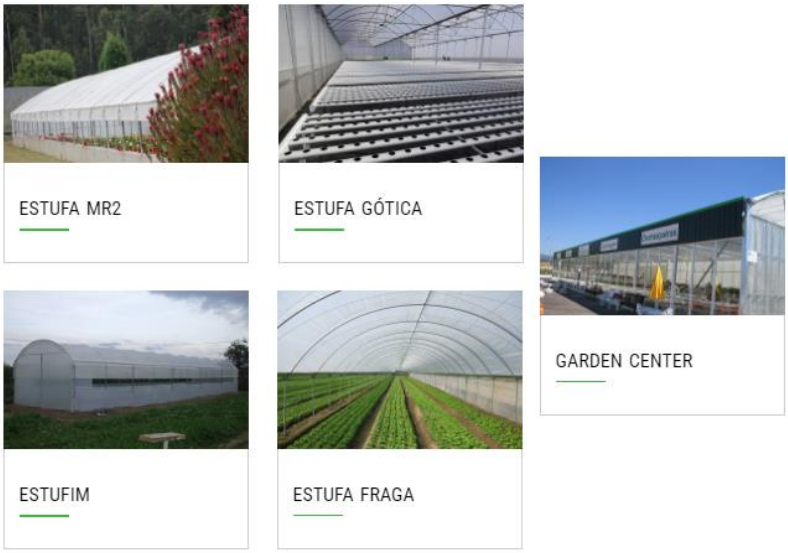

Figura 2 - Estufas comercializadas PRILUX

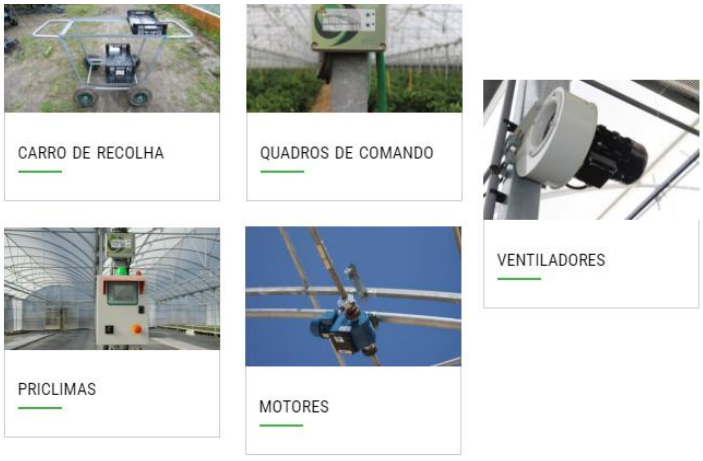

Figura 3 - Atuadores da empresa PRILUX

1.6 O que existe de novo neste projeto

Neste projeto existe a possibilidade para haver uma manutenção e gestão da estufa a longa distancia, não precisando assim o utilizador se deslocar tantas vezes para o fazer, para isso o utilizador conta com uma ligação em tempo real a um servidor *web*.

Este sistema é de custo reduzido tornando-se mais acessível a sua aquisição pelo público em geral, trazendo vantagens na gestão de uma estufa, tais como:

- Gestão da estufa à distância, reduzindo assim as deslocações à mesma;
- Controlo dos consumos de recursos naturais;
- Adaptabilidade e expansibilidade de atuadores usados numa estufa

1.7 Metodologia Utilizada

Em primeiro lugar é preciso afirmar que este projeto irá usar o método de desenvolvimento ágil, em que o projeto irá evoluir consoante os resultados obtidos nos testes, testes esses que serão feitos ao longo do seu desenvolvimento.

Para fazer a programação no Arduíno irá ser usado Arduíno programming language que dará acesso a todas as suas funcionalidades que podem ser programadas, sendo o programa usado e facultado pelos próprios desenvolvedores de Arduíno.

1.8 Resultados Esperados

Os resultados que são pretendidos com este projeto são nomeadamente a redução do consumo da água, o que leva a uma melhor utilização deste recurso natural usado, e a manutenção da melhor temperatura para o desenvolvimento das plantas otimizando assim ao máximo a sua produção.

É pretendido também que o utilizador deste sistema possa fazer a manutenção da estufa à distância, facilitando assim a possibilidade da execução de outras tarefas tão ou mais importantes para esse utilizador, o que resulta em menos tempo gasto na manutenção da estufa.

Num caso mais grave em que a estufa esteja situada num lugar ventoso e aconteça um caso em que o utilizador tenha deixado as "janelas" abertas, nesse caso remotamente, pode fechar as mesmas sem se deslocar ao local e assim salvando a estufa de ter maiores danos, ou pode deixar o sistema decidir o que fazer quando isso acontece.

Todos estes pontos contribuirão para um maior lucro, pois os produtos crescem melhor e mais fortes levando a uma melhor qualidade, gerando assim um aumento nas receitas e a diminuição da mão de obra e custos associados à exploração da estufa.

CAPITULO 2 – EQUIPAMENTOS E SISTEMAS UTILIZADOS

2.1. Equipamento e breve explicação

O equipamento usado neste projeto é o seguinte:

1x Arduíno Uno
1x Potenciómetro
3x LED
1x ESP 8266 ou modulo de Wi-fi
1x Sensor humidade (DHL)
2x Sensor Temperatura (TMP 36)
5x Resistências:

- 1K ohms
- 2K ohms

1x Conversor 5Volt para 3.3Volt
1x BreadBoard
Jumper's

Neste ponto vai ser abordado com uma breve descrição, cada um dos componentes mencionado em cima, para uma melhor compreensão do material usado.

1. Arduino

O arduino uno é um microcontrolador baseado no *ATmega 328P*. Este tem 14 entradas/saídas digitais, 6 entradas analógicas, uma ligação USB, um botão de *reset* e uma entrada de alimentação. Todas as entradas/saídas digitais e analógicas trabalham com voltagens lógicas de 0 a 5Volt.

Para a programação do mesmo basta simplesmente ligar o arduino a um computador através de uma porta USB. Usando o *Arduino Software,* um IDE (Ambiente de Desenvolvimento Integrado) gratuito fornecido pelos próprios desenvolvedores do Arduino, é possível programar o mesmo com uma linguagem C proprietária. Este pode ser programado para funções específicas que podem assim resolver alguns problemas do dia-a-dia.

Devido á facilidade de programação do mesmo é possível ligar a este os mais diversos tipos de sensores e atuadores que existem, podendo encontrar assim as suas bibliotecas na internet facilitando assim o seu uso.

Com este é possível realizar as tarefas pretendidas facilitando muitas delas que por vezes possam ser maçadoras para o ser humano. É preciso falar também um pouco de que graças a esta tecnologia, foi possível uma evolução imensa no ramo da robótica, da ciência e da domótica onde graças a este pequeno microcontrolador é possível realizar vários projetos como, controlo

de acesso por leitor de RFID, um relógio de cozinha, um detetor de fumo, assim como uma estufa (o caso deste projeto), entre outros (Arduino).

2. Sensores

Em primeiro é preciso referir que um sensor é um dispositivo usado para ler o meio ambiente onde se encontra e lendo esses dados, tanto físico como químico, podendo ser medido analogicamente ou digitalmente. Os tipos de sensores usados neste projeto são sensores de temperatura, usado para se poder medir a temperatura dentro e fora da estufa, potenciómetros, para fazer a simulação da velocidade do vento e para poder fazer simulações da temperatura interior, outro sensor usado é de humidade do solo para se poder saber quando se deve regar a plantação.

Neste projeto são usados potenciómetros para simular a velocidade do vento devido ao elevado custo deste sensor. É usado o potenciómetro para simular a temperatura de dentro da estufa para a simulação de todas as possibilidades que acontecem dentro da estufa.

2.1. Potenciómetro

O Potenciómetro é usado para fazer a simulação da velocidade do vento que pode haver, assim pode ter-se uma maior aproximação do que acontece numa estufa verdadeira, pois o vento é essencial para ter um controlo sobre a estufa, a nível de climatização assim como da segurança. Este sensor também é utilizado para a simulação da temperatura interior da estufa a fim de poderem ser testadas todas as condições.

2.2. Sensor de Temperatura

Os sensores de Temperatura são para detetar a temperatura que está a cada momento na estufa e fora dela, assim podemos saber se a estufa se encontra com uma temperatura adequada.

2.3. Sensor de humidade

Sensor de humidade do Solo é um sensor que vai indicar, se o solo onde se encontram as plantas está húmido, seco e/ou mesmo submerso permitindo assim controlar os sistemas de rega.

3. Resistências

As resistências são usadas com o fim de fazer com que os *Led's* não queimem quando os ligarmos a uma tensão de 5Volt. Estas também são usadas para o módulo de Wi-fi.

