Desenvolvimento de uma rede de sensores sem fios para monitorização de peixes

Daniel Roitman Pozzatti

Dissertação para a obtenção de Grau de Mestre em Engenharia Electrónica e Telecomunicações

Orientador: Prof. Doutor Peter Stallinga

2014

Desenvolvimento de uma rede de sensores sem fios para monitorização de peixes

Declaração de autoria de trabalho

Declaro ser o autor deste trabalho, que é original e inédito. Autores e trabalhos consultados estão devidamente citados no texto e constam da listagem de referências incluída.

Daniel Roitman Pozzatti

[1]Esta dissertação foi escrita ao abrigo do novo acordo ortográfico.

Agradecimentos

Gostaria de agradecer ao meu orientador Peter Stallinga, pelos conhecimentos gentilmente partilhados e pela disponibilidade demonstrada no decorrer do desenvolvimento deste projeto. Agradeço também a Paulo Zaragoza Pedro da Caviar Portugal com quem a convivência e discussão de várias ideias ao longo deste último ano permitiram a realização de um projeto mais consistente.

A nível mais pessoal quero agradecer à minha namorada Mariana Boavida e à minha mãe Rosana Roitman pelo amor e apoio demonstrados em todos os momentos da minha vida.

Quero ainda agradecer a todos os meus amigos e colegas que fizeram parte desta minha passagem pela Universidade do Algarve.

Resumo

Este trabalho teve como principal objetivo desenvolver e implementar uma rede de sensores sem fios, capaz de monitorizar tanques de aquacultura. Dada a necessidade de um sistema de baixo custo e elevada flexibilidade para monitorização dos tanques de cultivo de esturjão Beluga por parte da empresa Caviar Portugal, foi efetuado um estudo detalhado sobre sensores, rede de sensores sem fios e sobre o protocolo ZigBee, permitindo assim desenvolver um módulo de hardware capaz de efetuar o pretendido. Estes módulos são constituído por um Arduino, um dispositivo de comunicação sem fios e um conjunto de sensores. Os dados são recolhidos, verificados e enviados para um computador central com ligação à Internet. A comunicação entre o computador central e os tanques é da responsabilidade dos módulos wireless XBee, dispositivos que implementam o protocolo ZigBee. Toda a informação recolhida nos tanques de cultivo é armazenada numa base de dados PostgreSQL, para posterior consulta. O protótipo desenvolvido mostrou-se fiável e adequado para a supervisão dos tanques de cultivo, no entanto não é perfeito e há aspetos que têm de ser melhorados, nomeadamente desenvolver uma placa de circuito impresso única e aprimorar a interface do sistema com o utilizador.

Palavras-chave: Rede de Sensores sem Fios, Sensores, Aquacultura, Monitorização, Arduino, Xbee.

Abstract

This study aimed to develop and implement a wireless sensors network, capable of monitoring aquaculture ponds. Given the need for a low cost and high flexibility for monitoring the cultivation tanks sturgeon By the company Beluga Caviar Portugal, was conducted a detailed study of sensors, wireless sensor network and the ZigBee protocol, thereby developing a module hardware capable of performing intended. These modules are constituted by a Arduino a device for wireless communication and a set of sensors. The data are collected, scanned and sent to a central computer with internet connection. Communication between the central computer and tanks is the responsibility of the XBee wireless modules, devices that implement the ZigBee protocol. All information collected in tanks cultivation is stored in a PostgreSQL database for later retrieval. The prototype was proven to be reliable and suitable for the supervision of cultivation tanks, however is not perfect and there are aspects that have to be improved, in particular develop a plate single printed circuit board and improve the system interface with the user.
Keywords: Wireless Sensor Network, Sensor, Aquaculture, Monitoring, Arduino, Xbee.

Lista de Figuras

Lista de Tabelas

INDÍCE

Capítulo 1

Introdução

1.1 Enquadramento e motivação

O projeto intitula-se desenvolvimento de uma rede de sensores sem fios para monitorização de peixes. Esta dissertação insere-se no projeto de Mestrado em Engenharia Eletrónica e Telecomunicações da Faculdade de Ciências e Tecnologias da Universidade do Algarve.

Pretende-se, com esta dissertação, criar um protótipo para utilização numa rede de sensores sem fios, capaz de monitorizar tanques de cultivo de Esturjão Beluga para produção de caviar, de forma flexível e adaptável.

O projeto teve a duração de um ano e apresenta inovações no campo dos sistemas sem fios para monitorização de tanques de aquacultura.

O trabalho decorreu nas instalações da Caviar Portugal, localizada na própria Universidade do Algarve.

A Caviar Portugal

A Caviar Portugal – Acipenser Lda - é uma empresa portuguesa pioneira na criação de Esturjão para a produção de caviar. É uma empresa incubadora na Universidade do Algarve, situada no Centro de Empresas da Universidade do Algarve, Campus de Gambelas, em Faro.

A sua missão é ambiciosa. Pretende ser uma referência na produção de Esturjão e caviar a nível regional e nacional, satisfazendo as necessidades do mercado interno nestes produtos de elevado valor.

A Caviar Portugal aposta na investigação e desenvolvimento de novos produtos a partir do Esturjão e otimização dos métodos produtivos de forma a estar na vanguarda da produção de diversas espécies de Esturjão.

O objetivo maior da criação desta empresa é contribuir, a longo prazo, para a reintrodução das espécies autóctones de Esturjão (atualmente extintas no meio natural), nos rios e estuários de Portugal e promover a utilização do Esturjão na alimentação, indústria farmacêutica e cosmética.

Este projeto ganhou o Prémio Especial do Mar no Concurso "Ideias em caixa 2010" promovido pela Universidade do Algarve e a Caixa Geral de Depósitos.

O Esturjão

Por ser um dos peixes comercialmente mais valiosos do mundo, o Esturjão está em

risco de desaparecer e neste momento praticamente só é possível criá-lo em regime de aquacultura.

No seu ambiente natural, com temperaturas negativas, os esturjões podem demorar 20 anos até atingir a sua maturação sexual mas, no Algarve devido à temperatura amena, o seu crescimento pode ser muito mais rápido.

O Esturjão atlântico, espécie extinta em Portugal desde a década de 80, já habitou nos estuários do Guadiana e do Arade.

Os responsáveis pelo projeto estimam que, em 2015, já seja possível a venda de caviar produzido no Algarve em alguns restaurantes e lojas *gourmet*.

O tipo de aquacultura feita pela Caviar Portugal é diferente da tradicional em esteiro ou em tapada (tanques de terra). É uma tecnologia completamente diferente, com sistemas de recirculação de água, em ambiente completamente controlado. É uma aquacultura industrial.

Foi-me permitido fazer parte deste projeto, onde testamos tecnologia e técnicas produtivas num empreendimento pioneiro e inovador em Portugal.

1.2 Objetivos da dissertação

Com base na informação acima transcrita foi-me solicitado que desenvolvesse um sistema para monitorização dos tanques. Apesar de no mercado existirem sistemas para monitorização de piscicultura, nenhum deles se enquadra diretamente na solução que era necessária. Os sistemas existentes no mercado apresentam um custo elevado, complexa instalação e não tem flexibilidade de modificação.

O objetivo desta dissertação é desenvolver um protótipo para um sistema inovador,que com um baixo custo de produção, seja consiga atender as seguinte características:

- Fácil instalação, em que qualquer operador com a mínima instrução seja capaz de colocar o equipamento a funcionar.
- Flexível, podendo ser modificado a qualquer momento, seja na sua localização ou substituição dos sensores nele contidos.
- Sem fios, não necessitando de fios ou cabos de comunicação.
- Capacidade de monitorizar os tanques, recolhendo e armazenando os respetivos parâmetros dos tanques de forma a melhorar o controlo de qualidade da água e diminuir a taxa de mortalidade do esturjão, melhorando o seu desenvolvimento.

Além de ser capaz de monitorizar os tanques, recolhendo e armazenando os respetivos parâmetros necessários para assegurar a qualidade da água, diminuindo assim a taxa de mortandade e aumentando o desenvolvimento do Esturjão.

A monitorização dos tanques deve seguir algumas especificações que este projecto deve ter em conta:

- Monitorizar a temperatura entre -10 °C e 60 °C, com uma resolução nunca inferior a 0,5 °C.
- Medir o nível dos tanques, verificando se o volume de água é inferior ou superior a valores pré determinado.

- Determinar o pH da água periodicamente.
- Ser capaz de disponibilizar os dados dos tanques para o utilizador, minuto a minuto.

O projeto deu passos concretos para tornar realidade um negócio de produção do precioso caviar em Portugal, que pode vir a ter um retorno milionário.

Capítulo 2

Estado da arte

2.1 Rede de sensores sem fios

Rede de Sensores sem Fios (RSSF) é uma tecnologia emergente que promete funcionalidades sem precedentes para monitorizar, instrumentar e possivelmente controlar o mundo físico. [Loureiro, 2008]

Uma RSSF pode ser definida como sendo uma rede de nós sensores ou simplesmente sensores, de baixo consumo energético, distribuídos em locais de interesse, capaz de formar uma rede, com conectividade sem fios. Uma rede deste tipo deve analisar o ambiente em que o equipamento está inserido com pouco ou nenhum impacto físico nesse ambiente. Os "nós sensores" devem ser capazes de medir variáveis num determinado ambiente, por meio de sensores ou transdutores além de transmitir os dados medidos por meio de ondas eletromagnéticas para outro ponto da rede. Para o efeito, um nó sensor deve possuir, pelo menos, um sensor, um transmissor de rádio, um micro-controlador e uma fonte de energia. Uma rede RSSF possui um nó coordenador (*sink node*), responsável por receber os dados dos nós sensores e encaminhar para um computador ou outro dispositivo capaz de armazenar ou reencaminhar os dados recebidos, interpreta-los e se necessário tomar decisões.

Na Figura 2.1 podemos ter uma visão geral da arquitetura e organização de uma RSSF típica:

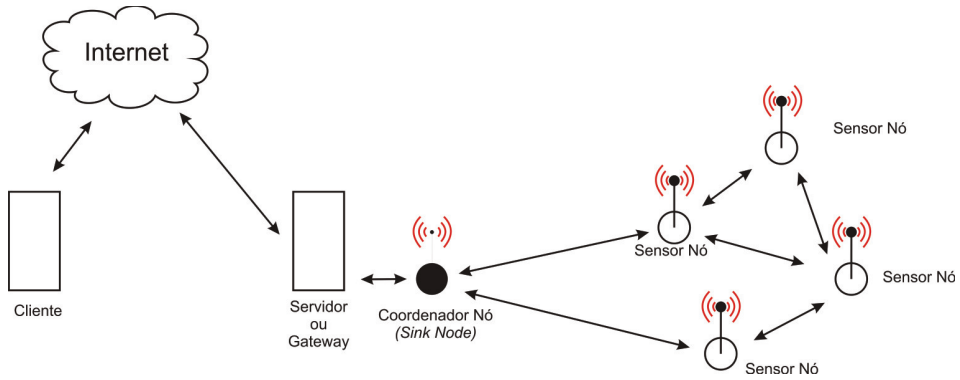


Figura 2.1: Visão geral da arquitetura e organização de uma RSSF.

2.2 Padrão ZigBee/IEEE 802.15.4

2.2.1 Introdução

De acordo com Drew Gislason em *ZigBee Wireless Network*, apesar da existência de vários padrões para dispositivos sem fios, todos os padrões existentes esforçavam-se por obter altas taxas de transferência e maximizar os recursos disponíveis, tendo como objetivo a transmissão de dados na Internet ou *stream* de vídeo.

Em meados de 1999 verificou-se a necessidade de criar um padrão de redes sem fios para ser utilizado em dispositivos de monitorização e controlo. Estes dispositivos deveriam ser capazes de: formar redes com grande quantidade de dispositivos; cobrir grandes dimensões de forma simples e autónoma; funcionar durante vários meses ou anos utilizando apenas um par de baterias AA; comunicar com micro-controladores de apenas 8 *bits* e, ainda assim, ser altamente fiável, durável, com uma boa relação custo-benefício e uma comunicação de dados segura.

Assim, o *Institute of Electrical and Eletronics Engineers* (IEEE) criou o padrão IEEE 802.15.4 para redes sem fios de baixa velocidade e baixo consumo energético.

O padrão IEEE 802.15.4 define duas camadas de baixo nível: PHY (*Physical Layer*) e MAC (*Medium Access Control*), deixando as camadas superiores livres para implementação de acordo com a necessidade de utilização. PHY é a camada responsável pela interligação do protocolo com o rádio de comunicação e por fornecer serviços para a camada MAC, camada essa imediatamente superior que controla o acesso à rede.

Utilizando o padrão IEEE 802.15.4, a ZigBee Alliance criou o padrão ZigBee que utiliza as 2 camadas de baixo nível do IEEE 802.15.4 e implementa mais cinco camadas de nível superior como podemos observar na Figura 2.2.

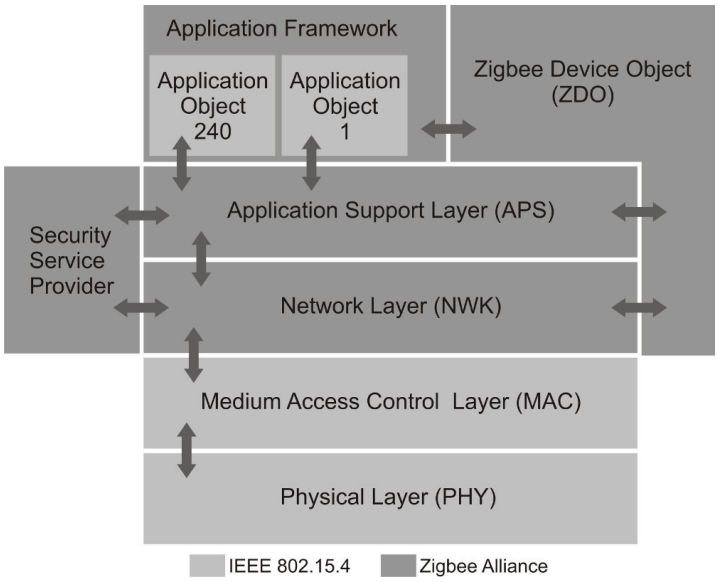


Figura 2.2: Diagrama da Arquitetura e Pilha do Protocolo ZigBee – Fonte: Adaptado de [Gislason, 2008]

2.2.2 Organização e arquitetura ZigBee/IEEE 802.15.4

O padrão ZigBee não utiliza exatamente as sete camadas do modelo OSI (*Open Systems Interconnection*[1], no entanto possui alguns elementos semelhantes como as três primeiras camadas PHY, MAC, NWK (*Network Layer*). As demais camadas do modelos OSI estão englobadas nas camadas APS (*Application Support Layer*) e ZDO (*ZigBee Device Object*) do modelo ZigBee. [Gislason, 2008]

A camada PHY tem como principal responsabilidade transformar os pacotes de dados digitais em ondas eletromagnéticas, para posteriormente serem enviadas pelo transmissor de radio. Entretanto, também é responsável pela deteção do canal de comunicação e seleção do mesmo. A seleção do canal é feita de acordo com os canais disponíveis para cada banda de frequência. Na Tabela 2.1 podemos verificar, quais são as bandas de frequências, assim como os seus canais e o tipo de modulação do sinal.

Tabela 2.1: Especificações da Camada Física.

PHY (MHz)	Bandas de Frequências (MHz)	Modulação
868/915	868-868.6	BPSK
	902-928	BPSK
2450	2400-2482.5	O-QPSK

A camada MAC é responsável por definir que dispositivos serão FFD (*Full Fuction Device)* ou RFD (*Reduced Function Device*). Os dispositivos FFD operam como *routers* ou coordenadores da rede, enquanto os dispositivos RFD trabalham apenas como pontos finais. O modo de operação dos dispositivos da rede define uma das principais características do padrão ZigBee que é o baixo consumo de energia. Ainda é responsabilidade da Camada MAC descobrir e identificar os nós da rede, através do envio de sinalizadores (*Beacons*) e receção das respetivas respostas, confirmando assim presença dos nós na rede.

A camada NWK é responsável deve garantir a capacidade de *routing*, permitindo então a formação de redes em malha (*Mesh Networking*). Essa característica permite ainda a utilizar de transmissões em massa para toda a rede (*broadcast*), determinar rota para a entrega de pacotes para um único nó na rede (*unicast*) e ainda a confirmar a entrega dos pacotes de dados ao destino.

Na Figura 2.3 é possível observar as três topologias que uma rede ZigBee pode adotar e os três modos de operação que os dispositivos podem assumir, que explico de seguida:

Os três modos possíveis de operação, citamos a seguir:

- Coordenador - Dispositivo FFD que tem a responsabilidade de criar e gerir a rede, efetuar o reconhecimento dos nós, associar e desassociar novos dispositivos na rede e normalmente ainda funciona como *gateway*, comunicando com um computador ou outro dispositivo específico através de comunicação RS-232.

- *Router* - Dispositivo FFD que tem a função de dispositivos finais, mas também possuem as propriedades de um *router*, podendo assim reencaminhar dados de nós vizinhos sem a necessidade de comunicar com o dispositivo coordenador. Graças a

[1]Modelo de referência da ISO (*International Standards Organization)*), desenvolvido com o objetivo de permitir a comunicação entre máquinas heterogéneas e definindo diretivas genéricas para redes de computadores independentemente da tecnologia de implementação

existência dos dispositivos *router* podemos aumentar substancialmente o alcance da rede ZigBee.

- Dispositivo Final - Dispositivo que pode ser FFD ou RFD, dotado da opção de funcionar em modo *sleep*, onde o dispositivo fica inativo quando não tem dados para enviar ou receber, comutando para o modo ativo quando necessário. Essa propriedade faz com que este tipo de dispositivo seja o mais eficiente energeticamente. Normalmente são os nós onde estão associados os sensores e atuadores.

Dependendo do modo de operação dos dispositivos, uma rede ZigBee pode ser:

- Estrela (*Star*) - Topologia mais simples formada apenas por um coordenador e dispositivos finais. O coordenador inicia a rede e os dispositivos finais comunicam diretamente com o coordenador. Toda a informação da rede passa pelo dispositivo coordenador.
- Árvore (*Tree*) - Topologia onde o coordenador inicia a rede, depois de inicializada, novos dispositivos podem associar-se à rede, seja pelo coordenador ou *routers*, esse tipo de topologia permite expandir a dimensão da rede e a comunicação entre dispositivos pode acontecer sem a necessidade de passar pelo coordenador.
- Malha (*Mesh*) - "Topologia em que a rede pode ajustar-se automaticamente, tanto na sua inicialização como na entrada ou saídas de dispositivos na rede, auto-organizando-se para otimizar o tráfego de dados" [Messias, 2013]. Desta forma pode escolher o melhor caminho para a comunicação dos dados e na falha de um caminho pode optar por outro caminho alternativo. Assim sendo apesar de ser a topologia mais complexa, é a mais completa e fiável.

A Camada APL (*Application Layer*) e as suas respetivas sub camadas: (APS, ZDO e *Application Framework*), são as camadas de mais alto nível especificadas pelo padrão ZigBee. São responsáveis por filtrar os pacotes referentes aos respetivos nós registados na rede, por fornecer a confirmação da receção, por efetuar retransmissões automáticas de pacotes quando não há confirmações de entrega, por manter as tabelas de endereçamento e por guardar o estado corrente dos nós, assim como as suas características de segurança e agrupamento.

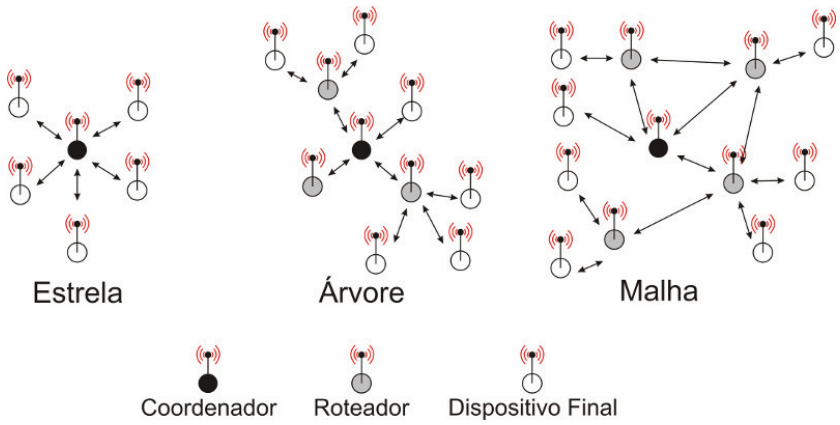


Figura 2.3: Tipos de Topologia de uma rede ZigBee.

2.3 Módulo RF Xbee

2.3.1 Introdução

Esta secção apresenta o módulo de comunicação para RSSF. Encontramos no mercado diversas empresas que disponibilizam dispositivos baseados no padrão ZigBee. Utilizaremos nesta dissertação o módulo Xbee fabricado pela *Digi International*, pois é um módulo com pequena dimensão, baixo consumo energético, robusto e durável, com grande quantidade de material didático e informativo disponível, de fácil aquisição no mercado português e presente na Faculdade de Ciências e Tecnologia da Universidade do Algarve para testes e experiências.

2.3.2 Características do módulo Xbee

O Xbee é composto basicamente por um micro-controlador que contém o *firmware* com a implementação do protocolo ZigBee, especificações referentes ao dispositivo e ao seu comportamento, e por um transcetor de rádio que tem a função de enviar e receber sinais de rádio-frequência.

Cada Xbee possui 2 endereços distintos, o MY (16 bits) e o número de série ID (64 bits). Fazendo uma analogia entre os endereços das RSSFs e as redes TCP/IP. Podemos associar o endereço MY ao endereço IP, que é atribuído a cada dispositivo toda a vez que o mesmo se conecta à rede, enquanto o ID é único e invariável de forma que podemos associa-lo ao endereço MAC de uma placa de rede.

Uma rede Xbee contém um Coordenador, que tem como endereço de rede MY o valor zero, como já mencionado na secção 2.2.2 é o dispositivo que coordena a rede. Entretanto pode conter diversos outros dispositivos *routers* ou finais na mesma rede.

A *Digi International* disponibiliza os Xbee's em dois modelos distintos: Xbee Série 1 e Xbee Série 2. A principal diferença entre as duas séries é que a Série 1 utiliza estritamente o protocolo IEEE 802.15.4 e os dispositivos da Série 2 utilizam o protocolo ZigBee, permitindo assim a formação de redes do tipo malha.

Além das duas séries de dispositivos, existem vários modelos e encapsulamentos distintos, variando de acordo com o tipo de antena, alcance do sinal, frequência de operação, etc.

A Figura 2.4 mostra alguns dos modelos dos rádios Xbee Série 2 produzidos pela *Digi International*.

