

Mestrado em Engenharia Eletrónica

Unidade Curricular de Sistemas Embebidos

Relatório do projeto

*IOT Smart Home*



Docente: Prof. Rui Manuel Rodrigues Rocha

Discentes: André Oliveira, nº 83873

Markiyan Pyekh, nº 96959

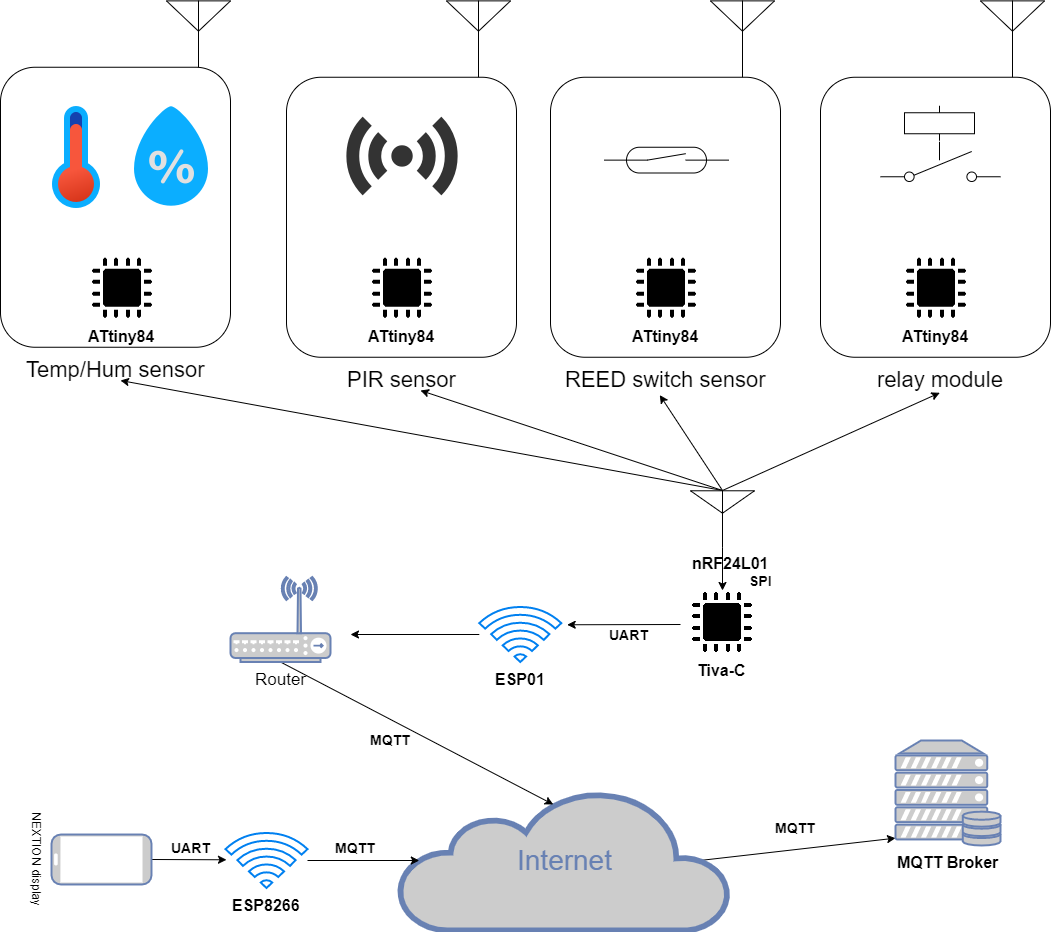
Dezembro 2019

Índice

Índice das imagens

Introdução

O presente projeto foi elaborado no âmbito da Unidade Curricular (UC) de Sistemas Embebidos integrada no plano curricular do Mestrado em Engenharia Eletrónica (MEE). Neste projeto pretendemos desenhar e desenvolver um sistema embebido com o principal propósito de monitorização e controlo de parâmetros de uma casa/quarto ou outro espaço de convivência ou trabalho. A ideia principal consiste na existência de um nó roteador de controlo (Gateway) e vários nós secundários: uns realizam o trabalho de atuadores (relé), e outros de recolha de informação (sensores de temperatura, humidade, estado das portas/janelas etc.). Iremos utilizar Broker MQTT gratuito disponibilizado pelo serviço on-line chamado CloudMQTT. No esquema abaixo podemos observar a visão geral do sistema.

 As perspetivas futuras do presente projeto visam a implementação de uma interface gráfica WEB, e ainda uma versão para dispositivos móveis. Para além disso, também é possível adicionar novas funcionalidades ao nível de software, nomeadamente execução de tarefas consoante certos acontecimentos prévios. O exemplo de uma destas funcionalidades seria o acendimento automático das luzes como resultado da leitura do parâmetro medido pelo sensor de luminosidade.

Requisitos

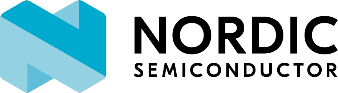
A elaboração do presente sistema pretende cumprir requisitos de dois tipos: funcionais e não funcionais.