4. Led's

Os três *Led's* são usados para simular 3 atuadores diferentes, já que os atuadores para o controlo da estufa são dispendiosos. Estes irão simular a abertura de uma torneira fazendo assim

irrigação por gotejamento, simular a abertura e ou fecho de uma janela e ligar e/ou desligar as luzes artificiais da estufa.

5. Modulo Wi-fi (ESP8266)

O módulo Wi-fi (designado por ESP8266) é um componente com TCP/IP integrado que dá a qualquer microcontrolador acesso a uma rede Wi-fi, após ser programado para tal.

O ESP8266 é capaz de hospedar ou mesmo ir buscar todas as funções da rede Wi-fi. Este modulo vem pré-programado com um conjunto *firmware* de comando AT (Tabela 1). Com este é possível fazer qualquer coisa como se fosse usado um *Wi-fi Shield's*.

Este módulo permite ser integrado com os sensores e dispositivos específicos de outra aplicação.

Usando este modulo é possível fazer a manutenção a longa distância que este projeto necessita.

Comando	Valores	Descrição	Exemplo	Resposta
AT	-	Verifica se o ESP está OK	AT	OK
AT+RST	-	Reinicia o ESP	AT+RST	OK
AT+CWMODE	1 = Station, 2 =AP, 3=Ambos	Define o modo de operação Wi-fi	AT+CWMODE=3	Query, Sta, ap, both
AT+CWLAP	-	Lista de redes disponíveis	AT+CWLAP	query
AT+CWJAP	SSID, Senha	Ligar a uma rede	AT+CWJAP="SSID","PASSWORD"	query
AT+CWQAP	-	Desligar da rede	AT+CWQAP	query, OK
AT+CIPSTATUS	-	Retorna o status do Wi-fi	AT+CIPSTATUS	AT+CIPSTATUS
AT+CWJAP?	-	Verifica a rede ligada	AT+CWJAP?	Query, ok
AT+CIFSR	-	Retorna o IP do ESP na rede	AT+CIFSR	AT+CIFSR 192.168.0.106, OK

Tabela 1 -Tabela de Comandos AT

6. Conversor de 5Volt para 3.3Volt (AMS1117)

Para termos esta tensão de 3.3Volt usamos um conversor que vai converter os 5Volt de entrada nos 3.3Volt de saída, alimentando assim os dispositivos que precisam desta tensão para o seu funcionamento.

7. BreadBoard

A *BreadBoard* serve para fazer a montagem dos circuitos elétricos, esta facilita a inserção de componentes pois não precisam de ser soldados.

8. *Jumper's* ou cabos elétricos

Os cabos elétricos são usados para ligar dois pontos elétricos, este tem uma pequena peça plástica isolante que contém uma peça metálica no seu interior, responsável pela condução de eletricidade.

9. ThingSpeak

9.1. O que é?

O ThingSpeak é um site online (Figura 4) que permite receber e/ou enviar dados no novo mundo da internet das coisas, onde também se pode enquadrar o Arduíno. Com este site, existe a possibilidade de receber os dados que são enviados do Arduíno e avaliá-los minuto a minuto. Caso alguma coisa esteja a correr mal o utilizador pode, a partir do mesmo, escolher a melhor solução, isto caso o utilizador não queira que este sistema responda automaticamente e deseje ser ele o próprio a decidir o que fazer.

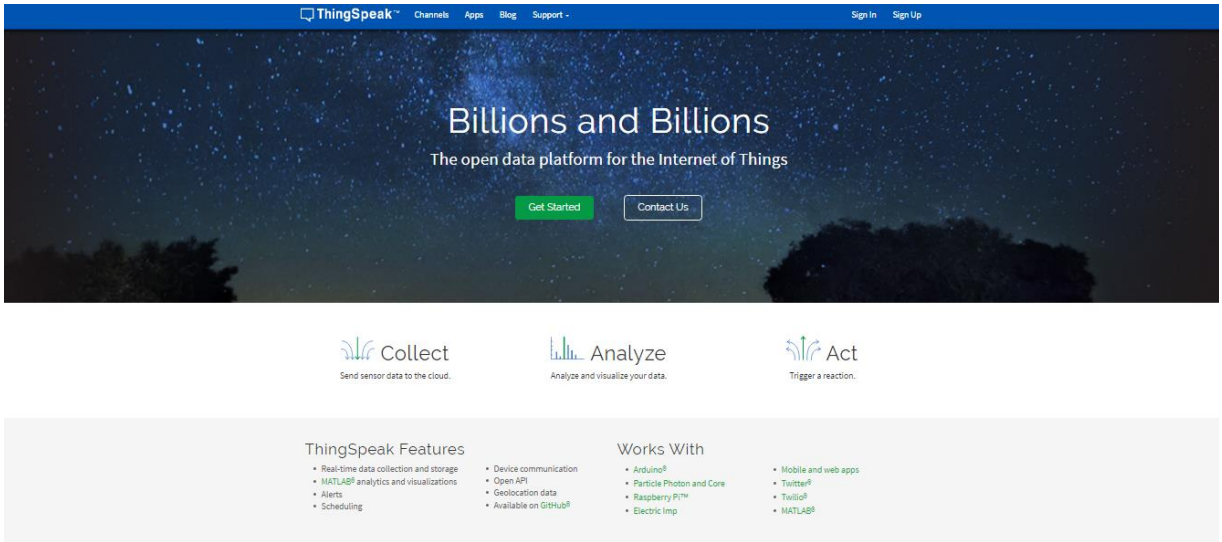

Figura 4 - Pagina Inicial ThingSpeak

9.2. Porquê a escolha deste?

A escolha do uso desta plataforma foi pela facilidade de acesso da mesma e pela facilidade de uso que esta nos dá, para além de ser completamente gratuita.

Este site permite ativar/desativar um atuador, ver a temperatura em tempo real, ver a velocidade do vento e a humidade do solo sem o utilizador ter de se deslocar à estufa. Com este o utilizador tem um trabalho mais facilitado, só se deslocando à estufa quando realmente é

preciso. Sem isto ele também se deslocava quando realmente era preciso, mas agora são menos vezes.

9.3. Para que é usado?

É usado para a ativação ou desativação do(s) atuador(es) e para a monitorização contínua dos valores dos sensores.

Para a análise dos resultados são usados gráficos dos mesmos tendo assim uma perceção de como os valores estão a evoluir ao longo do tempo. Para a ativação dos atuadores assim como a sua desativação é usada uma pequena página web que permite enviar comandos para o Arduino.

9.4. Como proceder à criação

9.4.1. Channel

Para proceder a criação dum *channel* no *ThingSpeak*, em primeiro lugar é preciso ter conta no site, de seguida escolhe-se a opção *channels* e faz-se a criação do mesmo como se pode ver na Figura 5.

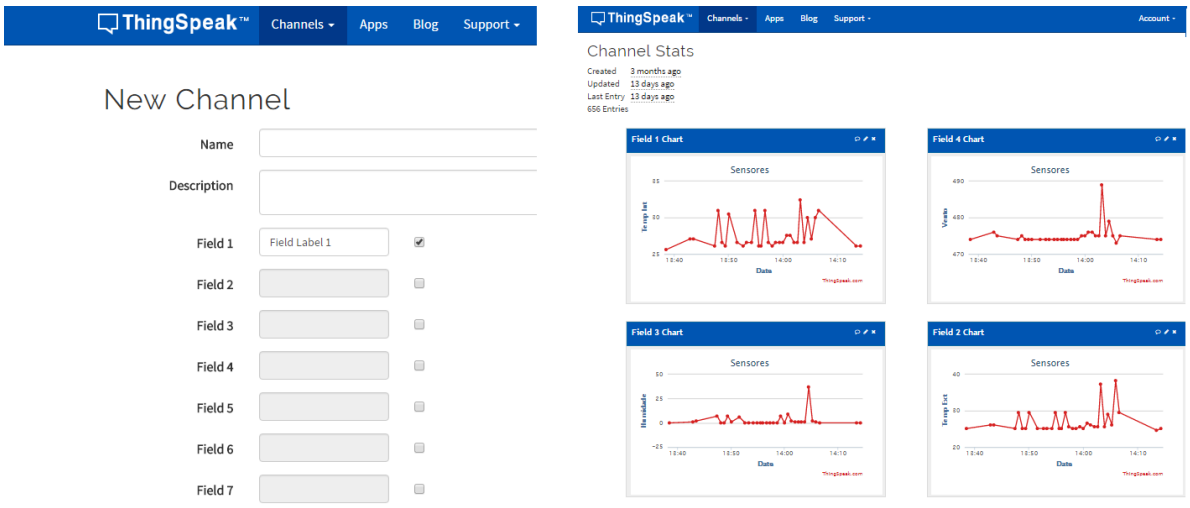

Figura 5 - Criação dum Channel

Como se pode ver, a criação do mesmo é simples pois basta colocar o nome nos *Field's,* escolher o nome para este, caso se queira escrever uma descrição do mesmo e de seguida salvar o *channel*.