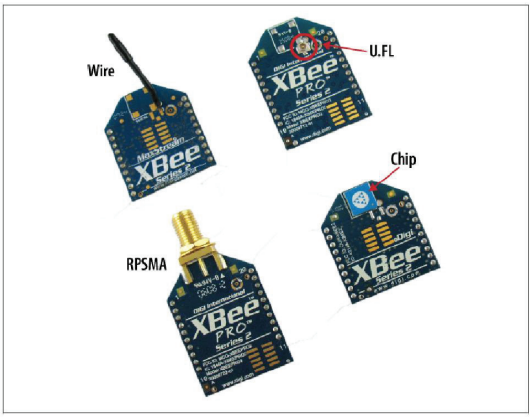


Figura 2.4: Modelos de dispositivos Xbee Série 2 – Fonte: Retirado de [Faludi, 2010].

Conclui-se que existe uma grande variedade Xbee's disponíveis no mercado, tornando interessante comparar os modelos existentes de modo a considerar alguns aspetos antes de escolher o modelo a utilizar. Na Tabela 2.2 podemos comparar os módulos Xbee existentes.

- **Alcance da antena**: É importante considerar a relação entre o alcance que os módulos podem atingir e as respetivas antenas utilizadas. Os módulos com antenas do tipo *chip* proporcionam um menor alcance, ocupam menos espaço e têm um menor custo face aos outros modelos.
- **Potência de transmissão vs. Consumo energético**: O modelo Xbee Pro têm uma potência de transmissão superior ao modelo Xbee proporcionando assim um maior alcance de transmissão. Importante dizer que maior potência significa também maior consumo energético, fazendo com que os módulos Xbee tenham um consumo de energia significativamente menor do que os modelos Xbee Pro.
- **Custo**: Como já referido, os modelos com antena externa e modelos da série Xbee Pro têm valores mais elevados.

Nesta dissertação a diferença de preço entre os modelos não é fator decisivo, uma vez que o preço do produto final será influenciado maioritariamente pelo valor dos sensores

Tabela 2.2: Comparação entre os módulos Xbee Série1 e Série2 – Fonte: Adaptado de [Digi, International, 2012]

Característica	**Xbee Série2**	**Xbee Pro Série2**	**Xbee Série1**	**Xbee Pro Série1**
Potência de Transmissão	1,25 mW	50 mW	1 mW	63 mW
Sensibilidade de Receção	-96 dBm	-102 dBm	-100 dBm	-102 dBm
Alcance Interno	40 m	90 m	30 m	90 m
Alcance Máximo	120 m	1,6 km	90 m	1,6 km
Consumo Máximo	40 mA	295 mA	45 mA	215 mA
Topologia	Ponto a Ponto, Ponto Multi-ponto, Malha	Ponto a Ponto, Ponto Multi-ponto, Malha	Ponto a Ponto, Ponto Multi-ponto	Ponto a Ponto, Ponto Multi-ponto
Tipos de Antenas	Wire, Chip, Conector UFL, RPSMA	Wire, Chip, Conector UFL, RPSMA	Wire, Chip, Conector UFL, RPSMA	Wire, Chip, Conector UFL, RPSMA
Valor do Equipamento	17.00 €	28.00 €	19.00 €	32.00 €

e não pelo valor dos dispositivos de comunicação. Desta forma foi escolhido o modelo Xbee Série2, com antena do tipo *chip* ou *wire*. A escolha foi efetuada devido ao facto de ser os dispositivos com consumo de energia e com menores dimensões.

2.3.3 Modo de operação do Xbee

Os módulos Xbee podem operar em 2 modos distintos: Modo transparente (AT) ou Modo *Application Programming Interface* (API)

- Modo AT: No modo de operação AT os dados são enviados ou recebidos *bit* a *bit*. Os dados recebidos da UART (*Universal Asynchronous Receiver/Transmitter*), serão diretamente transmitido por RF (Rádio Frequência) para o Xbee de destino. Já os dados recebidos vão diretamente do canal RF para o pino de saída de dados do microprocessador DO (TX). Os Xbee dispõem de buffers de transmissão e receção para melhorar a performance da comunicação. [Messias, 2013]. O modo AT é um modo de simples configuração e utilização, envolve poucos dados de endereçamento e não precisamos de nos preocupar com a interpretação dos dados a enviar, sendo o suficiente para várias aplicações de RSSF. Entretanto o modo AT não permite utilizar as entradas e saídas digitais do Xbee, não permite identificar o endereço de origem da mensagem recebida, nem receber mensagem de confirmação ou de falha de transmissão.
- Modo API: No modo API os dados enviados e recebidos estão contidos em pacotes de dados (*frames*), esses pacotes definem operações ou eventos no Xbee. Com este modo de operação obtemos algumas vantagens em relação ao AT, passo a citar:
  - Transmite dados para múltiplos destinos sem necessidade de reconfiguração do *firmware* do rádio Xbee. Apenas é necessário especificar o endereço de destino na *frame* de envio.
  - É possível solicitar confirmação de receção quando enviamos uma mensagem.
  - Cada *frame* recebida contem o endereço de origem da mensagem.
  - Possibilidade de configuração remota dos nós da rede.

A Figura 2.5 dá uma ideia geral da estrutura de uma *frame* API.

O primeiro *byte* é um delimitador de início e possui um valor fixo 0xE7, qualquer byte que chegue antes do delimitador é descartado. Seguidamente os *bytes* 2 e 3 indicam o tamanho da *frame* a receber.

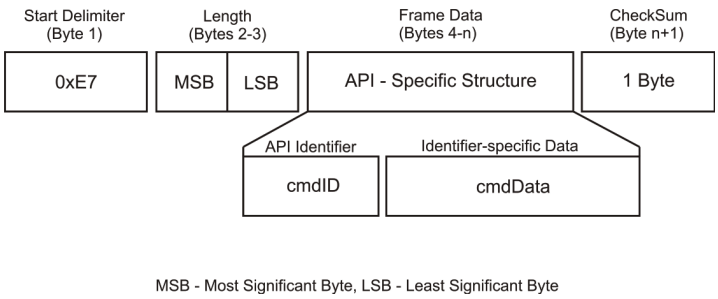


Figura 2.5: Estrutura Geral de uma *frame* API – Fonte: Adaptado de [Digi, International, 2012]

Do *bytes* 4 ao *n* contém a informação que desejamos que seja enviada, além de um identificador API e outro específico de mensagem. Esses dois identificadores tem como função determinar o tipo de mensagem que está a ser enviada ou recebida.

Por último, temos um *byte* para verificação da integridade da mensagem. A verificação é feita a partir de uma soma de verificação (*checksum*). Caso o valor do um *checksum* esteja incorreto uma *frame* de status é retornada indicando erro. É de notar que a parte mais importante do modo API é a *frame data* e, assim sendo, vamos aprofundar um pouco mais o estudo do mesmo:

A *Digi International* especificou 14 tipos de *frames* para utilização no modo de operação API.

Na Tabela 2.3 podemos observar quais são esses tipos e seus respetivos valores:

Tabela 2.3: Tipos de *frames* do modo API – Fonte: Adaptado de [Digi, International, 2012]

Tipos de *Frame* API	**Valor**
Modem Status	0x8A
AT Command	0x08
AT Command . Queue Parameter Value	0x09
AT Command Response	0x88
Remote Command Request	0x17
Remote Command Response	0x97
ZigBee Transmit Request	0x10
Explicit Addressing ZigBee Command Frame	0x11
ZigBee Trasmit Status	0x8B
ZigBee Receive Packet (A0=0)	0x90
ZigBee Explicit Rx Indicator (A0=1)	0x91
ZigBee IO Data Sample	0x92
Xbee Sensor Read Indicatoor (A0=0)	0x94
Node Identification Indicator	0x95

Cada tipo de *frame* tem uma finalidade específica e uma estrutura própria de dados, mas 2 tipos distintos podem relacionar-se entre si. Por exemplo, uma *frame Remote Command Request* pode trocar informação com uma *Remote Command Response*.

A estrutura das *frames* utilizadas nesta dissertação serão explicadas seguidamente:

***ZigBee Transmit Request* (0x10) -** *frame type* utilizada para enviar ate 72 bytes de informação em *broadcast* (todos os dispositivos da rede) ou a um único dispositivo especifico da rede.

A estrutura completa da *frame* 0x10, assim como todos os dados necessários para o seu envio pode ser visto na Figura 2.6.

***Frame Type*-** Define o tipo da *frame*, neste caso deve ser 0x10.

***Frame* ID-** Identificador da *frame*, uma mensagem de resposta retornará com a mesma *frame* ID, de forma a confirmar que a mensagem foi entregue ao destino. Caso o *frame* ID seja determinado como 0x0, a solicitação de confirmação será ignorada.

***64-bit Destination Address*-** Endereço de destino, tem um comprimento de 64 *bits* e é único para cada dispositivo Xbee.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x16	
	Frame-specific Data	Frame Type	3	0x10	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following address is also supported: 0x000000000000FFFF - Broadcast address
			6	0x13	
			7	0xA2	
			8	0x00	
			9	0x40	
			10	0x0A	
			11	0x01	
			LSB 12	0x27	
		Reserved	13	0xFF	Set to 0xFFFE.
			14	0xFE	
		Broadcast Radius	15	0x00	Sets maximum number of hops a broadcast transmission can occur. If set to 0, the broadcast radius will be set to the maximum hops value.
		Transmit Options	16	0x00	Bitfield: bit 0: Disable ACK bit 1: Don't attempt route Discovery. All other bits must be set to 0.
		RF Data	17	0x54	Data that is sent to the destination device
			18	0x78	
			19	0x44	
			20	0x61	
			21	0x74	
			22	0x61	
			23	0x30	
			24	0x41	
	Checksum		25	0x13	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Figura 2.6: API ZigBee *frame Transmit Request* - Fonte: Retirado de [Digi, International, 2012].

RF *Data*- Mensagem de dados a ser enviada.

***Zigbee Receive Packet* (0x90) -** *frame type* de receção de dados. A estrutura completa da *frame* 0x90, assim como todos os dados necessários para a sua receção podem ser observados na Figura 2.7.

***64-bits Source Address*-** Campo que indica o endereço da fonte que transmitiu o pacote de dados.

***Received Data*-** Mensagem de dados recebida.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x11	
	Frame-specific Data	Frame Type	3	0x90	
		64-bit Source Address	MSB 4	0x00	
			5	0x13	64-bit address of sender. Set to 0xFFFFFFFFFFFFFFFF (unknown 64-bit address) if the sender's 64-bit address is unknown.
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender
			LSB 13	0x84	
		Receive Options	14	0x01	0x01 - Packet Acknowledged 0x02 - Packet was a broadcast packet 0x20 - Packet encrypted with APS encryption 0x40 - Packet was sent from an end device (if known) Note: Option values can be combined. For example, a 0x40 and a 0x01 will show as a 0x41. Other possible values 0x21, 0x22, 0x41, 0x42, 0x60, 0x61, 0x62.
		Received Data	15	0x52	Received RF data
			16	0x78	
			17	0x44	
			18	0x61	
			19	0x74	
			20	0x61	
	Checksum		21	0x0D	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Figura 2.7: API ZigBee *frame Receive Packet* - Fonte: Retirado de [Digi, International, 2012].

2.4 Microcontrolador ATMEGA 328

2.4.1 Introdução

Um microcontrolador é um circuito integrado de pequenas dimensões que contém os elementos essenciais de um computador: Processador, memória, periféricos, interfaces de comunicação, temporizadores, (...)

O ATMEGA 328 é um micro-controlador de 8 *bits* baseado na arquitetura RISC *Reduced Instruction Set Computer*, é produzido pela *Atmel Corporation* e faz parte da família de microcontroladores AVR.

Uma vez que, devido às suas características e boa relação custo/benefício, foi o microcontrolador utilizado no protótipo final.

2.4.2 Características do ATMEGA 328

- Baixa potência.
- Alto desempenho.

- Arquitetura RISC avançada.
- Executa 131 instruções, a maior parte num único ciclo de relógio.
- 32x8 registadores de trabalho.
- Taxa de transferência de até 20 MIPS (*Million Instructions Per Second*) a 20 MHz.
- 32 kbytes de memória de programa flash de auto programação.
- 1 kbyte de memória EEPROM.
- 2 kbytes de memória SRAM.
- Ciclos "write/erase": memória flash 10.000 vezes, EEPROM 100.000 vezes.
- Bits de bloqueio para proteção do software.
- Tensão de operação: 1,8 V - 5,5 V.
- 23 entrada/saídas programáveis.
- 8 canais ADC (*Analog/digital converter*) com precisão de 10 *bits* na versão TQFP[2] ou 6 canais ADC com precisão de 10 *bits* na versão PDIP[3].
- Intefaces *Serial*: USART (*Universal Synchronous Asynchronous Receiver Transmitter*), SPI (*Serial Peripheral Interface*) *Master/Slave* e *two-wire* compatível com o protocolo I^2C.
- Comparador Analógico.
- Oscilador interno (excluindo a necessidade de qualquer outra fonte externa de *clock*).
- Fontes de interrupções internas e externas.
- Seis modos *sleep*: Idle, ADC *noise reduction*, *power-save*, *power-down*, *standby* e *extended standby*.
- Tem incorporado um circuito digital ALU (*Arithmetic Logic Unit*) de alto desempenho. Este está conectado diretamente aos 32 registadores, o que torna possível aceder a dois deles num único ciclo de *clock*.
- Para maximizar o desempenho de processamento, a CPU (*Central Processing Unit*) utiliza a *Harvard Architecture*.[4] Desta forma, enquanto uma tarefa está a ser realizada, a seguinte é solicitada na memória de programas, o que possibilita a execução de uma instrução por ciclo de *clock*.

[2] *Thin Quad Flat Package*, encapsulamento quadrado utilizado nos circuitos integrados em que os seus terminais se estendem igualmente pelos 4 lados do chip.

[3] *Dual In-Line Package*, encapsulamento utilizado nos circuitos integrados em que os terminais são duas linhas paralelas.

[4] Arquitetura de Computador que separa as memórias de dados e programas, permitindo o acesso a ambas em simultâneo.

2.5 Amostragem no tempo

Amostragem é o processo de representar um sinal analógico (contínuo no tempo), em um conjunto discreto de amostras periódicas. Este processo tem como objetivo formar uma função $\hat{x}$(t), a partir da função inicial conhecida x(t). Para conseguir determinar $\hat{x}$(t), multiplica-se a função inicial x(t) por uma série periódica de impulsos unitários (impulsos de Dirac), como descrito na Figura 2.8.

2.5.1 Conversor A/D

É um Dispositivo capaz de converter um sinal analógico em um sinal digital utilizando a técnica de amostragem. Conforme descrito na Figura 2.9, um conversor A/D efetua uma amostragem do sinal, seguidamente faz uma quantificação de forma a aproximar o valor amostrado de um dos 2^n (n - número de *bit's* do conversor) níveis possíveis e atribui um código binario para o respetivo nível. A quantidade de níveis do ADC, determina a sua resolução, como pode ser observado na Figura 2.10

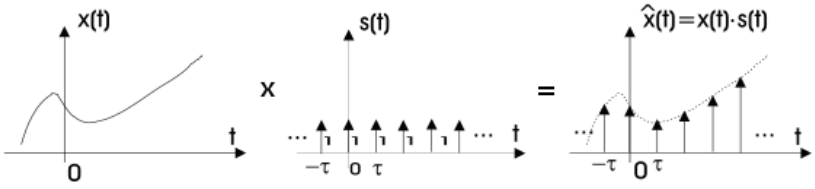


Figura 2.8: Processo de Amostragem - Fonte: Adaptado de [Jesus, 2005].

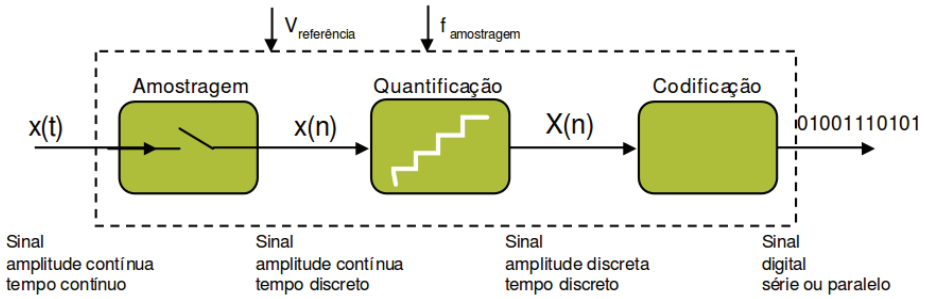


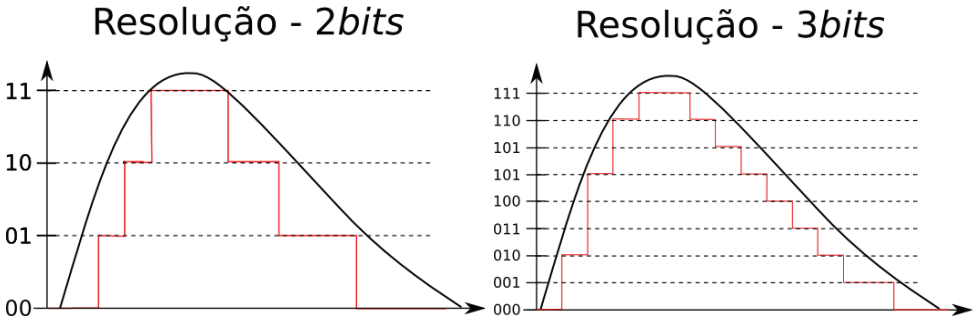
Figura 2.9: Processo de funcionamento de um conversor A/D - Fonte: Retirado de [Tavares and Silva, 2005].

2.5.2 Sobreamostragem (*Oversampling*)

Oversampling é o processo de amostrar um sinal com uma frequência significativamente maior do que a de Nyquist.[5] Tendo como objetivo aumentar a resolução do conversor A/D.

Para que o processo de oversampling funcione corretamente, o sinal a ser convertido não deve variar durante a amostragem, por isso é necessário haver algum ruído no

[5]Teorema de Nyquist prova que é necessário amostrar um sinal, com uma frequência duas vezes superior a maior frequência contida no sinal, para que o mesmo possa ser recuperada. Esta frequência é denominada frequência de Nyquist.



(a) Resolução ADC 2 *bits* (b) Resolução ADC 3 *bits*

Figura 2.10: Resolução de um conversor A/D para 2 e 3 *bits*

sinal, distribuído de forma uniforme entre as amostras. Pode parecer contraditório mas como o ruído idealmente tem 1 *bit*, este vai incidir apenas no LSB (*Least Significant Bit*), fazendo com que o valor final seja um valor médio entre todas as amostras. Se o sinal retornar sempre o mesmo valor, o valor médio será igual a uma amostra, não acrescentando nada ao sinal sobreamostrado.

2.6 Plataforma Arduino

2.6.1 Introdução

Arduino é uma plataforma *open-source* para desenvolvimento de protótipos eletrónicos. A ideia surgiu em Itália pelas mãos do professor Massímo Banzi, que necessitava de uma placa para projetos escolares menos dispendiosa do que as existentes no mercado. A plataforma Arduino é composta por *hardware* e *software*. O *hardware* é composto por um micro-controlador ATMEGA, *chip* de comunicação USB (*Universal Serial Bus*), reguladores de tensão, oscilador, entre outros. O *software* consiste numa IDE (*Integrated Development Environment*) onde é possível compilar e introduzir códigos escritos em C/C++ para placa de desenvolvimento.

2.6.2 Hardware Arduino

Existem no mercado vários modelos de Arduino e a implementação do hardware varia em cada versão. Nesta dissertação foi utilizado o Arduino Duemilanove, assim falaremos sobre este em específico.

Na Figura 2.11 podemos observar o Arduino Duemilanove e os seus principais componentes.

Característica principais

A Tabela 2.4 apresenta as principais características do Arduino Duemilanove.

Alimentação

O Arduino Duemilanove pode ser alimentado pela porta USB ou por uma fonte de alimentação externa. Apesar de a placa operar com tensões externas de 6 a 20 V, se a

tensão for menor que 7 V, o regulador de tensão pode fornecer menos de 5 V e a placa pode operar de forma instável. Por outro lado, se a fonte externa fornecer mais de 12 V o regulador de tensão pode sobreaquecer e avariar o circuito.

Memória

A memoria é nativa do ATmega328, como já mencionado na Secção 2.4 este possui 32 kB de memória *flash*, 2 kB de SRAM e 1 kB de EEPROM que pode ser utilizada para guardar dados permanentemente. Os dados gravados nesta memória permanecem mesmo que a placa seja desligada, tal coisa não acontece com a memória SRAM, que apaga cada vez que o circuito é desligado.

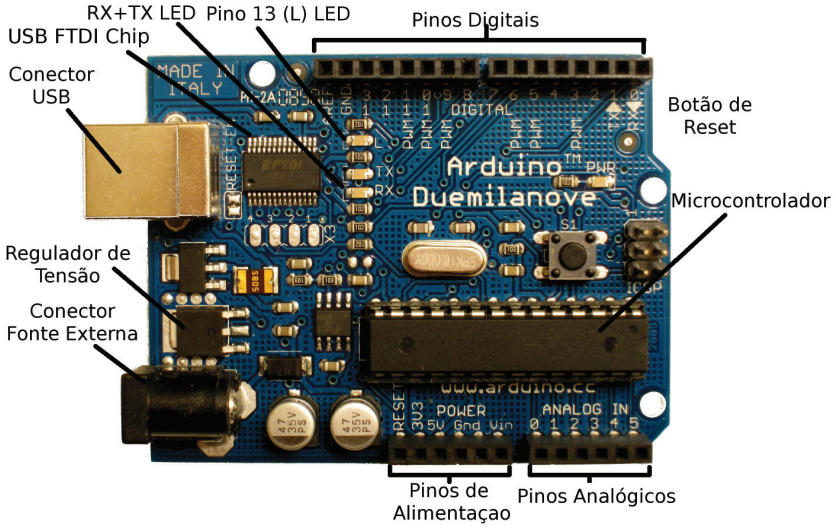


Figura 2.11: Placa Arduino Duemilanove - Fonte: Adaptado de [Arduino, 2013].