Quanto aos requisitos funcionais, o sistema:

* Deve ser capaz de medir a temperatura e humidade;
* Deve adquirir a informação com periodicidade de 7 segundos;
* Deve ser capaz de detetar e reportar o movimento de forma imediata;
* Deve ser capaz de monitorizar e reportar de forma imediata o estado de uma porta/janela;
* Deve disponibilizar um GUI portátil que disponibilize a seguintes possibilidades:
  + Visualizar os parâmetros dos sensores acima mencionados;
  + Controlar um atuador(relé);
  + Configurar o ponto de acesso (AP) e dados do servidor (MQTT Broker);
* Deve utilizar o protocolo MQTT na comunicação entre o Gateway e o Broker;
* Deve implementar o protocolo de comunicação por comandos AT entre o Gateway e o modulo Wi-Fi;
* Deve implementar um sistema operacional em tempo real (RTOS), em particular o FreeRTOS;
* Deve garantir que a informação transmitida pelos nós
* Não pode perder informação destinada à um certo nó, quando este encontra-se offline;

Por outro lado, para obedecer aos requisitos não funcionais, o sistema:

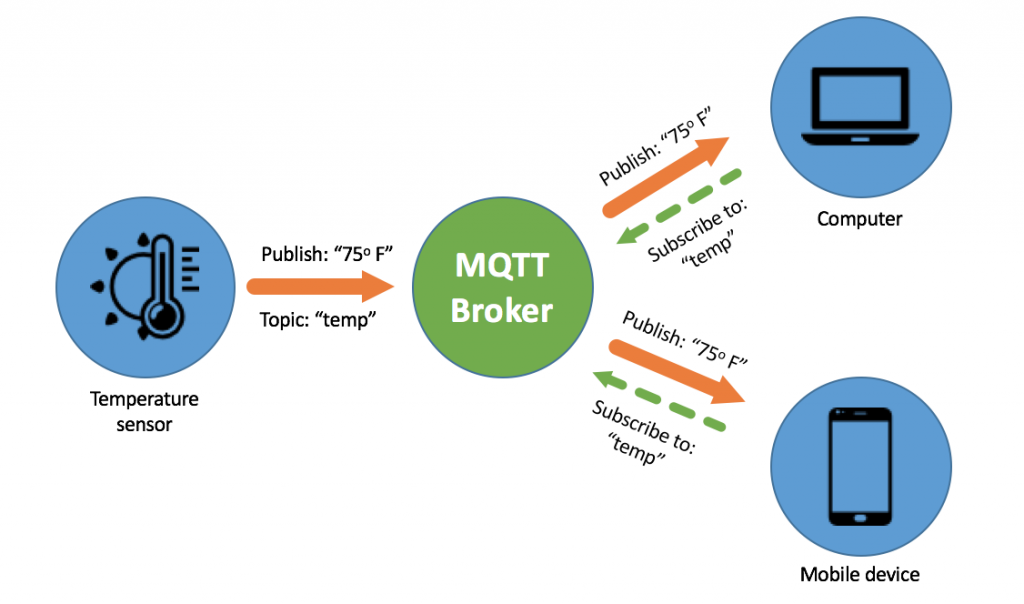
* Deve transmitir informação entre os nós e o Gateway via radio (nRF24L01);
* Deve ser utilizado modulo Wi-Fi (ESP01) para permitir o acesso a Internet no Gateway;
* Deve permitir com que o GUI consiga correr igualmente a partir de baterias internas ou via alimentação externa utilizando ligação USB;
* Deve alimentar todos os nós (temperatura/humidade, sensor das portas/janelas, sensor do movimento) em exceção do atuador (relé) com uma pilha (Button cell) de 3V;
* Deve implementar um conversor de corrente 220-5V para alimentar o modulo do relé;
* Deve implementar o ATtiny84 - controlador AVR de 8 bits em cada um dos nós;
* Deve utilizar o display (Nextion NX4832T035) como meio de comunicação entre o utilizador e restantes componentes do sistema;
* Deve utilizar SoC (ESP8266) como microcontrolador principal da interface gráfica do utilizador;
* Deve medir temperatura e humidade utilizando sensor DHT11;
* Deve utilizar TM4C123GXL Tiva-C (placa de desenvolvimento) como núcleo do Gateway central;
* Deve alimentar o Gateway com um transformador 220-5V;
* Deve permitir consumos de corrente inferior a 1mA durante o período de stand-by;



Software

1. Protocolo MQTT

MQTT (Message Queing Telemetry Transport) é um protocolo de comunicação com baixos requisitos ao nível da largura de banda e também ao nível de hardware, sendo extremamente simples e leve. Este baseia-se nos princípios (machine-to-machine), que é usado em situações de pequena largura de banda e dados do utilizador. Estas características tornam este protocolo ideal para as aplicações “Internet of Things” (Internet das coisas) um mundo de equipamentos conectados, além das aplicações mobile onde banda e potência da bateria são relevantes.

O MQTT utiliza o paradigma Publisher/Subscriber (pub/sub) para a troca de mensagens. O paradigma pub/sub implementa um middleware denominado de broker. O broker é responsável por receber, e reenviar as mensagens recebidas dos Publishers para os Subscribers.

O Publisher é responsável por se ligar ao broker e publicar as mensagens. Já o Subscriber é responsável por se ligar ao broker e receber as mensagens que ele tiver interesse.

O paradigma pub/sub utiliza o conceito de tópicos para processar as mensagens, em que cada mensagem é enviada para um determinado tópico. Diferente de outros protocolos de mensagem, o Publisher não envia a mensagem diretamente ao Subscriber, mas sim ao broker.

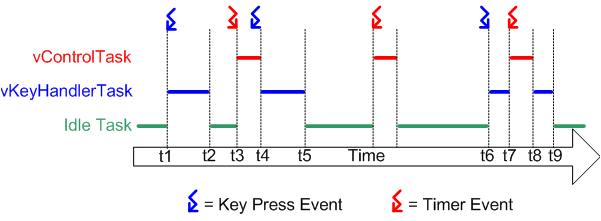
O publisher envia a mensagem para o broker em um determinado tópico. O broker é responsável por receber a mensagem do Publisher e fazer uma pré-filtragem das mensagens e enviá-las para os Subscribers que estivem registrados em um determinado tópico.

