INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
DEPARTAMENTO DE ÁREAS ACADÊMICAS IV
BACHARELADO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Gustavo Mota Martins de Souza

SEGUE O BICHO: DESENVOLVIMENTO DE UMA COLEIRA PARA MONITORAMENTO DE ANIMAIS SILVESTRES

Goiânia
2023

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[X] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico/Tecnológico - Tipo: _____	

Nome Completo do Autor: Gustavo Mota Martins de Souza
Matrícula: 20171010700395
Título do Trabalho: Segue o bicho: Desenvolvimento de uma coleira para monitoramento de animais silvestres

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Goiânia, 17 / 10 / 2024.
Local Data

Gustavo Mota . M de Souza

_____
Assinatura do Autor e/ou Detentor dos Direitos Autorais

Gustavo Mota Martins de Souza

SEGUE O BICHO: DESENVOLVIMENTO DE UMA COLEIRA PARA MONITORAMENTO DE ANIMAIS SILVESTRES

Trabalho de Conclusão de Curso de Graduação em Bacharelado em Engenharia de Controle e Automação do Instituto Federal de Educação Ciência e Tecnologia de Goiás como requisito para a obtenção do título de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Carlos Roberto da Silveira Junior

Goiânia
2023

So895s Souza, Gustavo Mota Martins de
Segue o bicho: desenvolvimento de uma coleira para monitoramento de animais silvestres / Gustavo Mora Martins de Souza. – Goiânia: Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia, 2024.
95 f.: il.

Orientador: Prof. Dr. Carlos Roberto da Silveira Junior.

TCC (Trabalho de Conclusão de Curso) – Bacharelado em Engenharia do Controle e Automação, Departamento de Áreas Acadêmicas IV, Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

1. Sistema inteligente. 2. Geolocalização. 3. Radiocomunicação. 4. Bioma Cerrado. I. Silveira Junior, Carlos Roberto da Silveira (orientador). II. Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia. III. Título.

CDD 005.133

Ficha catalográfica elaborada pela Bibliotecária Karol Almeida da Silva CRB1/ 2.740
Biblioteca Professor Jorge Félix de Souza,
Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

ATA DE DEFESA DE TCC 2 - 2023/2

ATA DA SESSÃO PÚBLICA ONLINE DE APRESENTAÇÃO E DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO DE BACHARELADO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO, DAS ÁREAS ACADÊMICAS IV, DO INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS, CAMPUS GOIÂNIA.

Aos quinze dias do mês de dezembro de dois mil e vinte e três, a partir das dez horas, no formato híbrido, foi realizada a sessão pública de apresentação e defesa do Trabalho de Conclusão de Curso intitulado "**SEGUE O BICHO: DESENVOLVIMENTO DE UMA COLEIRA PARA MONITORAMENTO DE ANIMAIS SILVESTRES** ", de autoria do graduando **Gustavo Mota Martins de Souza (matrícula 20171010700395)**, do curso de Bacharelado em Engenharia de Controle e Automação, como requisito final para obtenção do título de Engenheiro de Controle e Automação.

A banca examinadora foi composta pelos seguintes membros: **Prof. Dr. Carlos Roberto da Silveira Junior** (Orientador/Presidente) (SIAPE: 1485525), do IFG – Campus Goiânia, coordenação de Engenharia Elétrica, **Prof. Claudio Afonso Fleury** (SIAPE: 1125472), do IFG campus Goiânia, coordenação de Engenharia de Controle e Automação e **Prof. Dr. Josemar Alves dos Santos Junior** (SIAPE: 1948498), IFG, Campus Inhumas, coordenação de Engenharia de Controle e Automação, **Dr. Ricardo Henrique Fonseca Alves** (CPF 030.455.821-41), Engenheiro Eletricista - Petrobrás.

Após a apresentação, os autores foram questionados pela banca examinadora, cujo parecer final foi:

( ) Aprovado (x) Aprovado com correções ( ) Aprovado com restrições ( ) Reprovado.

A presente ata foi lavrada após encerrar-se a sessão, sendo assinada eletronicamente pela banca examinadora.

Observações:_____

O trabalho corrigido deverá ser apresentado ao Professor Orientador no prazo máximo de 30 dias a partir da data de defesa.

FECHAMENTO: Nada mais havendo a tratar, o presidente da banca avaliadora, Prof. Carlos Roberto da Silveira Junior, encerrou esta sessão pública de apresentação e defesa do Trabalho de Conclusão de Curso (TCC) às **11h 30min**.

Eu, ***Leonardo Costa de Paula***, Coordenador do curso de Bacharelado em Engenharia de Controle e Automação, sem mais nada a constar neste documento, lavrei a presente ATA assinada digitalmente.

Banca Examinadora:

Prof. Dr. Carlos Roberto da Silveira Junior
(Orientador/Presidente)

Prof. Dr. Claudio Afonso Fleury

Prof. Dr. Josemar Alves dos Santos Junior

Dr. Ricardo Henrique Fonseca Alves

Goiânia, 15 de dezembro de 2023.

Documento assinado eletronicamente por:
- **Josemar Alves dos Santos Junior**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 10/01/2024 00:14:26.
- **Claudio Afonso Fleury**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 05/01/2024 10:03:05.
- **Carlos Roberto da Silveira Junior**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 21/12/2023 14:52:58.
- **Leonardo Costa de Paula**, **COORDENADOR(A) DE CURSO - FUC1 - GYN-CCBECA**, em 21/12/2023 14:36:06.

Este documento foi emitido pelo SUAP em 13/12/2023. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifg.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 489746
Código de Autenticação: 0f2457342e

Dedico esse trabalho a todas as crianças que um dia
sonharam em se tornarem cientistas

AGRADECIMENTOS

Agradeço especialmente à minha família. Foram essas pessoas que me incentivaram quando eu necessitava de motivação e estiveram nos momentos difíceis que inevitavelmente todos atravessam em algum momento. Expresso minha gratidão ao meu orientador Carlos Roberto da Silveira Júnior, por estar sempre disponível para prestar auxílio no projeto e por ter me aberto as portas para os frutos que esse projeto possa gerar no futuro. Dirijo meus agradecimentos a todos os meus professores, por terem compartilhado um pouco do seu conhecimento comigo. Aos meus colegas de curso, que contribuíram para o meu desenvolvimento pessoal e profissional ao longo dos anos e que compartilharam bons e maus momentos durante a graduação. Aos meus amigos que, mesmo com cada um tendo rotinas diferentes, sempre puderam dedicar um pouco do seu tempo junto comigo. Aos meus colegas de trabalho com os quais convivi diariamente e que, com essa convivência, me ensinaram o que é necessário para ser um bom profissional. Por fim, minha gratidão a todas as pessoas que, em algum momento, compartilharam algum tempo comigo. Mesmo não estando presentes, deixaram um pouco de si em minhas memórias.

"O perigo de verdade não é que computadores passem a pensar como humanos, mas sim que humanos passem a pensar como computadores." (Sydney J. Harris)

RESUMO

O Cerrado é um dos biomas brasileiros com maior diversidade, caracterizado por um alto valor de endemismo. Apesar de ser o segundo maior bioma do Brasil, a rápida e intensa perda de cobertura vegetal tornou-se uma preocupação crescente. Essas características conferem ao Cerrado o status de Hotspot mundial de biodiversidade, ou seja, áreas naturais ao redor do globo que possuem grande diversidade ecológica, porém estão em risco de extinção. o Que serve como Alerta sobre os perigos da conversão de áreas de vegetação natural em pastagens e áreas de cultivo, e seus impactos diretos na biodiversidade. Nesse contexto, medidas conservacionistas precisam ser minuciosamente consideradas. Neste projeto, o monitoramento do comportamento de animais silvestres é considerado uma medida es- sencial, proporcionando um aprimoramento do conhecimento sobre a relação das espécies monitoradas com o ecossistema e permitindo a definição de políticas públicas que minimizem os impactos antrópicos. Para isso foi realizado o desenvolvimento de uma coleira de monitoramento de animais silvestres, empregando a comunicação via radiofrequência e geolocalização, juntamente com a comunicação via celular. O sistema estabelece conexão com as antenas de celulares para transmitir os dados de localização do animal, que são armazenados na nuvem. Esses dados armazenados geram representações gráficas e mapas, os quais podem ser acessados remotamente. O desenvolvimento do protótipo emprega módulos eletrônicos comerciais, que se encontram no mercado brasileiro, criando um produto de baixo custo.