9.4.2. TalkBack

Fazer a criação do *TalkBack* escolhe-se a opção "*Apps*" e logo de seguida escolhe-se a opção "*TalkBack*" e vai aparecer como na Figura 7.

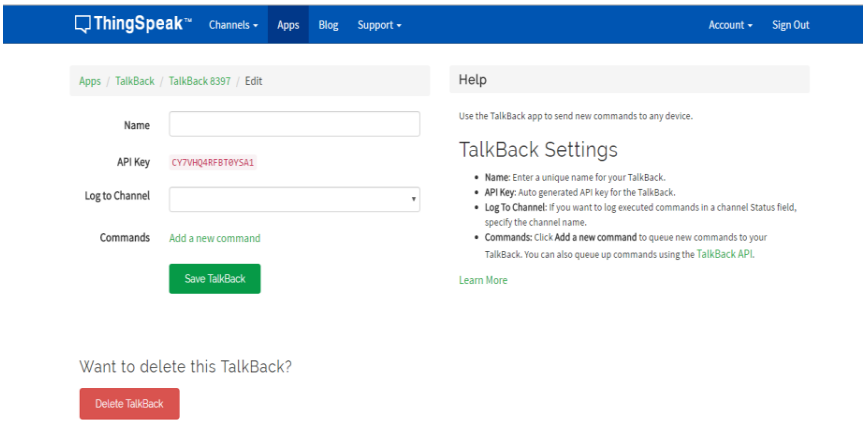

Figura 7 - TalkBack do ThingSpeak

Neste caso basta escolher o nome para este e o *channel* onde se quer fazer a ligação deste e como neste caso não queremos, não adicionamos novos comandos pois estes virão a posteriori através da página web a seguir.

9.4.3. Página Web

Para fazer a criação da página web que é usada para este projeto, escolhe-se a opção "*Apps*" depois escolhe-se a opção "*Plugins*" e começa se uma nova e obtém-se o que se pode ver na Figura 8.

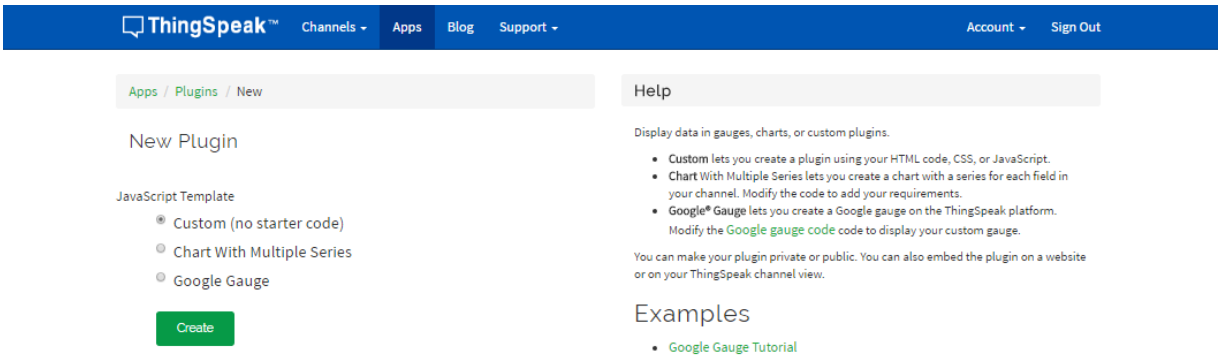

Figura 8 - Criação de um Plugin e escolha da Linguagem

Neste caso vai-se escolher a opção "*Custom(no starter code)*", para assim se poder fazer a página ao gosto do utilizador, por mais simples que seja. De seguida obtém-se o que se pode ver na Figura 9.

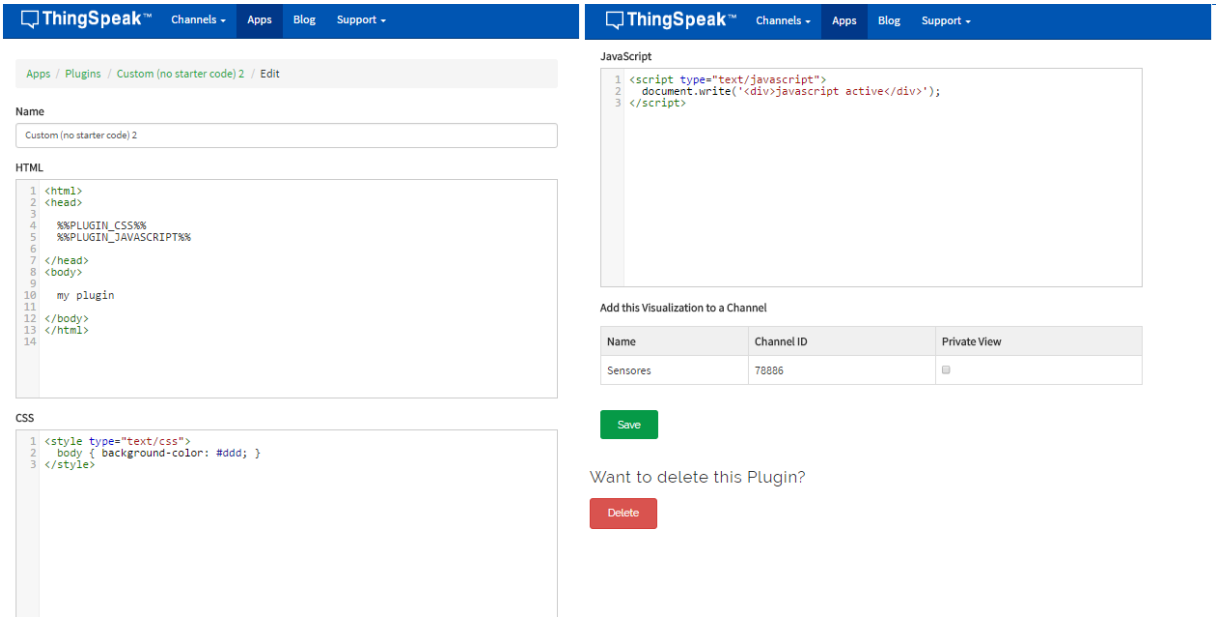

Figura 9 - Criação da Página Web

Para a criação da página web para o controlo da estufa, como se pode ver, pode-se usar Código HTML, CSS e JavaScript para a sua criação. No caso deste trabalho como só se quer algo muito simples o que se faz é uma página simples com um titulo e botões para o que se quer controlar assim fazendo com que o utilizador possa fazer todas as operações (Figura 10).

Figura 10 - Interface Final

CAPITULO 3 – IMPLEMENTAÇÃO DA SOLUÇÃO

3.1. Explicação da parte eletrónica do projeto

Neste capítulo vai ser explicado como são feitas as ligações entre o Arduíno e os sensores utilizados e a ligação para o modulo de Wi-fi. Também neste capítulo serão apresentados excertos de código de forma a demonstrar como é a comunicação feita entre o Arduíno e os sensores assim como a ligação ao ThingSpeak através do modulo de Wi-fi e a sua configuração.

A Figura 11, mostra a ligação feita do Arduíno ao sensor de Temperatura e ao Potenciómetro. Neste projeto é usado um sensor para ver a temperatura exterior e um potenciómetro para simular a temperatura interior de uma estufa a fim de ver se todas as condições estão corretas.

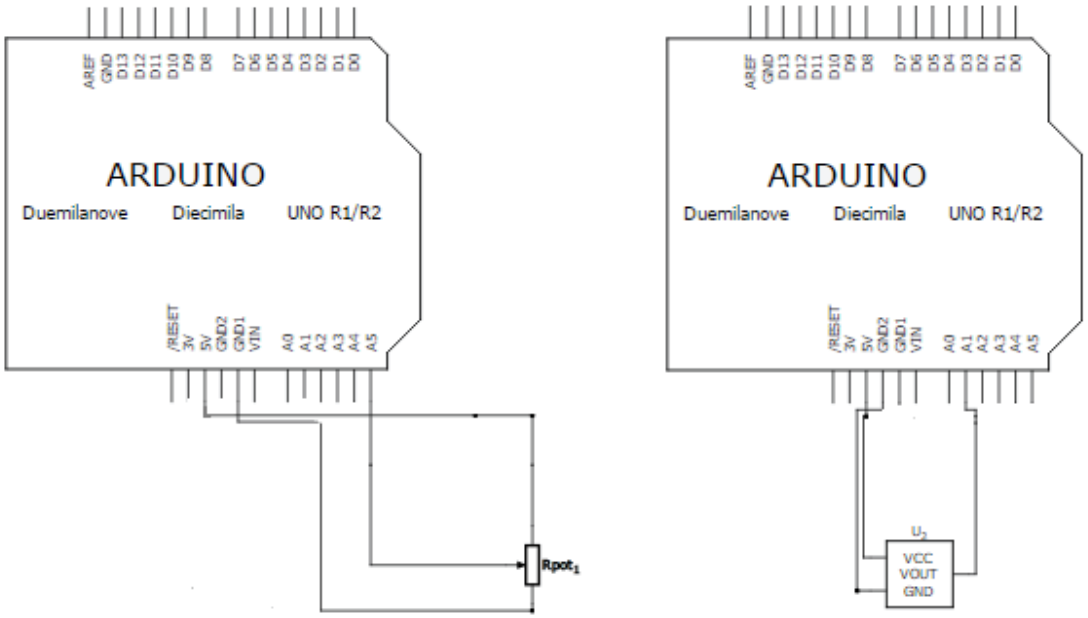

Figura 11 – Arduino, Sensor de Temperatura e Potenciómetro

Para verificar se a plantação necessita de rega utiliza-se o sensor de humidade (Figura 12), este vai fornecer ao utilizador a informação da humidade do solo onde se encontra a plantação, com este e possível fazer uma rega mais eficiente e assim as plantas tem um crescimento melhor.