1. FreeRTOS

FreeRTOS é um Sistema Operacional (SO) de tempo real desenhado para ser implementado em sistemas embebidos. Este SO foi desenhado para ser pequeno e simples. O núcleo em si consiste precisamente de 3 ficheiros escritos em C. FreeRTOS dispõe métodos para múltiplas tarefas ou “threads”, mutexes, semáforos e timers de software. O modo “tick-less” permite utilizar este SO em sistemas com consumos reduzidos. Não existem funcionalidades avançadas tipicamente encontradas em sistemas operativos como Linux ou Microsoft Windows, tais como drivers de periféricos, gestão de memória avançada, contas de utilizador ou “networking”. A ênfase está na compacidade e velocidade de execução. O FreeRTOS pode ser pensado como uma 'biblioteca de threads' em vez de um SO, embora a interface da linha de comandos e os complementos de abstração de I/O do tipo POSIX estejam disponíveis. Também é suportado por bibliotecas SSL / TLS populares, como wolfSSL.

O FreeRTOS implementa várias threads fazendo com que o programa host chame um método de tick da thread em intervalos curtos regulares. O método de tick de encadeamento alterna tarefas, dependendo da prioridade e de um esquema de agendamento round-robin. O intervalo usual é de 1/1000 de segundo a 1/100 de segundo, por meio de uma interrupção de um timer de hardware, mas esse intervalo geralmente é alterado para se adequar a um aplicativo específico.

O diagrama abaixo demonstra como as tarefas seriam agendadas por um sistema operacional em tempo real. O RTOS criou uma tarefa de idle que será executada apenas quando não houver outras tarefas capazes de fazê-lo. A idle task do RTOS está sempre em um estado em que é capaz de executar.



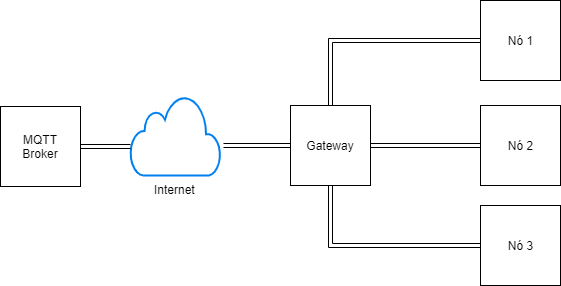
Melhorar a parte final

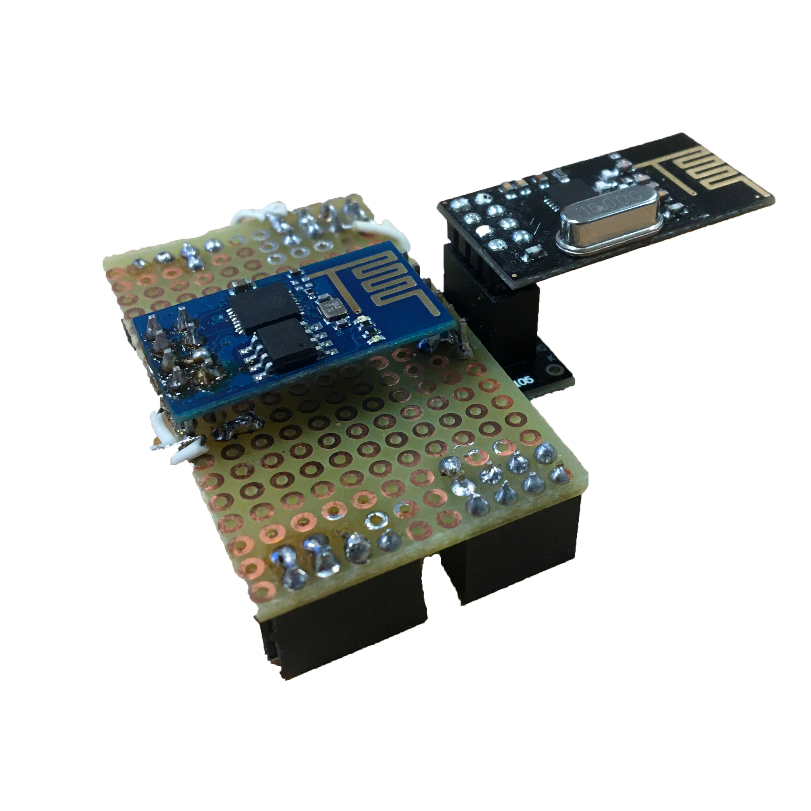
Descrição dos nós

Nesta secção iremos falar detalhadamente sobre a estrutura interna e características técnicas de cada uma das partes do sistema em questão. No total foram desenvolvidos 5 módulos, cada um com funcionalidades distintas.

1. Gateway

Gateway ou por outras palavras, coração do sistema tem como principal função fazer a gestão de informação que por um lado vem do Broker MQTT e tem que ser enviada para um nó específico que se relaciona com a mesma. Da mesma forma, a informação proveniente dos nós tem que ser processada e posteriormente publicada no tópico especialmente designado para a mesma. Esquema abaixo representa de uma forma esquemática o princípio lógico do gateway.