Tabela 2.4: As principais características do Arduino Duemilanove - Fonte: [Arduino, 2013]

Arduino Duemilanove	
Microcontrolador	ATmega328
Tensão de funcionamento	5V
Tensão de alimentação (recommended)	7-12V
Tensão de alimentação (limits)	6-20V
Pinos de entrada/saída digital	14 (6 podem ser saídas PWM)
Pinos de entrada analógica	6
Corrente DC por pino I/O	40 mA
Corrente DC para o pino 3.3V	50 mA
Memória flash	32 kB - 2 kB utilizados para o bootloader
SRAM	2 kB
EEPROM	1 kB
Frequência de Clock	16 MHz

Entradas e saídas

O Duemilanove possui catorze pinos digitais, que podem ser configurados para funcionar como entrada ou saída. Funcionando a 5 V são capazes de fornecer ou receber uma corrente máxima de 40 mA. Entre os 14 pinos digitais alguns possuem funções especiais, como por exemplo:

- Os pinos de receção e transmissão de dados 0 (RX) e 1 (TX), são capazes de efetuar uma comunicação do tipo *Serial* TTL (*Transistor-Transistor Logic*)[6], enviando dados pelo pino 1 (TX) e recebendo pelo pino 0 (RX).
- Os pinos 3, 5, 6, 9, 10 e 11, podem fornecer uma saída analógica PWM (*Pulse Width Modulation*) de 8-bits, sendo capazes de fornecer uma onda quadrada com *duty cicle* variável, permitindo assim o controlo da tensão fornecida.
- Nos pinos 10 (SS) (*Slave Select)*), 11 (MOSI) (*Master Out Slave In*), 12 (MISO) (*Master In Slave Out*), 13 (SCK) (*Serial Clock*) é possível realizar uma comunicação SPI.

Para além dos pinos digitais acima citados, contamos ainda com seis entradas analógicas. Cada uma destas possui o seu próprio conversor A/D (Analógico/Digital) de 10 *bits*. Os conversores A/D fornecem 1024 níveis discretos com tensões de referência variável: 5 V; 3,3 V; 1,1 V, ou uma tensão externa. A tensão é escolhida utilizando a função *analogReference*().

Software Arduino

A Arduino desenvolveu o seu próprio ambiente de programação, onde facilmente conseguimos escrever códigos e enviar para o Duemilanove ou qualquer outro modelo produzido. O ambiente pode ser instalado em muitos sistemas operativos através do site: *http://arduino.cc/en/Main/Software*. A linguagem de programação é própria, mas utiliza a linguagem C++ como referencia. Toda a estrutura da declaração de variáveis, ponteiros, vetores, funções é idêntica a do C++. O *software* dispõe de vários exemplos comentados e um vasto manual facilitando assim a aprendizagem da programação.

Shield

Apesar de todas as funcionalidades já citadas, o Arduino pode ter as suas capacidades expandidas ao utilizar *shields*. *Shields* são placas que podem ser conectadas no topo do Arduino, sendo fáceis de montar e com um menor custo de produção.

Existem no mercado *shields* para as mais diversas funções, como por exemplo para efetuar comunicação móvel, adicionar um display, ler cartões de memória, adicionar uma placa *ethernet*, entre outras. Nesta dissertação utilizaremos o *Xbee Shield* para facilitar a ligação entre o Arduino Duemilanove e o Xbee, dispensando assim uma placa de circuito externa capaz de fornecer as condições necessárias para a comunicação entre o Duemilanove e o Xbee.

[6]Tipo de comunicação que trabalha a 5 V e utilizam transístores bipolares em sua construção.

Xbee Shield

A placa Xbee Shield foi desenvolvida pela *Libellium Company* e permite conectar de forma simples e segura o Xbee ao Arduino, facilitando a utilização do Arduino em RSSF. O Xbee Shield possui dois *jumpers*[7], que nos permite selecionar se a comunicação *serial* deve ser efetuada entre o Xbee e o microcontrolador ou entre o Xbee e a porta USB. Dependendo da posição dos *jumpers* é possível utilizar o Arduino mais Xbee como um dispositivo autónomo, capaz de ler sensores, ativar atuadores e enviar dados remotamente numa RSSF, ou quando conectado a um computador através da porta USB, seja utilizado como um *gateway*, encaminhando as informações da RSSF para o computador. O *shield* disponibiliza um botão de *reset* e conexões para os pinos analógicos e digitais, com o intuito de substituir os que ficaram ocultos no Arduino. Um regulador de tensão garante o fornecimento de apenas 3,3 V para o rádio Xbee permitindo operar de forma segura e estável.

Na Figura 2.12 podemos observar o Xbee Shield para melhor entender a informação acima descrita.

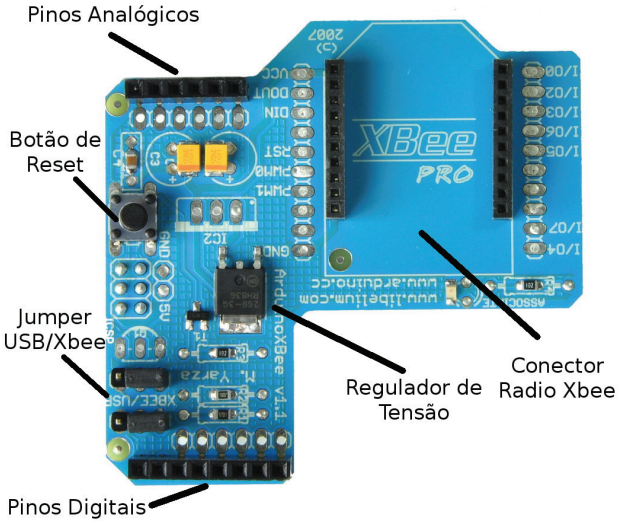


Figura 2.12: Placa Xbee Shield e seus principais componente

.

2.7 Sensores e atuadores

2.7.1 Introdução

"Sensores e atuadores são dispositivos capazes de traduzir uma informação, “sinal”, do domínio físico para o domínio eletrónico". [Stallinga, 2011]. Sensores e atuadores são utilizados quotidianamente em tudo que nos rodeia, seja um simples atuador que acende a lâmpada quando entramos na casa de banho, ou num caso mais complexo, um sensor capaz de medir os batimentos cardíacos de um doente durante uma cirurgia.

[7] Pequena peça que curta eletricamente os pontos a qual esta conectada.

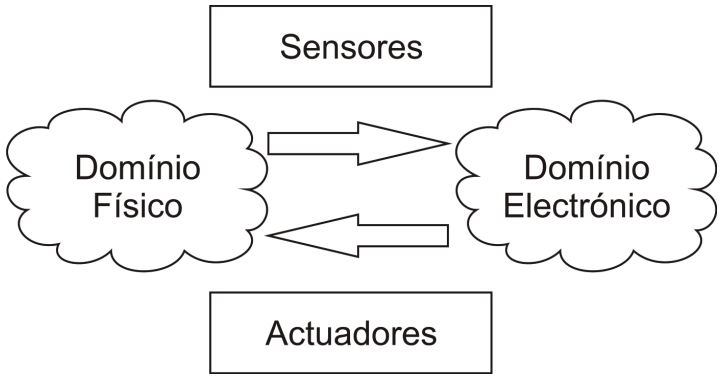


Figura 2.13: Transdutores entre o domínio físico e eletrónico – Fonte: Adaptado de [Stallinga, 2011]

.

2.7.2 Sensores

Dispositivos capazes de interpretar uma grandeza física, medindo-a e convertendo-a numa informação que possa ser compreendida por um observador ou dispositivo eletrónico.

Assim como qualquer instrumento de medição, um sensor tem características importantes que devem ser tidas em consideração na hora da escolha do sensor mais indicado para uma dada aplicação. Algumas dessas características são:

Gama de Funcionamento - é o intervalo da grandeza física que um instrumento consegue medir. "A gama de medida relativa será a comparação relativa entre valor mínimo e valor máximo medíveis"[Azinheira, 2006].

Exatidão (*Accuracy*) - é o majorante do erro absoluto, ou seja, a diferença máxima entre o valor real e o valor indicado na saída do sensor.

Histerese - "um sensor deve ser capaz de seguir as mudanças do parâmetro de entrada indiferente da direção em que a variação é feita. A histerese é a medida dessa propriedade"[Brown, 2012].

Sensibilidade - é definida como a inclinação da curva característica de saída (dY/dX). Por outras palavras, determina o quão sensível um sensor é a um respetivo parâmetro físico.

Tempo de resposta - os sensores não variam instantaneamente quando ocorre uma variação no parâmetro de entrada. Mudam para o novo estado ao longo do tempo, esse período de tempo é chamado tempo de resposta.

Além das características mencionadas anteriormente, pode ser importante ter em consideração também outras características como por exemplo: preço; facilidade de manutenção; dimensões; encapsulamento; durabilidade; tempo de vida útil, entre outras. Apesar da quantidade de características, os sensores podem ser de dois tipos distintos, analógicos ou digitais. Essa distinção é feita de acordo com a forma como o sensor responde à variação do parâmetro físico.

Os sensores analógicos são assim chamados porque se baseiam em eletrónica analógica. "Sinais analógicos são aqueles que, mesmo limitados entre dois valores de tensão,

podem assumir infinitos valores intermediários". [Patsko, 2006]. Teoricamente, para cada mínima variação da característica física à entrada do sensor, haverá uma variação correspondente em tensão na saída do sensor. Por exemplo o LM35, um dispositivo cujo a tensão de saída varia 10 mV para cada variação de 1 °C. Se submetermos o dispositivo a uma fonte de calor, o valor da tensão a saída do dispositivo irá aumentar gradualmente, até um limite máximo como pode ser observado na reta (a) da Figura 2.14.

Um sensor digital tem na sua saída um valor discreto. Na lógica binária (base do funcionamento dos sistemas digitais), assumem valores "0" ou "1" para determinar se o mesmo está ligado ou desligado. Como no exemplo da *Figura* 2.15 onde temos uma chave digital. Quando a mesma está fechada, assume um valor *high* determinando que o circuito está ligado, no momento em que é aberta assume um valor *low*, determinando que o circuito está desligado.

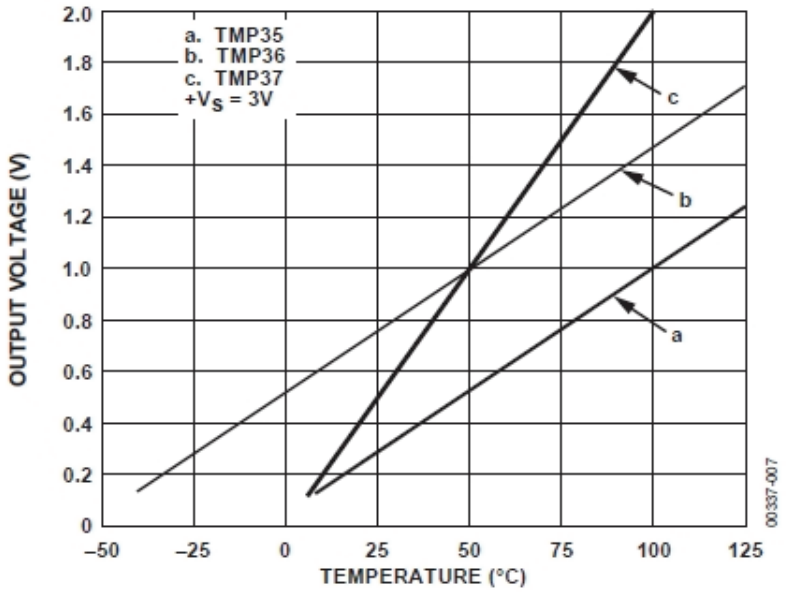


Figura 2.14: Gráfico da relação Tensão vs. Temperatura dos sensores a. TMP35, b. TMP36, c. TPM37 - Fonte: Retirado de [Analog Devices, 2013]

.

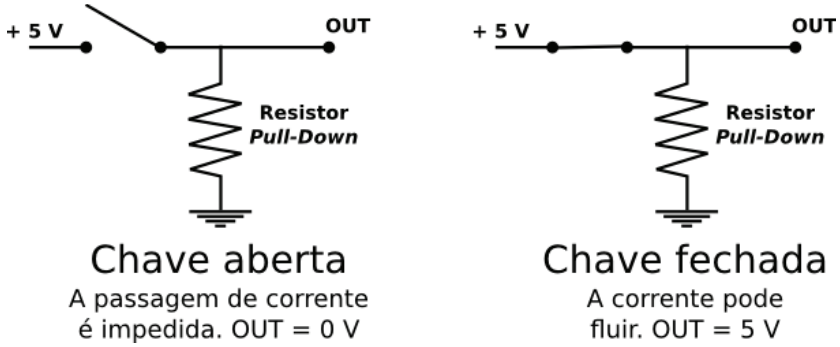


Figura 2.15: Esquema de funcionamento de uma chave digital.

2.7.3 Sensores analógicos

De forma a completar o estudo dos sensores analógicos falaremos de alguns sensores específicos utilizados nesta dissertação, assim como o seu princípio de funcionamento.

Sensores de temperatura

"A temperatura é um dos parâmetros mais importantes mas também mais complexos da física"[Azinheira, 2006]. Todos os dias controlamos e medimos a temperatura de diversos meios e objetos e para esse fim há varios sensores disponíveis no mercado, entre os quais:

Termístores - sensores resistivos de material semicondutor destinado a medir a variação de temperatura. Sua resistência varia fortemente com a temperatura, dando origem a umas das suas principais características, ser um sensor com excelente sensibilidade e elevado sinal de saída.

Nos termístores, a relação entre a temperatura e a resistência pode ser aproximada de uma curva exponencial. Essa curva pode ser positiva ou negativa, dependendo se o termístor é do tipo PTC (*Positive Temperature Coefficient*) ou NTC (*Negative Temperature Coefficient*). Na Figura 2.16 é possível observar a relação temperatura vs. resistência para termístores NTC e PTC.

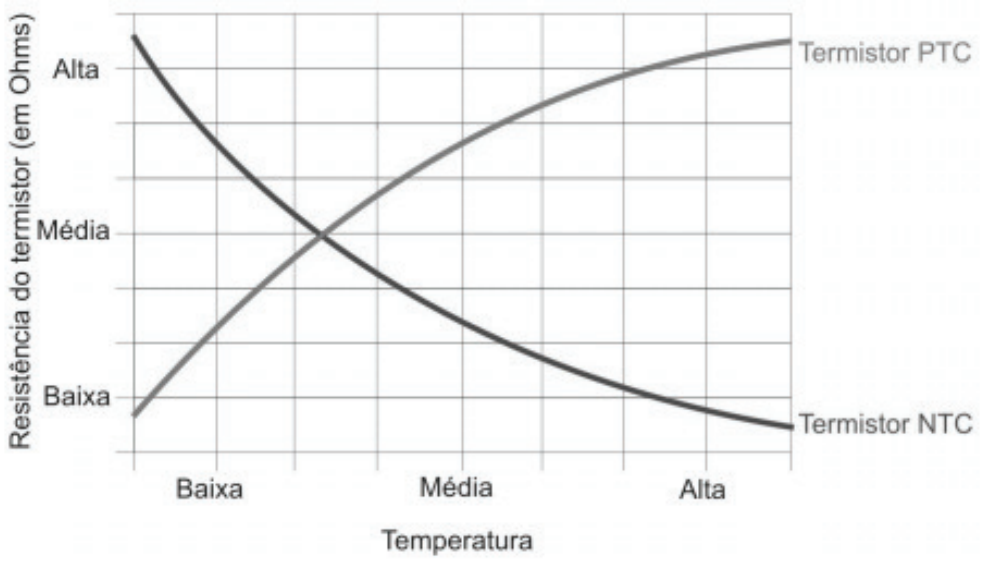


Figura 2.16: Relação resistência vs. temperatura dos termístores PTC e NTC - Fonte: Retirado de [Patsko, 2006]

Circuito Integrado LM35 - sensores que utilizam o mesmo princípio de funcionamento dos termístores, o comportamento dos materiais semicondutores serem dependentes da temperatura. A vantagem desses circuitos é que num único encapsulamento, contém todos os componentes necessários para determinar a temperatura. Um simples exemplo dessa aplicação é um díodo de junção pn, onde a relação tensão-corrente depende da temperatura de forma exponencial. Essa relação pode ser calculada a partir da equação de "Ebers-Moll":

$$I(V,T) = I_s(T)\left[e^{\frac{qV}{kT}} - 1\right] \Rightarrow V = \frac{kT}{q}\log\left[1 + \frac{I}{I_s(T)}\right] \tag{2.1}$$

$I_s \rightarrow$ Corrente de saturação
$k \rightarrow$ Constante de Boltzmann
$q \rightarrow$ Carga do eletrão
$T \rightarrow$ Temperatura

O LM35 é um circuito integrado complexo que utiliza de díodos, transístores e amplificadores para obter uma relação entre a tensão e a temperatura linear de 10 mV/°C. Na Figura 2.17, é possível observar o circuito interno do integrado LM35.

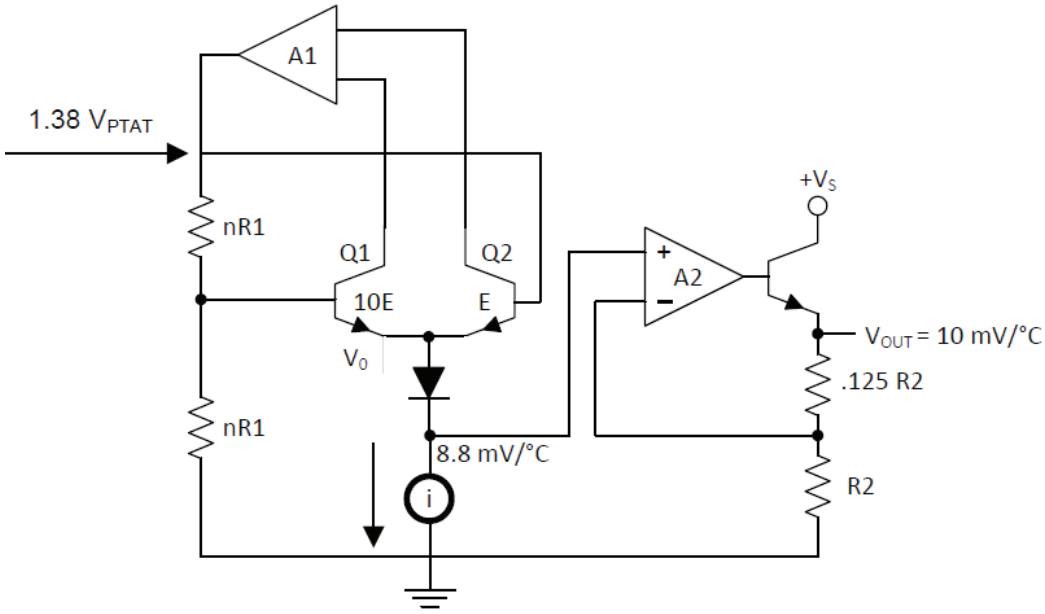


Figura 2.17: Esquema do circuito interno do LM35 - Fonte: Retirado de [Texas Instruments, 2013]

Circuito Integrado LM34 - Integrado de funcionamento idêntico ao do LM35. Como se pode confirmar na Figura 2.18, as únicas alterações são os valores de alguns componentes, de forma a fornecer na saída uma relação de 10 mV/°F. Uma vantagem do LM34 face ao LM35 é a possibilidade de medir valores de temperaturas abaixo de 0 °C de forma direta. Para o para o LM35 0 °C corresponde a 0 mV e para o LM34 temos que 0°F corresponde a 0 mV e se a relação entre as escalas de temperatura é dada pela equação:

$$C = \frac{F - 32}{1{,}8} \tag{2.2}$$

$C \rightarrow$ Temperatura em graus Celcius
$F \rightarrow$ Temperatura em graus Fahrenheit

Desta forma, ao utilizar o LM34 podemos medir temperaturas mínimas na faixa de -17,7 °C.

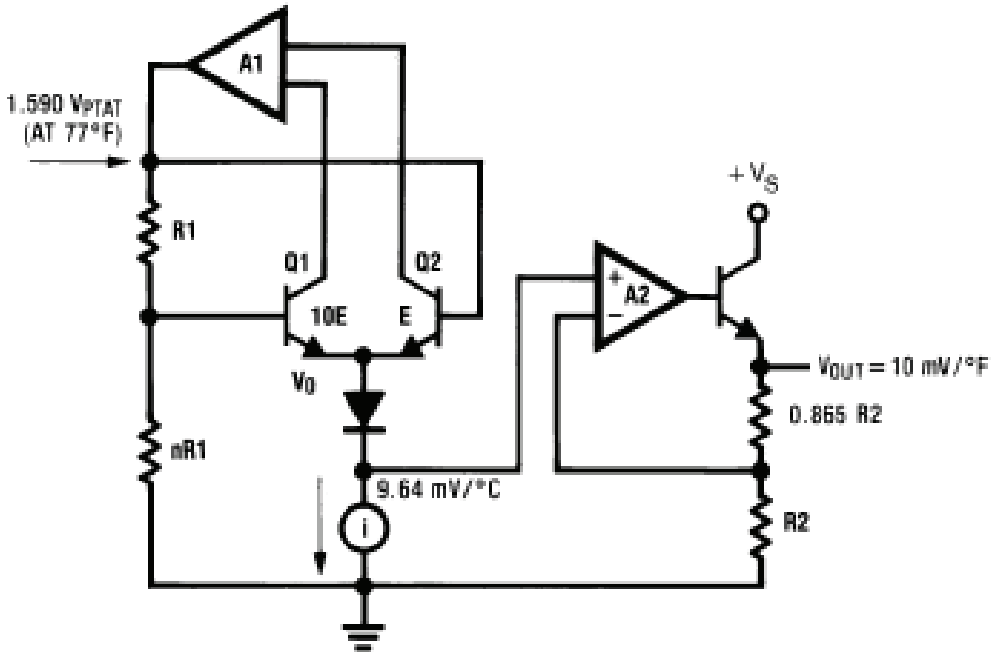


Figura 2.18: Esquema do circuito interno do LM34 - Fonte: Retirado de [Texas Instruments, 2013]

.

Sensor pH - O termo pH (Potencial de Hidrogénio), deriva da combinação "p" potencial e "H" símbolo químico do elemento hidrogénio. Tem como objetivo indicar a acidez, neutralidade ou alcalinidade de uma solução aquosa.

Segundo [Stallinga, 2012], matematicamente o "p" é definido como o logaritmo de base 10 da concentração de iões de hidrogénio em solução. Em água a molécula de hidrogénio está associada à molécula de água da seguinte maneira,

$$\mathrm{pH} \equiv -\log_{10}\left(\frac{[\mathrm{H_3O^+}]}{\mathrm{mol/L}}\right) \tag{2.3}$$

A partir da equação anterior podemos observar que, devido ao sinal negativo, quanto maior o valor do pH menor será a concentração de iões.