Palavras-chave: Monitoramento; Sistema inteligente; Geolocalização; Radiocomunicação; Comunicação GPRS.

ABSTRACT

The Cerrado is one of the most diverse Brazilian biomes, characterized by a high value of endemism. Despite being the second largest biome in Brazil, the fast and intense loss of vegetation cover has become a growing concern. These characteristics confer to the Cerrado the status of World Hotspot of biodiversity, that is Rais around the globe that have great ecological diversity, but are at risk of extinction. Warning us about the dangers of converting areas of natural vegetation into pastures and cultivation areas, and their direct impacts on biodiversity. In this context, conservationist measures need to be thoroughly considered. In this project, the monitoring of the behavior of wild animals is considered a measure providing an improvement of knowledge about the relationship of species monitored with the ecosystem and allowing the definition of public policies that mini mizem the anthropic impacts. For this was carried out the development of a collar monitoring of wild animals, using radio frequency communication and geolocation, together with cellular communication. The system establishes connection with cellular antennas to transmit the location data of the animal, which are stored in the cloud. These stored data generate graphical representations and maps, which can be accessed remotely. The development of the prototype employs electronic modules, which are in the Brazilian market, creating a low-cost product.

Keywords: Monitoring; Smart system; Geolocation; Radio communication; GPRS communication

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CETAS	Centro de Triagem de Animais Selvagens
CETAS-GO	Centro de Triagem de Animais Selvagens de Goiâs
GPRS	*General Packet Radio Service*
GSM	*Global System for Mobile*
IBAMA	Instituto Brasileiro do Meio Ambiente
ICMBio	Instituto Chico Mendes de Conservação de Biodiversidade
IDE	*Integrated Development Environment*
IOT	*Internet Of Things*
MMA 178	Ministério do Meio Ambiente, portaria n° 178
MVP	Mínimo Produto Viável
NMEA	*National Marine Electronics Association*
ONGs	Organizações Não Governamentais
PCB	Placa de circuito impresso
PWM	*Pulse Width Modulation*
SMS	*Short Message Service*
UHF	*Ultra High Frequency*
VHF	*Very High Frequency*

LISTA DE SÍMBOLOS

mAh	MiliAmpère-Hora
mA	MiliAmpère

SUMÁRIO

1 INTRODUÇÃO

Estima-se que 20% das espécies existentes no planeta façam parte do ecossistema brasileiro. Apenas no que diz respeito às aves, o Brasil possui 1971 espécies (BUCHERONI, 2021), além de 848 espécies de répteis, 1188 de anfíbios (ICMBio RAN, 2022), e 775 espécies de mamíferos (ABREU *et al.*, 2022). Apesar da riqueza de vida, o número de animais na fauna brasileira já foi consideravelmente maior, uma vez que várias espécies já foram extintas e muitas outras estão atualmente ameaçadas de extinção.

Ao longo da história do planeta, houve cinco eventos de extinção em massa, nos quais diversas espécies desapareceram em um período relativamente curto. Estima-se que 99,9% das espécies que já existiram na Terra foram extintas. Além disso, muitos cientistas afirmam que enfrentamos a sexta extinção em massa, com espécies à desaparecer 100 vezes mais rapidamente do que no passado. De acordo com um estudo conduzido por Matt Davis, Søren Faurby e Jens-Christian Svenning, serão necessários milhões de anos para que os mamíferos se recuperem das extinções causadas pelas atividades humanas (WILCOX, 2018).

A classificação do status de conservação da fauna é divulgada pelo Ministério do Meio Ambiente, por meio da Lista de Espécies da Fauna Brasileira Ameaçadas de Extinção, resultado da avaliação das espécies da fauna conduzida pelo ICMBio (Instituto Chico Mendes de Conservação de Biodiversidade), órgão ambiental do governo brasileiro. A lista mais recente foi publicada no dia 08/06/2022 no Diário Oficial da União, através da portaria MMA 178 (Ministério do Meio Ambiente, portaria n° 178). O levantamento indica que o país possui 1.249 espécies e subespécies da fauna ameaçadas de extinção, onde que 144 animais tiveram seu status de ameaça reduzido, enquanto 219 novos animais foram incluídos (PIMENTA, 2022).

Nesse contexto, o monitoramento da fauna silvestre assume um papel crucial, uma vez que é um programa que tem com objetivo avaliar as populações de animais silvestres em determinado habitat (AMBPLUS, 2023). Esse procedimento é geralmente exigido durante o processo de licenciamento ambiental de empreendimentos, sejam eles empresas privadas ou construções públicas, que possam gerar poluição ou potencialmente afetar o meio ambiente. Esses monitoramentos buscam avaliar o impacto da instalação ou operação desses empreendimentos em áreas que abrigam a fauna silvestre.

Geralmente, o monitoramento é exigido pelo órgão regulador do meio ambiente no estado em questão e deve ser realizado de acordo com as normas estabelecidas pelo IBAMA (Instituto Brasileiro do Meio Ambiente) e pelo respectivo órgão estadual em ques-

tão (AMBPLUS, 2023). É de suma importância analisar os dados recebidos para detectar quaisquer alterações no ecossistema, uma vez que a construção desses empreendimentos pode impactar o local de procriação desses animais ou até mesmo resultar na redução de espécies ou indivíduos na região.

O monitoramento da fauna silvestre possibilita uma compreensão aprofundada da utilização e efetividade dos corredores ecológicos, que desempenham o papel crucial de aumentar o tamanho e as chances de sobrevivência de populações de diferentes espécies. Adicionalmente, eles permitem a recolonização com populações de espécies localmente reduzidas e contribuem para a diminuição da pressão sobre as áreas protegidas (ARRUDA; NOGUEIRA DE SÁ, 2003). Em áreas de domínio particular, tais ações correspondem a Áreas de Proteção Ambiental, que consistem em faixas de fragmentos de áreas naturais destinadas à preservação e conservação da fauna e flora.

Além disso, nos casos de acidentes ou captura de animais selvagens em ambientes urbanos, os animais são encaminhados ao CETAS (Centro de Triagem de Animais Selvagens) para avaliação, cuidados, adaptação e posterior reintrodução no meio ambiente. O monitoramento subsequente à soltura dos animais selvagens assume papel relevante, uma vez que fornece informações valiosas sobre sua readaptação ao habitat, monitora a população em áreas específicas e avalia o comportamento, entre outros aspectos relevantes. Nesse sentido, o monitoramento oferece subsídios essenciais para aprimorar a eficácia das ações de reintrodução de animais selvagens.

Portanto, o monitoramento da fauna silvestre possui uma importância estratégica significativa, tanto para o empreendimento quanto para a conservação da biodiversidade. Desse modo, torna-se um procedimento essencial em áreas onde empresas operam e que também servem como habitat para animais silvestres (AMBPLUS, 2023), o que permite uma avaliação precisa da diversidade de espécies e animais na região. Tal abordagem facilita o desenvolvimento de estratégias de preservação ou mesmo ações de resgate direcionadas a espécies ameaçadas. Com essa abordagem, ameaças à biodiversidade podem ser prontamente enfrentadas, o que mitiga o risco de extinção de espécies de extrema importância para o equilíbrio ecológico (AMBPLUS, 2023).

1.1 OBJETIVOS

O objetivo central deste projeto consiste na criação de um colar de custo acessível destinado ao monitoramento de animais selvagens, que visa garantir adaptabilidade às necessidades dos parceiros envolvidos.

Os objetivos específicos são:

(a) Fazer o teste dos componentes para entender como cada módulo funciona, além de verificar se eles não possuem nenhum defeito.

(b) Integrar todos os módulos em um único sistema consolidado.

