16º Congresso de Inovação, Ciência e Tecnologia do IFSP - 2025

SISTEMA AUTOMATIZADO DE MONITORAMENTO SEM FIO UTILIZANDO PROTOCOLO DE COMUNICAÇÃO ZIGBEE E BANCO DE DADOS SQL.

RAMON OLIVEIRA ANTONIO[1], MARCOS VINÍCIUS ALVES DE OLIVEIRA[2]

[1]Graduando em Engenharia Elétrica, Bolsista PIBIFSP, IFSP, Campus Votuporanga, ramon.antonio@aluno.ifsp.edu.br.
[2]Mestre em Engenharia Elétrica e professor no IFSP - Campus Votuporanga, marcos.oliveira@ifsp.edu.br

Área de conhecimento (Tabela CNPq): 3.04.05.02-5 Automação Eletrônica de Processos Elétricos e Industriais.

RESUMO: O avanço da Internet das Coisas (IoT) e das tecnologias de comunicação sem fio tem possibilitado sistemas de monitoramento em tempo real, embora muitos ainda apresentem limitações em escalabilidade e integração de dados. Este trabalho apresenta o desenvolvimento de um sistema automatizado de monitoramento sem fio baseado em módulos ZigBee (XBee Pro S2B), Arduino Mega e banco de dados SQL, configurado em topologia coordenador–escravo. A integração foi realizada por meio de scripts em Python e PHP, permitindo inserção e consulta de dados no MySQL, além da criação de uma interface web para visualização em tabelas e gráficos. Os testes com sensor ultrassônico demonstraram comunicação estável entre os módulos, inserção consistente das leituras no banco de dados e atualização em tempo real na interface web, sem perdas significativas ou interferências relevantes na faixa de 2,4 GHz. O desempenho do banco de dados confirmou consultas rápidas e geração eficiente de históricos, enquanto a interface web local mostrou-se eficaz para acompanhamento em tempo real e exportação dos dados. Conclui-se que a arquitetura modular proposta é confiável, escalável e flexível, apresentando potencial para aplicações em monitoramento distribuído e supervisão remota.

PALAVRAS-CHAVE: Banco de dados SQL, Comunicação sem fio, Internet das Coisas, Monitoramento distribuído, Redes ZigBee, Sensores.

AUTOMATED WIRELESS MONITORING SYSTEM USING ZIGBEE COMMUNICATION PROTOCOL AND SQL DATABASE

ABSTRACT: The advancement of the Internet of Things (IoT) and wireless communication technologies has enabled real-time monitoring systems, although many still face limitations in scalability and data integration. This work presents the development of an automated wireless monitoring system using ZigBee (XBee Pro S2B) modules, an Arduino Mega, and an SQL database, configured in a coordinator–end device topology. Integration was achieved through Python and PHP scripts for data insertion and querying in MySQL, along with a web interface for visualization in tables and graphs. Tests with an ultrasonic sensor demonstrated stable communication between modules, consistent data recording in the database, and real-time updates on the web interface, with no significant packet losses or interference in the 2.4 GHz band. The database performance ensured fast queries and efficient

generation of historical records, while the local web interface proved effective for real-time monitoring and data export. The results indicate that the proposed modular architecture is reliable, scalable, and flexible, highlighting its potential for distributed monitoring and remote supervision applications.

KEYWORDS: Distributed Monitoring, Internet of Things, Sensors, SQL Database, Wireless Communication, ZigBee Networks.

INTRODUÇÃO

O avanço da comunicação sem fio e da Internet das Coisas (IoT) tem permitido o desenvolvimento de sistemas de monitoramento capazes de coletar, transmitir e armazenar dados em tempo real, sendo aplicados em áreas como automação, saúde, agricultura de precisão e gestão predial, trazendo otimização de recursos e redução de custos (NGUYEN et al., 2021).

Redes de sensores sem fio (*Wireless Sensor Networks* — WSN) integram diversos dispositivos para medição e controle distribuído, com baixo custo de implementação e manutenção. Entre as tecnologias utilizadas, destaca-se o ZigBee, devido a baixo consumo de energia, confiabilidade e custo acessível (AGARWAL et al., 2013). Conforme **??**), "a tecnologia ZigBee é projetada para aplicações que requerem redes seguras, baixa taxa de dados, longa duração de bateria e baixa complexidade", tornando-a adequada para monitoramento contínuo.