As moléculas de água dissociam-se, onde as ligações se quebram ocasionalmente devido a energia térmica. Isso acontece da seguinte forma,

$$\mathrm{H_2O} \rightleftharpoons \mathrm{H^+ + OH^-} \tag{2.4}$$

Agora assumindo que tínhamos 2 moléculas de água, podemos reescrever a equação anterior,

$$2H_2O \rightleftharpoons H_3O^+ + OH^- \tag{2.5}$$

A constante de reação é dada pela seguinte equação,

$$K = \frac{[H_3O^+][OH^-]}{[H_2O][H_2O]} \tag{2.6}$$

A concentração de água pode ser considerada constante, sendo definida da seguinte maneira,

$$K_w = [H_3O^+][OH^-] \tag{2.7}$$

Considerando que a temperatura ambiente é de 298 kelvin (aproximadamente 25 °celcius), o valor de K_w é constante,

$$K_w = 1{,}00 \times 10^{-14}\ \mathrm{mol}^2/\mathrm{L}^2 \tag{2.8}$$

O sensor de pH determina a concentração de iões de hidrogénio, medindo o potencial elétrico de uma determinada reação química, desde que o potencial elétrico seja dependente da concentração de iões. Normalmente utiliza-se a seguinte dissociação de iões de silício e água,

$$SiO_2 + 4H_3O^+ \rightleftharpoons Si^{4+} + 6H_2O \tag{2.9}$$

Na Figura 2.19 podemos observar o diagrama esquemático de um sensor de pH numa solução aquosa. Ele mede a diferença de potencial da reação química que ocorre dos dois lados da membrana de vidro. Num dos lados, o pH é constante e de valor neutro (pH = 7), do outro lado está exposto a solução aquosa. Idealmente a membrana é um isolante, não permitindo a condução elétrica, no entanto na realidade ocorre de forma diferente. Para que o sensor funcione, pequenas correntes passam sob a forma de protões através da membrana. O voltímetro mede o potencial elétrico originado pela diferença de potencial químico que ocorre entre os dois lados da membrana.

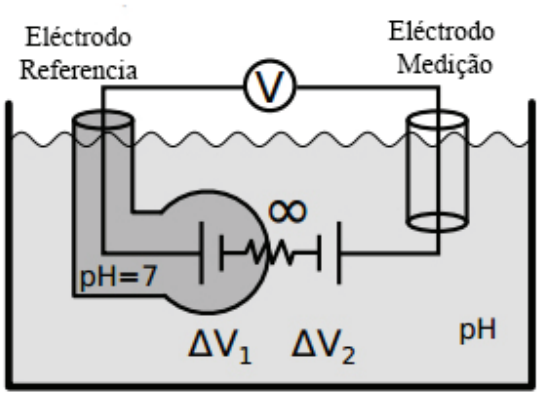


Figura 2.19: Esquema de funcionamento de um sensor de pH - Fonte: Retirado de [Stallinga, 2012]

O potencial químico μ_a é equacionado da seguinte forma,

$$\mu_a = \mu_{\mathrm{A},0} + RT \ln\left(\frac{[A]}{\mathrm{mol/L}}\right) \tag{2.10}$$

$R \rightarrow$ Constante molar
$T \rightarrow$ Temperatura absoluta

O potencial elétrico relaciona-se com o potencial químico da seguinte maneira,

$$V = \Delta V_1 - \Delta V_2 = \tag{2.11}$$

$$V = \Delta V_1 - \Delta V_2 = \frac{RT}{qN_A}\left(\ln\left(\frac{[\mathrm{H_3O^+}]}{\mathrm{mol/L}}\right) - \ln\left(10^{-7}\right)\right) = \frac{RT}{qN_\mathrm{A}}\ln(10) \times (\mathrm{pH} - 7) \tag{2.12}$$

Com base nas equações anteriores podemos determinar a sensibilidade do sensor. Para uma temperatura constante de 298 K temos,

$$S \equiv \frac{\mathrm{d}V}{\mathrm{d}pH} = \frac{RT}{qN_\mathrm{A}}\ln(10) = 59{,}13 \text{ mV/pH} \tag{2.13}$$

Sensor de ORP (*Oxidation Reduction Potential*) - Também conhecido como "potencial redox"ou oxidação-redução é a espontaneidade de uma espécie química adquirir eletrões e assim ser reduzido. Para que uma reação redox aconteça, deve haver uma espécie que ceda eletrões (redutor) e uma que aceite eletrões (oxidante). Cada espécie química possui um potencial redox próprio e quanto maior for este valor, maior será a tendência da espécie para adquirir eletrões.

O Sensor de ORP tem como função medir esse potencial de redução e utiliza o mesmo conceito do sensor de pH. Um elétrodo de referência (normalmente Ag/AgCl) é protegido por uma membrana de vidro, o outro lado está exposto à solução aquosa. O sensor mede o potencial de redução entre o elétrodo de referência e a solução aquosa.

Sensor de O_2 (elétrodo de Clark) - Utilizando o método polarográfico, um sensor de oxigénio pode efetuar a medição do oxigénio duma solução efetua-se por intermédio de vário tipos de elétrodos. O elétrodo de Clark é o mais utilizado para este processo e o seu funcionamento pode ser verificado de seguida:

Como demonstrado na Figura 2.20 o sensor é composto por um elétrodo de Pt (cátodo) e um elétrodo de referência Ag (ânodo), ambos imersos numa solução de cloreto de potássio. Uma membrana de *Polytetrafluoroethylene* (PTFE ou "Teflon') ou *Fluorinated ethylene propylene* (FEP), esta membrana isola eletricamente a solução deixando passar o oxigénio. Quando o oxigénio passa a membrana, acontecem as seguintes reações:

No cátodo de Pt:

$$\mathrm{O_2(g) + 4H^+(aq) + 4e^- \rightarrow 2H_2O} \tag{2.14}$$

No ânodo de Ag:

$$Ag(s) + Cl^{-}(aq) \rightarrow AgCl(s) + e^{-} \tag{2.15}$$

onde g significa gás e aq aquoso.

A célula eletroquímica é polarizada com uma pequena tensão (normalmente de 0,6 V) e a corrente desta reação é diretamente proporcional a concentração de oxigénio. Na Figura 2.21 podemos ver a possível relação entre a corrente e a concentração de oxigénio.

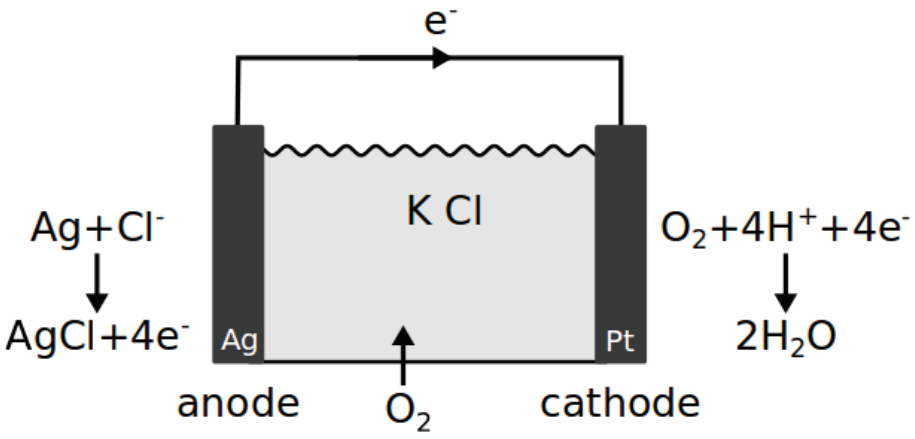


Figura 2.20: Esquema de funcionamento de um sensor de Clark - Fonte: Retirado de [Stallinga, 2012]

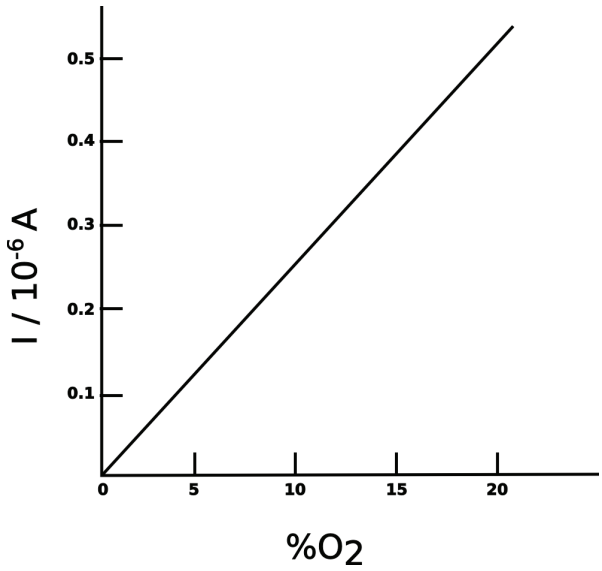


Figura 2.21: Relação entre a intensidade de corrente e a concentração de O_2 (para uma tensão na gama 0,4 - 0,7V) de um elétrodo de Clark imerso em solução aquosa. - Fonte: Adaptado de [Barreto, 2007]

2.7.4 Sensores digitais

Como mencionado anteriormente, sensores digitais são sensores que retornam na sua saída um valor digital. Além de uma chave digital que tem uma lógica binária simples, capaz de assumir apenas dois valores, um sensor digital pode ser muito mais complexo.

De forma a completar este conceito vamos demonstrar o funcionamento de alguns dos sensores digitais utilizados nesta dissertação.

Sensor integrado DS18B20

Apesar de neste sensor o processo de monitorização da temperatura ocorrer de forma semelhante à do LM34 ou LM35 onde há um sinal analógico, referente a uma temperatura, a diferença principal está no facto de conter um conversor analógico digital no seu interior, fazendo a conversão analógica-digital intrinsecamente. O DS18B20 além de possuir um ADC, é programável, podendo assumir de 9 a 12 *bits* de resolução. Outra vantagem deste integrado é o facto de utilizar o protocolo *1-wire* [8] para comunicação.

As principais vantagens na utilização do sensor DS18B20 são:

- O sinal resultante ser menos vulnerável a ruídos e interferências.
- Poder ser ligado diretamente a circuitos digitais.
- Comunicação de vários sensores num único fio, devido ao facto de cada sensor possuir um endereço próprio.

Na Figura 2.22 é possível observar o diagrama de blocos completo do sensor DS18B20.

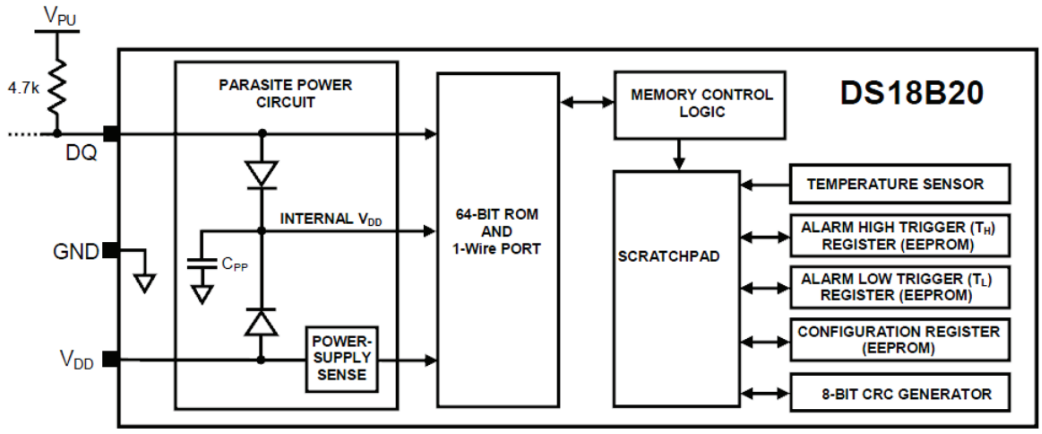


Figura 2.22: Diagrama de blocos do sensor de temperatura DS18B20 - Fonte: Retirado de [Maxim Integrated, 2013]

.

2.7.5 Sensor de ultrassom

São sensores que utilizam ondas sonoras de alta frequência, para detetar objetos ou movimento. Como demonstrado na Figura 2.23, um emissor, normalmente um piezoelétrico, excita uma película capaz de emitir uma onda ultrassónica de curta duração.

[8] Protocolo de comunicação desenvolvido pela Dallas Semiconductor, tem como principal característica enviar dados de vários sensores utilizando apenas um fio para comunicação

Essa onda propaga-se no ar e reflete-se no objeto a ser detetado, retornando então uma onda de reflexão que será captada pelo recetor, podendo assim detetar um objeto ou até mesmo a distância entre o objeto e o sensor.

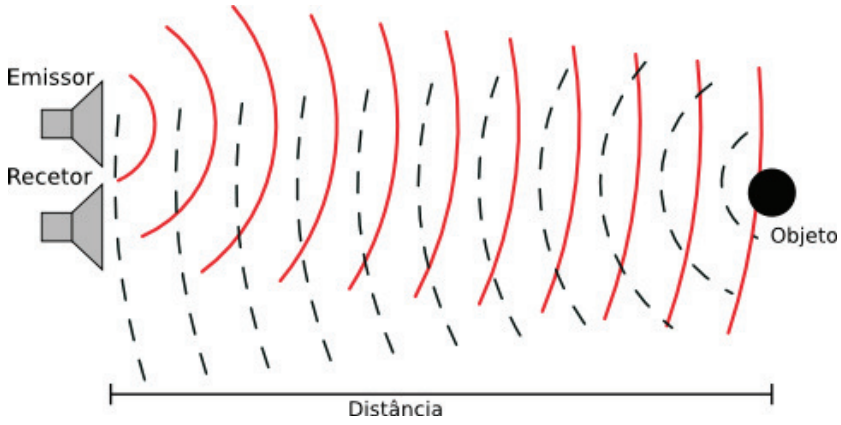


Figura 2.23: Princípio de funcionamento de um sensor ultrassom

.

2.7.6 Atuadores

Atuadores são dispositivos que funcionam de forma inversa aos sensores. Em vez de traduzir uma grandeza física num sinal elétrico, é capaz de transformar um sinal elétrico numa grandeza física, ou ainda transformar uma forma de energia noutra. Como por exemplo um aquecedor, capaz de transformar energia elétrica em energia térmica ou num caso mais complexo um LCD (*liquid crystal display*) que transforma um sinal eletrónico em luminosidade, produzindo assim imagens.

2.7.7 Interruptor eletromecânico "Relé")

Dispositivo que responde a um pequeno sinal, sendo capaz de ligar ou desligar um componente externo como por exemplo uma lâmpada. Um relé é composto por uma bobina e um conjunto de contactos, como pode ser observado na Figura 2.24

Quando uma pequena corrente circula pelos terminais da bobina, é gerado um campo magnético no seu núcleo, este que atrai ou repele o induzido, que por sua vez liga ou desliga um circuito externo.

Apesar de o relé ser um dispositivo com mais de um século, ainda hoje é utilizado em muitas aplicações, principalmente quando é necessário que circuitos de baixa corrente, sejam capazes de atuar em circuitos de corrente elevada.

2.7.8 Eletroválvula

Dispositivo composto por um solenóide e uma válvula, destinado a abrir e fechar um fluxo. Quando uma corrente percorre os terminais do solenóide, esta gera um campo magnético capaz de abrir ou fechar a válvula.

Na Figura 2.25 é possível observar algumas variedades de eletroválvulas.

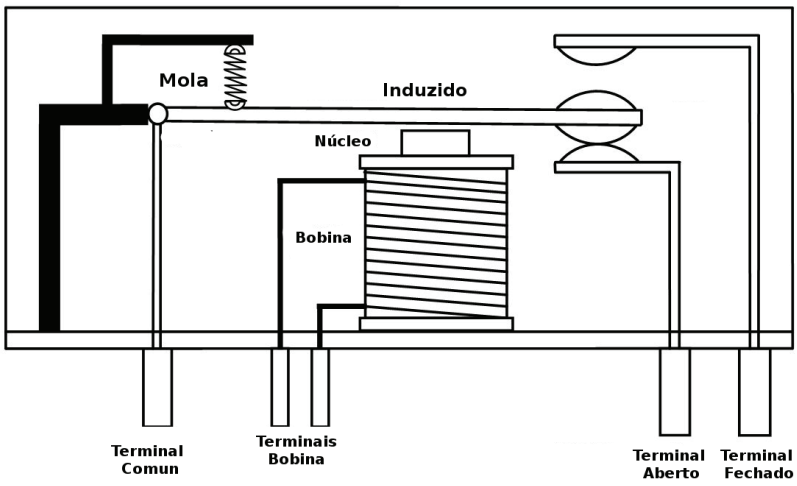


Figura 2.24: Princípio de funcionamento de um interruptor eletromecânico - Fonte: Adaptado de [Glolab, 2013]

.

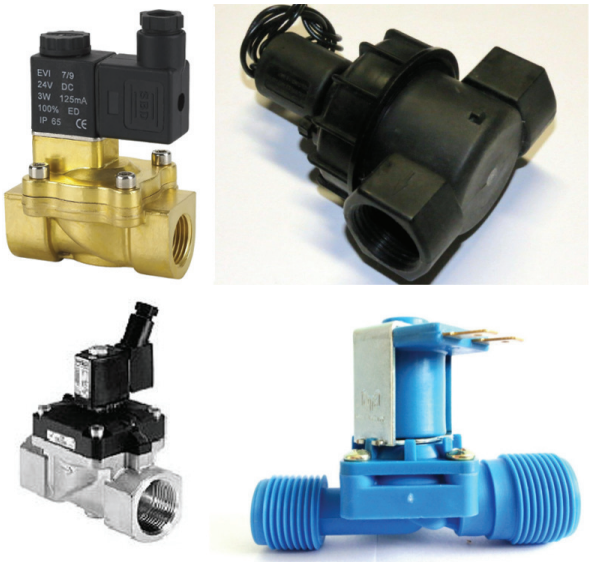


Figura 2.25: Exemplos de algumas eletroválvulas

.

Capítulo 3

Escolha dos sensores

Para o desenvolvimento dos módulos de monitorização dos tanques foi fundamental tomar algumas decisões sobre a seleção dos sensores. Visando um produto final flexível, resistente e com o menor custo possível, foram efetuados testes e comparações, que serão discutidos neste capítulo.

3.1 Sensores de temperatura

Foram selecionados três sensores de temperatura para possível utilização no protótipo. Dois sensores analógicos (LM35 e LM34) e um digital DS18B20. Na Tabela 3.1 é possível equiparar os sensores. O objetivo era encontrar um sensor de precisão (máximo 0,5 °C), de fácil ligação ao Arduino, resistente à água e de baixo custo.

Tabela 3.1: Principais característica dos Sensores de Temperatura.

Características Sensores Temperatura			
	LM35	LM34	DS18B20
Calibração Direta	°C	°F	°C
Fator Escalar	10 mV/°C	10 mV/°F	
Variável			
Precisão	0,5 °C	0,5 °C	0,5 °C
Variação da Temperatura ºC	0 °C a 150 °C	-18 °C a 145 °C	-55 °C a 125 °C
Variação da Temperatura ºF	32 °F a 302 °F	0 °F a 300 °F	–67 °F a +257 °F
Tensões de Funcionamento	4 V a 30 V	5 V a 30 V	3 V a 5,5 V
Valor Médio	1,5 €	1,50 €	3,95 €

Uma vez que o custo destes sensores são relativamente baixos, foi possível adquirir amostras dos três tipos para testes no laboratório, com o objetivo de perceber qual se adaptava melhor ao protótipo final. Nas Secções adiante falaremos sobre os testes e utilização destes sensores.

3.1.1 Sensor LM35

O LM35 é um sensor analógico capaz de determinar a temperatura, como já referido anteriormente na Secção 2.7, é um circuito integrado que usa da sensibilidade dos materiais semicondutores para determinar a temperatura. É composto por um revestimento

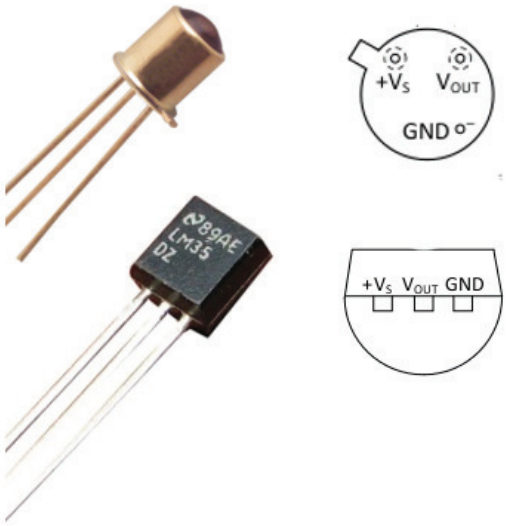


Figura 3.1: Integrado LM35, encapsulamentos e pinos de conexão - Fonte: Adaptado de [Texas Instruments, 2013]

.

plástico ou metálico (dependendo do tipo de encapsulamento) e três pinos de ligação. Cada um destes pinos tem uma função específica como podemos observar na Figura 3.1.

No pino V_s ligamos a tensão para a alimentação do sensor, no GND ligamos a terra e o V_{out} ligamos a uma das entradas lógicas do Arduino.

No Capítulo 2.6 citamos que o Arduino possui em cada uma das suas entradas analógicas um ADC de 10 *bits*, ou seja, oferece 1024 níveis de quantificação.

Ao utilizar o LM35 ligado ao Arduino, temos uma conversão A/D. Essa conversão utiliza um valor em tensão como referência, que tem como padrão o valor de 5 V.

Sendo a resolução em temperatura definida por,

$$\Delta T = \frac{\Delta V}{S} \tag{3.1}$$

$T \rightarrow$ Temperatura
$V \rightarrow$ Tensão
$S \rightarrow$ Sensibilidade

Ao efetuar uma leitura ao LM35 numa porta analógica do Arduino, a resolução do ADC é de 5 V/1024 $\simeq$ 5 mV.

O sensor varia linearmente 10 mV/°C. Por outras palavras, quando temos 0 °C, obtemos uma tensão correspondente a 0 V a saída do sensor, para 1 °C teremos uma tensão de 10 mV, para 25 °C vamos obter 250 mV, e assim por diante.

Pela Equação 3.2 conseguimos deduzir que a menor quantidade mensurável é de 0,5 °C.

$$\frac{1°\text{C}}{x} = \frac{10\text{mV}}{5\text{mV}} \tag{3.2}$$

Conforme a Tabela 3.1, a temperatura máxima suportada pelo LM35 é de 150 °C, que corresponde a uma tensão de 1,5 V. Sendo a tensão de referência do ADC 5 V, só

atingiremos o nível 1024 do ADC quando a tensão for igual a 5 V. Tendo em conta que 5 V corresponde a uma tensão de 500 °C. Concluímos que estamos a utilizar apenas 30% dos níveis de quantificação do ADC, pois 150/500=0,3.

Ao efetuar leituras com o LM35, com a tensão de referência igual a 5 V, foram detetados alguns problemas, listados a seguir.