(c) Desenvolver um protótipo de coleira eletrônica de geolocalização e radiocomunicação destinado à monitorização da posição do animal.

(d) Elaborar um artigo científico fundamentado nos resultados obtidos ao longo do projeto.

Nesse capítulo foram apresentados contextos e dados que explicam a importância do monitoramento dos animais. No próximo capítulo, abordar-se um contexto mais específico, que apresentará a importância do monitoramento no bioma Cerrado.

2 MONITORAMENTO DA FAUNA DO CERRADO

O Cerrado destaca-se como a maior e mais rica savana da região neotropical, sendo que essa região compreende uma região extensa biogeograficamente que abrange desde a América Central até a América do Sul (inclusive o Brasil), incluindo parte do sul do México e da península da Baixa Califórnia, e ilhas do Caribe. Além disso o Cerrado abriga aproximadamente de 20% a 50% da biodiversidade brasileira e além de possuir uma vasta variedade de animais endêmicos, que são as espécies nativas, restritas a determinada região geográfica, ou seja, ocorrem exclusivamente em uma certa região (MYERS, 2000; KLINK; MACHADO, 2005). Com uma área de cerca de dois milhões de quilômetros quadrados, ocupa aproximadamente 22% do território nacional e engloba 8 das 12 bacias hidrográficas do país, o que o caracteriza como a "caixa d'água do Brasil"(RIBEIRO; WALTER, 2008).

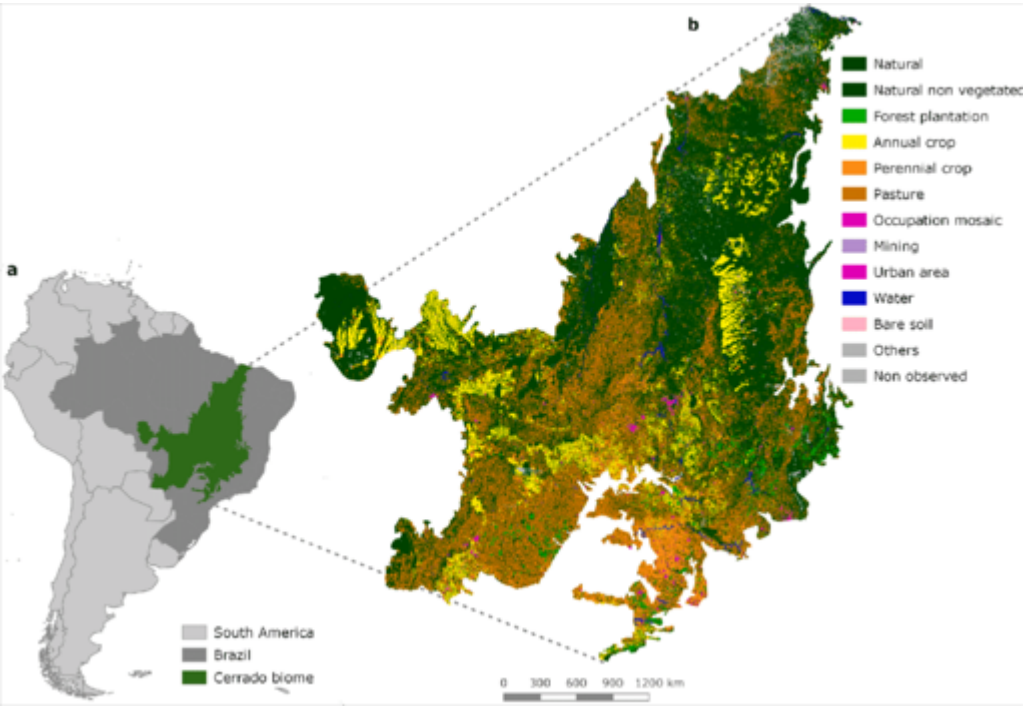
Quanto à composição florística, o Cerrado se manifesta como um mosaico. o que favorece a formação de micro-habitats específicos, os quais, por sua vez, impulsionam a diferenciação de nichos ecológicos para as diversas espécies nativas, o que atende suas distintas exigências ecológicas (DA SILVA RODRIGUES, 2007; PAULA HANNA VALDUJO *et al.*, 2009).

Quase metade da área original do Cerrado já sofreu alterações devido, principalmente, ao avanço dos limites agrícolas e da urbanização (STRASSBURG, 2017), já chegou ao ponto em que a atividade pecuarista representa a principal forma de ocupação do solo nesse bioma (SANO *et al.*, 2008). Conforme mapeamento do Terraclass até 2013, a conversão de áreas nativas do Cerrado, quando somadas, abrangem quase metade do bioma (TERRACLASS, 2013). Em contraste, apenas 9,6% desse bioma está protegido dentro de Unidades de Conservação (FRANÇOSO *et al.*, 2015), o que é consideravelmente baixo em relação à importância desse bioma. A Figura 1 apresenta o mapa da cobertura vegetal obtido pelo Terraclass, em 2013.

A fragmentação e perda de habitat representam um dos principais obstáculos enfrentados por esses animais, o que limita a conexão entre os fragmentos, o que dificulta a dispersão e causa o isolamento, o que resulta em perdas significativas na riqueza e abundância dessas comunidades. (GALETTI; ALVES-COSTAS; CAZETTA, 2013; MAGLE; THEOBALD; CROOKS, 2009; BATEMAN; FLEMING, 2012).

Por conta da grande ocupação humana e sua alta densidade populacional cada vez maior, os conflitos entre os humanos e os animais silvestres são bem amplos, o que inclue atropelamentos, ataques a animais domésticos, caça ilegal, dentre outros (FORMAN *et al.*, 2002; GEHRT; RILEY, 2010; URBANEK; NIELSEN; WILSON, 2009; MAGLE;

Figura 1 – Mapa do uso e cobertura da terra no Cerrado .



Fonte: TerraClass (2013).

THEOBALD; CROOKS, 2009). Esses conflitos geram uma percepção errônea das pessoas em relação aos animais, que chegam a ser perseguidos por retaliação, o que acarreta na redução quantitativa de populações em regiões onde há uma maior presença humana. Esse cenário modifica o comportamento de espécies diretamente afetadas, as quais evitam o contato com o humano, o que deixa ainda mais difícil monitorá-las (MCKINNEY, 2006).

De outro modo, os desafios para a conservação das espécies nativas do Cerrado evidenciam a importância da conservação dos remanescentes naturais, nos quais há elevada riqueza e composição das espécies (BINI *et al.*, 2006; FRANÇOSO *et al.*, 2015; AZEVEDO *et al.*, 2016). Além disso, a manutenção e a implementação de novas áreas protegidas mostram-se como meios eficazes de conservação da fauna e flora do Cerrado (BENSUMAN, 2006; FRANÇOSO *et al.*, 2015). Uma notícia promissora é que o crescimento do número de inventários e estudos de monitoramento padronizados tem aumentado o conhecimento sobre as histórias de vida de espécies, o que permite a ampliação dos registros de ocorrência das espécies e novas descrições, com finalidade de impulsionar e viabilizar ações conservacionistas (CARDOSO; POMBAL, 2010; ANDRADE; VAZ-SILVA, 2012; FRANÇOSO *et al.*, 2015; VAZ-SILVA *et al.*, 2015; ANDRADE *et al.*, 2016; ANDRADE *et al.*, 2020;).

A conservação e preservação da fauna nativa, principalmente daquelas localizadas

em áreas vizinhas a regiões de grande densidade populacional, ou seja, sob forte pressão antrópica, faz-se necessária e depende inicialmente da identificação dos fragmentos e ambientes mais utilizados por determinados grupos. Deve-se identificar os principais efeitos da pressão antrópica nessas comunidades, além de identificar o uso de fragmentos antropizados, os quais mascaram os efeitos deletérios dos processos antrópicos ao disponibilizar recursos e abrigos às espécies com maior facilidade (FURLONGER; DEWAR; M.B., 1987; DONCASTER; DICKMAN; D.W., 1990; STATHAM, M.; STATHAM, H. L., 1997).