A coleta eficiente de dados deve ser acompanhada de armazenamento estruturado. Bancos de dados SQL permitem consultas complexas, criação de históricos e integração com análises, agregando valor aos sistemas distribuídos (KARTHIKEYAN, 2010).

Este trabalho apresenta o desenvolvimento de um sistema automatizado de monitoramento sem fio utilizando módulos ZigBee e banco de dados SQL, garantindo transmissão confiável, registro persistente e consultas históricas, detalhando a arquitetura, configuração dos módulos e integração com o banco de dados.

MATERIAIS E MÉTODOS

O sistema de monitoramento sem fio foi desenvolvido utilizando Arduino Mega, módulos XBee Pro S2B e um banco de dados SQL hospedado localmente via XAMPP, permitindo aquisição, transmissão e visualização de dados de sensores em tempo real.

O desenvolvimento iniciou-se pela integração do Arduino Mega com um sensor ultrassônico e um módulo XBee configurado como escravo. O módulo coordenador foi conectado ao computador via adaptador USB. A configuração foi realizada no XCTU, definindo PAN ID, endereços únicos e modo transparente (AT) para transmissão direta dos dados. A Figura 1 ilustra a interface utilizada.

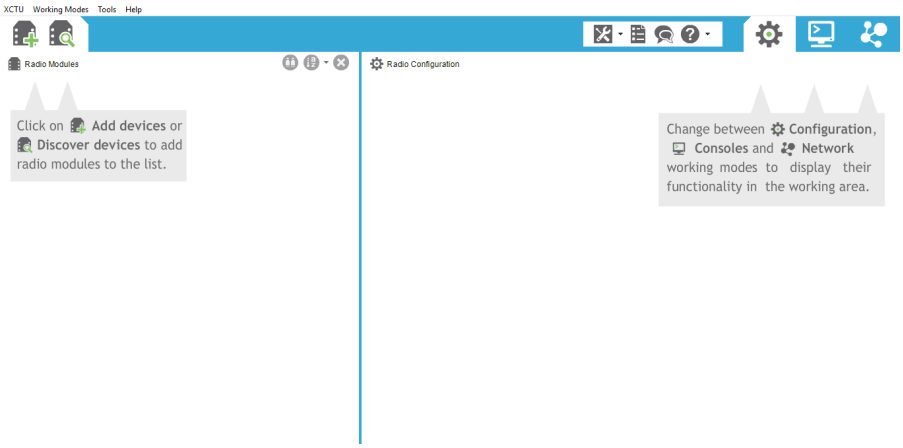


Figura 1: Interface do XCTU utilizada para configuração dos módulos XBee. (Os autores, 2025)

O módulo escravo foi acoplado ao Arduino Mega via shield, garantindo comunicação serial direta. O coordenador recebeu os dados e enviou ao banco MySQL no XAMPP via scripts PHP. A Figura 2 mostra a configuração física.

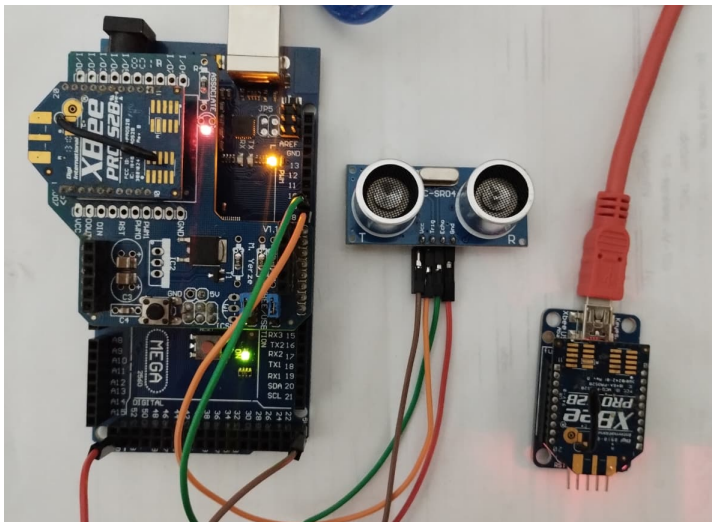
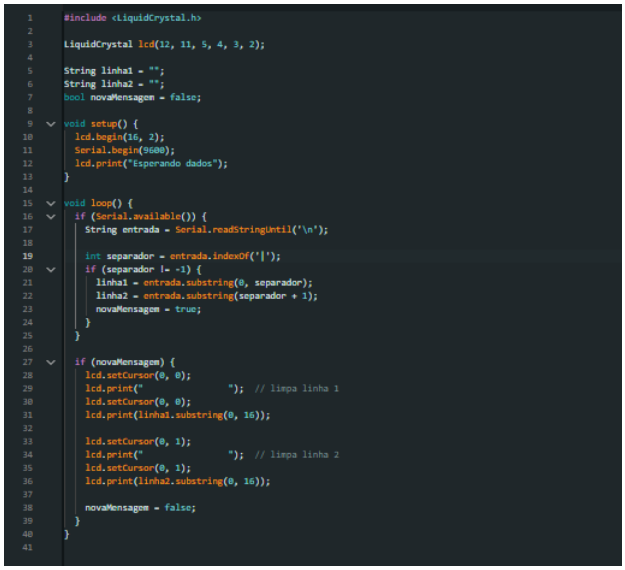


