**Plataforma computacional – Software**

O microcontrolador que se decidiu usar, para controlar todos os dispositivos (o modulo do Sigfox, o sensor de temperatura e o sensor ultrassónico), foi o lpc1769. Esta escolha deve-se ao facto de ser um microcontrolador que contem periféricos que são importantes (como o timer e i2c), o que permite ter um maior número de funcionalidades, e tem um sistema de interrupções para cada periférico, permitindo assim manter o microcontrolador em estado idle, e acordar na aparição de um evento.

Esta escolha poderia ser alterada, se no final se observar que existem outros microcontroladores que satisfaçam nossas necessidades do sistema embebido, por um preço mais baixo ou com um menor custo de bateria.

Primeiramente foi necessário realizar os programas que fossem controlar o sensor ultrassónico e o sensor de temperatura.

Sensor de Temperatura

Para comunicar com o sensor de temperatura é usado um porto, do GPIO0, do microcontrolador. A transmissão, realiza-se só com um pino, o que é necessário controlar os tempos das ações. Para controlar esses tempos, é usado o periférico SystemTick. Com o SystemTick é possível programar um contador com um certo valor, e quando esse valor chegar a 0, este emite uma interrupção. Assim, nestes momentos onde existe delays, o microcontrolador encontra-se no estado idle, há espera de uma interrupção. Isto permite uma poupança de bateria nesses intervalos de tempo. O uso de um sensor de temperatura, foi para garantir que os resíduos que se situam dentro do contentor, não comecem a fermentar, causando problemas de salubridade. Para este controlo, uma forma era usar o alarm que o DS18B20+ disponibiliza. Este permite configurar um intervalo de temperatura. Depois de o microcontrolador enviar o comando Alarm Search, consegue observar se a temperatura encontra-se dentro dos limites configurados. A outra forma é ler a temperatura da memória do sensor, usando o comando Read Scratchpad. A primeira operação tem uma menor duração e é menos dispendiosa de energia. Mas no Alarm Search, é necessário configurar o sensor com os limites, limites esses, que poderão ter sido pensados para de dia, e quando chega à noite, apesar de os residios se encontram a fermentar, a temperatura não passa o limite, devido à temperatura ambiental ter diminuído. Desta forma é lido sempre a temperatura, e são enviados a outra entidade para realizar esses cálculos.

Sensor Ultrassónico

Com o sensor Ultrassónico é também utilizado o GPIO0. Este usa dois portos: i) um para provocar o envio da onda ultrassónico do sensor, ii) e o outro que é notificado quando a onda chega ao recetor do sensor. Para saber o tempo do tempo que a onda demorou a voltar, é usado o TIMER0. Neste caso não foi usado o SystemTick, devido a este ser um contador decrescente.

External Interrupt

Reparou-se que ao colocar o microcontrolador a acordar periodicamente, isto poderia causar desperdícios de recursos. Um desses casos acontece quando o microcontrolador acorda durante a noite, hora que possivelmente ninguém coloca lixo, para obter a quantidade de lixo. A partir da segunda leitura, a quantidade de lixo mantinha-se o mesmo, o que causaria um desperdício de energia, a sua leitura no período da noite. O outro caso pode acontecer se o contentor encher de uma forma exponencial e o microcontrolador não detetou esse aumento devido a não ter passado o tempo para acordar. Este rápido aumento pode causar os resíduos transbordarem do contentor. Assim, para resolver estas duas inconveniências, foi acrescentado uma nova funcionalidade, esta que acorda o microcontrolador com uma interrupção externa. Coloca-se uma peça de hardware na tampa do contentor, que quando esta abrisse, acorda o microcontrolador. Assim, durante a noite, se ninguém abrir a tampa este mantinha-se no estado idle. No caso de encher rapidamente o contentor, este tem conhecimento porque foi acordado quando abriram a tampa. Para assegurar o correto funcionamento do microcontrolador, este adormece com um timer se o cidadão não baixar a tampa. Devido a este acordar com a tampa quando abre, se ninguém baixasse, o microcontrolador nunca mais acordava, o que causava a não realização das leituras.

Com isto foi criada a classe ExternalInterrupt de modo a ativar a interrupções externas. E foi adicionado ao TIMOR0 a possibilidade ativar uma interrupção quando o contador atingisse um dado valor.

RTC

Como já foi referido anteriormente, se um contentor não ultrapassar um certo threshold, este pode nunca ser recolhido. Para resolver este problema, foi colocado um sensor de temperatura que obtinha a informação se os resíduos estavam a fermentar, através da temperatura, causando problemas de salubridade. Mas esta verificação tem que ser realizada periodicamente, algo que nós retiramos com a ativação do microcontrolador com a abertura da tampa. Assim, se ninguém abrir a tampa, o microcontrolador não vai ler a temperatura ambiental. Para resolver este problema, foi adicionado um RTC. Este permite ter o conceito de dia, e foi adicionado uma interrupção que se atividade quando passa um dia. Podemos assim observar que o microcontrolador acorda todos os dias para ir ler a temperatura, mesmo que a tampa não seja levantada.

Sigfox

Por fim foi necessário enviar estas informações para o controlador de Sigfox [arduino]. Para isso foi usado o protocolo de comunicação I2C. Este permite realizar escritas e leituras nos periféricos slaves. Quando o microcontrolador quiser escrever no controlador de Sigfox, inicia uma transação I2C com sinal de escrita ativo.

O controlador de Sigfox usa para comunicar com o master 4 portos. Dois para realizar as transações I2C (SCL e SDA), um como output para notificar que está busy (e pode realizar comunicações I2C) e outro de input que serve para sinalizar o controlador de Sigfox para obter a configuração do microcontrolador (através de uma comunicação bidirecional com a Cloud Sigfox). Assim, para o correto funcionamento, quando um master observa que o slave se encontra no estado busy, espera que este termine a sua tarefa.

De modo a que as mensagens trocadas entre o dispositivo e a cloud fossem corretamente interpretadas, foi necessário estabelecer uma estrutura para cada mensagem (uplink e downlink) essa estrutura pode-se observar na Figura 1.