É possível selecionar organismos modelo para a implementação do monitoramento via rádio-colares, uma vez que algumas espécies apresentam ampla distribuição pelo bioma, áreas de vida consideráveis e são afetadas significativamente pelos efeitos da atividade humana. A monitorização pós-soltura revela-se crucial por permitir ter mais informações a respeito da readaptação desse animal no habitat, monitoramento de população em determinadas áreas, avaliação de comportamento, dentre outros. O monitoramento fornece subsídio para proporcionar maior eficiência na soltura de animais silvestres.

Desse modo, as alterações ambientais causadas pelos seres humanos ao longo da história têm dificultado a coexistência harmoniosa entre o homem e a natureza, o que afeta os ecossistemas e impede que alguns serviços naturais sejam oferecidos. Os animais desempenham papel essencial na prestação desses serviços, uma vez que eles contribuem para a produção de alimentos, o controle de pragas e a manutenção do equilíbrio ecológico (DELTA S, 2021).

Nesse contexto, o monitoramento de animais selvagens é fundamental para avaliar os impactos ambientais causados por diferentes tipos de empreendimentos e compreender como essas atividades afetam a fauna local. Além disso, o monitoramento aumenta o conhecimento sobre a biodiversidade em regiões pouco estudadas. Esses estudos são conduzidos de maneira sazonal ao longo de todas as fases de construção e operação dos empreendimentos, em conformidade com as normas do IBAMA (DELTA S, 2021).

Diante dessa importância, os pesquisadores começaram a monitorar sistematicamente os movimentos dos animais, para isso, se utiliza de diversas técnicas para tornar esse monitoramento possível. Por muito tempo, os pesquisadores dependiam de seguir e observar os animais ou capturá-los, marcá-los e esperar pela recaptura. Contudo, com o avanço da tecnologia, o monitoramento remoto tornou-se uma ferramenta essencial para a obtenção de informações sobre os hábitos e movimentos das espécies animais na natureza, com mínima interferência humana (ECO, 2015).

Nesse cápitulo foram abordados a importância do monitoramento no Cerrado. No

próximo capítulo, serão apresentadas algumas tecnologias utilizadas para realizar esses monitoramentos.

3 TECNOLOGIAS UTILIZADAS PARA MONITORAMENTO DE ANIMAIS SILVESTRES

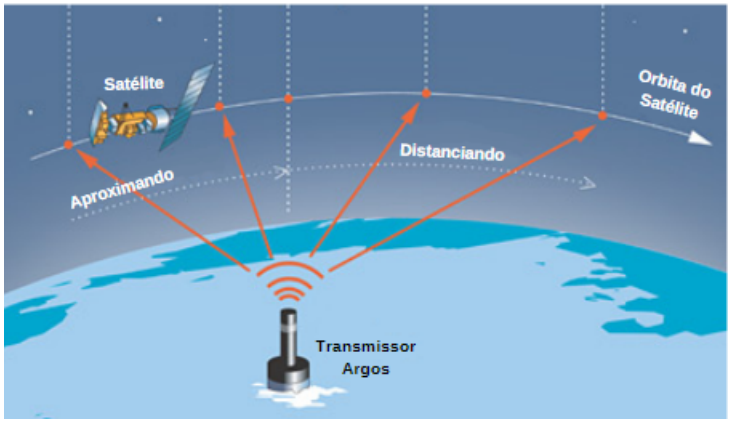
A primeira das tecnologias utilizadas para monitoramento animais silvestres, foi o rastreamento por rádio, essa mesma está em uso desde os anos 50 e ainda hoje é a técnica mais comum e barata, uma vez que permite monitorar desde insetos, como abelhas, até grandes mamíferos, como baleias (ECO, 2015). Os procedimentos de rastreamento desse tipo são conhecidos como radiotelemetria. Esse método usa dois dispositivos: o primeiro trata-se de um transmissor que é ligado ao animal através de dispositivos externos, como, por exemplo, colares, ou, então, implantado no animal cirurgicamente. Esse transmissor armazena informações e as emite em forma de VHF (*Very High Frequency*), um sinal de alta frequência que nada mais é que uma onda eletromagnética, a mesma onda usada nas estações de rádio. Normalmente, cada transmissor é pré-programado com uma frequência única, o que faz com que cada aparelho transmita um sinal diferente, tal como se fossem várias estações de rádio diferentes, o que permite localizar cada animal individualmente (ONÇAFARI, 2020).

O segundo dispositivo é um receptor que, geralmente, é conectado a antenas direcionais capazes de captar o sinal VHF emitido pelo transmissor, como se fosse um rádio doméstico que sintoniza em uma estação (ONÇAFARI, 2020).Porém, para que esse receptor detecte o sinal do colar, é necessário que o equipamento esteja a poucos quilômetros do animal. Caso o animal esteja dentro do alcance, o radiotransmissor enviará um sinal para o receptor, que responderá em forma de bipes (ONÇAFARI, 2020). Após ouvir o bipe, é necessário fazer uma triangulação, isto é, dividir a área de um terreno em vários triângulos, técnica usada em vários campos de conhecimento, como na Topográfia e na matemática, por exemplo. A triangulação ajuda a encontrar o animal em menor tempo.

Já no final da década de 1970, surge um novo método para a realização dos monitoramentos: através de uma parceria Franco Americana foi criado o sistema Argos, que utiliza de uma telemetria proporcionada pelo sistema de satélites Argos, esse monitoramento é conhecido como rastreamento por satélite ou então rastreamento via Argos. Esta técnica também usa de VHF, mas diferente do rastreamento a rádio, o transmissor envia um sinal eletromagnético de alta frequência para um satélite. Quando utiliza essa técnica, o usuário dessa tecnologia pode monitorar o animal à distância. Assim, o acompanhamento pode ser realizado por um computador ou um celular conectado ao satélite (PALMINTERI, 2019). A Figura 2, que foi retirada do próprio manual da Argos, permite a visualizar o funcionamento desse tipo de monitoramento.

De acordo com o manual da Argos, os satélites estão em uma órbita polar a uma altitude de 850 km. Os satélites veem os Polos Norte e Sul em cada revolução orbital.

Figura 2 – Rastreamento via Argos. Um dos tipos de monitoramento mais usados em países polares



Fonte: Argos User's Manual (2007-2016)

Os planos orbitais giram em torno do eixo polar na mesma velocidade que a Terra demora para girar ao redor do Sol, ou seja, uma revolução por ano. Cada revolução orbital atravessa o plano equatorial em tempos solares locais fixos. Portanto, cada satélite passa dentro da visibilidade de qualquer transmissor praticamente na mesma hora local todos os dias. O tempo necessário para completar uma revolução em torno da Terra é de aproximadamente 100 minutos. (ARGOS, 2007-2016). Mas, apesar de todos esses benefícios, por usar satélites em órbita polar, os monitoramentos realizados em coordenadas mais próximas da Linha do Equador têm uma cobertura inferior se comparados às coordenadas em alta latitude. Assim, é necessário pelo menos quatro transmissões para calcular a localização (PALMINTERI, 2019). Além disso, essa forma de monitoramento precisa de uma assinatura de canal, o que faz com que a mesma tenha um custo elevado se comparada às outras tecnologias de monitoramento, já que é usado apenas para programas relacionados à proteção ambiental, programas de educação ambiental e programas governamentais (ECO, 2015).

Na década de 90, surge a mais nova tecnologia de monitoramento, conhecida como rastreamento por localização ou rastreamento por GPS. Essa técnica também usa de satélites para realizar os monitoramentos, mas, diferentemente, o colar não é mais um transmissor, e sim um emissor de rádio. Nessa forma de monitoramento, os dispositivos armazenam as coordenadas geográficas, obtidas pela triangulação feita pelos próprios satélites, em intervalos pré-determinados e com grande precisão (ONÇAFARI, 2020). Os dados podem ser acessados pelo usuário através do próprio dispositivo, ou via UHF (*Ultra High Frequency*), através de um computador, ou *smartphone* conectado à internet. A diferença dessas duas formas de acessar os dados, é que ao usar o UHF (*Ultra High Frequency*), o usuário precisa chegar a uma distância de até 12 metros do animal para estabelecer uma

conexão remota e obter as informações armazenadas (ONÇAFARI, 2020).