- 0,5 °C é a resolução mínima aceite neste projeto, conforme citado na Secção 1.2.
- Valores instáveis e não correspondentes a temperatura, como pode ser observado na Figura 3.2, onde o valor da temperatura deveria ser de 31,9 °C.
- Aproveitamento de apenas 30% dos níveis de quantificação do ADC. Pois uma vez que o sensor é capaz de medir temperaturas até 150 °C, que corresponde a uma tensão de 1,5 V e o ultimo nível do ADC corresponde a 5 V, temos então 1,5 V / 5 V = 0,3.
- Impossibilidade de medir temperaturas negativas.

Uma forma a superar alguns dos problemas citados é utilizar uma tensão de referência diferente de 5 V. Ao utilizarmos a tensão de 1,1 V para o ADC, maximizamos para 100% a sua utilização. Outra grande vantagem foi a estabilização e certeza dos valores recebidos. Isso acontece porque a tensão de 5 V fornecida pelo Arduino para o ADC, vem diretamente da ligação USB ao computador. Dependendo da porta USB, os valores da tensão fornecida e o nível de ruído encontrado, está acima do aceitável para utilização neste trabalho. Quando utilizada outra tensão disponível no Arduino, esta é fornecida por um regulador de tensão, oferecendo valores muito mais precisos e estáveis que no caso do 5 V.

Na Tabela 3.2 podemos verificar os valores da tensão fornecida pelo Arduino (5 V; 3,3 V; 1,1 V), quando ligado a computadores diferentes e constatar a desigualdade na tensão de 5 V.

Esta variação na tensão influência a leitura da temperatura no Arduino, da seguinte maneira:

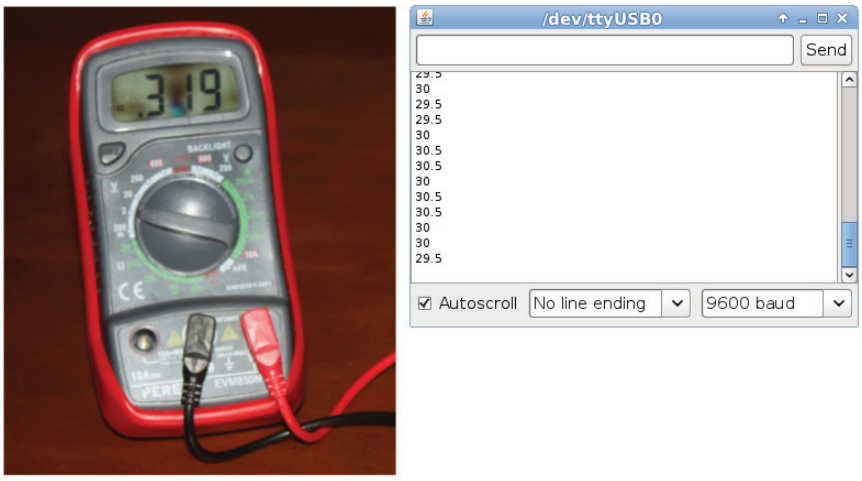


Figura 3.2: Leitura da temperatura com o LM35 ligado ao Arduino, comparado com a leitura está de um multímetro.

	Tensão Fornecida pelo Arduino		
	1,1 V	3,3 V	5 V
Computador 1	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,96 V $\pm$ 0,03
Computador 2	1,099 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,89 V $\pm$ 0,02
Computador 3	1,099 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,98 V $\pm$ 0,02
Computador 4	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,81 V $\pm$ 0,03
Computador 5	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,84 V $\pm$ 0,02
Computador 6	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,94 V $\pm$ 0,02
Computador 7	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,82 V $\pm$ 0,02
Computador 8	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,91 V $\pm$ 0,02
Computador 9	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,88 V $\pm$ 0,02
Computador 10	1,098 V $\pm$ 0,001	3,23 V $\pm$ 0,01	4,79 V $\pm$ 0,02

Tabela 3.2: Diferença entre os valores da tensão do Arduino em computadores distintos.

Supondo que a temperatura é de 25 °C. Como já citamos, o sensor varia linearmente 10 mV/°C, logo a saída do sensor teríamos 250 mV.

O Arduino vai receber este valor 250 mV e vai traduzir para um dos 1024 níveis do seu ADC. No caso da tensão de referência ser 5 V teríamos $\frac{250\ \text{mV}}{5\ \text{V}/1024} \simeq 50$. Entretanto se a tensão de referência for 4,81 V e fizermos os mesmos cálculos como no caso anterior, o nível referente será o 53.

Como já referido para cada nível do ADC temos aproximadamente 0,5 °C, quando solicitamos o valor da temperatura ao Arduino, com base nesse valor ele retornará os valores da temperatura, que neste caso seriam, 25 °C e 26,5 °C, causando um erro de 1,5 °C.

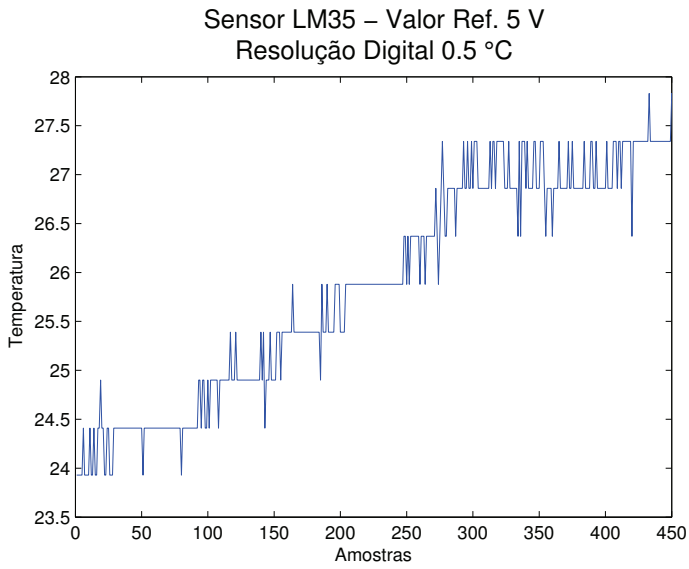
Outra técnica utilizada para melhoria da resolução de um ADC é o *oversampling*. Com o intuito de testar se este procedimento traz alguma melhoria significativa para a determinação da temperatura com o sensor LM35, resolvemos desenvolver um código para o Arduino, capaz de recolher o maior número possível de amostras durante um segundo e calcular a sua média.

O tempo de um segundo foi escolhido pois, como o sensor estará submerso num tanque com grande quantidade de água, o tempo necessário para uma variação significativa da temperatura será superior a um segundo.

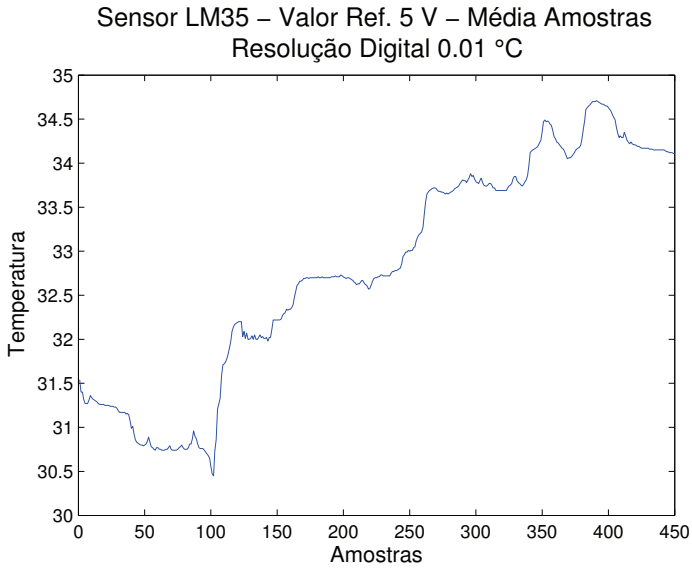
De acordo com [Arduino, 2013] o tempo necessário para ler uma entrada analógica é cerca de cem microssegundos, ou seja a sua taxa máxima de leitura é de dez mil vezes por segundo.

O sensor foi então ligado ao Arduino, primeiramente recolhemos amostras únicas a cada um segundo. Seguidamente utilizamos um código capaz de recolher dez mil amostras e calcular a sua média. Ambos os procedimentos foram executados periodicamente durante cerca de sete minutos e meio, gerando um total de 450 amostras e a partir dos valores obtidos, foram gerados dois gráficos que podem ser visualizado na Figura 3.3.

A Figura 3.3 (a) mostra a variação da temperatura quando recolhemos amostras a cada um segundo, enquanto na Figura 3.3 (b) temos a variação do mesmo sensor, entretanto efetuando a média de dez mil amostras a cada segundo. Com base nos gráficos é possível concluir que a utilização de *oversampling* durante a determinação da temperatura com um sensor LM35, aumenta significativamente a resolução do ADC.



(a) Valores sem utilização de *buffer*



(b) Valores com utilização de *buffer*

Figura 3.3: Gráfico dos valores da temperatura do LM35

De forma a determinar qual a contribuição do ruído para a melhoria da resolução do ADC, foi desenvolvido uma função em Matlab capaz de simular um ADC. O código simula a receção de uma tensão de 250 mV DC, adiciona ruído, recolhe dez mil amostras, calcula a sua média e quantifica para um dos níveis do ADC. O ruído utilizado é do tipo gaussiano, com média nula e valores de variância entre 0 e 0,1 V^2. Para cada variância, os cálculos eram efetuados e os valores dispostos em um gráfico, podendo ser visualizados na Figura 3.4.

Conforme podemos observar na Figura 3.4, quando estamos entre 2 níveis de quantificação do ADC, de acordo com que o ruído vai aumentando o erro digital vai diminuindo, isso acontece gradualmente até a variância atingir o valor de 0,01 V^2, a partir deste valor

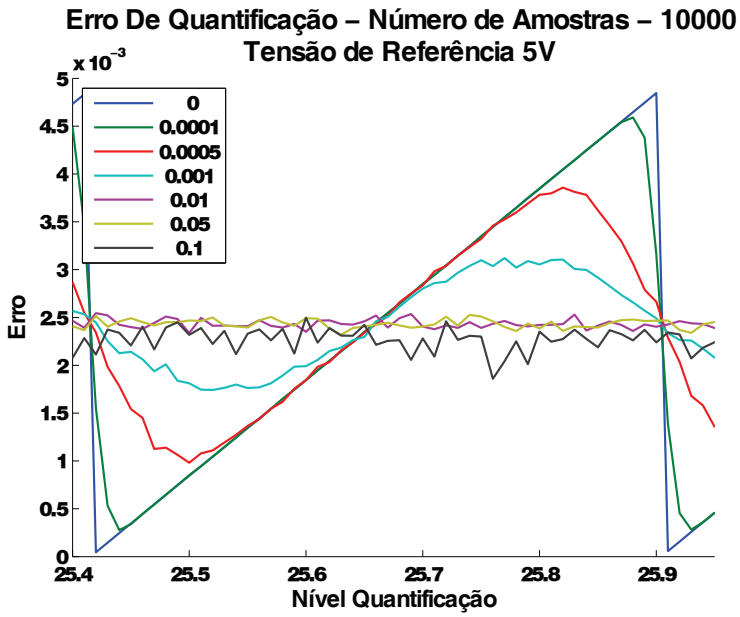


Figura 3.4: Simulação de um ADC, no qual ao sinal de entrada é adicionado ruído gaussiano de média nula e variância com valores entre 0 e 0.1 V^2

o erro digital começa aumentar. Este comportamento já era previsto, pois uma vez que temos um sinal de 0,25 V e estamos a adicionar um ruído com desvio-padrão igual ou superior a 0,1 V, o ruído passa a ser mais significativo que o sinal de entrada.

Apesar de a Figura 3.3 (b) comprovar que o *oversampling* é capaz de melhorar a resolução do ADC e a Figura 3.4 possibilitar determinarmos qual a contribuição do ruído para a melhoria da resolução. Não é possível garantir que os valores obtidos pelo sensor LM35 quando utilizado *oversampling*, eram realmente os valores esperados, uma vez que não foi comparado com nenhum valor padrão.

Com o intuito de perceber como deveria ser a distribuição do sinal de um sensor com ruído, foi efetuada uma simulação de Monte Carlo. Após verificar como deveria ser a distribuição de um sinal, cuja a temperatura média fosse de 28,8 °C, sentiu-se a necessidade de comparar os resultados da simulação, com resultados práticos utilizando o sensor LM35.

Para tal comparação o LM35 foi submetido a uma banho termostático, como pode ser observado na Figura 3.5. Este banho permitiu uma temperatura estável de 28,8 °C, esta temperatura foi obtida por um termómetro de mercúrio, devidamente calibrado.

Foi criado um código para o Arduino, capaz guardar os valores obtidos pelo sensor LM35, para posterior análise. Com base nos dados foi gerado um histograma, calculada a média e o desvio padrão. Os respetivos resultados podem ser observados na Figura 3.6.

Outro ensaio efetuado, foi submeter todos os sensores a um mesmo banho termostático, durante um mesmo período de tempo, com o objetivo de comparar os valores de leitura entre os três sensores simultaneamente com um padrão. Os resultados deste experimento pode ser observado na Tabela 3.3.

Com base nas experiências acima descritas, conseguimos concluir que os sensores apresentavam ruído, que a utilização de *oversampling* melhorou significativamente a resolução do ADC e que o cálculo da média das amostras está muito próximo do valor

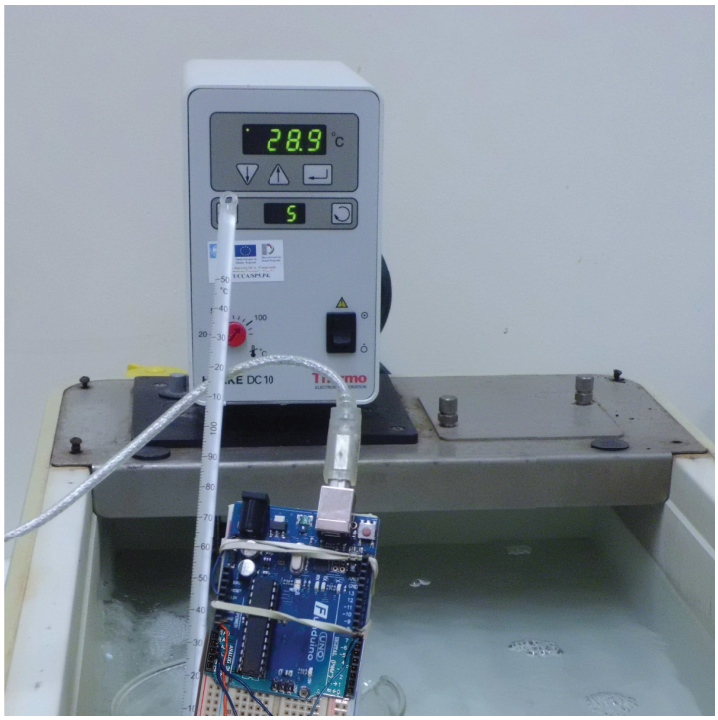


Figura 3.5: Banho termostático com os sensores, para obtenção de uma temperatura estável

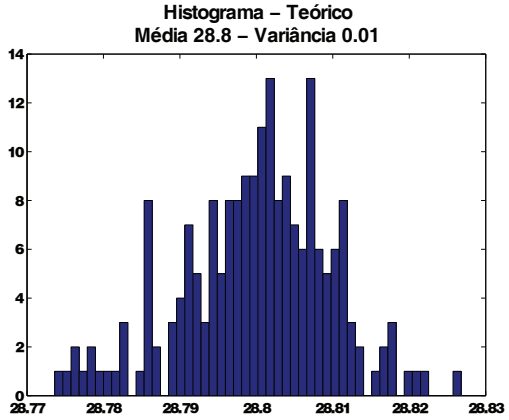
	Banho Termostático	
	Tensão de Referência ADC	
Sensor	5V	1,1V
Padrão	28,800 °C	28,800 °C
LM35	27,280 °C	28,370 °C
LM34	27.800 °C	28,890 °C
DS18B20	28,920 °C	28,888 °C

Tabela 3.3: Temperatura dos Sensores em Banho Termostático

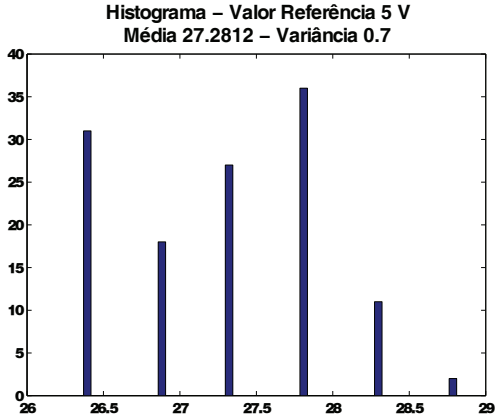
esperado, principalmente quando utilizamos 1,1 V como tensão de referência. Os métodos já apresentados podem ser utilizados de forma fiável, garantindo uma boa estimativa da temperatura, no entanto em nenhum dos casos podemos afirmar que a distribuição é do tipo gaussiano.

3.1.2 Sensor LM34

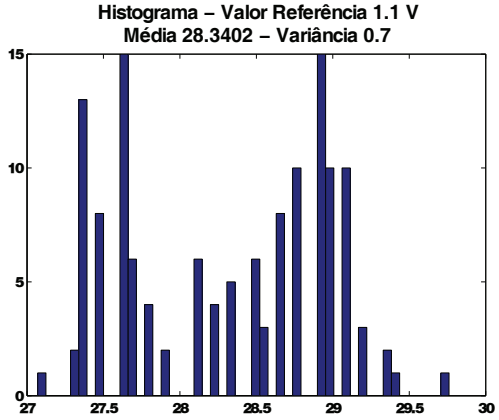
O sensor LM34 é um circuito integrado semelhante ao LM35, a única diferença é que o LM35 está linearmente calibrado para fornecer 10 mV/°C, enquanto o LM34 fornece 10 mV/°F. Essa diferença é que o torna mais apropriado para o projeto do que o LM35. Possibilitando medir temperaturas abaixo de 0 °C, recorrendo apenas a Equação 2.2 citada na Secção 2.7.3.



(a) Histograma teórico, com média de 28.8 °C e desvio-padrão 0,1 $°C^2$



(b) Histograma, sensor LM35, tensão de referência 5 V. Média 27,289 °C e Variância 0,656 $°C^2$



(c) Histograma, sensor LM35, tensão de referência 1,1 V. Média 28,340 °C e Variância 0,670 $°C^2$

Figura 3.6: Histogramas a partir das amostras de temperatura recolhidas com um sensor LM35 para diferentes tensões de referência do ADC.

3.1.3 Sensor DS18B20

Este sensor é do tipo digital, composto por um circuito integrado capaz de traduzir uma temperatura para um valor com 12 *bits* de resolução. Funcionando sobre o protocolo *1-wire*, é possível conectar vários sensores numa única porta digital do Arduino. A ligação ao Arduino necessita de um resistor *pull-up*, para de evitar oscilações e garantir apenas os 2 valores de tensão 0 V e 5 V, para os estados lógicos.

O protocolo *1-wire*, utiliza o princípio mestre-escravo nas suas comunicações. O mestre, neste caso o microcontrolador, inicia e controla a comunicação, solicitando informação aos dispositivos escravos, que aqui serão os sensores DS18B20, cada dispositivo escravo tem um endereço único de 64 *bits*, este endereço permite ao mestre, solicitar a informação a um escravo específico, apesar de todos estarem ligados um único fio. O protocolo utiliza uma *frame* específica para comunicação utilizando, entretanto o Arduino tem uma biblioteca própria para a comunicação de dispositivos *1-wire*.

Os pinos do DS18B20, assim como o método de ligação ao Arduino pode ser visualizados na Figura 3.7.

Apesar de um custo minimamente mais elevado que os anteriores, este componente supera os demais apresentados nos seguintes aspetos:

- Tem um ADC integrado no próprio dispositivo. Esta característica é de grande importância em aplicações onde o sensor não pode estar acoplado à placa do circuito, dependendo de um fio de comprimento significativo para conexão com o circuito. Este tipo de utilização introduz ruído e perda na qualidade do sinal analógico, comprometendo a sua utilização. No caso do DS18B20 como a conversão ADC acontece dentro do próprio circuito integrado, o sinal transmitido será pouco afetado pelo ruído da linha.
- Tem uma boa resolução do ADC interno. Utilizando o seu próprio ADC resolução de que tem 12 *bits* o sensor não precisa do conversor do Arduino de apenas 10 *bits* de resolução, conseguindo uma boa precisão sem necessitar de nenhum algoritmo para tal.
- Conté, um encapsulamento à prova de água. Entre os três sensores mencionados acima, este foi o único encontrado com proteção para ser utilizado submerso em água.

Baseado nos testes e qualidades acima descritas, este foi o sensor que melhor se adaptou a todas as condições necessárias para este projeto, sendo assim o sensor de temperatura escolhido.

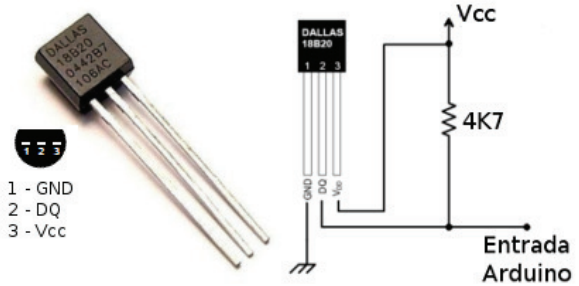


Figura 3.7: Integrado DS18B20, encapsulamento e pinos de conexão

.

3.2 Escolha do sensor de ultrassom

Foi determinado que, de entre todos os métodos para medição de nível de água, iria ser utilizada a medição por ultrassom. Tomámos esta decisão por não necessitar de contacto direto com os tanques, como acontece com outros métodos, por exemplo com boias, réguas de nível e sensores capacitivos.

Existem no mercado alguns modelos de dispositivos para medição por ultrassom, sendo que o modelo HC-SR04 destaca-se dos demais por ter um baixo custo, boa performance e fácil integração com Arduino. Podendo medir distâncias entre 2 cm e 400 cm, com uma resolução de 3 mm, determinamos que este sensor seria o ideal para a aplicação indicada.