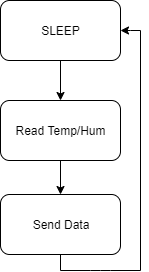




Este modulo foi construído a base de uma placa de desenvolvimento TM4C123GXL Tiva-C. Utiliza nRF24L01 para realizar a comunicação com os nós e ESP01 para garantir a comunicação com o broker MQTT. Microcontrolador corre Sistema Operativo FreeRTOS, o que permite a execução de várias tarefas de uma forma distribuída e equilibrada. Cada tarefa tem uma prioridade definida, o que permite que as tarefas com maior prioridade sejam executadas em primeiro lugar.

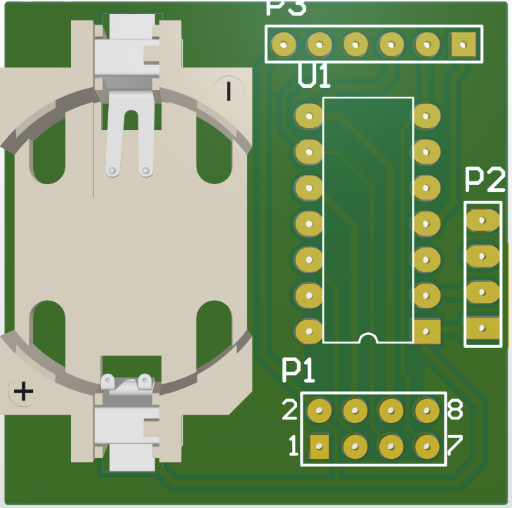
Por mais cenas aqui… por exemplo descrever tarefas…

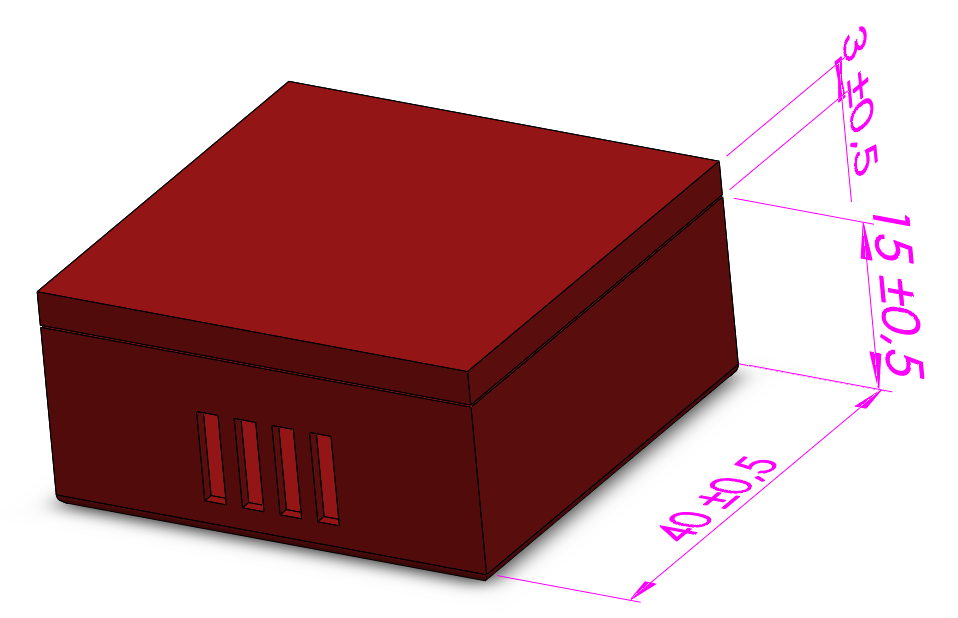
1. Sensor de Temperatura/Humidade



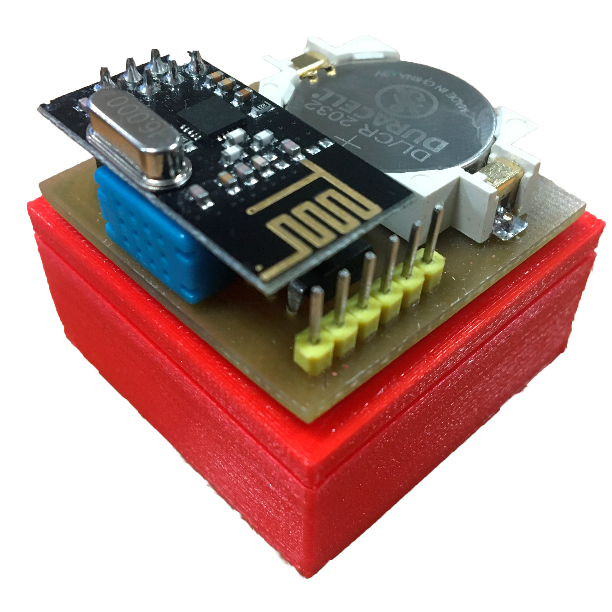
Nó responsável pela medição de temperatura e humidade. Tem como microcontrolador ATtiny84 e transfere a informação para o Gateway através de nRF24L01. A comunicação entre os mesmo é realizada através do protocolo SPI. Corre numa frequência de 8MHz. Maior parte do tempo encontra-se em modo stand-by (LPM POWER DOWN). Cada 8 segundos (tempo máximo para configurar o WDT) acorda, faz a leitura dos parâmetros do sensor de temperatura e humidade (DHT11) e envia-os para o Gateway (descrição do funcionamento demonstrado esquematicamente na figura ao lado). Para notificar o Gateway, empacota os dados numa estrutura (struct), liga o transceiver radio, e efetua a transmissão do pacote gerado. A leitura da informação do sensor de temperatura (DHT11) é feita apenas por um fio. Esta não pode ser feita com frequência menor a 1seg. Esta restrição do sensor não causa nenhumas dificuldades na implementação do modulo, visto que o aquisition rate é de 8 segundos.