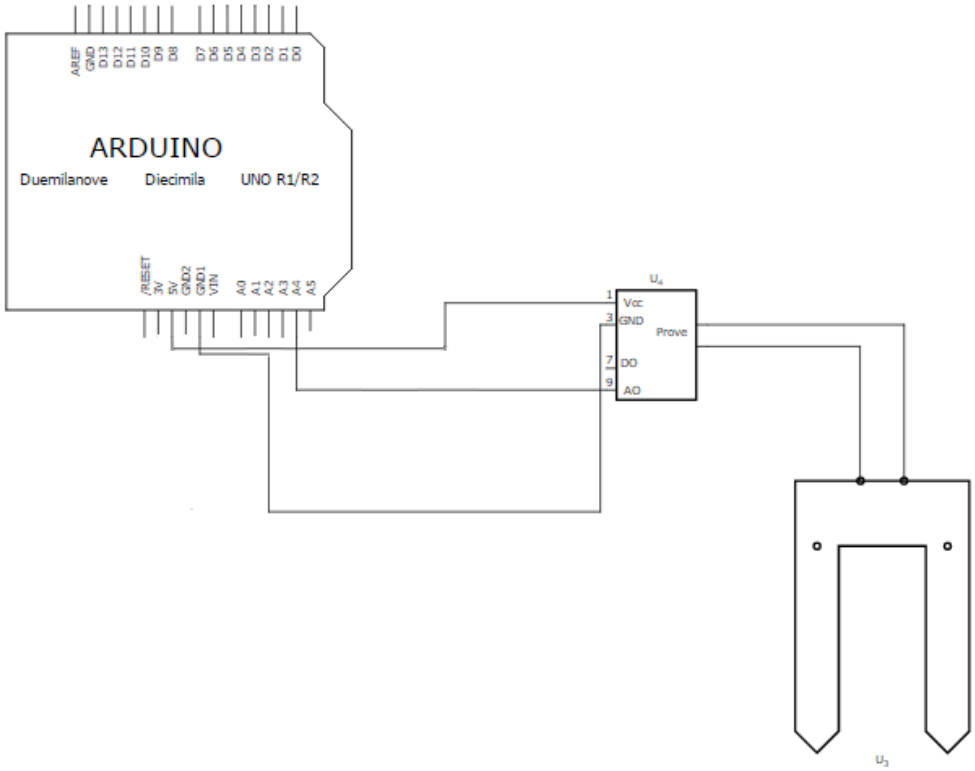

Figura 12 - Arduino e Sensor de Humidade

Para fazer a ligação do Arduíno ao modulo Wi-fi ou ESP8266, e feita da maneira indicada na Figura 13. Este permite ligar à internet e assim conseguir enviar os dados recebidos dos sensores já apresentados anteriormente e enviá-los para o site ThingSpeak.

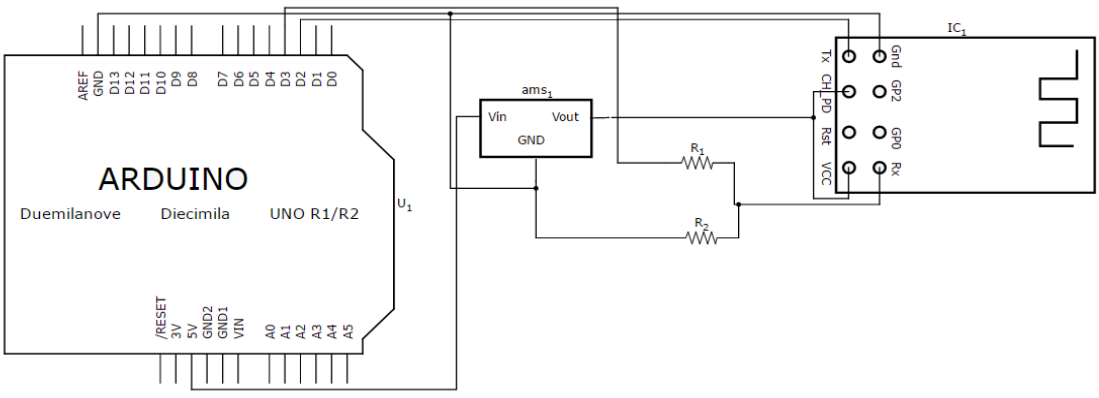

Figura 13- Arduino e Esp 8266

Este também permite receber comandos enviados do site ThingSpeak, como se pode verificar na página criada (Figura 14), com o fim de ligar e/ou desligar os atuadores que se quer, assim como receber a temperatura máxima e mínima que a estufa não deve ultrapassar.

Figura 14 - Pagina Web Criada

Por fim, em substituição dos atuadores, são usados neste projeto Leds para fazer a simulação dos mesmos, devido ao seu preço avultado. Os Leds, simulam um expressor, uma janela e a iluminação artificial. Na Figura 15 mostra-se como é simples as ligações dos Leds.

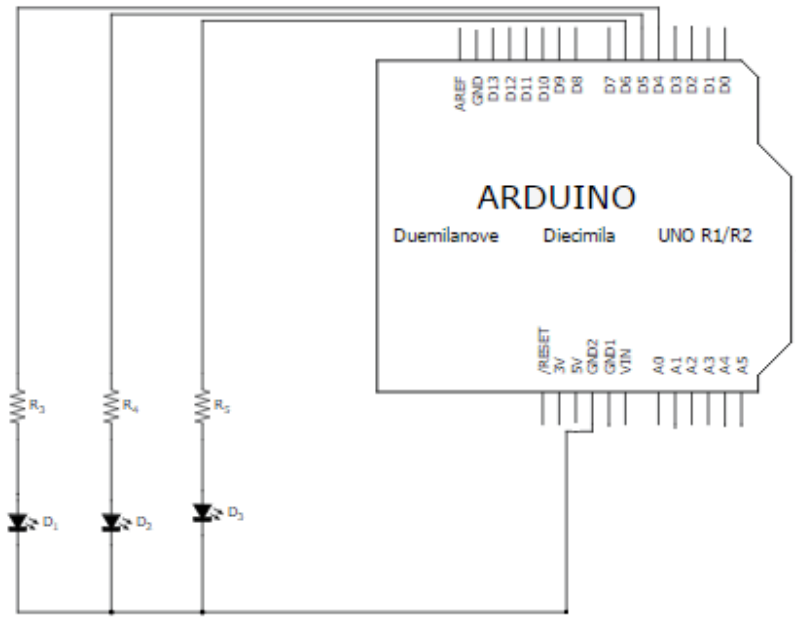

Figura 15 - Arduino e Led's

Por fim, na Figura 16 mostra o esquema do projeto todo.