Como podemos observar na Figura 3.8, este sensor possui 4 pinos de ligação. O pino *trigger* é ligado a uma porta digital do Arduino e é por onde se envia um pulso com uma duração de 10 μs. O pino *echo* é o pino recetor, por onde esperamos o retorno do pulso enviado. A distância é calculada com base no tempo entre o envio e a receção do pulso. Como já citado, o sensor envia pulsos ultrassónicos e estes propagam-se pelo ar. A velocidade do som a propagar pelo ar é de 340 m/s, assim sendo podemos calcular a distância entre o objeto e o sensor através da seguinte equação:

$$v = \frac{2d}{t} \Rightarrow d = \frac{vt}{2} \Rightarrow d = \frac{(340 \text{ m/s}) \times t}{2} \Rightarrow d = (170 \text{ m/s}) \times t \tag{3.3}$$

Havia dúvidas sobre como o sensor iria responder num ambiente aquático, mas os testes efetuados com o HC-SR04 resultaram muito bem, medindo com a mesma precisão a distância com obstáculos sólidos ou líquidos.

Durante os testes verificou-se a necessidade de efetuar um tratamento dos dados recebidos pelo sensor. Isto porque, em alguns momentos, durante a aquisição dos dados, detetámos alguns valores discrepantes. Depressa nos apercebemos de que estas alterações ocorriam durante a manutenção dos tanques, quando algo - por exemplo a mão de quem fazia a manutenção - passava entre o sensor e a face da água, causando leituras erradas.

A solução encontrada foi recolher 100 amostras consecutivamente, utilizar um algoritmo *Bubble sort* para organizar os dados por ordem crescente e seguidamente eliminar os dados discrepantes. Para remoção dos dados discrepantes, utilizamos o mesmo princípio do *Box-Plot* contido na Figura 3.9. Determinamos uma caixa com limite superior

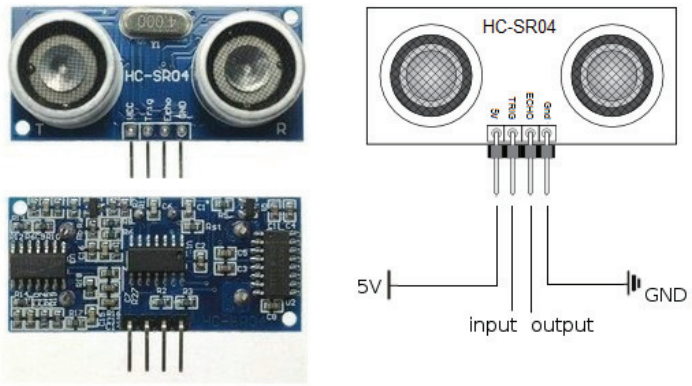


Figura 3.8: Sensor de ultrassom HC-SR04 e modo de ligação

.

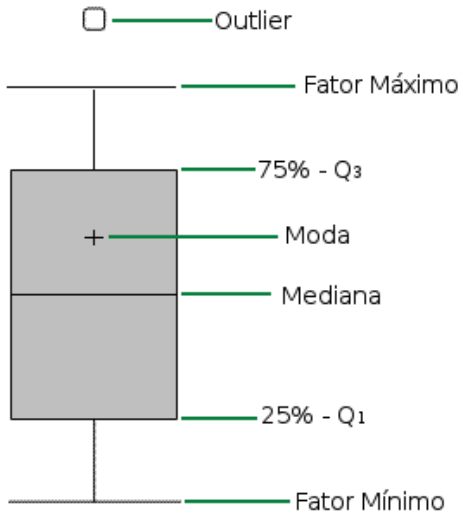


Figura 3.9: Exemplo de determinação de uma *box-plot* - Fonte: Adaptado de [Medina and Chwif, 2006]

.

e inferior e todos os valores fora dessa "caixa" são descartados, como explica Medina and Chwif em Modelagem e Simulação de Eventos Discretos. Desta forma aumentamos a fiabilidade na interpretação dos dados, resolvendo assim a questão acima referida.

3.3 Escolha do sensor pH/ORP

O sensor de pH, também conhecido por elétrodo de pH, é um sensor analógico que traduz a quantidade de pH para uma tensão elétrica, mais precisamente 59,2 mV por unidade de pH, conforme demonstrado na Secção 2.7.3 Existem no mercado uma infinidade de sensores para este efeito, onde as maiores diferenças entre os modelos estudado, são: tempo de funcionamento sem necessidade de calibração e tempo de resposta. Com valores que podem variar das dezenas até as centenas de euros, nenhum destes elétrodos estão preparados para ligação direta com o Arduino. Isso deve-se ao facto de que a impedância interna do sensor em relação à impedância do Arduino ser tão elevada que a tensão gerada pelo sensor perderia-se. Outra questão é a baixa tensão produzida pelo sensor, com uma tensão de apenas 59,2 mV por unidade de pH, qualquer ruído na linha ou ligação degradaria o sinal, impossibilitando uma leitura correta. Sem mencionar a necessidade de medir tensões negativas, quando o pH é inferior a 7.

Para possibilitar a ligação de um elétrodo de pH ao Arduino, é necessária a utilização de um circuito amplificador e/ou conversor de impedância. Surgiu então a ideia de desenvolver um circuito capaz de acoplar o sensor ao Arduino, mas uma vez que já há no mercado soluções implementadas e testadas para este propósito, (e este não era o objetivo desta dissertação), optamos por utilizar de um circuito comercializado.

O produto mais interessante encontrado para a ligação entre o elétrodo e o Arduino, foi o dispositivo *Micro footprint pH monitoring subsystem* produzido pela empresa Americana *Atlas Scientific*. Este circuito é capaz de comunicar diretamente com o Arduino por uma comunicação *serial*, efetuar a calibração do sensor epossui *LED* para verificar

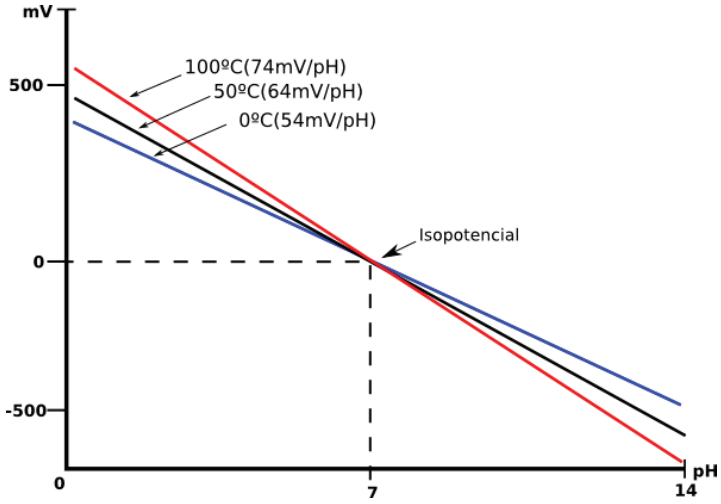


Figura 3.10: Influência da temperatura no pH - Adaptado de [Rodrigues, 2010]

.

o seu estado de funcionamento, sendo o produto escolhido para integrar no protótipo. No entanto este produto não é vendido na Europa e o custo de importação do mesmo inviabilizou a sua utilização.

Como segunda opção decidiu-se utilizar a placa pH/ORP *adapter*, produzida pela Phidgets Inc. com vários representantes na Europa, inclusive em Portugal. Este adaptador, além de ter uma alta impedância de entrada, amplifica e equaliza a tensão referente ao valor do pH. Este mesmo adaptador pode ser utilizado com elétrodos de ORP, mudando apenas a posição de um interruptor onde é possível selecionar a opção pH ou ORP. A única desvantagem deste adaptador é não efetuar a calibração do elétrodo de pH. Para contornar esta situação foi desenvolvida uma função para o Arduino, capaz de calibrar o sensor, utilizando soluções tampão de pH 7 e pH 10. Esta função solicitar ao utilizador que introduza o sensor na solução com pH 7, seguidamente solicita a introdução do sensor na solução de pH 10 e efetua os cálculos necessários para calibração.

Outra questão abordada sobre a aquisição de dados com o sensor de pH foi a temperatura. Uma vez que o valor do pH depende da temperatura, conforme demonstrado na Equação 2.12 e o ambiente onde o protótipo está a funcionar sofre variações da mesma, houve á necessidade de realizar a compensação automática do valor do pH para a temperatura no instante da aquisição do valor.

O sensor de pH, funciona de forma linear e a influência da temperatura no valor do pH é diretamente proporcional à inclinação da reta como podemos observar na Figura 3.10.

O sensor de ORP não necessita de calibração, desta forma foi necessário implementar apenas uma função simples para o Arduino, capaz de recolher os dados e enviar para o dispositivo coordenador da rede.

Capítulo 4

Montagem do protótipo

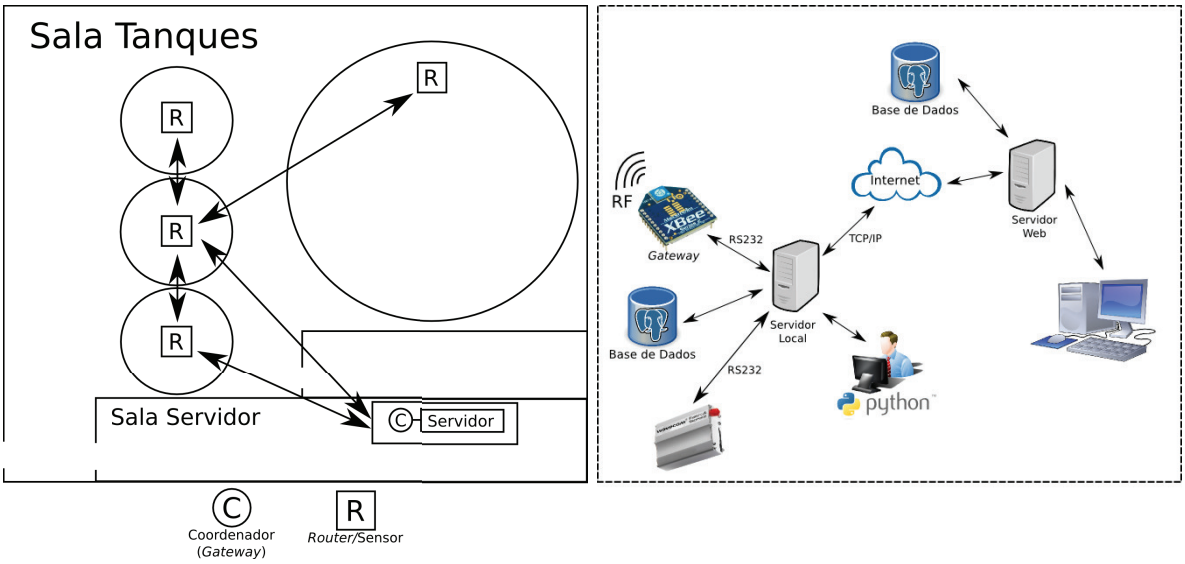
Neste capítulo é demonstrada a arquitetura da RSSF, montada no laboratório da empresa Caviar Portugal para monitorização dos tanques. Seguidamente é demonstrado o esquema de ligação dos sensores e toda a informação necessária para montagem do protótipo.

4.1 Topologia da RSSF e arquitetura do sistema

A análise do ambiente que desejamos monitorizar é de grande importância para a decisão do tipo de configuração a utilizar e respetiva localização dos equipamentos. Uma vez que o laboratório da Caviar Portugal está em constante modificação e os tanques são mudados com alguma frequência, houve a necessidade de que o protótipo fosse flexível, podendo funcionar em qualquer local do laboratório sem a necessidade de ser reconfigurado. Com o objetivo de maximizar a área de cobertura da RSSF e não perder qualidade no sinal, todos os nós da RSSF são do tipo *router/sensor*. Este tipo de nó funciona como *router*, podendo reencaminhar dados de outros nós ao mesmo tempo que é capaz de receber informações dos sensores e enviar para o coordenador da rede.

O espaço da Caviar Portugal contava com 4 tanques, dispostos conforme nos mostra a Figura 4.1 (a). Cada tanque recebe um módulo *router/sensor* que é composto por: Arduino, Xbee Shield, Xbee rádio e sensores. Dado o facto que todos os módulos são do tipo router/sensor, mesmo que futuramente algum dos tanques seja desativado ou novos sejam instalados, a RSSF adaptar-se-á sem a necessidade de novas configurações. Os *routers* aceitam automaticamente o novo equipamento na rede, atualizam as respetivas tabelas de rotas e começam a enviar dados pela RSSF. Todos os dados recebidos pelos *routers/sensores* são encaminhados para o coordenador da rede que está ligado a um servidor local, situado na "sala servidor", como se pode verificar na Figura 4.1 (a).

Os dados recebidos pelo coordenador "*gateway*" do sistema, são tratados e armazenados na base de dados por uma aplicação em Python. Esta aplicação tem como função verificar a integridade dos dados, alertando o utilizador do sistemas através de mensagens de texto (SMS) o utilizador, caso algum parâmetro não esteja dentro dos valores predefinidos. O envio do alerta SMS é efetivado pelo *modem* GSM (*Global System for Mobile Communications*) conectado ao servidor local por uma ligação *serial* RS232. A Figura (b) demonstra toda a estrutura do sistema de monitorização, sendo que alguns elementos serão explicadas nos capítulos seguintes, como por exemplo a utilização do "servidor web".



(a) Topologia da RSSF (b) Arquitetura do Sistema

Figura 4.1: (a) Topologia da RSSF e (b) Arquitetura do sistema de monitorização

4.2 Montagem do protótipo

Neste secção será abordada a montagem do protótipo utilizado nos tanques para aquisição dos dados, assim como o método de ligação dos sensores e conexões entre os dispositivos.

4.2.1 Esquema de ligação do circuito

A Figura 4.2 demonstra o esquema de ligação dos sensores ao Xbee Shield, que está conectado ao Arduino.

O circuito e composto por dois adaptadores pH/ORP, sendo capazes de operar com sensores de pH ou ORP. Estes sensores encontram-se ligados respetivamente nas portas A0 e A1.

No pino D2 encontramos conectado o sensor de temperatura DS18B20, que conta ainda com um resistor de 4,7 kΩ de *pull-up*.

Nos pinos D5 e D6 estão conectados respetivamente os pinos *Trigger* e *Echo* referentes ao sensor de ultrassom.

Já no pino D9 temos conectado um *buzzer*, cuja a função é tocar, caso de algum sensor retornar valores fora dos limites determinados.

Por fim temos os pinos D11, D12 e D13, onde podemos encontrar os LED's de *status*.

Todos os outros fios que encontramos no esquema são referentes as tensões de alimentação ou ligações a terra, estando respetivamente ligados ao pinos Vcc ou GND.

Os primeiros ensaios com os protótipos foram efetuados utilizando *protoboards*.[1] Mas devido à falhas na conexão entre o Xbee Shield e os fios dos sensores, encontrou-se a necessidade de utilizar fichas para as ligações. Sendo que do lado dos sensores utilizaremos fichas do tipo "macho" e do lado do Arduino "fêmea".

[1]Placa com furos e conexões condutoras para montagem de circuitos elétricos experimentais.

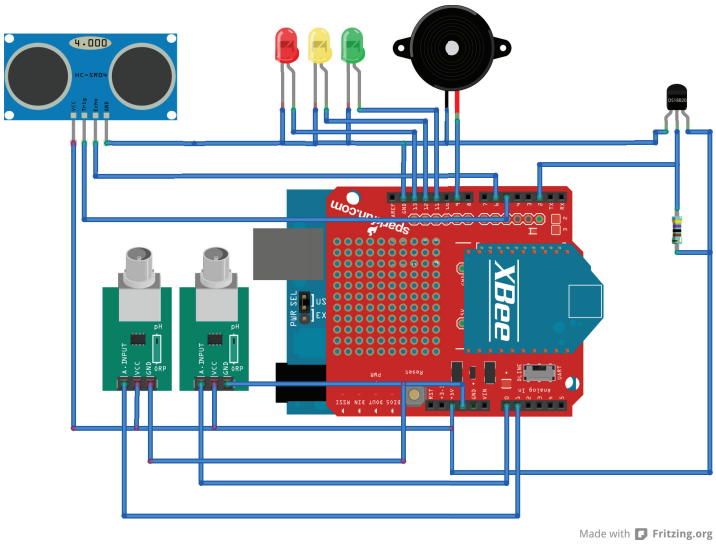


Figura 4.2: Esquema de ligação do circuito interno do protótipo para aquisição dos dados nos tanques.

4.2.2 Ligação dos sensores

Os conectores foram escolhido com base na sua existência no laboratório e possível reutilização, como será discutido nesta secção.

Sensor DS18B20

Necessitando apenas de três terminais para a conexão do DS18B20, foi utilizado o conector de áudio "*Jack 3.5 mm stereo*".

O conector é dividido em três partes, cada umas destas partes contém a ligação a um dos três fios do sensor, conforme Figura 4.3 (a). O conector "macho" Figura 4.3 (b), foi soldado diretamente ao sensor de temperatura, enquanto a "fêmea" Figura 4.3 (c), encontra-se soldada juntamente com o resistor *pull-up*, para posterior ligação ao Xbee Shield.

(a) Pinos de Ligação Jack 3.5 mm "macho"

(b) Sensor após solda do Conector

(c) Fêmea 3,5 mm para ligação ao Xbee Shield

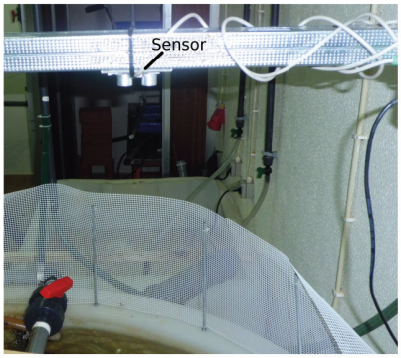
Figura 4.3: Demonstração do processo de ligação da ficha para conexão do sensor de temperatura ao protótipo.

Sensor HC-SR04

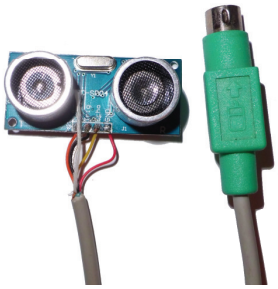
Este sensor possui quatro pinos metálicos, como já demonstrado na Figura 3.8, podendo ser soldado a um circuito impresso ou ligado a uma *protoboard*. Entretanto neste projeto esta ligação não será possível, necessitamos de um cabo para que o sensor fique instalado por cima do tanque. Sendo capaz de medir a distância até a linha da água, conforme Figura 4.4 (a). Necessitando de 4 pinos de ligação e um cabo de pelo menos 1,5 m de comprimento, optamos por utilizar conectores PS/2, pelo facto de ter uma ficha robusta, que atende as necessidade e há em demasia no laboratório sem utilização. Foram removidas de ratos de computadores, já trazem um fio de boa qualidade, resistente e com blindagem.

Conforme Figura 4.4 (b) os pinos metálicos foram removidos, dando lugar aos cabos com fichas PS/2. Na Figura 4.4 (c) podemos ainda verificar os pinos de ligação da ficha PS/2.

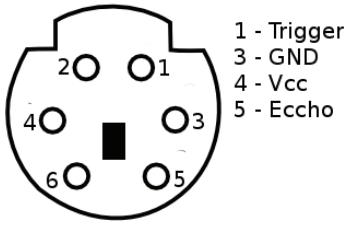
Após fixação do sensor no suporte, é necessário medir a distância do sensor até a borda do tanque, seguidamente determinar o tamanho do tanque e seu diâmetro, conforme Figura 4.5. Estes dados devem ser submetidos no Arduino para que seja possível determinar o nível e o volume de água no tanque. O processo de introdução



(a) Imagem do sensor montado sobre o tanque, para monitorização do volume de água no tanque



(b) Conector PS/2 para ligação do sensor de ultrassom



(c) Pinos utilizados na ligação

Figura 4.4: Demonstração do processo de ligação da ficha PS/2, para conexão do sensor de ultrassom ao protótipo.

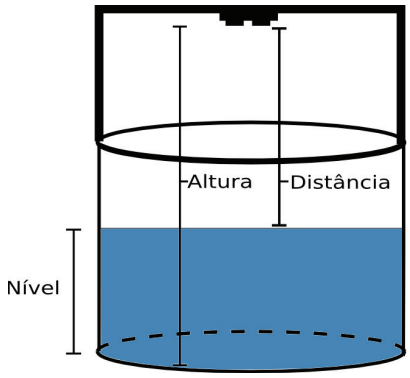


Figura 4.5: Esquema de Medidas, para instalação do sensor de Ultrassom no tanque.

das dimensões do tanque no Arduino será explicado no Capítulo 6.

Para determinar o volume de água no tanque, foram utilizadas as seguintes equações:

$$V = \pi r^2 (h - d) \tag{4.1}$$

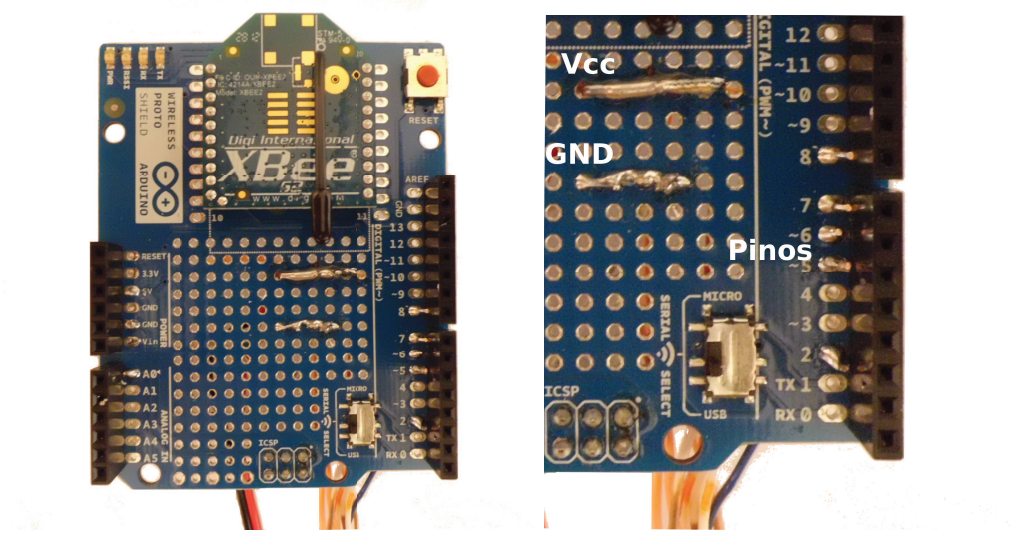
$h \rightarrow$ Altura do sensor
$d \rightarrow$ Distância a água
$r \rightarrow$ Raio do tanque

4.2.3 Ligação do Xbee Shield

Optou-se por não utilizar nenhuma placa de circuito ou *shield* para ligação definitiva dos sensores ao circuito, todas as ligações foram efetuadas no Xbee Shield conforme Figura 4.6 (a). Os fios referentes aos pinos do Arduino, foram soldados nas suas respetivas posições, enquanto os fios GND e Vcc foram soldados em duas linhas de contactos distintas, Figura 4.6 (b). Após todos os conectores estarem soldados ao Xbee Shield, este foi respetivamente ligado ao Arduino.