Em modo SLEEP, o nó apresenta consumos extremamente baixos (menos de 1mA), permitindo assim uma life-time elevada utilizando apenas uma pilha de 3V (formato CR2032 aprox. 230mAh). No momento de transmissão, o sistema consome cerca de 3mA. A transmissão é feita no período menor do que um segundo. O tempo de estimado de vida útil de uma pilha é cerca de 3 meses, o que não é pouco, mas mesmo assim é um dos aspetos a melhorar. Para aumentar o tempo entre a troca dos elementos de alimentação, podemos diminuir o aquisition rate do sensor, passando de 8 para pelo menos 16 segundos; adicionar mais uma pilha em série, duplicando a capacidade (2 x 230mAh) ou colocar um acumulador de Li-Po juntamente com o circuito de monitorização de carga (um dos pontos a melhorar).

 Na figura a esquerda podemos observar uma visualização em 3D da placa desenvolvida. Do lado esquerdo da placa encontra-se o socket para colocar a pilha do formato CR2032 ou semelhantes. Com P3 esta indicado um conjunto do headers que têm como principal objetivo permitir a reprogramação do microcontrolador ATtiny84 dentro do circuito (in circuit programming). O próprio microcontrolador assenta no local assinalado com U1. Headers assinalados com P1 servem para colocar o modulo radio (nRF24L01). O próprio sensor de temperatura e humidade assenta no local indicado como P2.



Na figura a direita esta representado o modelo desenhado em 3D para acomodar a PCB do sensor. A caixa tem dimensões 40x40x19 mm. Uma das faces laterais inclui aberturas para facilitar a livre circulação do ar, permitindo as leituras mas precisas do sensor.



Versão final do modulo em cima da própria caixa impressa em PLA. Para diminuir o tamanho do modulo, sem alterar a lista dos componentes a utilizar, podemos: trocar o ATtiny84 por ATtiny85 num enclosure SOIC8; nRF24l01 pelo mesmo mas uma versão SMD; mudar o formato das pilhas para mais pequenas, mas duplicar a quantidade das mesmas.

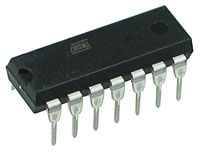
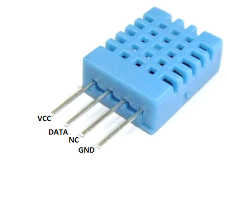
**Componentes utilizados:**

-Socket para CR2032 ou semelhantes;

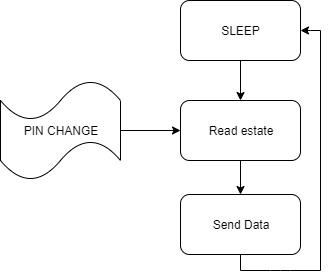
-ATtiny84 – microcontrolador AVR de 8 bits;

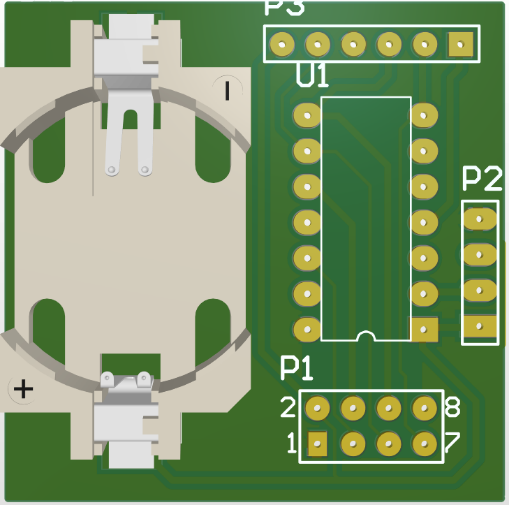
-nRF24L01 Nordic semiconductor radio transceiver;

-DHT11 sensor de temperatura e humidade;