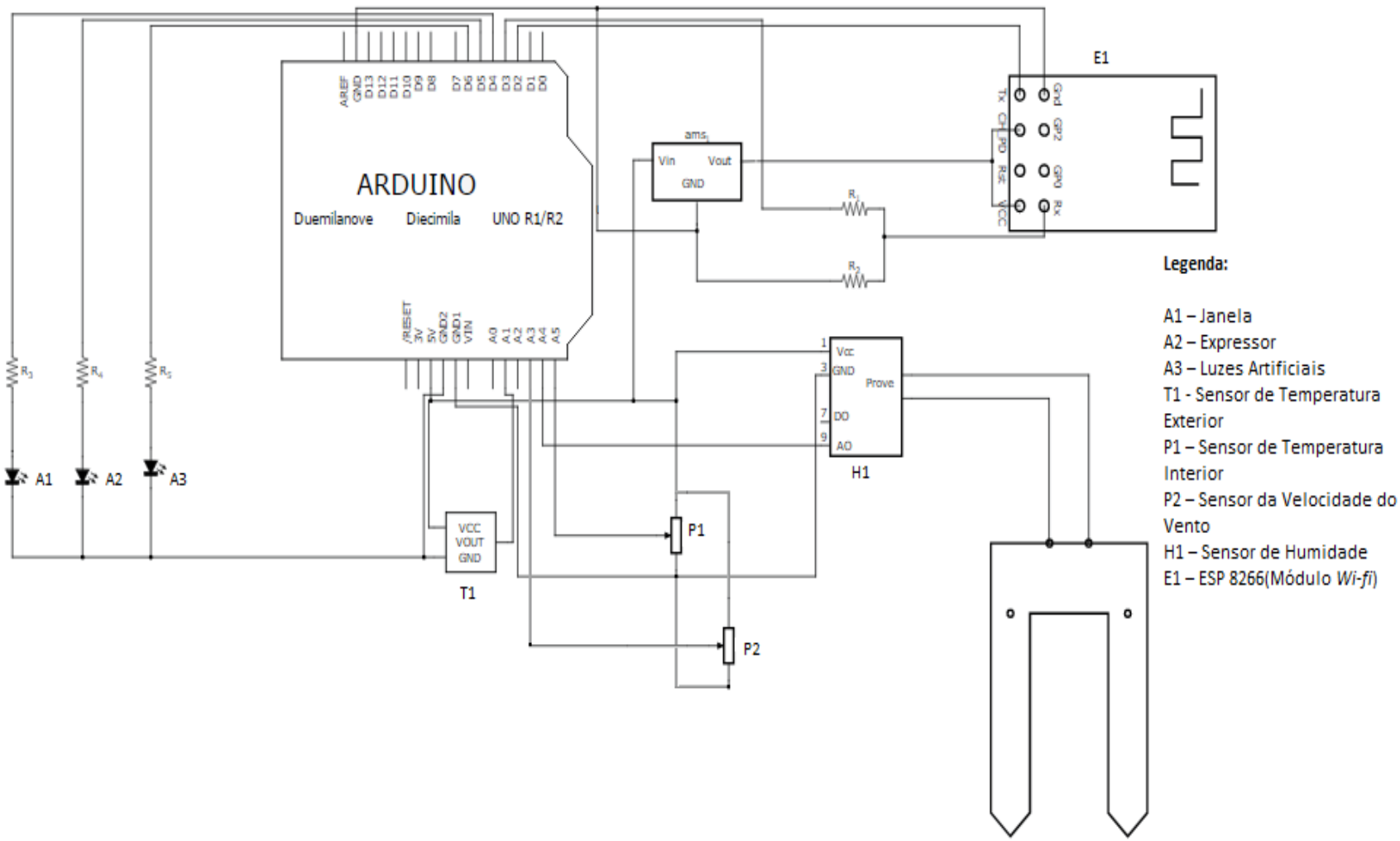

Figura 16 -Esquema Completo

3.2. Algoritmo do Programa de controlo de uma estufa

Apresenta-se de seguida o algoritmo usado para a resolução do problema proposto. Na Tabela 1 encontram-se as condições e como este algoritmo atua consoante as mesmas.

1. Função Setup()

1. Definir porta digital 4 como SAIDA (Luzes artificiais)
2. Definir porta digital 5 como SAIDA (Expressor)
3. Definir porta digital 6 como SAIDA(Janela)
4. Definir porta digital 8 como SAIDA (Sensor de humidade do solo)
5. Iniciar a porta serie como Baud Rate de 9600
6. Iniciar o software serial com Baud Rate de 9600 nas portas 2 e 3
7. Enviar para a software serial AT+RST
8. Definir no Esp a rede a utilizar
9. Definir no Esp o modo como Station
10. Definir no Esp para mostrar o ip da rede
11. Definir Esp para múltiplas conexões
12. Ler valores da EEPROM

2.Funçao loop()

1. Ler sensor de humidade
2. Ler potenciómetro
3. Fazer a média da velocidade do Vento
4. Ler a Temperatura de Fora
5. Ler a Temperatura de Dentro
6. Fazer a média da Temperatura de Fora
7. Fazer a ligação ao ThingSpeak
8. Se modo automático estiver ativo
   8.1. Processar modo automático
9. Esperar 20 segundos

No algoritmo acima no ponto 10, pode ser ver 20 segundos, este valor é usado para fazer testes para não se tornar longa a espera para se poder ver o que acontece, mas o ideal será usar o tempo de 60 segundos.

3.Função LigarAoThingSpeak()

1. Enviar o comando «AT+CIPSTART=4,"TCP", "184.106.153.149",80» para o esp
2. Se for encontrado "Error" no esp

2.1. Escrever AT+CIPSTART error
2.2. Terminar função

3. Definir comando a enviar «GET https://api.thingspeak.com/update?api_key=apiKey&field1=TemperaturaDentro&field2=TemperaturaFora&field3=humidade&field4=Vento&talkback_key=talkback_key HTTP/1.0\r\n\r\n»
3.1. Enviar comando «AT+CIPSEND=4, Tamanho do Comando no ponto 3», para o esp
3.2. Esperar 1 segundo
3.3. Se for encontrado no esp o caracter '>' então
3.3.1. Enviar comando do ponto 3
3.3.2. Escrever comando enviado
3.4. Enquanto Tempo + Tempo Procura for Maior que millis() fazer
3.4.1. Enquanto a serial estiver Available
3.4.1.1. Ler caracter e colocar numa variável
3.4.1.2. Se variável for igual a '$' então
3.4.1.2.1. Ler caracteres até ao fim da linha "\r" e colocar numa variável
3.4.1.2.2. Escrever variável recebida
3.4.1.2.3. Iniciar modo Manual
3.4.1.3. Se variável for igual a '<' então
3.4.1.3.1. Ler caracteres até ao fim da linha "\r" e colocar numa variável
3.4.1.3.2. Escrever variável recebida
3.4.1.3.3. Converter variável recebida para número
3.4.1.3.4. Guardar número na posição 1 da EEPROM
3.4.1.4. Se variável for igual a '>' então
3.4.1.4.1. Ler caracteres até ao fim da linha "\r" e colocar numa variável
3.4.1.4.2. Escrever variável recebida
3.4.1.4.3. Converter variável recebida para número
3.4.1.4.4. Guardar número na posição 0 da EEPROM
3.4.1.5. Se variável igual a '~' então
3.4.1.5.1. Ler caracteres até ao fim da linha "\r" e colocar numa variável
3.4.1.5.2. Se variável for igual a "Manual" então
3.4.1.5.3. Ligar Modo Manual e desligar Modo Automático
3.4.1.5.4. Se variável for igual a "Auto" então
3.4.1.5.4.1. Ligar Modo Automático e desligar Modo Manual
3.5. Enviar o comando «AT+CIPCLOSE» para o esp

4. Função Manual

1. Colocar o comando recebido numa variável
2. Se variável igual a "ON1" então
2.1. Ligar LED1
3. Se variável igual a "ON2" então

3.1. Ligar LED2
4. Se variável igual a "ON3" então
   4.1. Ligar LED3
5. Se variável igual a "OFF1" então
   5.1. Desligar LED1
6. Se variável igual a "OFF2" então
   6.1. Desligar LED2
7. Se variável igual a "OFF3" então
   7.1. Desligar LED3

O utilizador do sistema com esta função pode ligar os atuadores que pretender, a rega, abrir a janela ou ligar as luzes artificiais. Como o *thingspeak* só envia um comando de cada vez, se o utilizador pretende-se ativar vários atuadores duma só vez, este sistema irá ligar um de cada vez consoante recebe o comando, tendo assim que esperar para ligar o próximo.

5. Função AUTO()

1. Fazer VlmI = ValorLimiteInf + 2
2. Fazer VlmS = ValorLimiteSup - 1
3. Se modo automático ligado então
   3.1. Se MediaVento for maior que 600 então
      3.1.1. Fechar Janelas
      3.1.2. Janela é falso
   3.2. Se MediaVento for menor que 550 então
      3.2.1. Janela é Verdadeiro
4. Se millis() menos TempoTemp for maior que IntTemperatura então
   4.1. TempoTemp vai ser igual ao millis
   4.2. Se ValorTempD for menor ou igual que ValorLimiteInf
      4.2.1. Ligar Luzes Artificiais
      4.2.2. Se ValorTempF maior ou igual que VlmI
         4.2.2.1. Se Temperatura a subir então
            4.2.2.1.1. Se Janela Verdadeiro então
               4.2.2.1.1.1. Abrir Janela
         4.2.2.2. Se não Se Temperatura a descer então
            4.2.2.2.1. Fechar Janela
   4.3. Se ValorTempD for maior ou igual que VlmS
      4.3.1. Desligar Luzes Artificiais
      4.3.2. Fechar Janelas
5. Se millis() menos Tempo humidade for maior que IntHumidade então
   5.1. TempoHumidade vai ser igual ao millis
   5.2. Se humidade for menor ou igual que 500 então

5.2.1. Ligar expressor

Com o modo acima obtemos uma resposta dos atuadores como se pode observar na Tabela 2. Este modo consegue responder a qualquer situação, adaptando-se ao ambiente dentro da estufa.