O rastreamento por GPS permite ao usuário obter dados sobre a localização de animais independentemente da condição climática, com uma frequência que varia de minutos a semanas (ECO, 2015). Além disso, como o GPS é apenas um receptor, ele requer uma quantidade menor de energia, e a bateria pode ser de pequeno porte. Como não precisa de transmissor, geralmente essa opção de monitoramento é a mais barata (PALMINTERI, 2019).

Outro benefício, se comparado com o rastreamento por satélite, diz respeito à frequência e precisão dos dados. Por exemplo, o Sistema Argos, nas regiões tropicais, possui uma cobertura menor. Geralmente, o Argos fornecerá poucas estimativas de localização por dia, o que resulta em um raio de precisão de 250 metros ou mais. Em contraste, o GPS tem um raio de precisão de 30 metros (PALMINTERI, 2019). Uma taxa de erro de 250 metros pode ser aceitável ao rastrear uma baleia, mas é muito alta se o objetivo for calcular o uso de um corredor florestal por um animal (PALMINTERI, 2019).

Um exemplo de aplicação do sistema de rastreamento via GPS foi uma parceria realizada entre instituições públicas e privadas, com pesquisadores da Embrapa Pantanal, ao qual os mesmos desenvolveram uma versão brasileira de colares GPS. O que o configura como a primeira vez que essa tecnologia foi fabricada no País. Essa tecnologia foi testada em campo para verificar a eficácia da coleta de dados, a durabilidade do aparelho e a adequação deste equipamento em diferentes animais (DICHOFF, 2017). Com isso, a Tabela 1 mostra um resumo que explica como cada tecnologia funciona, além de suas vantagens e desvantagens.

Nos tempos atuais, também existe soluções de monitoramento que usam de inteligência artificial para facilitar no monitoramento dos animais, em 2021 o Instituto Baleia Jubarte teve a ideia de usar câmeras térmicas acopladas em um software especialista em monitoramento. Através da aplicação de Machine Learning, esses softwares conseguem analisar uma quantidade imensa de informações, o que possibilita que esse sistema acione um alarme caso perceba a presença de baleias em um raio de 2 km (FRANÇA, 2022). Grandes empresas da área de tecnologia, como a Microsoft, também têm desenvolvido softwares de Machine Learning e Inteligência Artificial para monitorar os animais. Uma dessas ferramentas é utilizada para combater a extinção dos pinguins. Para isso, são coletadas imagens de 40 locais que servem de habitat para algumas espécies de pinguins. Com base nessas imagens, é criado um modelo de Machine Learning para reconhecer os animais nas fotos e, assim, realizar a contagem com base na densidade populacional observada nas imagens (WAKKA, 2020).

Tabela 1 – Tabela das diferentes tecnologias.

Tecnologia	**Descrição**	**Vantagens**	**Desvantagens**
Rastreamento via rádio	Precisa de dois dispositos para funcionar. O primeiro é um transmissor, emite sinais em forma de VHF. E o segundo dispositivo é um receptor capaz de captar o sinal VHF.	Baixo custo, consome pouco.	Precisa que o operador esteja a poucos quilometros do animal a ser observado.
Rastreamento via Argos	O transmissor envia um sinal para um satélite Argos que está em orbita planar.	precisão maior que o rastreamento a rádio	dos monitoramentos e o que possui o custo mais elevado, e por o sátelite estar em orbita polar a precisão em locais proximos da linha do equador e baixa.
Rastreamento por GPS	O colar recebe as informações de triangulação de varios satélites GPS, que podem ser acessados remotamente.	Mais preciso que todos os outros métodos de rastreamento; É monitorado remotamente	Custo mais elevado se comparado ao rastreamento via rádio, é o tipo de rastreamento que mais consome energia.

Assim, é possível observar que os dispositivos de rastreamento moderno, são capazes de determinar exatamente onde um animal está no momento da medição. Quando se utiliza dos dados que são obtidos através dessas tecnologias, é possível determinar os movimentos migratórios e diários do animal, o tamanho da área que esse animal circula, além de saber quais os seus habitats naturais (ECO, 2015). O uso desses dados permitem, indicar o impacto, positivo ou negativo, do desenvolvimento humano sobre determinado habitat, podem também indicar novas formas de manejo das populações estudadas, ou ainda, verificar se em alguma área há um número suficiente de indivíduos de uma espécie que permita a sua sobrevivência (ECO, 2015).

Apesar do conhecimento de que o mercado brasileiro fabrica equipamentos eficazes para a realização dos monitoramentos, a produção nacional dessa tecnologia não consegue suprir a demanda, uma vez que a mesma é fabricada por um número limitado de empresas, o que dificulta a obtenção desses equipamentos para ONGs e instituições menores, além de elevar consideravelmente o preço. Para determinar o preço do equipamento, foi realizado um levantamento com a empresa Tigrinus Equipamentos para Pesquisa, uma das empresas

mais reconhecidas no mercado brasileiro, os preços obtidos estão demonstrados na Tabela 2.

Tabela 2 – Tabela de solução comercial.

Nosso N°	**Modelo**	**Descrição**	**Valor Unitário**
RT001A	T1.2V	Módulo GPS-VHF (TX-RX)-GPS 66 Canais, armazena coordenadas e transmite com download e upload remoto (alcance do UHF até 150 m), com comunicação com Transceptor UHF (RT001R). Acompanha transmissor VHF (faixa de frequência de 150mHz) adaptado com sensor de atividade/-mortalidade. Alimentado por bateria de Lithium SAFT. Resinado, com formato personalizado (colar, poliamida e couro) para pequenos carnívoros, peso 60g - 70g.	R$ 4.440,00
RT001VHF	T1.2V	Módulo transmissor VHF, faixa de frequência de 150mHz, adaptado com sensor de atividade/mortalidade, Alimentado por bateria de Lithium SAFT. Resinado, com formato personalizado (colar nylon/poliamida e couro), peso aprox. de 40g.	R$ 890,00
RT001R	T1.2V	Tranceptor UHF TX-RX - download e upload de coordenadas de colares-GPS, Compatível com 1 (um) ou mais módulos-GPS Tigrinus RT001A. Display iluminado, com bateria recarregável (incluso) e carregador.	R$ 2.950,00
RT002B	T1.2V	Antena direcional Yagi 5, elementos - com cabeamento - para download de dados (UHF). Compatível para uso no item RT001R.	R$ 700,00
RC-02	DJ-T11X	Receptor RX-VHF. Ampla faixa de VHF, múltipla programação.	R$ 5.900,00
RT002A	T1.2V	Antena direcional Yagi 3 elementos - com cabeamento - para localização de VHF (rastreamento). Compatível para uso no item RC-02.	R$ 700,00

Observa-se na Tabela 2 que o preço varia de acordo com o equipamento, porém alguns desses equipamentos como o RT002B e RT002A, foram feito para melhorar a leitura dos sinais e facilitar o monitoramento, os quais são usados mais como equipamentos complementares, e por isso são comprados em conjunto. Então suponha-se que em um projeto sejam comprados os equipamentos RT001A e RT002A, o preço do equipamento ficaria R$ 5.100,00 (cinco mil e cem reais).

Nesse capítulo foi apresentado as tecnologias usadas para a realização de monitoramento animal, assim como foram apresentadas as previsão do preço de prototipagem do colar que será desenvolvido. No próximo capítulo será abordado a metodologia que será

usada para a execução e organização do projeto.

4 FUNDAMENTAÇÃO TEÓRICA

4.1 PLATAFORMA ARDUINO

O Arduino, além de ser um microcontrolador, é uma plataforma de prototipagem eletrônica de código aberto amplamente utilizada em projetos de eletrônica, robótica e automação. Foi criado por uma equipe de pesquisadores composta por David Cuartielles, David Mellis, Gianluca Martino, Massimo Banzi e Tom Igoe (LIMA CARLOS, 2021), que buscaram desenvolver uma solução acessível e de fácil utilização para entusiastas, estudantes e profissionais da área.