Figura 2: Configuração dos módulos XBee: escravo acoplado ao Arduino Mega e coordenador conectado ao PC via USB.

Os códigos nas Figuras 3 e 4 realizam a aquisição, transmissão e armazenamento dos dados. O Arduino captura e segmenta leituras enviadas via XBee, enquanto o Python insere os dados no banco MySQL, tratando exceções para maior robustez.



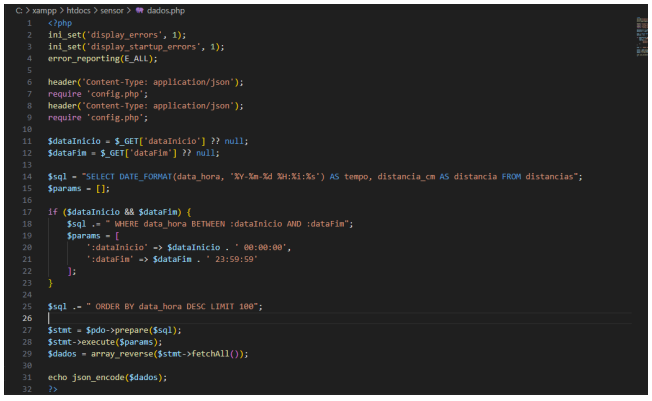
```arduino
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

String linha1 = "";
String linha2 = "";
bool novaMensagem = false;

void setup() {
  lcd.begin(16, 2);
  Serial.begin(9600);
  lcd.print("Esperando dados");
}

void loop() {
  if (Serial.available()) {
    String entrada = Serial.readStringUntil('\n');

    int separador = entrada.indexOf('|');
    if (separador != -1) {
      linha1 = entrada.substring(0, separador);
      linha2 = entrada.substring(separador + 1);
      novaMensagem = true;
    }
  }

  if (novaMensagem) {
    lcd.setCursor(0, 0);
    lcd.print("                ");  // limpa linha 1
    lcd.setCursor(0, 0);
    lcd.print(linha1.substring(0, 16));

    lcd.setCursor(0, 1);
    lcd.print("                ");  // limpa linha 2
    lcd.setCursor(0, 1);
    lcd.print(linha2.substring(0, 16));

    novaMensagem = false;
  }
}
```

Figura 3: Código Arduino.

```python
import serial
import mysql.connector
from mysql.connector import Error

# CONFIGURE AQUI
porta_serial = 'COM3'
baud_rate = 9600

# Conexão com o banco de dados
def conectar_bd():
    return mysql.connector.connect(
        host='localhost',
        user='root',
        password='',  # padrão do XAMPP é senha vazia
        database='sensor_db'
    )

# Insere no banco
def inserir_distancia(distancia):
    try:
        conn = conectar_bd()
        cursor = conn.cursor()
        query = "INSERT INTO distancias (distancia_cm) VALUES (%s)"
        cursor.execute(query, (distancia,))
        conn.commit()
        cursor.close()
        conn.close()
        print(f"Distância {distancia} cm inserida no banco.")
    except Error as e:
        print(f"Erro ao inserir no banco: {e}")

# Leitura da serial
try:
    with serial.Serial(porta_serial, baud_rate, timeout=1) as ser:
        print(f"Conectado em {porta_serial}, aguardando dados...\n")
        while True:
            linha = ser.readline().decode('utf-8').strip()
            if linha.startswith("Distância:"):
                try:
                    valor = linha.split(":")[1].strip().replace(" cm", "")
                    distancia = float(valor)
                    inserir_distancia(distancia)
                except ValueError:
                    print("Erro ao converter valor:", linha)
except serial.SerialException:
    print(f"Erro: Porta {porta_serial} não encontrada.")
except KeyboardInterrupt:
    print("Finalizado pelo usuário.")
```

Figura 4: Código Python.