Tabela de Condições							
Temperatura		Vento (Lido do Sensor)	humidade Solo (Lido do Sensor)		Torneira	Janela	Luzes Artificiais
Dentro (°C)	Fora (°C)						
15	17	600	650		Desligado	Desligado	Desligado
10	18	100	150		Ligado	Ligado	Ligado
25	25	150	600		Desligado	Desligado	Desligado
16	8	200	550		Desligado	Desligado	Ligado
25	20	200	450		Ligado	Desligado	Desligado
18	20	500	650		Desligado	Ligado	Ligado

Tabela 2 - Tabela de Condições

6. Função ConfigurarESP ()

1. Colocar valor de millis () numa variável
2. Enviar comando recebido para esp
3. Criar variável para guardar a resposta
4. Enquanto variável (ponto 1) mais Espera for maior que millis ()

4.1. Enquanto a software serial for Available

4.1.1. Ler Caracter e colocar numa variável

4.1.2. Adicionar à variável (ponto 3) a variável (ponto 4.1.1.)

4.1.3. Escrever a resposta (ponto 3)

A função apresentada anteriormente é utilizada para enviar dados para o ESP (Modulo *Wi-fi*) para fazer a configuração do mesmo. Esta recebe comando de configuração, após enviado esta recebe uma resposta.

7. Função Temperatura Fora

1. Ler valor do Sensor de Temperatura
2. Converter o valor lido em milivolt
3. Converter para graus Celsius
4. Colocar a temperatura em graus celsius numa variável

8. Função Temperatura Dentro

1. Ler o Valor do Potenciómetro
2. Converter o valor para a escala de leitura do sensor de Temperatura
3. Converter o valor para milivolt
4. Converter o valor em milivolt para graus celsius
5. Colocar a temperatura em graus celsius numa variável

9. Sensor de Humidade

1. Ligar Sensor de humidade
2. Esperar 10 segundos
3. Ler o sensor de humidade
4. Fazer 1023 - Valor humidade lido
5. Se Valor humidade lido menor ou igual a 300 então
   5.1. Escrever Solo Seco
6. Se Valor humidade lido maior que 300 e menor ou igual a 700 então
   6.1. Escrever Pouca Humidade
7. Se Valor humidade lido que 700 e menor ou igual a 950 então
   7.1. Escrever Bastante Humidade
8. Se Valor humidade lido maior que 950 então
   8.1. Escrever Solo Submerso

10.Função EEPROM

1. Ler a posição 0 da Eeprom
2. Ler a posição 1 da Eeprom
3. Se primeiro endereço for diferente de 0
   3.1. Colocar numa variável que representa o Limite inferior
4. Se segundo endereço for diferente de 0
   4.1. Colocar numa variável que representa do Limite Superior

Os limites aqui apresentados nesta função, são usados para delimitar a temperatura do interior da estufa.

11. Função MediaVento

1. Colocar o valor lido do potenciómetro numa variável
2. Se a variável for menor que Valor lido do Potenciómetro
   2.1. Fazer MediaVento = MediaVento * 0.99 + Varivel * 0.01
3. Se não variável for maior que Valor lido do Potenciómetro
   3.1. Colocar valor lido do potenciómetro na variável

12. Função MediaTemperatura

1. Se a Media igual a 0 então
   1.1. Media é igual a Valor Sensor de Temperatura
2. Se não
   2.1. Media = Media * 0.9 + Valor Sensor de Temperatura * (1 - 0.9)

Nesta função pode se tirar que quando o valor do sensor for menor que a Média aqui calculado então significa que a temperatura se encontra a subir e quando o valor do sensor for maior que a Média a temperatura se encontra a descer.

13.Função Limite

1. Se variável for para guardar na posição 1 então
   1.1. Colocar valor recebido numa variável representada pelo Limite Superior
   1.2. Gravar na posição 1 da Eeprom
2. Se variável for para guardar na posição 0 então
   2.1. Colocar valor recebido numa variável representada pelo Limite Inferior
   2.2. Gravar na posição 0 da Eeprom

Esta função recebe os valores da temperatura recebidos do site *ThingSpeak*, estes são guardados na *Eeprom*, e colocados em variáveis diferentes para serem usadas delimitando uma temperatura máxima e uma temperatura mínima.

3.3. Explicação de Partes do Código

Sensor de Temperatura Fora

Para se obter a temperatura em Graus através do sensor é apenas possível após algumas conversões. Em primeiro é preciso dizer que este sensor trabalha com baixa voltagem, dos 2.7 Volt até 5.5 Volt, ou seja, a temperatura vai deste os -40 °C ate aos 125 °C, este sensor tem uma precisão de +- 2 °C.

Ao se ler o sensor de temperatura este vai dar um valor que esta entre 0 a 1023 pois esta é a gama de valores lidos na porta analógica do Arduíno. O valor lido tem de se converter para mili Volt, onde torna-se mais fácil a conversão, para Graus Celsius, a escala deste é por cada 10 mili Volt é 1°C.

Mili Volt = (Valor Lido do Sensor * Voltagem) / 1024

A Voltagem aqui usada é 5Volts = 5000 mili Volt.

Após obter o valor em milivolt faz-se

Graus Celsius = (Mili volt - *offset*) * 100

O *offset* é 0.5, o valor do especifico que pode ser visto na *datashit* (Anexo B) do sensor usado.

Assim obtém-se o valor em graus pretendido.

Sensor de Temperatura Dentro

Após uma cuidada análise e alguns testes aos valores lidos do sensor de temperatura, para obter o valor em graus através do potenciómetro, tem de se ter atenção os cálculos feitos no sensor de temperatura. Quando a temperatura está nos 120 °C o valor lido no potenciómetro é de 358.5, o valor quando está nos 0 °C é de 105.4. Assim sabe-se que os valores da temperatura que se quer ler esta compreendido entre estes dois valores e assim basta fazer o seguinte:

Em primeiro é preciso ler o potenciómetro, de seguida usando uma simples regra de três simples obtém-se

Valor Pretendido = (Valor lido * 358.5) / 1024

Após isto basta repetir os passos para o calculo do sensor de temperatura, ou seja,

Milivolt = (Valor Pretendido * Voltagem) / 1024

Graus Celsius = (Milivolt - offset) * 100

Assim obtendo o valor em graus que se pretende.

CAPITULO 4 - CONCLUSÃO

Com este projeto é possível concluir-se que existe a possibilidade de criação de um sistema autónomo, que visa ajudar os agricultores que tenham estufas a terem um maior controlo de todos os parâmetros importantes dessas estufas.

Este sistema usa um microcontrolador (Arduíno), sensores para detetar a temperatura, a humidade e a velocidade do vento, um modulo *Wi-fi* e um servidor web.

Destaca-se a possibilidade de o utilizador poder monitorizar na sua estufa minuto a minuto informando-o das condições climatéricas existentes de forma a que possa intervir sempre que achar necessário. Se o utilizador preferir pode ligar o modo automático que possibilita a este sistema adaptar-se às condições existentes e intervindo quando é necessário.

Com o sistema apresentado existe a possibilidade de ter uma estufa viável para a produção de grande qualidade, quer seja usado para um negócio ou para uso próprio, tendo assim maior aproveitamento da plantação.

Após a conclusão deste, verifica-se que este protótipo tem tudo para ser implementado numa estufa real, visto ter um custo reduzido e não só poder trazer benefícios económicos e de bem-estar para o seu proprietário, mas também para o meio ambiente pois, racionaliza o uso de um bem escasso como a água.