Integrado, em uma única placa, um microcontrolador Atmel AVR com comunicação serial, eles facilitaram o uso do hardware, onde, diferente dos outros microcontroladores disponíveis no mercado, uma placa Arduino não precisaria de gravadores externos e processos complexos para fazer a gravação dos programas (LIMA CARLOS, 2021). A ideia é que o usuário só precise de um cabo USB e uma ideia de programa. A linguagem de programação utilizada é baseada em C/C++. Porém, eles criaram uma IDE (*Integrated Development Environment*) extremamente simples e intuitiva, que oferece uma curva de aprendizado suave para iniciantes. Essa IDE fornece uma plataforma bastante amigável e, o mais importante, o código é carregado do ambiente de programação diretamente para a placa, sem programas intermediários (ELETROGATE, 2023).

Além disso, tanto o hardware quanto o software do Arduino são **open source**, o que significa que o código, os esquemas e o projeto são disponibilizados ao público de forma que qualquer pessoa possa usar, o que permite que os usuários compartilhem e reutilizem programas e projetos (MCROBERTS, 2015). Essa característica de código aberto proporciona uma grande flexibilidade e liberdade criativa para os desenvolvedores.

Devido a essa flexibilidade e acessibilidade, o Arduino se tornou uma plataforma popular tanto para iniciantes quanto para especialistas na área de eletrônica e programação. A sua comunidade de desenvolvedores é extremamente ativa e engajada, o que proporciona um ambiente colaborativo e de aprendizado contínuo. Essa comunidade oferece suporte, compartilha projetos, tutoriais e soluções, o que permite que os usuários explorem e ampliem as possibilidades do Arduino em seus projetos.

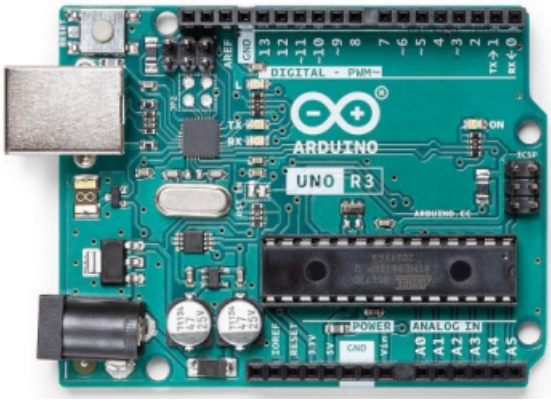
Através do compartilhamento de conhecimento e da colaboração entre os membros da comunidade, novas ideias e inovações surgem constantemente. Os desenvolvedores têm a oportunidade de aprender com os outros, obter feedback sobre seus projetos e contribuir para o avanço do ecossistema do Arduino. Essa comunidade ativa é um dos principais aspectos que fazem do Arduino uma plataforma dinâmica e em constante evolução.

Existem várias versões e modelos de placas Arduino disponíveis, cada uma com suas características específicas. A mais comum é o Arduino Uno, lançado em 2010, e é a placa que geralmente é utilizada na maioria dos projetos. No entanto, também é possível encontrar outras versões como o Arduino Due, Leonardo, Duemilanove, Mega, Nano, Micro e o Zero, por exemplo (MCROBERTS, 2015). Essa variedade de opções permite aos usuários escolher a placa que melhor se adequa às necessidades do projeto em termos de recursos, tamanho, capacidade de processamento e interfaces disponíveis. Cada modelo de placa tem suas vantagens e usos específicos, o que amplia as possibilidades de desenvolvimento e implementação de projetos com o Arduino.

4.1.1 Arduino Uno

O Arduino UNO é uma das placas mais populares e mais utilizadas da família Arduino devido à sua simplicidade e facilidade de uso. Ele é uma placa com um microcontrolador baseado no chip Atmega328P produzida pela Atmel, este possui uma arquitetura AVR de 8 bits e clock de 16 MHz. Além de dispor de 32Kb de memória flash e 2Kb de SRAM (MOTA, 2021). A Figura 3 apresenta a placa Arduino Uno.

Figura 3 – Placa Arduino Uno



Fonte: (ARDUINO, 2023b)

Ele conta com 14 pinos digitais numerados de 0 a 13. Esses pinos podem ser configurados como entradas ou saídas digitais, o que proporciona versatilidade para diversas aplicações. Além disso, 6 desses pinos também suportam a função de saída PWM (*Pulse Width Modulation*), o que permite o controle de intensidade de sinais analógicos (SOUZA, 2013). O Arduino UNO possui também 6 pinos analógicos de entrada, numerados de A0 a A5. Eles que são especialmente projetados para a leitura de valores analógicos, proveni-

entes de sensores ou potenciômetros. Apesar de serem chamados de pinos analógicos, eles também podem ser utilizados como pinos digitais. Além desses pinos digitais e analógicos, o Arduino UNO também possui pinos de alimentação, que permitem a conexão de uma fonte de energia externa. O *pino Vin* é a entrada de alimentação, onde pode ser fornecida uma tensão entre 7 e 12 volts. Além disso, o *pino 5V* fornece uma saída de 5 volts, que pode ser utilizada para alimentar componentes eletrônicos, utilizada para alimentar componentes eletrônicos conectados à placa. O *pino GND* é utilizado como referência de terra.

O Arduino UNO possui pinos dedicados para comunicação serial. O *pino 0* (RX) é utilizado para receber dados, enquanto o *pino 1* (TX) é utilizado para transmitir dados, esses pinos possibilitam uma interação eficiente com outros dispositivos. Adicionalmente, o Arduino UNO também possui uma porta USB que permite a comunicação com o computador, o que facilita o processo de programação e depuração.

E por fim, ele possui outros pinos adicionais, como o pino RESET, que é utilizado para reiniciar o microcontrolador, e os pinos de alimentação para 3.3V e GND (ARDUINO, 2023c). A seguir, será apresentada a Tabela 3, que explica sobre os pinos analógicos, e a Tabela 4, que explica sobre os pinos digitais. Ambas foram retiradas do datasheet presente no site oficial do Arduino e apresentam com mais clareza cada pinagem (ARDUINO, 2023d).

Tabela 3 – Tabela dos pinos analógicos do Arduino UNO.

Pinagem	**Função**	**Tipo**	**Descrição**
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

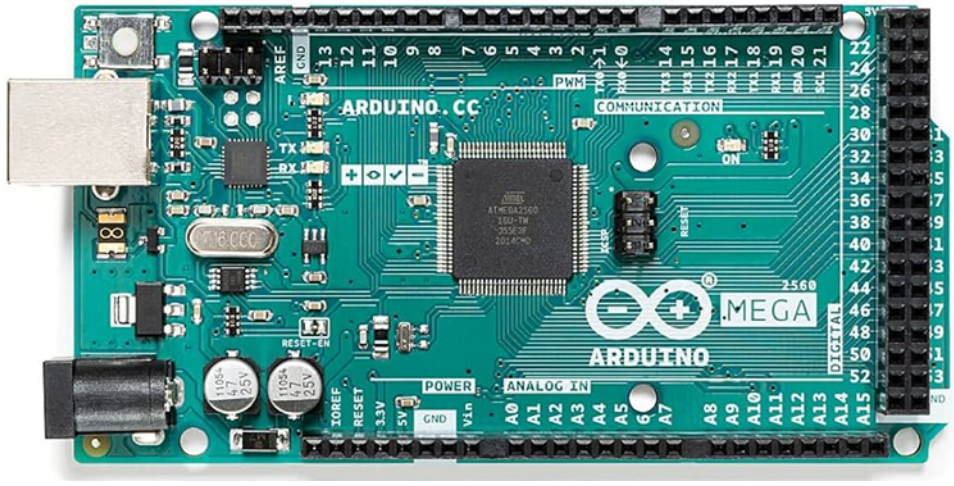
Tabela 4 – Tabela dos pinos digitais do Arduino UNO.