A comunicação com o banco MySQL é feita via scripts PHP, que permitem consultas em JSON filtradas por período, garantindo eficiência e controle. A Figura 5 apresenta o código dessa integração.



```php
<?php
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);

header('Content-Type: application/json');
require 'config.php';
header('Content-Type: application/json');
require 'config.php';

$dataInicio = $_GET['dataInicio'] ?? null;
$dataFim = $_GET['dataFim'] ?? null;

$sql = "SELECT DATE_FORMAT(data_hora, '%Y-%m-%d %H:%i:%s') AS tempo, distancia_cm AS distancia FROM distancias";
$params = [];

if ($dataInicio && $dataFim) {
    $sql .= " WHERE data_hora BETWEEN :dataInicio AND :dataFim";
    $params = [
        ':dataInicio' => $dataInicio . ' 00:00:00',
        ':dataFim' => $dataFim . ' 23:59:59'
    ];
}

$sql .= " ORDER BY data_hora DESC LIMIT 100";

$stmt = $pdo->prepare($sql);
$stmt->execute($params);
$dados = array_reverse($stmt->fetchAll());

echo json_encode($dados);
?>
```

Figura 5: Código de integração com o banco de dados. (Os autores, 2025)

O desenvolvimento seguiu uma abordagem modular: testes de aquisição, transmissão via XBee, recepção pelo coordenador e armazenamento no banco, finalizando com interfaces web para visualização em tempo real.

Foram definidas métricas para avaliar o desempenho: **latência de transmissão**, pela diferença entre aquisição e registro no banco; **taxa de perda de pacotes**, comparando leituras enviadas e armazenadas; **disponibilidade**, verificando operação contínua por 24 horas; **consumo de recursos**, monitorado no servidor em diferentes taxas de aquisição; e **precisão**, aferida pela comparação com valores de referência manuais. Essas métricas validaram a eficiência e robustez do sistema, orientando melhorias e futuras expansões.

RESULTADOS

As medições realizadas pelo sensor ultrassônico conectado ao Arduino Mega, transmitidas sem fio pelos módulos XBee Pro S2B, foram armazenadas em um banco de dados MySQL e disponibilizadas por uma interface web local, permitindo monitoramento e análise em tempo real.

Na Figura 6, observa-se a interface do phpMyAdmin, integrado ao XAMPP, exibindo inserções automáticas das leituras e evidenciando a consistência dos dados coletados.

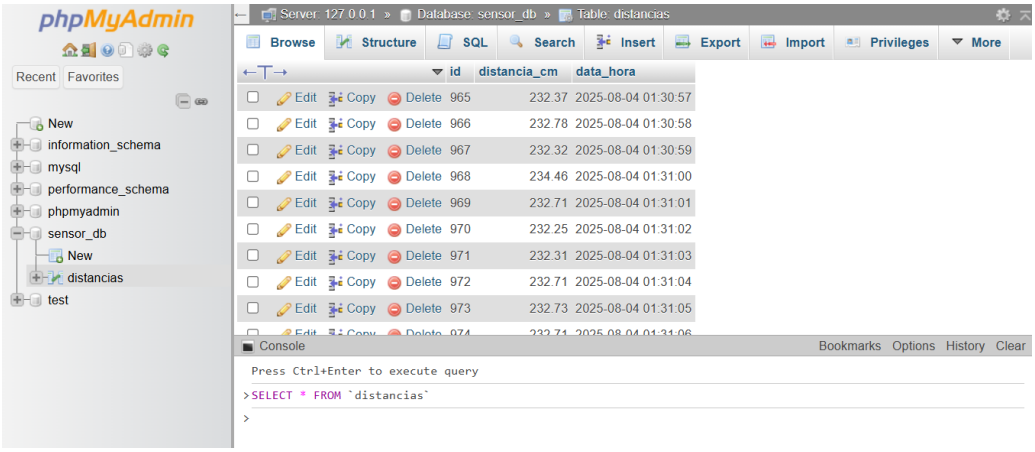


Figura 6: Interface do banco de dados MySQL no XAMPP mostrando registros das leituras. (Os autores, 2025)

A Figura 7 apresenta a interface web local (localhost), onde as medições são exibidas em tabelas e gráficos de tendência, incluindo registros de data e hora, facilitando o acompanhamento histórico e o monitoramento em tempo real.