BIBLIOGRAFIA

Referências

Arduino, c. (s.d.). *Arduino cc.* Obtido de Arduino cc: https://www.arduino.cc

ESTUFASMINHO, S. (12 de Julho de 2016). *EM, S.A.* Obtido de http://www.estufasminho.pt/

HortaNova. (16 de 07 de 2016). *HortaNova.* Obtido de HortaNova: http://www.hortanova.pt

Prilux. (17 de 7 de 2016). *PRILUX.* Obtido de PRILUX: http://www.prilux.pt

Anexo A

Código Do Projeto

Código do Projeto

```
#include <SoftwareSerial.h>

#include <EEPROM.h>


SoftwareSerial ser(2, 3);

//Modos

boolean autom = true;

boolean manu = false;

boolean Janela = true;

boolean Temp = true;


// Atuadores

// Janela, expressor, luzes artificiais

#define LED1 4

#define LED2 5

#define LED3 6


// Sensores Usados Neste Projeto

#define SensorHumidade A4

#define SHativa 8

#define SensorTemperaturaFora A1

//#define SensorTemperaturaDentro A1

#define SensorVento A5


// Variáveis globais

// dos sensores para o programa todo

int   ValorHumidade = 0;
```

```
float ValorTempF = 0;
float ValorTempD = 0;
int   Vento = 0;

//Codigo de Ligação ao ThingSpeak
String apiKey = "JPAWNFZPYBGY8EHY";
String TalkbackK = "0H8XVB9HP2PHX0ZC";

// Tempo Para Receber o Talkback
long TempoProcuraM = 1000; //em milisegundos
//Prever temperatura a subir ou a descer
float Media ;
float TemperaturaAtual;
float VentoM ;
//Valores para os limites
float ValorLimiteInf = 12.00;
float ValorLimiteSup = 25.00;
float VlmI ;
float VlmS ;
//Tempo anterior
long TempoTemp = 0;
long TempoHumidade = 0;
//Intervalo de tempo
long IntTemperatura = 1000;
long IntHumidade = 1000;
//Comando recebido
String cmmmd = "";
```

```
char argv[] = "";
//Guardar valores recebidos para os limites
int ji ;
int js ;
unsigned long Tempo = millis();

void setup() {
  //Definir os Led's, para os se poderem manipular
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  //Poder ativar o sensor de umidade
  pinMode(SHativa, OUTPUT);

  // permite fazer o debug da serial
  Serial.begin(9600);
  // permite fazer o debug do software serial
  ser.begin(9600);

  // faz reset ESP8266
  ser.println("AT+RST");
  //Liga a Rede Aqui definida
  ConfigurarESP("AT+CWJAP=\"Karma\",\"123456789polo**\"\r\n", 10000);
  //X
  ConfigurarESP("AT+CWMODE01\r\n", 1000);
  //MOSTRAT IP
  ConfigurarESP("AT+CIFSR\r\n", 1000);
```

```
  //configura para multiplas conexoes
  ConfigurarESP("AT+CIPMUX=1\r\n", 1000);
  //Buscar valores dos limites se possivel
  Eeprom();
}

void loop() {
  //Ver Sensor de Humidade
  LigarSHumidade();
  Serial.print(ValorHumidade);      //Para Debug
  Serial.println();
  //Vento Sensor do vento  (Potenciometro)
  Vento = analogRead(SensorVento);
  Serial.print(Vento);            // Para Debug
  Serial.println();
  MediaVento();
  //TEMP FORA
  TemperaturaFora();
  //Temp dentro
  TemperaturaDentro();
  MediaTemperatura();
  Serial.println();
  LigarAoThingSpeak();
  Auto();
  delay(20000);
}
```

```
void LigarAoThingSpeak() {
  //TCP connection
  String cmd = "AT+CIPSTART=4,\"TCP\",\"";
  cmd += "184.106.153.149"; // api.thingspeak.com
  cmd += "\",80";
  ser.println(cmd);

  if (ser.find("Error")) {
   Serial.println("AT+CIPSTART error");
   return;
  }

  //STRING PARA ATUALIZAR CHANNEL E RECEBER O TALKBACK
  String getStr = "GET https://api.thingspeak.com/update?api_key=";
  getStr += apiKey;
  getStr += "&field1=";
  getStr += ValorTempD;
  getStr += "&field2=";
  getStr += ValorTempF;
  getStr += "&field3=";
  getStr += ValorHumidade;
  getStr += "&field4=";
  getStr += Vento;
  getStr += "&talkback_key=";
  getStr += TalkbackK;
  getStr += " HTTP/1.0\r\n\r\n";
```

```
// send data length
cmd = "AT+CIPSEND=4,";
cmd += String(getStr.length());
ser.println(cmd);

delay(1000);

if (ser.find(">")) {
  ser.print(getStr);
  Serial.print(getStr);
}

while ( (Tempo + TempoProcuraM) > millis()) {

  while (ser.available()) {

    char a = ser.read();

    if ( a == '$') {

      cmmmd = ser.readStringUntil('\r');
      Serial.println(cmmmd);
      if (manu) {
        Inicia(cmmmd);
      }
    }
    if ( a == '<') {
```

```
  cmmmd = ser.readStringUntil('\r');
  Serial.println(cmmmd);
  char buf[cmmmd.length()];
  cmmmd.toCharArray(buf, cmmmd.length());
  //Converter para int
  ji = atoi(buf);
  Serial.println("Limite Inferior Alterado: ");
  Serial.print(ji);
  int i = 0;
  Limite(ji, i);
}
if ( a == '>') {
  cmmmd = ser.readStringUntil('\r');
  Serial.println(cmmmd);
  char buf2[cmmmd.length()];
  cmmmd.toCharArray(buf2, cmmmd.length());
  //Converter para Int
  js = atoi(buf2);
  Serial.println("Limite Superior Alterado: ");
  Serial.print(js);
  int i = 0;
  Limite(i, js);

}
if ( a == '~') {
  cmmmd = ser.readStringUntil('\r');
  Serial.println(cmmmd);
```

```
      if (cmmmd == "Manual") {
        manu = true;
        autom = false;
      }
      if (cmmmd == "Auto") {
        manu = false;
        autom = true;
      }
     }

   }
  }

  Serial.println();
  ser.println("AT+CIPCLOSE");
  // alert user
  Serial.println("AT+CIPCLOSE");
}

String Inicia (String Comando) {
  String Cmd = Comando;
  if (manu) {
    Serial.println("Aguarde..");
    if (Cmd == "ON1") {
      digitalWrite(LED1, HIGH);
    }
    if (Cmd == "ON2") {
```

```
      digitalWrite(LED2, HIGH);
    }
    if (Cmd == "ON3") {
      digitalWrite(LED3, HIGH);
    }
    if (Cmd == "OFF1") {
      digitalWrite(LED1, LOW);
    }
    if (Cmd == "OFF2") {
      digitalWrite(LED2, LOW);
    }
    if (Cmd == "OFF3") {
      digitalWrite(LED3, LOW);
    }
  }
}

void Auto() {
  if (autom) {
    Serial.println("A iniciar Modo Automatico....");
    VlmI = ValorLimiteInf + 2;
    VlmS = ValorLimiteSup - 1;
    if (VentoM < 500.00) {
      digitalWrite(LED3, LOW);
      Janela = true;
      Serial.println("Abrir Janelas");
    } else if (VentoM > 550.00) {
```

```
  Serial.println("Fechar Janela!!");
  Janela = false;
}
if ((millis() - TempoTemp) > IntTemperatura) {
  Serial.println("A ver Temperatura...");
  TempoTemp = millis();
  if (ValorTempD < ValorLimiteInf) {
    digitalWrite(LED3, HIGH);
    if (ValorTempF > VlmI) {
      if (Temp) {
        if (Janela) {
          digitalWrite(LED1, HIGH);
        }
      } else {
        digitalWrite(LED1, LOW);
      }
    }
  } else if ( ValorTempD > VlmS) {
    digitalWrite(LED1, LOW);
    digitalWrite(LED3, LOW);
    Serial.println("Janela e Luzes Desligados...");
  }
}

if ((millis() - TempoHumidade) > IntHumidade) {
  Serial.println("A ver Humidade...");
  TempoHumidade = millis();
```

```
    if (ValorHumidade < 500) {
      digitalWrite(LED2, HIGH);
    } else if (ValorHumidade >= 500) {
      digitalWrite(LED2, LOW);
    }
   }

  }
}

String ConfigurarESP(String Comando, const int Tespera) {
  String resposta = "";
  long int TMP = millis();
  ser.print(Comando);
  while ( (TMP + Tespera) > millis()) {

    while (ser.available()) {
      char h = ser.read();
      resposta += h;
    }
  }
  Serial.println(resposta);
  return resposta;
}

void TemperaturaFora() {
  int volts = analogRead(SensorTemperaturaFora);
```

```
  volts = analogRead(SensorTemperaturaFora);
  volts = analogRead(SensorTemperaturaFora);
  Serial.println(volts);
  float volt = ((volts * 5.0) / 1024.0);
  float Celsius = (volt - 0.5) * 100;
  ValorTempF = Celsius;
  Serial.print("Valor Temperatura de Fora.:");
  Serial.print(ValorTempF);
  Serial.print("\n");
}

void TemperaturaDentro() {
  int tmp = analogRead(SensorVento);
  tmp = analogRead(SensorVento);
  tmp = analogRead(SensorVento);
  Serial.println(tmp);
  float tmp36 = (tmp * 358.5) / 1024.0;
  float volt = (tmp36 * 5.0);
  volt = volt / 1024.0;
  float Celsius = (volt - 0.5) * 100.0;
  ValorTempD = Celsius;
  Serial.print("Valor Temperatura de Dentro.:");
  Serial.print(ValorTempD);
  Serial.print("\n");
}

void LigarSHumidade() {
```

```
 digitalWrite(SHativa, HIGH);
 delay(100);
 int VH = analogRead(SensorHumidade);
 ValorHumidade = 1023 - VH;

 if (ValorHumidade <= 300) {
  Serial.print("Sem Humidade / Solo Seco, Valor Humidade.:");
  Serial.print(ValorHumidade);
  Serial.println();

 }
 if ((ValorHumidade > 300) && (ValorHumidade <= 700)) {
  Serial.print("Pouca Humidade, Valor Humidade.:");
  Serial.print(ValorHumidade);
  Serial.println();
 }
 if ((ValorHumidade > 700) && (ValorHumidade <= 950)) {
  Serial.print("Bastante Humidade, Valor Humidade.:");
  Serial.print(ValorHumidade);
  Serial.println();
 }
 if (ValorHumidade > 950) {
  Serial.print("Demasiada Agua / Solo Submerso, Valor Humidade.:");
  Serial.print(ValorHumidade);
  Serial.println();
 }
}
```

```
void MediaTemperatura() {
  // Prever a subida de temperatura
  if (Media == 0) {
    Media = ValorTempF;
  } else {
    Media = Media * 0.9 + ValorTempF * (1 - 0.9);
  }
  if (ValorTempF > Media) {
    Temp = true;
    Serial.print("Temperatura a subir... A media.:");
    Serial.print(Media);
    Serial.println();
  }
  if (ValorTempF < Media) {
    Temp = false;
    Serial.println("Temperatura a descer... A media.:");
    Serial.print(Media);
    Serial.println();
  }
}

void MediaVento() {
//Prever a velocidade do vento
  if (VentoM == 0) {
    VentoM =  Vento;
  } else  {
```

```
    VentoM = VentoM * 0.99 +  Vento * (1 - 0.99);
  }
  Serial.print("Media do Vento.:");
  Serial.print(VentoM);
  Serial.println();
}

void Limite(int li, int ls) {
  if (li == 0) {
    //Se o limite inferir for 0 entao o valor que temos e o superior
    ValorLimiteSup =ls;
    EEPROM.write(1, ls);
    Serial.println("Gravado na memória...");
  }
  if (ls == 0) {
    // Se o limite superior for 0 entao o valor que temos e o inferior
    ValorLimiteInf = li;
    EEPROM.write(0, li);
    Serial.println("Gravado na memória...");
  }
}

void Eeprom() {
  int var1 = EEPROM.read(0);
  int var2 = EEPROM.read(1);
  Serial.println("Valor Lidos da memoria...");
  if (var1 != 0) {
```

```
    ValorLimiteInf = var1 * 1.0;
    Serial.print("Limite Inferior.:");
    Serial.print(var1);
    Serial.println();
   }
   if (var2 != 0) {
    ValorLimiteSup =var2 * 1.0;
    Serial.print("Limite Superior.:");
    Serial.print(var2);
    Serial.println();
   }
}
```