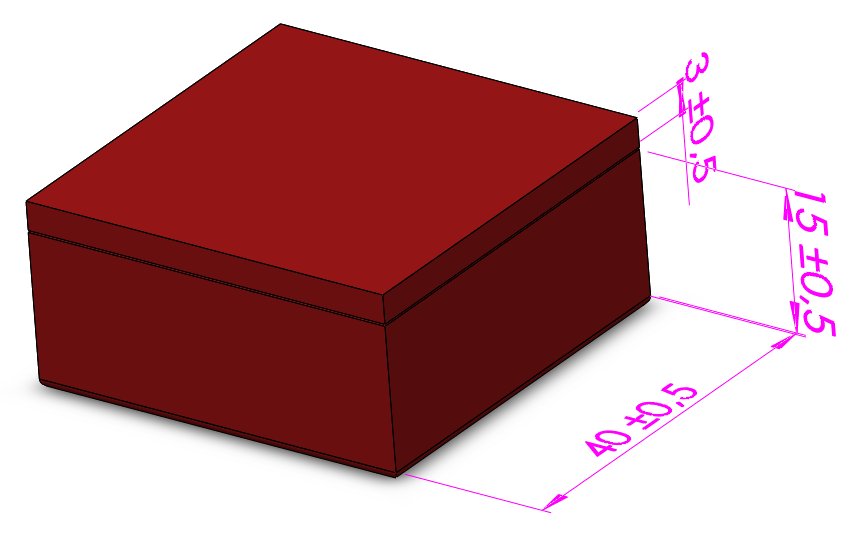


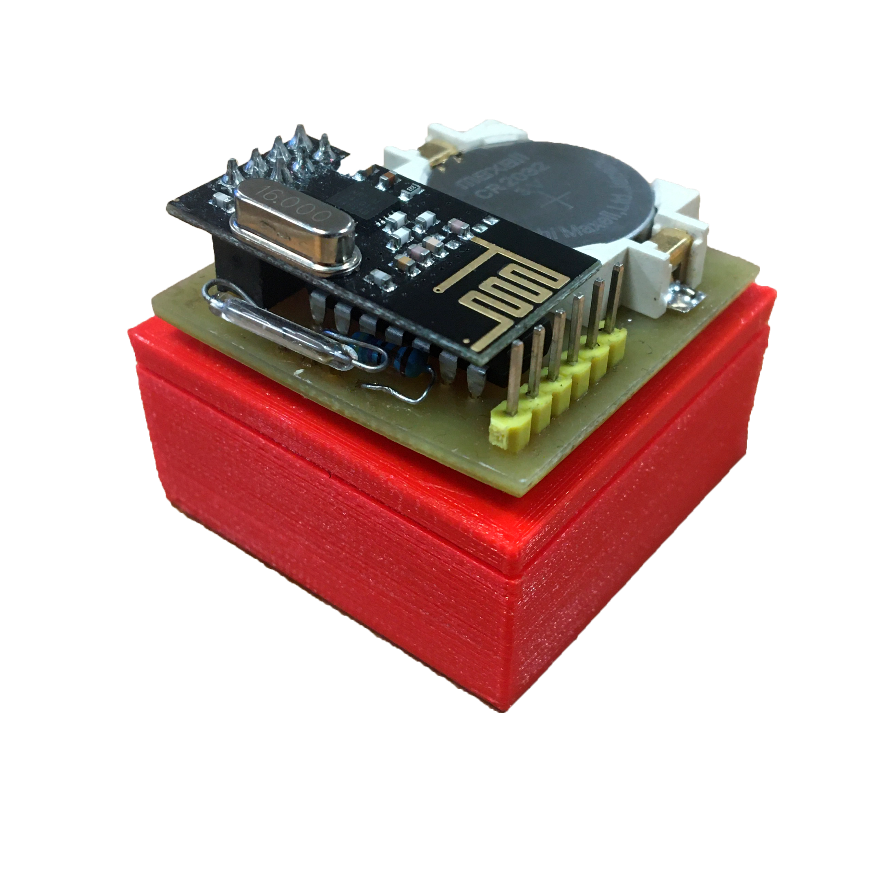
1. Sensor REED

Nó responsável pela monitorização do estado de uma porta ou janela. Deteta se a mesma esta fechada ou aberta. Tem como microcontrolador ATtiny84 e transfere a informação para o Gateway através de nRF24L01. A comunicação entre os mesmo é realizada através do protocolo SPI. Corre numa frequência de 8MHz. Maior parte do tempo encontra-se em modo stand-by (LPM POWER DOWN). Implementa uma rotina de interrupção acionada pela troca do estado do pino. Esta rotina faz restart to microcontrolador, forçando assim a sida do modulo do estado SLEEP. Se esta não for acionada dentro de 8 segundos, o modulo corda, faz a leitura do estado do pino onde o REED switch encontra-se ligado e envia a informação sobre o mesmo para o Gateway (descrição do funcionamento demonstrado esquematicamente na figura ao lado). Esta abordagem permite uma robustez do sistema, garantido os consumos baixos da bateria e ao mesmo tempo exclui trocas do estado não registadas. Para notificar o Gateway, empacota os dados numa estrutura (struct), liga o transceiver radio, e efetua a transmissão do pacote gerado.

Em modo SLEEP, o nó apresenta consumos extremamente baixos (menos de 1mA), permitindo assim uma life-time elevada utilizando apenas uma pilha de 3V (formato CR2032 aprox. 230mAh). No momento de transmissão, o sistema consome cerca de 3mA. A transmissão é feita no período menor do que um segundo. No entanto calcular a vida útil de uma bateria torna-se quase impossível devido ao facto de haver interrupções que retiram o sistema do estado SLEEP n número de vezes. Devido a este facto, podemos apenas estimar o tempo da vida útil da bateria apenas em condições ideais (onde não acontecem interrupções). Sendo assim, este pode chegar a 3 meses, o que não é pouco, mas mesmo assim é um dos aspetos a melhorar. Para aumentar o tempo entre a troca dos elementos de alimentação, podemos diminuir o aquisition rate do sensor, passando de 8 para pelo menos 16 segundos (ou mesmo desligar o WDT, registando apenas as interrupções); adicionar mais uma pilha em série, duplicando a capacidade (2 x 230mAh) ou colocar um acumulador de Li-Po juntamente com o circuito de monitorização de carga (um dos pontos a melhorar).

Na figura a direita podemos observar uma visualização em 3D da placa desenvolvida. Do lado esquerdo da placa encontra-se o socket para colocar a pilha do formato CR2032 ou semelhantes. Com P3 esta indicado um conjunto do headers que têm como principal objetivo permitir a reprogramação do microcontrolador ATtiny84 dentro do circuito (in circuit programming). O próprio microcontrolador assenta no local assinalado com U1. Headers assinalados com P1 servem para colocar o modulo radio (nRF24L01). Os headers assinalados por P2 servem para ligar o REED switch juntamente com uma resistência de pull-down (470kOhm) entre pino de dados e GND.