4.3 Caixa de proteção

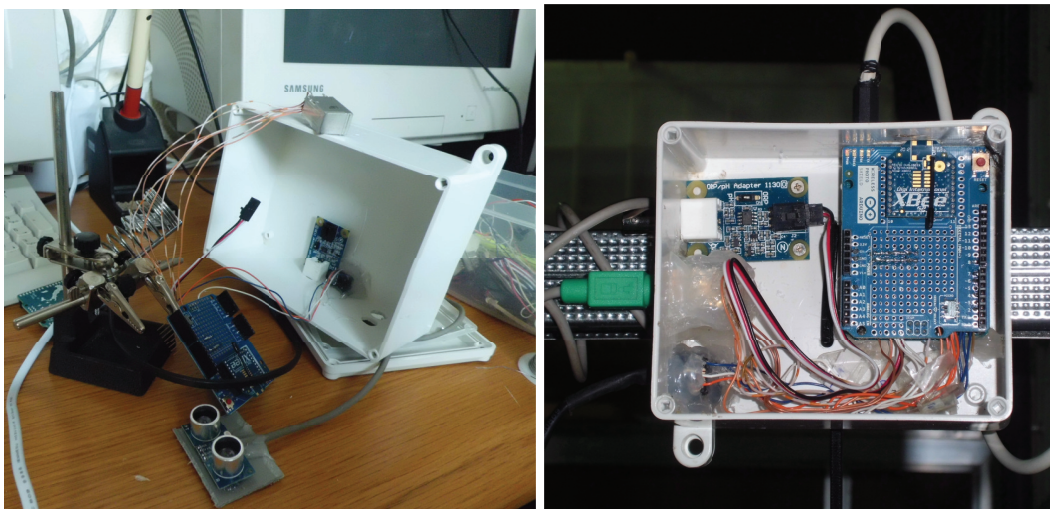
Estando em um ambiente com grande quantidade de agua e humidade, o circuito eletrónicos será acomodado em uma caixa de proteção. Esta caixa tem como objetivo oferecer uma melhor proteção e isolamento ao equipamento conforme podemos observar nas Figuras 4.7. A caixa é impermeável e todas as conexões foram isoladas com material vedante, evitando assim a degradação dos equipamentos eletrónicos.



(a) Xbee Shield com as respetivas conexões dos sensores soldadas a placa.

(b) Soldas dos conectores a placa Xbee Shield

Figura 4.6: Demonstração dos fios soldados diretamente no Xbee Shield



(a) Montagem do equipamento na caixa de proteção

(b) Equipamento em funcionamento, dentro de caixa de proteção

Figura 4.7: Demonstração do equipamento acomodado na sua respetiva caixa de proteção.

Capítulo 5

Software

Para o funcionamento do protótipo e gestão dos dados a monitorizar, foi necessário instalar alguns aplicativos já existentes, além de desenvolver rotinas de trabalho para o Arduino e um programa para gestão dos dados no servidor. Neste capítulo falaremos sobre a escolha do *software* utilizado e suas características. Todo o software utilizado neste projeto é *freeware*[1], este foi o principal fator de decisão na escolha dos mesmos. Cito a seguir os softwares escolhidos:

5.1 Sistema Operativo

O sistema operativo utilizado no servidor foi a distribuição Xubuntu 12.10, baseada no Ubuntu 12.10 a sua principal diferença é a utilização do ambiente gráfico Xfce, que necessita de muito menos recursos de sistema que a respetiva versão Ubuntu.

Foi necessária a instalação do aplicativo Wine, capaz de executar aplicações desenvolvidas para ambientes Windows, nomeadamente o X-CTU (software que permite configurar os Xbee's) e o Ambiente de desenvolvimento IDE Arduino, cuja a sua principal função é o desenvolvimento de códigos de trabalho para a placa Arduino.

5.2 Base de Dados

Devido a grande quantidade de informação a ser armazenada, houve a necessidade de utilizar uma base de dados. Para esta função foi escolhido o sistema de gestão PostgreSQL. Como [Lemes, 2004] cita em seu livro, PostgreSQL é uma base de dados completa para o desenvolvimento em que performance, portabilidade, custo e eficiência devem andar juntos. Sendo um dos sistemas de base de dados *open source*, mais robusto e eficiente do mercado, com mecanismo de controlo de concorrência de utilizadores, suporte a replicação assíncrona dos dados, tolerância a falhas, entre infinitas outras característica menos relevantes neste projeto. O PostgreSQL é ainda capaz de armazenar 32 *terabytes* de informação em uma única tabela, 1 *gigabyte* por campo da tabela e um número ilimitado de linhas por tabela.

Baseado nas qualidades acima descritas, o sistema de gestão para a base de dados escolhido foi o PostgreSQL.

[1] Designação dada a qualquer programa de computador, cuja a sua utilização não necessita de pagamentos de licenças

Para determinar a estrutura da base de dados, foi elaborado um modelo de dados conceptual E/R (Entidade/Relacionamento). Este modelo descreve os dados como entidades, atributos e relacionamentos. Designando entidade como um objeto ou conceito do mundo real, em que suas características são chamadas de atributos e a interação entre duas entidades relacionamento.

De forma a demonstrar o modelo E/R da base de dados desenvolvida para este projeto, na Figura 5.1 podemos observar o Diagrama do modelo E/R. A este modelo foram aplicadas as regras de normalização de base de dados, com o objetivo de reduzir a redundância dos dados mantendo assim a consistência da base de dados. Com base no modelo E/R criou-se a base de dados no PostgreSQL.

5.3 Realização do Código para o Arduino

Para o funcionamento do Arduino foi desenvolvido um código, dando-lhe a capacidade de coletar a informação dos sensores, tratar os dados, verificar se algum parâmetro está fora da gama de funcionamento normal e disponibilizar os dados para que o rádio Xbee envie para o coordenador.

Através do diagrama de fluxo presente na Figura 5.2, podemos ter uma melhor percepção do funcionamento do código implementado.

Seguindo o diagrama de fluxo, podemos observar que logo após a inicialização o dispositivo recebe uma ordem para ficar inativo durante dois segundos. Este período de inatividade tem como objetivo, dar tempo para o rádio Xbee assumir a rede Xbee, estando preparado para o envio e receção dos dados, antes da sua comunicação interna com o Arduino.

Houve a necessidade de um menu para configuração de alguns parâmetros do tanque, assim como calibração de alguns sensores. Dado a este facto, a cada ciclo de funcionamento o Arduino verifica se o seu tempo de funcionamento é superior a cinco segundos. Este tempo tem como objetivo certificar que o Arduino entre no menu de configuração apenas quando é inicializado. Todos os dados configurados através do menu, são armazenados na EEPROM, seguidamente os dados da EEPROM são associados a suas respetivas variáveis, evitando a utilização desnecessária da EEPROM.

Seguidamente o Arduino entra em um ciclo infinito capaz de solicitar informações aos sensores, verifica se o valor recebido está dentro dos padrões máximos e mínimos estipulados no menu de configuração, faz tocar um alarme caso necessário e preparar os dados para envio por rádio através do Xbee. É de citar que a *frame* de dados para o envio da informação através do rádio Xbee, necessita da informação no formato hexadecimal. Desta forma o array deve conter os dados separados carácter a carácter, para posterior conversão do seu código ASCII (*American Standard Code for Information Interchange*) para hexadecimal. Desta forma houve a necessidade de transformar os dados dos sensores que são do tipo *float*, em cadeias de caracteres, antes de envia-los para os seus respetivos lugares na frame de dados.

Por fim, os dados são enviados utilizando uma frame API do tipo *Transmit Request*, espera-se por uma resposta do destino e consoante a resposta, pisca o LED da cor determinada.

5.4 Desenvolvimento da Aplicação para o Servidor

Nesta secção falaremos sobre o programa desenvolvido para o coordenador da rede, este programa tem como objetivo, pegar em toda a informação recebida pelo coordenador, tratar os dados e armazenar a informação numa base de dados.

O coordenador funciona como um *gateway*, toda a informação recebida pelo seu rádio Xbee é encaminhada para o servidor conforme citado anteriormente em 4.1.

Para este programa utilizou-se a linguagem de programação Python, esta foi a linguagem escolhida, pois além de ser *freeware*, é uma linguagem de alto nível, orientada a objetos, com atribuição dinâmica de tipos de dados, de fácil percepção e com várias bibliotecas implementadas, inclusive para utilização com o Xbee.

A Figura 5.3 demonstra o funcionamento da aplicação desenvolvida, quando a aplicação é inicializada são efetuadas algumas configurações, como por exemplo velocidade de funcionamento da comunicação serial RS232. Após todas as configurações, inicia-se a comunicação entre o computador e o Arduino. Seguidamente a aplicação entra em um ciclo infinito, até que surja uma interrupção por parte do utilizador. Este ciclo infinito, tem como obrigação verificar se é possível efetuar uma ligação a base de dados, que está alojada em um servidor remoto. Caso não seja possível os dados são armazenados localmente e quando a ligação é estabelecida, os dados são atualizados para a base de dados na Internet.

Este procedimento foi necessário devido a inconsistência da ligação do laboratório da Caviar Portugal a Internet, assim garantimos que mesmo quando não exista ligação, os dados sejam armazenados localmente, e posteriormente quando reestabelecida a ligação, os dados sejam atualizados para a base de dados remota.

Ainda é de responsabilidade do ciclo infinito, receber a *frame* que chega no rádio Xbee do coordenador, recolher a informação referente a cada sensor contido no tanque, verificar se algum parâmetro esta fora da gama de valores dito como normais e caso necessário enviar uma mensagem de alerta, avisando ao utilizador de tal anomalia. Por fim os dados são disponibilizados para o utilizador, enquanto são armazenados na base de dados.

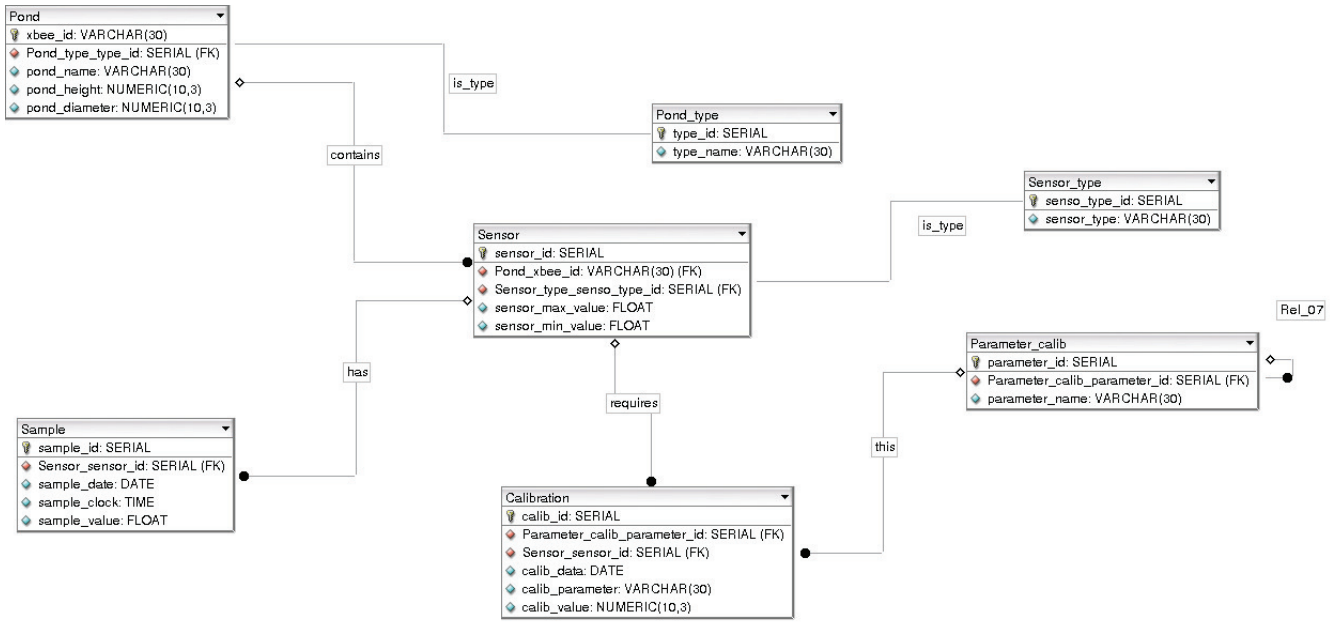


Figura 5.1: Diagrama do modelo E/R da base de dados

.

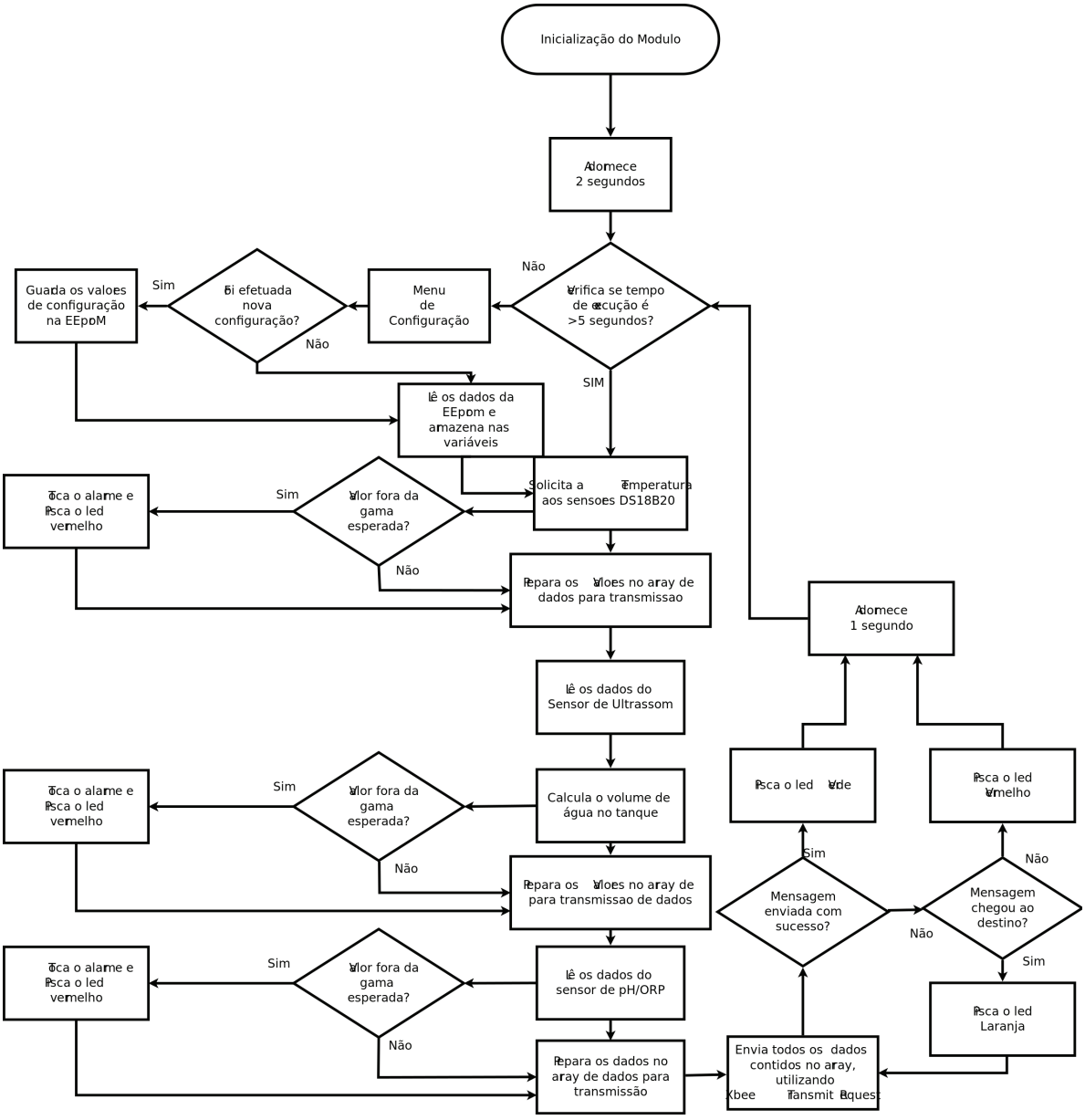


Figura 5.2: Diagrama de Fluxo do código de programação do Arduino

.

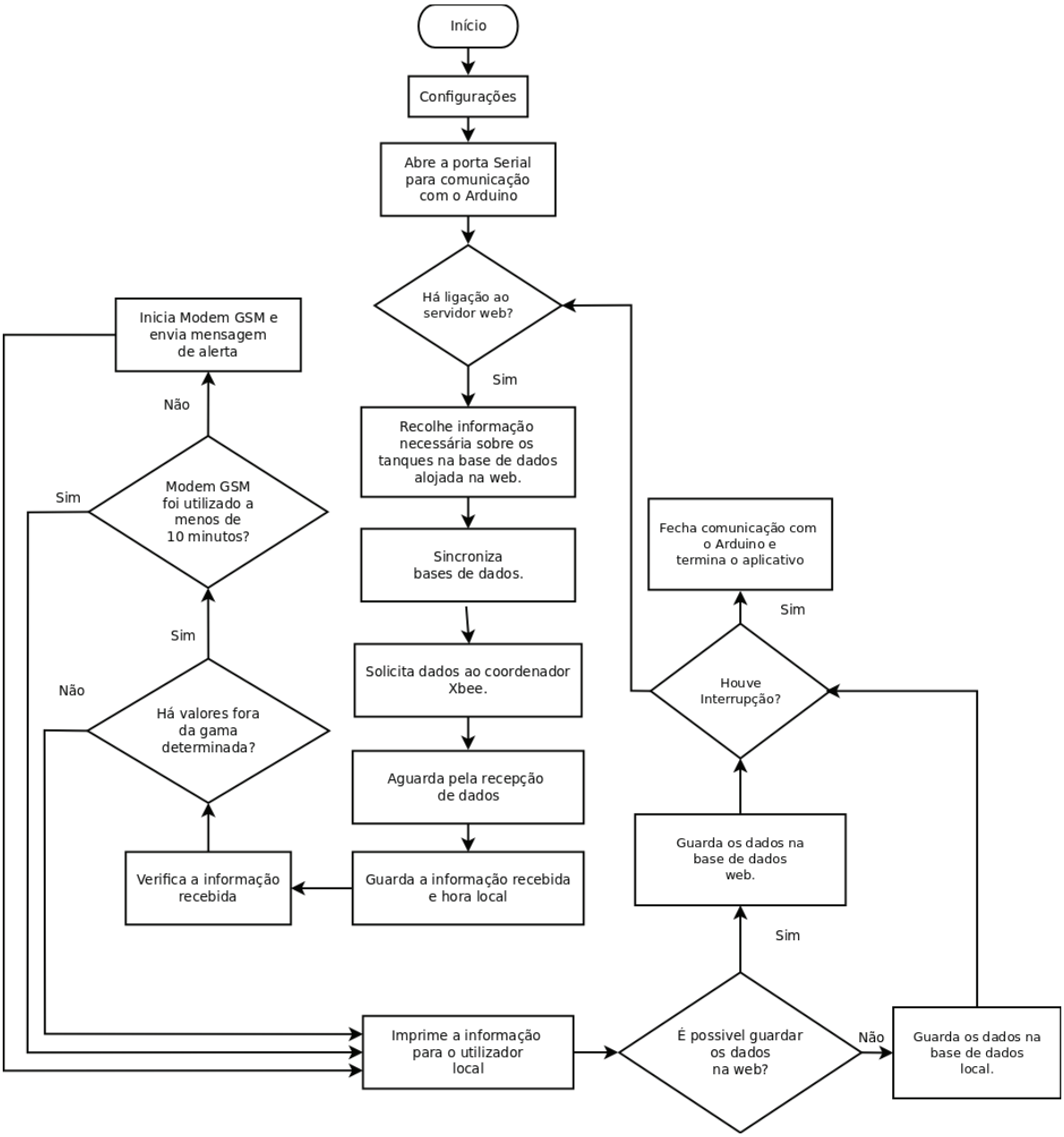


Figura 5.3: Diagrama de Fluxo da Aplicação desenvolvida em Python.

Capítulo 6

Ensaio do protótipo e análise do funcionamento

Após a montagem dos protótipos e implementação dos códigos e aplicações necessárias para o funcionamento da RSSF, foi feito um ensaio com todos os componentes da rede. Neste capítulo demonstraremos e analisaremos, todos os passos efetuados para que a RSSF seja capaz de efetuar a monitorização dos tanques.

6.1 Configuração dos rádios Xbee

A configuração dos rádios foram efetuadas com o auxílio da ferramenta X-CTU, disponibilizada pela própria Digi International, o X-CTU pode ser obtido gratuitamente no site do fornecedor. Entretanto esta aplicação está disponível apenas para o sistema operativo Windows. Para utilização em distribuições Linux (como é o caso do Xbuntu), há a necessidade de instalar o aplicação Wine, como referido no Capítulo 5.

Na Figura 6.1 temos as principais configurações do rádio Xbee router/sensor utilizado no protótipo.

Apesar da existência de vários modelos de *firmwares* disponíveis para o Xbee Serie 2, no testes efetuados no laboratório, a versão XB24-B demonstrou-se mais estável e consistente, frente a outras versões mais recentes como por exemplo a ZB24-ZB. Quando configurado nesta ultima versão, apesar de os dispositivos estarem na mesma PAN ID, não reconheciam o canal de comunicação determinado pelo coordenador da rede e não era possível a comunicação.

Como a versão XB24-B funcionou perfeitamente e atendeu todas as necessidade para a implementação da RSSF e esta foi a firmware escolhida. O modo de funcionamento utilizado foi o Zigbee Router/End Device API, que como já citado é capaz de funcionar como um dispositivo final, sem perder as funcionalidade de um dispositivo do tipo router.

A PAN ID escolhida foi a 2030 (apenas por comodidade, poderia ser utilizado qualquer valor), o modo de funcionamento API habilitado, a velocidade da comunicação serial foi configurada em 9600 e o nome do dispositivo foi dado de acordo com o tanque a monitorizar.