Pinagem	Função	Tipo	Descrição
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

4.1.2 Arduino Mega

O Arduino Mega é uma placa microcontroladora que se destaca por sua ampla capacidade de E/S (Entrada/Saída), além de oferecer mais memória que o Arduino UNO. Lançado como uma evolução natural para atender a demandas mais complexas em projetos eletrônicos, o Arduino Mega é baseado em um microcontrolador AVR ATmega2560 (SOUZA, 2014). A Figura 4 apresenta a exuberância de pinos que o Arduino Mega possui.

Figura 4 – Placa Arduino Mega



Fonte: (ARDUINO, 2023a)

Com 54 pinos digitais, numerados de 0 a 53, dos quais 15 destes podem ser utilizados como saídas PWM (*Pulse Width Modulation*) (SOUZA, 2014). Além disso, o Arduino Mega também possui 4 portas Receptoras (RX) e transmissoras (TX), usadas para realizar comunicações seriais (ARDUINO, 2023a), esses pinos digitais estão apresentados na Tabela 5. Possui também 16 pinos analógicos. Além dos pinos de alimentação, que permitem que ocorra a conexão da fonte de energia externa (ARDUINO, 2023a), os pinos digitais estão apresentados na Tabela 6. Essa riqueza de E/S permite conectar uma variedade mais extensa de dispositivos e sensores diretamente à placa, o que a torna adequada para projetos que exigem muitas conexões.

Tabela 5 – Tabela dos pinos digitais do Arduino Mega.

Pinagem	**Função**	**Tipo**	**Descrição**
1	D21/SCL	Digital Input/I2C	Digital input 21/I2C Dataline
2	D20/SDA	Digital Input/I2C	Digital input 20/I2C Dataline
3	AREF	Digital	Analog Reference Voltage
4	GND	Power	Ground
5	D13	Digital/GPIO	Digital input 13/GPIO
6	D12	Digital/GPIO	Digital input 12/GPIO
7	D11	Digital/GPIO	Digital input 11/GPIO
8	D10	Digital/GPIO	Digital input 10/GPIO
9	D9	Digital/GPIO	Digital input 9/GPIO
10	D8	Digital/GPIO	Digital input 8/GPIO
11	D7	Digital/GPIO	Digital input 7/GPIO
12	D6	Digital/GPIO	Digital input 6/GPIO
12	D5	Digital/GPIO	Digital input 5/GPIO
14	D4	Digital/GPIO	Digital input 4/GPIO
15	D3	Digital/GPIO	Digital input 3/GPIO
16	D2	Digital/GPIO	Digital input 2/GPIO
17	D1/TX0	Digital/GPIO	Digital input 1/GPIO
18	D0/Tx1	Digital/GPIO	Digital input 0/GPIO
19	D14	Digital/GPIO	Digital input 14/GPIO
20	D15	Digital/GPIO	Digital input 15/GPIO
21	D16	Digital/GPIO	Digital input 16/GPIO
22	D17	Digital/GPIO	Digital input 17/GPIO
23	D18	Digital/GPIO	Digital input 18/GPIO
24	D19	Digital/GPIO	Digital input 19/GPIO
25	D20	Digital/GPIO	Digital input 20/GPIO
26	D21	Digital/GPIO	Digital input 21/GPIO

Além da abundância de pinos, o Arduino Mega oferece uma maior capacidade de memória, com 256 KB de memória flash para o armazenamento de código e 8 KB de memória RAM para armazenamento temporário durante a execução do programa (ARDUINO, 2023a). Esses recursos adicionais são particularmente úteis em projetos que envolvem

Tabela 6 – Tabela dos pinos analógicos do Arduino Mega.

Pinagem	Função	Tipo	Descrição
1	NC	NC	Not Connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog	Analog input 0/GPIO
10	A1	Analog	Analog input 1/GPIO
11	A2	Analog	Analog input 2/GPIO
12	A3	Analog	Analog input 3/GPIO
13	A4	Analog	Analog input 4/GPIO
14	A5	Analog	Analog input 5/GPIO
15	A6	Analog	Analog input 6/GPIO
16	A7	Analog	Analog input 7/GPIO
17	A8	Analog	Analog input 8/GPIO
18	A9	Analog	Analog input 9/GPIO
19	A10	Analog	Analog input 10/GPIO
20	A11	Analog	Analog input 11/GPIO
21	A12	Analog	Analog input 12/GPIO
22	A13	Analog	Analog input 13/GPIO
23	A14	Analog	Analog input 14/GPIO
24	A15	Analog	Analog input 15/GPIO

programas mais extensos ou que requerem manipulação intensiva de dados (ALBUQUERQUE, 2023).

A placa também inclui outras características, como múltiplas portas de comunicação serial (UARTs), o que facilita a conexão com vários dispositivos simultaneamente. Além disso, possui um conector USB para programação e comunicação com o computador (ARDUINO, 2023a).

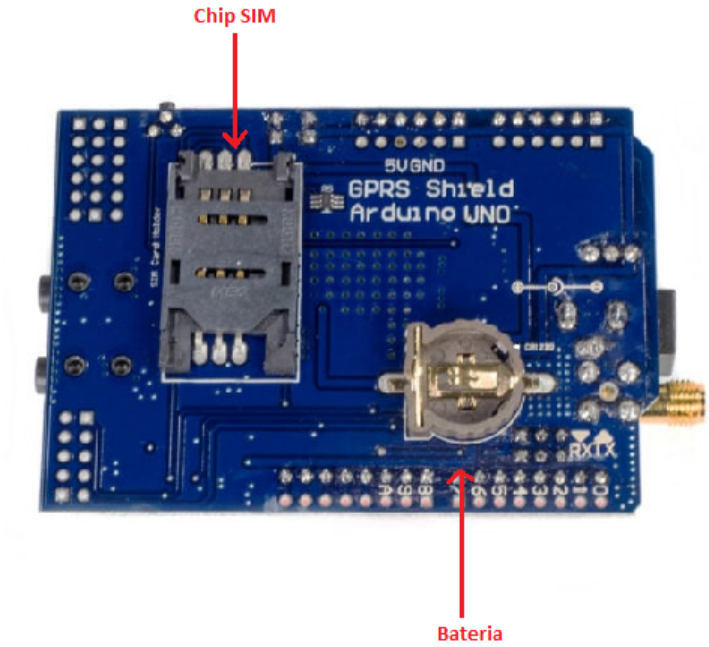
Devido à sua potência e versatilidade, o Arduino Mega é frequentemente escolhido em projetos que demandam uma quantidade substancial de I/O, processamento ou ambos. Seja para automação residencial, controle de robôs, sistemas de monitoramento, ou outras aplicações complexas, o Arduino Mega se destaca como uma opção robusta e flexível.

4.2 MÓDULO GPRS SIM900

O GPRS SIM900 é um módulo de comunicação que utiliza a tecnologia GPRS (*General Packet Radio Service*) para permitir a transmissão de dados por meio de redes celulares. Esse módulo é amplamente utilizado em projetos de IOT (*Internet Of Things*), sistemas de rastreamento veicular, monitoramento remoto e outras aplicações que exigem conectividade móvel.

O SIM900 é fabricado pela empresa SIMCom Wireless Solutions e oferece suporte a várias bandas de frequência, o que o torna compatível com redes GSM (*Global System for Mobile*) em diferentes regiões e países. Ele é baseado no chip quad-band GSM/GPRS SIM900, que permite a comunicação em quatro frequências diferentes: 850MHz, 900MHz, 1800MHz e 1900MHz (ELETROGATE, 2023a). Além disso, ele possui uma Ele possui uma interface serial que facilita a integração com microcontroladores, como o Arduino, e outros dispositivos eletrônicos (USER, 2020). A Figura 5 mostra a parte de inferior do módulo SIM900, parte essa, que se insere a bateria e o chip SIM. E a Figura 6 mostra a parte superior do módulo, aonde ficam as entradas e saídas do módulo.