Na figura a esquerda esta representado o modelo desenhado em 3D para acomodar a PCB do sensor. A caixa tem dimensões 40x40x19 mm. Uma das faces laterais situa o sensor magnético (REED switch) que é ativado no momento de apresentação de um íman numa distância inferior a 1cm.



Versão final do modulo em cima da própria caixa impressa em PLA. Para diminuir o tamanho do modulo, sem alterar a lista dos componentes a utilizar: podemos trocar o ATtiny84 por ATtiny85 num enclosure 8SOIC; nRF24l01 pelo mesmo mas uma versão SMD; mudar o formato das pilhas para mais pequenas, mas duplicar a quantidade das mesmas.

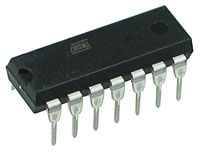
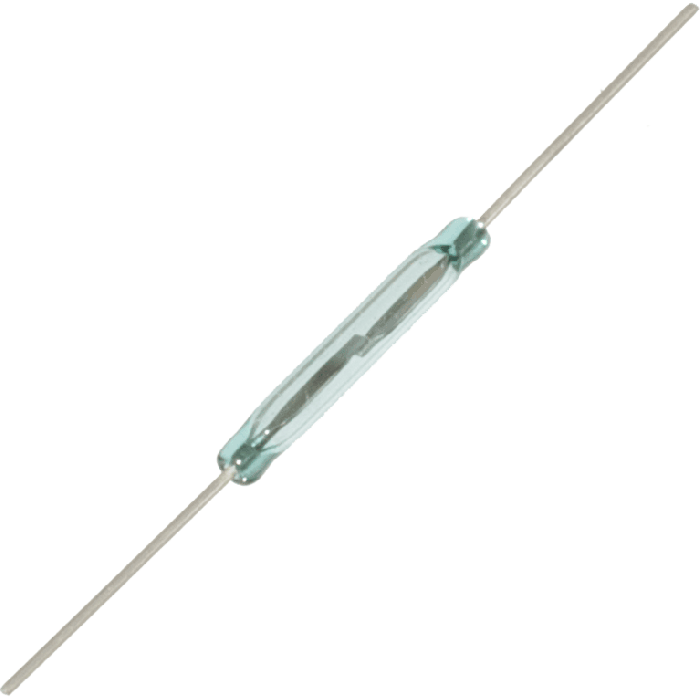
**Componentes utilizados:**

-Socket para CR2032 ou semelhantes;

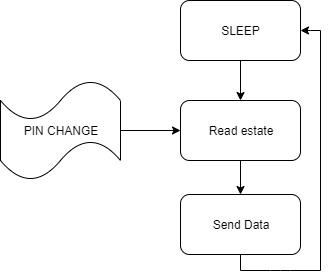
-ATtiny84 – microcontrolador AVR de 8 bits;

-nRF24L01 Nordic semiconductor radio transceiver;

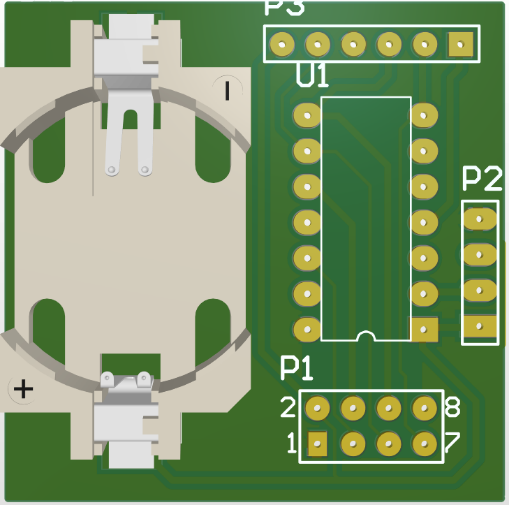
-REED switch (sensível ao campo magnético);

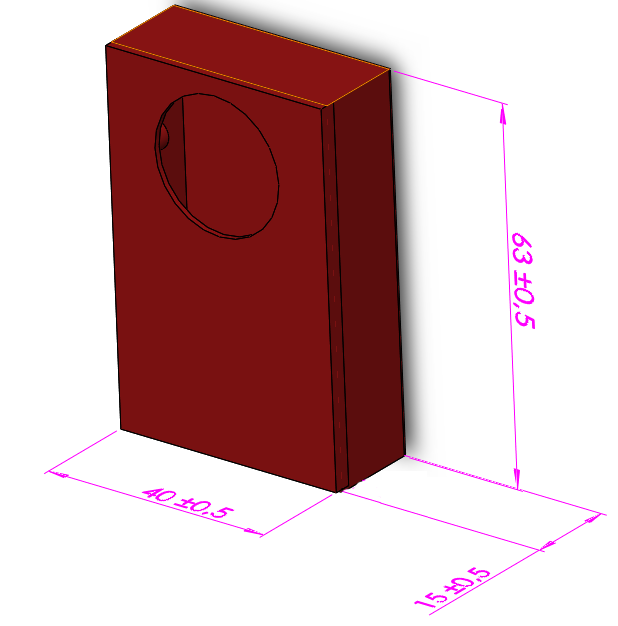


1. Sensor de movimento (PIR)

Nó responsável pela monitorização e deteção do movimento. Tem como microcontrolador ATtiny84 e transfere a informação para o Gateway através de nRF24L01. A comunicação entre os mesmo é realizada através do protocolo SPI. Corre numa frequência de 8MHz. Maior parte do tempo encontra-se em modo stand-by (LPM POWER DOWN). Implementa uma rotina de interrupção acionada pela troca do estado do pino. Esta rotina faz restart to microcontrolador, forçando assim a sida do modulo do estado SLEEP. Se esta não for acionada dentro de 8 segundos, o modulo corda, faz a leitura do estado do pino onde o PIR sensor encontra-se ligado e envia a informação sobre o mesmo para o Gateway (descrição do funcionamento demonstrado esquematicamente na figura ao lado). Esta abordagem permite uma robustez do sistema, garantido os consumos baixos da bateria e ao mesmo tempo exclui trocas do estado não registadas. Para notificar o Gateway, empacota os dados numa estrutura (struct), liga o transceiver radio, e efetua a transmissão do pacote gerado. A placa do sensor PIR fornece dois potenciómetros que permitem ajustar a sensibilidade do modulo e a duração do impulso na deteção do movimento. A principal limitação do sensor utilizado é o seu ângulo de visão, que consistem em <120º em menos de 7 metros de distância.