Devido ao facto de o laboratório estar ligado a um gerador para possíveis falhas de energia e os protótipos assim como o servidor ainda estarem ligados a um UPS (*Uninterruptible Power Source*), capaz de suportar o tempo de comutação entre a falha de energia e a ligação gerador. As configurações referentes a gestão de energia e modo

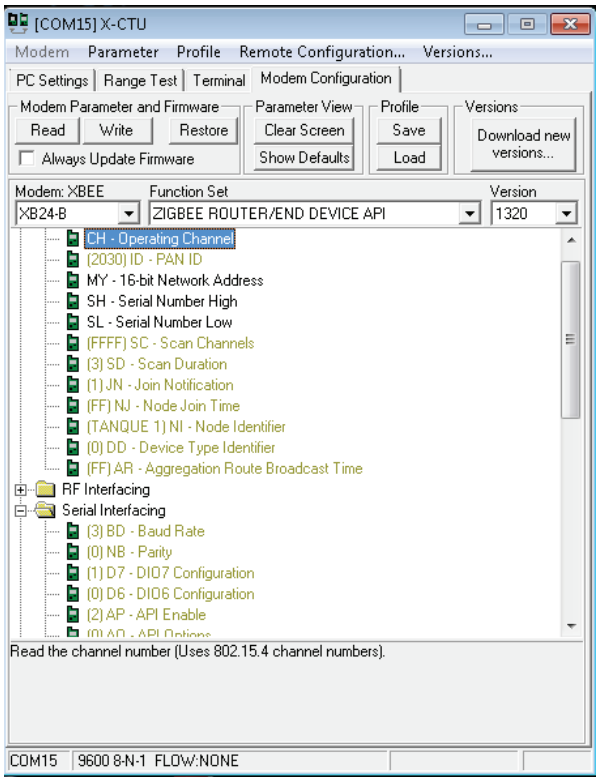


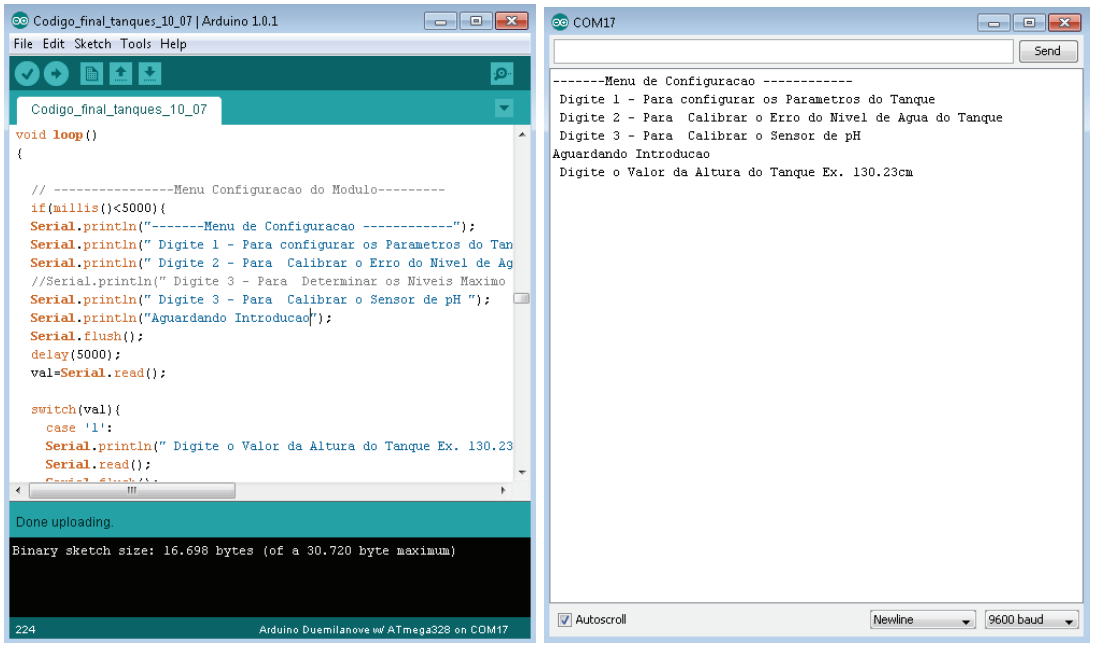
Figura 6.1: Configuração do Xbee através do X-CTU.

sleep, não foram utilizadas nesta dissertação. Deixando assim todos os dispositivos ativos continuamente, servido de caminho alternativo para que os dados dos tanques mais distantes cheguem ao coordenador. Este procedimento permitiu que o envio dos dados do "tanque 4", que está a uma distância considerável do coordenador, fossem reencaminhados por um dos outros tanques chegando até o coordenador. O reencaminhamento dos dados pode ser confirmado, quando desativamos todos os outros dispositivos da rede e o "tanque 4"perdeu sua comunicação com o coordenador, bastou ligar um dos dispositivos para recomeçar a comunicação.

A configuração do rádio Xbee do dispositivo coordenador da rede, difere da configuração dos demais rádios unicamente no tipo de funcionamento. Para este Xbee em questão, foi utilizada a opção Zigbee Coordinator API.

6.2 Introdução do código no Arduino

O Arduino disponibiliza a seu próprio ambiente de trabalho e programação, este ambiente pode ser utilizado em qualquer sistema operativo, de forma simples e prática. Este foi o ambiente utilizado para criar e compilar o código para o Arduino. Na Figura 6.2 (a) podemos ter uma visão geral do IDE. Além de criar, compilar o código e carrega-lo para a placa do Arduino, este Ambiente de programação ainda possibilita a troca de informações com o Arduino através de um monitor. Este monitor será utilizado para a configuração inicial do Arduino e calibração dos sensores, conforme Figura 6.2 (b) A introdução dos dados na placa Arduino é feita de forma simples, necessitamos apenas de selecionar o modelo Duemilanove e a porta de comunicação. Neste projeto, foram utilizadas algumas bibliotecas externas, como é o caso da XBee.h que [Faludi, 2010] cita



(a) Ambiente de programação Arduino (b) Monitor de comunicação disponível no próprio IDE

Figura 6.2: Demonstração do ambiente de programação e do seu monitor serial.

em seu livro e outras desenvolvidas pelo próprio autor, como por exemplo Ultrasonic.h. Todas estas bibliotecas devem estar associadas ao IDE do Arduino.

6.3 Instalação dos dispositivos nos tanques.

Após os dispositivos estarem devidamente abrigados em caixas, como foi demonstrado na Secção 4.3, foram instalados em uma barra horizontal, localizada por cima dos tanque, conforme pode ser visualizado na Figura 6.3. Os sensores foram ligados aos módulos e instalados estrategicamente nos tanques, conforme necessidade dos parâmetros a monitorizar. Apesar de os módulos ficarem por cima dos tanques, a instalação decorreu de forma simples e segura, podendo ser efetuada por qualquer utilizador, sem nenhuma experiência em eletrónica.

6.4 Execução da aplicação para recolha dos dados no servidor

Após todos os módulos devidamente instalados nos tanques, ligou-se o dispositivo coordenador da rede a uma das portas USB do servidor e iniciou-se a aplicação para monitorização dos tanques. Apesar de se poder executar a aplicação no terminal, apenas com a execução do seguinte comando "python server.py", foi criado um atalho, capacitando que a aplicação seja executada apenas com um clique duplo com o rato sobre a mesma. Devido ao facto de a aplicação efetuar todas as configurações necessárias para o seu funcionamento, até mesmo a deteção da porta USB por onde deve comunicar, tornou a aplicação muito simples de ser utilizada.

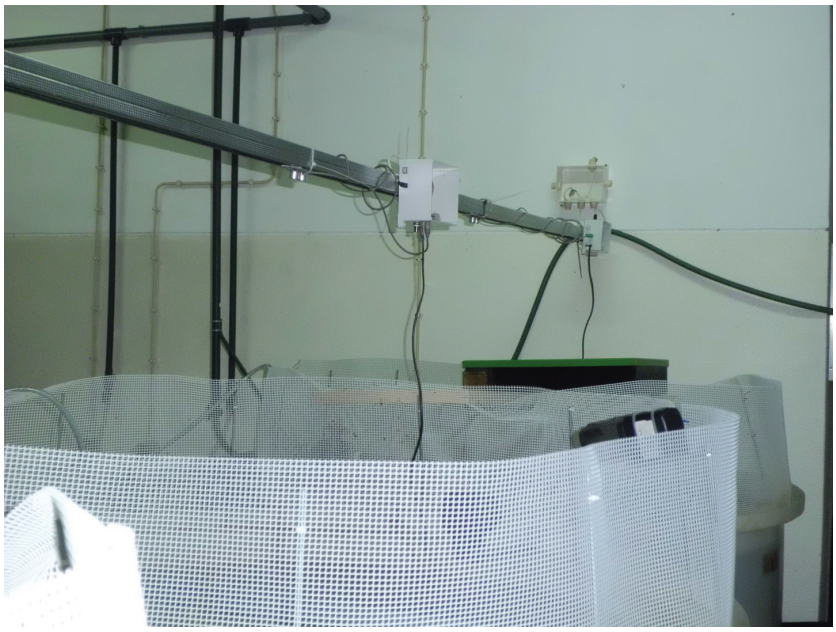


Figura 6.3: Instalação dos sensores por cima dos tanques.

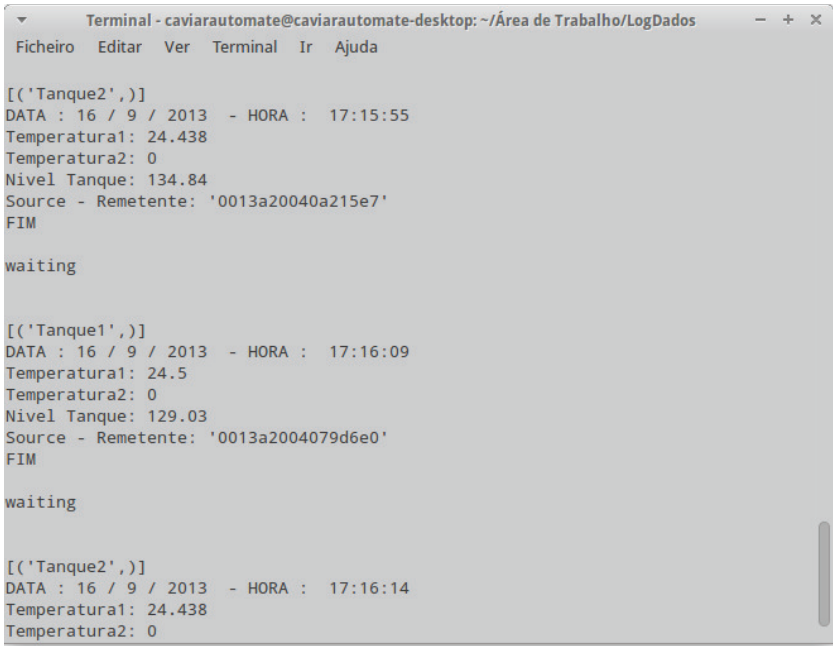


Figura 6.4: Servidor a efetuar a aquisição dos dados, de dois tanques de forma cíclica.

Apesar de não haver uma solicitação por parte do coordenador, para que os módulos envie os dados referentes aos sensores e ele aguarde pela receção, de forma periódica e sincronizada. Todos os dados referentes aos quatro tanques utilizados neste ensaio chegaram de forma correta ao servidor, podendo ser tratado e guardado na base de dados, conforme é possível observar na Figura 6.4

6.5 Testes e resultados obtidos

Com o objetivo de analisar o funcionamento do sistema de monitorização, foram efetuados alguns testes e ensaios finais, com o intuito de determinar se o protótipo correspondia ao que foi proposto.

6.5.1 Teste do alerta por mensagem de texto

Para testar o envio das mensagens de alerta, foi simulando um excesso de agua no "tanque 1", fazendo suir o nível do tanque para valores acima do estipulado. Mesmo com os quatro tanques a enviar informação em simultâneo para o coordenador, após o nível do tanque estar acima dos valores determinados, o tempo médio necessário para a receção da mensagem de alerta foi de um minuto e meio. Como o tempo necessário entre a falha no parâmetro e o tocar do alarme local ocorre de forma praticamente instantânea, o tempo médio do atraso na receção da mensagem, foi baseado no tocar do alarme. Foram efetuados dez ensaios consecutivos, que consistia em subir o nível do tanque e ao tocar do alarme, cronometrar o tempo até a receção da mensagem.

Dado ao facto de que o alerta tem como objetivo avisar um utilizador que não se encontra presente no laboratório, o tempo necessário para que o parâmetro em causa alcance valores drásticos, é muito maior, quando comparado com o tempo necessário para o envio da mensagem. O tempo de atraso entre o evento e a mensagem de alerta está dentro dos aceitáveis para este trabalho.

6.5.2 Falsos alertas no sensor de nível

Durante a implementação deste projeto deparamos com uma situação de alertas "fantasmas", referente ao nível do tanque. Estes alertas ocorriam principalmente durante a manutenção dos tanques como já comentado em 3.2. Durante os primeiros testes do sensor de ultrassom, havia a situação de imediatamente após algo passar entre o sensor e a linha da água, fazia disparar os alertas. Para determinar se o problema estava realmente solucionado, o sistema ficou montado e funcionando por vários dias consecutivos. Durante esse período não houve ocorrência de problemas com o disparo dos alarmes.

6.5.3 Falhas da conexão a Internet

Outra característica imposta durante o desenvolvimento deste projeto, foi a necessidade de um sistema que mesmo em caso de falhas, fosse na rede elétrica ou na conexão, fosse capaz de continuar recebendo os dados continuamente.

A solução encontrada para suprir as falhas de conexão, foi a implementação de uma rotina capaz de verificar se é possível o acesso a Internet e caso não seja, guarda os dados localmente, para posterior submissão na base de dados. Para testar o seu funcionamento, com o programa a funcionar e a guardar os dados diretamente na base de dados remota, desligo-se o cabo de rede. Apesar de o cabo estar desligado, a aplicação continuou a receber a informação dos tanques e a passou a guardar os dados localmente, no momento em que o cabo foi ligado e a conexão restabelecida, os dados anteriormente guardados, foram carregados para a base de dados remota de forma contínua e sem perda de integridade. Satisfazendo assim as expectativas impostas para este teste.

Capítulo 7

Conclusões e trabalhos futuros

Os testes efetuados com o protótipo final mostraram resultados satisfatórios, o desenvolvimento dos códigos propostos para o Arduino e para o servidor demonstram a importância do conjunto hardware-software no desenvolvimento de uma RSSF para monitorização de sistemas, neste caso de aquacultura. Apesar de não ser uma solução completa para a monitorização deste tipo de ambiente, os resultados finais foram satisfatórios, demonstrando que os objetivos impostos no inicio do projeto foram alcançados.

A escolha selecionada dos sensores, e dos equipamentos totalizou um custo médio por protótipo de 217,00 euros, este valor não inclui custo de mão de obra para montagem. Os valores dispostos na Tabela 7.1, demonstra que o protótipo é viável economicamente. Provando que é possível implementar um sistema de monitorização para qualquer ambiente, seja ele aquático ou não, com custos relativamente baixos, quando comparado com outras soluções existentes no mercado. Como exemplo, cito um sistema capaz de monitorizar parâmetros em tanques de aquacultura, proposto pela empresa *Itelmaltis Control Systems* que pode ser visualizada no Anexo 7.

Preço médio dos componentes do protótipo		
Componente	Quantidade	Valor
Arduino Duamilenove	1	20,00 €
Xbee Shield	1	15,00 €
Rádio Xbee S2	1	24,00 €
Ultrassom HC-SR04	1	9,00 €
pH/ORP Adapter	2	28,00 €
Elétrodo pH	1	27,00 €
Elétrodo ORP	1	60,00 €
Sensor DS18B20	1	6,00 €
Valor Total		217,00 €

Tabela 7.1: Custo médio para aquisição dos principais componentes do protótipo.

A utilização dos módulos Xbees e a implementação da RSSF, tornou capaz a criação de um sistema flexível e portável. Eliminando a excessividade de cabos existentes nos sistemas de monitorização.

Em relação a transmissão dos dados, foi possível determinar que o alcance da comunicação é uma característica dependente da configurações utilizada, podendo ser

adaptada conforme a necessidade do sistema a monitorizar.

O software apesar de não ter uma interface simples e amigável, nem permitir configurações e consultas por parte do utilizador. Cumpre as funções que lhe foram determinadas.

No geral podemos concluir que este projeto cumpre os objetivos inicialmente propostos. Entretanto tratando-se de um protótipo, há muita coisa por fazer e melhorar. Assim sendo apresento algumas soluções para trabalhos futuros,

- Adicionar sensores de Clark, para que seja possível a determinação da quantidade de oxigénio dissolvido na água, fator de grande importância na monitorização de sistemas de aquacultura.
- Adicionar atuadores para que o sistema além de monitorizar os parâmetros, também seja capaz de os controlar.
- Desenvolver um circuito eletrónico único, capaz de acomodar o Microcontrolador, rádio Xbee, e os demais componentes de hardware necessários para funcionamento do equipamento.
- Implementação de uma aplicação para dispositivos móveis, permitindo a monitorização dos tanques de qualquer localização.
- Melhorar a aplicação do servidor, adicionando uma interface gráfica para uma melhor monitorização dos tanques e interação com o utilizador.
- Apesar de não ser o ponto chave nesta dissertação, mas sendo uma das maiores qualidades das redes RSSF utilizando o protocolo Zigbee. Proponho também o estudo da eficiência energética e possível melhoria do sistema.
- Outro fator importante que poderia ser abordado no futuro, seria um estudo para utilização de uma comunicação de dados segura, uma vez que o Xbee permite criptografia na comunicação dos dados.

Bibliografia

Inc. Analog Devices. Datasheet Temperature Sensors, 2013. URL http://www.analog.com.

Arduino. Arduino Duemilanove, 2013. URL arduino.cc/en/Main/arduinoBoardDuemilanove.

José R. Azinheira. Sensores e Actuadores, 2006. URL http://lars.mec.ua.pt/public/LAR%20Projects/Humanoid/2004_DavidGameiro_FilipeCarvalho/Sensores/sa_folhas.pdf.

Maria C. Barreto. Método Polarográfico - Eléctrodo de Oxigénio Tipo Clark, 2007. URL http://www.barreto.uac.pt/bq2_prat/04_elect02.pdf.

Joseph J. Carr John M. Brown. *Introduction to Biomedical Equipment Technology, Third Edition*. 2012.

Atmel Corporation. CAVR121: Enhancing ADC resolution by oversampling, 2005. URL http://www.atmel.com/Images/doc8003.pdf.

Digi, International. XBee®/XBee-PRO® ZB RF Modules, 2012. URL http://www.digi.com.

Robert Faludi. *Building Wireless Sensor Networks* . 2010.

Drew Gislason. *Zigbee Wireless Networking*. 2008.

Corporation Glolab. Relays, The Electromechanical amplifier, 2013. URL http://www.glolab.com/relays/relays.html.

Sérgio M.M. Jesus. Processamento Digital de Sinal, 2005. URL http://w3.ualg.pt/~sjesus/aulas/pds.

Rye Rim Lee. A Study on IEEE 802.15.4 based Factory Automation System in Wireless Network, 2009.

Denise Lemes. *PostgreSQL - Conceitos e Aplicações*. Erica, 2004. ISBN 9788571948907.

Charles Borges de Lima. Os Poderosos microcontroladores AVR, 2009.

Otto Jacob Litjens. Automação de Estufas Agrícolas Utilizando Sensoriamento Remoto e o Protocolo Zigbee, 2009.

Antonio A. F. Loureiro. Redes de Sensores sem Fios, 2008. URL http://www.ic.unicamp.br/~cmbm/desafios_SBC/loureiroredesensores.pdf.

Ind. Maxim Integrated. Datasheet DS18B20, 2013. URL http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf.

Afonso C. Medina and Leonardo Chwif. Modelagem e Simulação de Eventos Discretos , 2006.

Antonio R. Messias. Controle remoto e aquisição de dados via XBee/ZigBee (IEEE 802.15.4), 2013. URL http://www.rogercom.com/.

Luís Fernando Patsko. Funcionamento e Utilização de Sensores, 2006. URL http://www.maxwellbohr.com.br/downloads/robotica/mec1000_kdr5000/tutorial_eletronica_-_aplicacoes_e_funcionamento_de_sensores.pdf.

Tadeu José Silva Rodrigues. Sensores de pH, 2010.

Marco Antonio Vieira Sales. Sistema de supervisão e controle da concentração de microalgas e temperatura na maturação de ostras em tanques de cultivo, 2004.

Patricio Serendero. Normalização Base de Dados, 2011.

Peter Stallinga. Instrumentation, 2011. URL http://www.stallinga.org/AcadActiv/Lectures/IALP/Theory/11_Instrumentation/Instrumentation.pdf.

Peter Stallinga. *Instrumentation Electronic*. 2012.

Vítor Grade Tavares and José Machado Silva. Conversores Analógico/Digital e Digital/Analógico, 2005. URL http://paginas.fe.up.pt/~jms/E3/conversoresAD.pdf.

Inc. Texas Instruments. Datasheet Temperature Sensors, 2013. URL http://www.ti.com.

Anexo A

Adelino Venturinha
Rua da Fonte Velha, Edif. Duas Palmeiras
Pechão, 8700-178 Olhão
T +351 289 821 003
F +351 289 812 358
E-Mail: adelinoventurinha@itelmatis.com

Para/To:
Paulo Pedro
Caviar Portugal
ppedro@caviarportugal.com

Assunto/Subject: ***Ref.ª:*** 4381180612R
Orçamento para sistema de monitorização de piscicultura

Exmo(s) Senhor(es),

Conforme solicitado, segue-se o orçamento para os trabalhos em epígrafe com as condições a seguir descriminadas:

1- Construção e fornecimento de um quadro elétrico para a monitorização de um sistema de piscicultura, com integração no S-Monitor, e com as seguintes características:
   a) Caixa em PVC;
   b) Módulo de entradas digitais;
   c) Módulo de entradas analógicas;
   d) Conversor RS485;

   Fornecimento dos seguintes equipamentos:
   e) Uma sonda de oxigénio dissolvido + temperatura com transmissor;
   f) Seis sondas de nível magnéticas;
   g) Dois transmissores para sondas PT100;
   h) Fonte de alimentação 230V/24V;
   i) Quatro Relés adequados para o efeito;
   j) Equipamento UPS;
   k) Modem GSM;

   Fornecimento e configuração do sistema de controlo remoto do sistema composto pelos seguintes itens:
   l) Licença de utilização do *software* de gestão técnica S-Monitor **50 variáveis**;
   m) Configuração de computador para o utilizador;

2- Inclusões e exclusões deste orçamento:
   a) Este orçamento não contempla mão-de-obra para a instalação do quadro elétrico e restantes equipamentos no local da instalação;
   b) Este orçamento contempla construção do quadro nas instalações da Itelmatis, e fornecimento do mesmo em conjunto com os restantes equipamentos;
   c) Este orçamento não contempla um PC, apenas contemplando a configuração do mesmo.

Alínea	Descrição	Valor da Proposta	Itens * Aceites
1	Construção e fornecimento de sistema de monitorização para piscicultura	€2.541,43	