Figura 7: Interface web local (localhost) para visualização das leituras em tempo real. (Os autores, 2025)

O banco de dados MySQL apresentou desempenho consistente, permitindo consultas rápidas e geração de históricos, em consonância com recomendações de (KARTHIKEYAN, 2010). A integração com scripts PHP e exportação em JSON possibilitou a atualização dinâmica da interface web.

Apesar de operar em 2,4 GHz, não foram identificadas interferências significativas, evidenciando a eficácia da seleção adaptativa de canal ((CHONG et al., 2015)). Pequenas variações observadas nas leituras derivam das limitações próprias do sensor, não da comunicação sem fio.

Os resultados apontam que o sistema é escalável, confiável e adequado para consultas em tempo real, alinhando-se às práticas descritas em (AGARWAL et al., 2013) e (NGUYEN et al., 2021). No

entanto, limitações como alcance restrito e sensibilidade a obstáculos ((ZHONG et al., 2018)) devem ser consideradas em expansões. Além disso, arquiteturas baseadas em APIs RESTful ou bancos NoSQL podem ampliar a escalabilidade em aplicações mais complexas ((AGARWAL et al., 2013)).

CONCLUSÕES

O sistema de monitoramento sem fio desenvolvido atingiu os objetivos propostos, demonstrando aquisição e transmissão confiáveis de dados de sensores, com registro persistente no banco de dados MySQL.

As interfaces web permitiram monitoramento em tempo real e controle remoto de sensores e atuadores, evidenciando a flexibilidade da solução. Testes adicionais serão realizados com mais sensores, integrando o sistema a uma unidade de aquaponia para acompanhamento contínuo.

A arquitetura modular mostrou-se escalável, possibilitando futuras expansões sem comprometer a performance. O sistema apresenta-se como uma solução integrada, confiável e adaptável para monitoramento distribuído e supervisão remota.

CONTRIBUIÇÕES DOS AUTORES

Marcos Vinícius Alves de Oliveira e Ramon Oliveira Antonio contribuíram com a elaboração e concepção do projeto. Ramon Oliveira Antonio realizou a pesquisa e o desenvolvimento do projeto. Todos os autores contribuíram com a revisão do trabalho e aprovaram a versão submetida.

AGRADECIMENTOS

Os autores agradecem ao IFSP – Campus Votuporanga por fornecer a estrutura adequada para o desenvolvimento do projeto e ao Programa Institucional de Bolsas de Iniciação Científica do IFSP pelo financiamento e bolsa do aluno.

REFERÊNCIAS

AGARWAL, Anshul et al. A Study of ZigBee Technology. **International Journal on Recent and Innovation Trends in Computing and Communication**, v. 1, n. 4, p. 287–292, 2013.

CHONG, Jo Woon et al. An Adaptive WLAN Interference Mitigation Scheme for ZigBee Sensor Networks. **International Journal of Distributed Sensor Networks**, v. 2015, p. 1–16, 2015. DOI: <10.1155/2015/851289>.

KARTHIKEYAN, Ramanathan. Routing Strategy Selection for Zigbee Mesh Networks. **International Journal of Communications, Network and System Sciences**, v. 3, p. 608–611, 2010. DOI: <10.4236/ijcns.2010.37081>.

NGUYEN, Cuong V. et al. ZigBee based data collection in wireless sensor networks. **International Journal of Informatics and Communication Technology**, v. 10, n. 3, p. 211–224, 2021. DOI: <10.11591/ijict.v10i3.pp211-224>.

ZHONG, Xiaolei et al. Research on Scalable Zigbee Wireless Sensor Network Expansion Solution. In: 032071. IOP Conference Series: Materials Science and Engineering. [S.l.: s.n.], 2018. v. 394. DOI: <10.1088/1757-899X/394/3/032071>.