Anexo B

Ficha Técnica do Sensor

TMP 35,36,37

Low Voltage Temperature Sensors

Data Sheet

TMP35/TMP36/TMP37

FEATURES

Low voltage operation (2.7 V to 5.5 V)
Calibrated directly in °C
10 mV/°C scale factor (20 mV/°C on TMP37)
±2°C accuracy over temperature (typ)
±0.5°C linearity (typ)
Stable with large capacitive loads
Specified −40°C to +125°C, operation to +150°C
Less than 50 µA quiescent current
Shutdown current 0.5 µA max
Low self-heating
Qualified for automotive applications

APPLICATIONS

Environmental control systems
Thermal protection
Industrial process control
Fire alarms
Power system monitors
CPU thermal management

GENERAL DESCRIPTION

The TMP35/TMP36/TMP37 are low voltage, precision centigrade temperature sensors. They provide a voltage output that is linearly proportional to the Celsius (centigrade) temperature. The TMP35/TMP36/TMP37 do not require any external calibration to provide typical accuracies of ±1°C at +25°C and ±2°C over the −40°C to +125°C temperature range.

The low output impedance of the TMP35/TMP36/TMP37 and its linear output and precise calibration simplify interfacing to temperature control circuitry and ADCs. All three devices are intended for single-supply operation from 2.7 V to 5.5 V maximum. The supply current runs well below 50 µA, providing very low self-heating—less than 0.1°C in still air. In addition, a shutdown function is provided to cut the supply current to less than 0.5 µA.

The TMP35 is functionally compatible with the LM35/LM45 and provides a 250 mV output at 25°C. The TMP35 reads temperatures from 10°C to 125°C. The TMP36 is specified from −40°C to +125°C, provides a 750 mV output at 25°C, and operates to 125°C from a single 2.7 V supply. The TMP36 is functionally compatible with the LM50. Both the TMP35 and TMP36 have an output scale factor of 10 mV/°C.

FUNCTIONAL BLOCK DIAGRAM

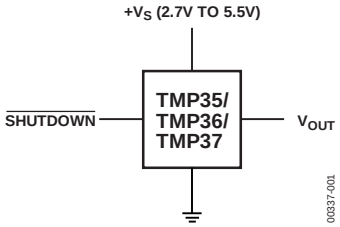

Figure 1.

PIN CONFIGURATIONS

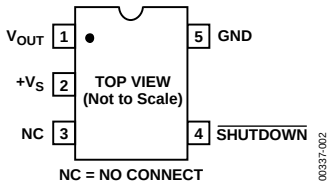

Figure 2. RJ-5 (SOT-23)

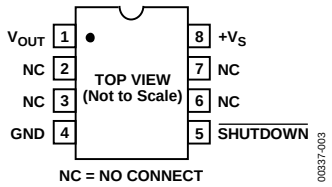

Figure 3. R-8 (SOIC_N)

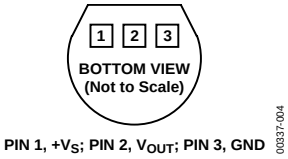

Figure 4. T-3 (TO-92)

The TMP37 is intended for applications over the range of 5°C to 100°C and provides an output scale factor of 20 mV/°C. The TMP37 provides a 500 mV output at 25°C. Operation extends to 150°C with reduced accuracy for all devices when operating from a 5 V supply.

The TMP35/TMP36/TMP37 are available in low cost 3-lead TO-92, 8-lead SOIC_N, and 5-lead SOT-23 surface-mount packages.

Rev. H **Document Feedback**

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700
Technical Support **www.analog.com**

FUNCTIONAL DESCRIPTION

An equivalent circuit for the TMP35/TMP36/TMP37 micropower, centigrade temperature sensors is shown in Figure 22. The core of the temperature sensor is a band gap core that comprises transistors Q1 and Q2, biased by Q3 to approximately 8 μA. The band gap core operates both Q1 and Q2 at the same collector current level; however, because the emitter area of Q1 is 10 times that of Q2, the V_{BE} of Q1 and the V_{BE} of Q2 are not equal by the following relationship:

$$\Delta V_{BE} = V_T \times \ln\left(\frac{A_{E,Q1}}{A_{E,Q2}}\right)$$

Resistors R1 and R2 are used to scale this result to produce the output voltage transfer characteristic of each temperature sensor and, simultaneously, R2 and R3 are used to scale the V_{BE} of Q1 as an offset term in V_{OUT}. Table 4 summarizes the differences in the output characteristics of the three temperature sensors.

The output voltage of the temperature sensor is available at the emitter of Q4, which buffers the band gap core and provides load current drive. The current gain of Q4, working with the available base current drive from the previous stage, sets the short-circuit current limit of these devices to 250 μA.

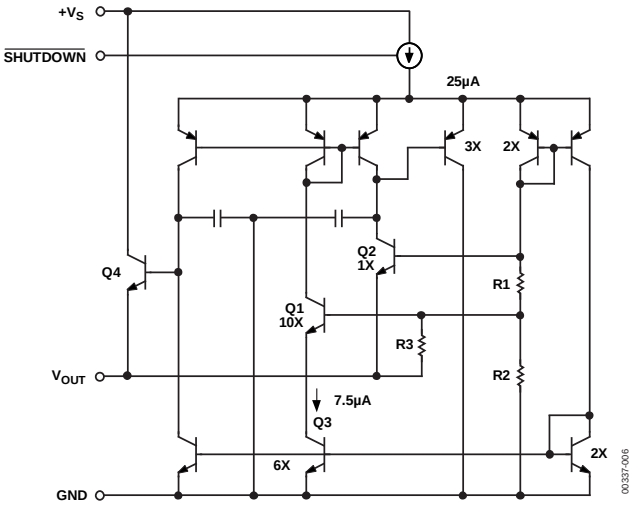

Figure 22. Temperature Sensor Simplified Equivalent Circuit

Table 4. TMP35/TMP36/TMP37 Output Characteristics

Sensor	Offset Voltage (V)	Output Voltage Scaling (mV/°C)	Output Voltage at 25°C (mV)
TMP35	0	10	250
TMP36	0.5	10	750
TMP37	0	20	500