Enquanto aos consumos de energia, estes são semelhantes ao modulo REED switch. O sensor PIR consome 50uA em estado standby, não interrompendo a monitorização do ambiente em que este é colocado. Para aumentar o tempo entre as trocas do elemento de alimentação pode ser desligado o WDT, passando a registar apenas as mudanças do estado do pino.

Na figura a direita podemos observar uma visualização em 3D da placa desenvolvida. Do lado esquerdo da placa encontra-se o socket para colocar a pilha do formato CR2032 ou semelhantes. Com P3 esta indicado um conjunto do headers que têm como principal objetivo permitir a reprogramação do microcontrolador ATtiny84 dentro do circuito (in circuit programming). O próprio microcontrolador assenta no local assinalado com U1. Headers assinalados com P1 servem para colocar o modulo radio (nRF24L01). Os headers assinalados por P2 servem para ligar o sensor PIR.

Na figura a direita esta representado o modelo desenhado em 3D para acomodar a PCB do sensor. A caixa tem dimensões 40x63x19 mm. A face frontal situa uma abertura para o sensor de movimento (PIR) que é ativado no momento de deteção do movimento.

Versão final do modulo. A caixa foi impressa em PLA.

Palha aqui

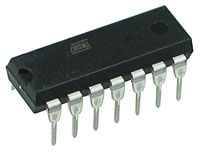
**Componentes utilizados:**

-Socket para CR2032 ou semelhantes;

-ATtiny84 – microcontrolador AVR de 8 bits;

-nRF24L01 Nordic semiconductor radio transceiver;

-Sensor PIR (sensível ao movimento);



1. Modulo do Relé

Dizer que:

Utiliza relé, como é obvio. Tem um conversor 220-5V…

Não é power eficiente, ou seja esta sempre ligado…

Guarda o ultimo estado do rele na memoria não volátil…

Ah… e no fim dizer que não foi conseguida a realização dentro do período de tempo estipulado….

----------------------PALHA PALHA PALHA------------------------

Interface gráfica de utilizador

Nesta secção iremos …

PALHA COM DESCRIÇAO DAS IMAGENS…

Dizer que comunica com esp por SPI

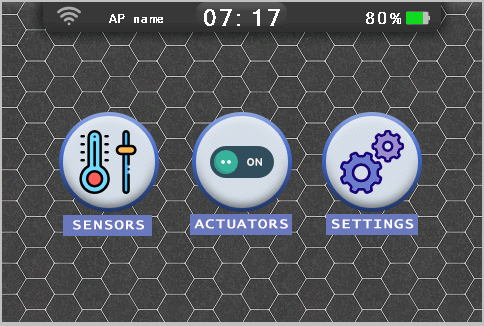
Mostra o estado da bateria..

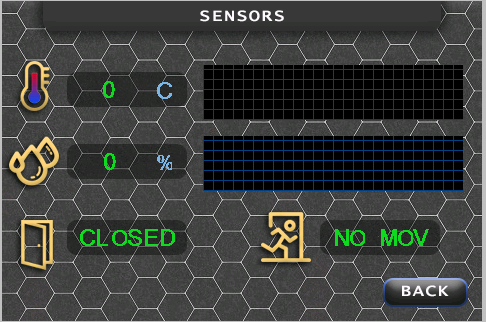
Horas…

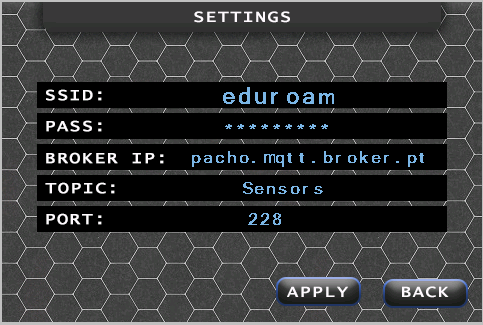
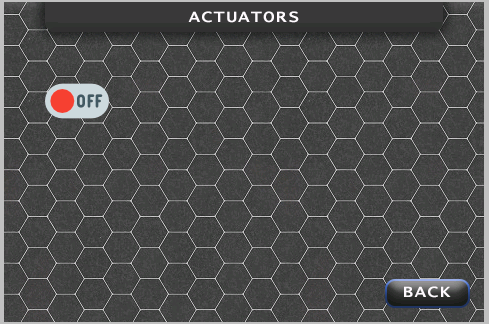
Tem 3 seções….

Bla bla bla.-…

Descrever o que esta nas imagens…







Conclusão

Bla bla bla….

Falar sobre as principais dificuldades:

-freertos, coordenação das tasks

-modulos – conseguir o low power consumption

Dizer quais dos requisitos não foram cumpridos…

Rele não funciona ( e pronto, é só isso).

Palha…)