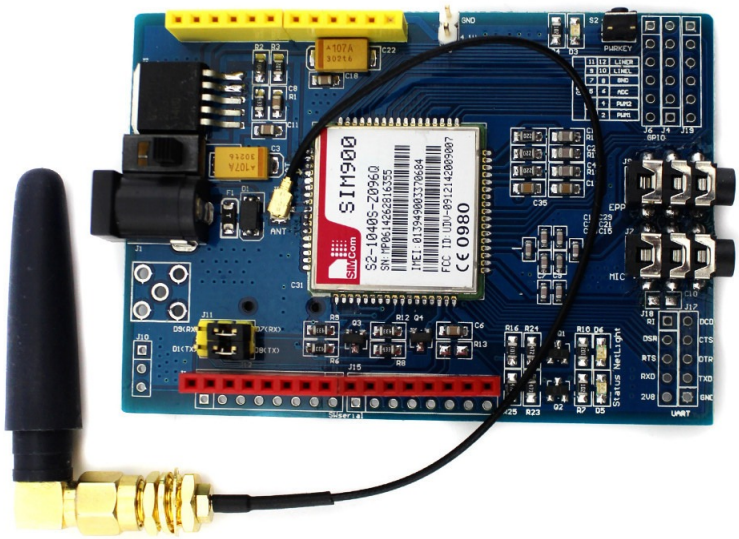
Figura 5 – Parte inferior do módulo SIM900 GSM GPRS SHIELD.



Fonte: (RANDOMNERDTUTORIAL, 2017)

Com o GPRS SIM900, é possível enviar e receber mensagens SMS, realizar chamadas telefônicas e estabelecer conexões de dados para transferência de informações pela internet. Esse módulo possui recursos avançados, como suporte a múltiplas conexões

Figura 6 – Parte superior do módulo SIM900 GSM GPRS SHIELD.



Fonte: (RANDOMNERDTUTORIAL, 2017)

TCP/UDP, autenticação de rede, criptografia e gerenciamento de energia (RANDOMNERDTUTORIAL, 2017).

A configuração e o controle do GPRS SIM900 são realizados por meio de comandos AT (Attention) enviados pela interface serial. Esses comandos permitem a configuração de parâmetros de rede, envio e recebimento de mensagens, controle do estado da conexão e outras funcionalidades.

O SIM900 também possui interfaces para conexão com cartões SIM, que são utilizados para autenticação na rede celular, e para conexão com dispositivos externos, como sensores, módulos GPS e cartões de memória. É importante mencionar que o SIM900 requer o uso de um plano de dados GPRS oferecido por uma operadora de telefonia móvel. Esse plano permite que o módulo estabeleça uma conexão de dados com a internet e transmita informações para servidores remotos (RANDOMNERDTUTORIAL, 2017).

Com sua capacidade de comunicação móvel, o GPRS SIM900 oferece uma solução eficiente e versátil para a implementação de sistemas de comunicação e monitoramento

remoto. Sua integração com plataformas como o Arduino possibilita a criação de projetos conectados e interativos, o que amplia as possibilidades de aplicações baseadas em IoT.

4.3 MÓDULO GPRS SIM800L

O GPRS SIM800L é um módulo de comunicação compacto e versátil, amplamente utilizado em aplicações de Internet das Coisas (IoT) e sistemas de monitoramento remoto que requerem conectividade móvel. Ele utiliza a tecnologia GPRS (General Packet Radio Service) para permitir a transmissão de dados por meio de redes celulares.

Com suas dimensões muito menores do que a dos GPRS apresentado anteriormente, o que o torna o componente ideal para diminuir o peso e o tamanho do protótipo, o SIM800L oferece suporte a várias bandas de frequência, o que faz com que ele se torna compatível com redes GSM/GPRS em diversas regiões do mundo. Esse módulo é amplamente valorizado por sua eficiência energética e confiabilidade na transmissão de dados, pois ele permite o envio e recebimento de informações de forma eficaz. A Figura 7 mostra o módulo SIM800l, a imagem, por mais que não esteja em escala, pode dar uma ideia do tamanho do componente.

Figura 7 – Módulo GPRS SIM800L.

Fonte: (COMPONENTS101, 2021)

Além de suportar chamadas telefônicas e o envio de mensagens SMS, o SIM800L possibilita a transferência de dados GPRS e oferece conectividade com a internet. Ele pode ser integrado a uma variedade de dispositivos e sistemas, que permite o monitoramento remoto e o controle de equipamentos por meio de uma conexão celular estável e confiável.

Para utilizá-lo de forma eficaz, é necessário fornecer a alimentação apropriada e conectar uma antena GSM externa para garantir a qualidade do sinal. O SIM800L pode

ser controlado e configurado por meio de comandos AT (Attention) enviados por uma interface serial, como a UART, o que facilita a integração com diversos dispositivos e sistemas.

É fundamental observar que o SIM800L requer um plano de dados GPRS oferecido por uma operadora de telefonia móvel para estabelecer uma conexão de dados com a internet e transmitir informações para servidores remotos. Sua capacidade de comunicação eficiente e confiável o torna uma escolha popular para uma variedade de projetos de IoT, o que proporciona conectividade móvel estável em um formato compacto e acessível.

Em síntese, o GPRS SIM800L é um módulo confiável e eficiente, especialmente adequado para aplicações de IoT e monitoramento remoto que exigem conectividade móvel em um formato compacto. Sua ampla compatibilidade e capacidade de transmissão de dados o tornam uma escolha popular para uma variedade de projetos e aplicações.

4.4 MÓDULO GPS NEO-6M

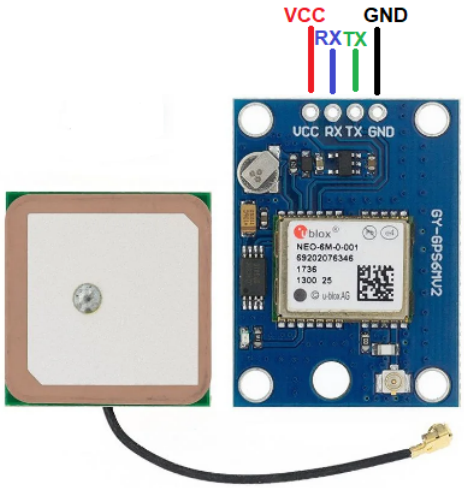
O NEO-6M é baseado no chip receptor de GPS u-blox NEO-6M, que apresenta uma alta sensibilidade e desempenho. Que utiliza a tecnologia de posicionamento global por satélite para receber sinais de satélites em órbita ao redor da Terra. Esses sinais são processados pelo módulo para calcular a posição geográfica exata do dispositivo. Ele suporta a recepção de sinais de diferentes constelações de satélite, como o GPS por exemplo. Além disso é um módulo GPS compacto e de baixo consumo de energia, projetado para fornecer informações precisas de localização e tempo. (U-BLOX, 2011).

Além das informações de localização, o GPS NEO-6M também fornece dados sobre a qualidade do sinal recebido dos satélites, como a quantidade de satélites em conecção e a intensidade do sinal. Essas informações podem ser úteis para avaliar a confiabilidade e a precisão das leituras do GPS. O NEO-6M fornece não apenas as coordenadas de latitude e longitude, mas também informações adicionais, como velocidade, direção, altitude e horário. Esses dados podem ser utilizados para rastrear e monitorar a posição de um objeto em movimento, criar sistemas de navegação, registrar rotas percorridas, entre outras aplicações (U-BLOX, 2011).

Para utilizar o GPS NEO-6M, é necessário conectar o módulo a um microcontrolador ou dispositivo que seja capaz de processar os dados recebidos. O módulo se comunica com o microcontrolador por meio de uma interface serial, que utiliza de comandos NMEA (*National Marine Electronics Association*) para transmitir os dados de posição, velocidade e tempo.

Uma das vantagens do NEO-6M é sua facilidade de uso e configuração. Ele é alimentado por uma tensão de 3.3V e possui poucos pinos de conexão, o que torna a integração com outros dispositivos relativamente simples. A Figura 8 apresenta as pinagens do NEO-6M. Além disso, o módulo possui uma antena embutida, o que elimina a necessidade de conexão de uma antena externa.

Figura 8 – Pinagem do módulo NEO-6M



Fonte: (ARDUCORE, 2023)

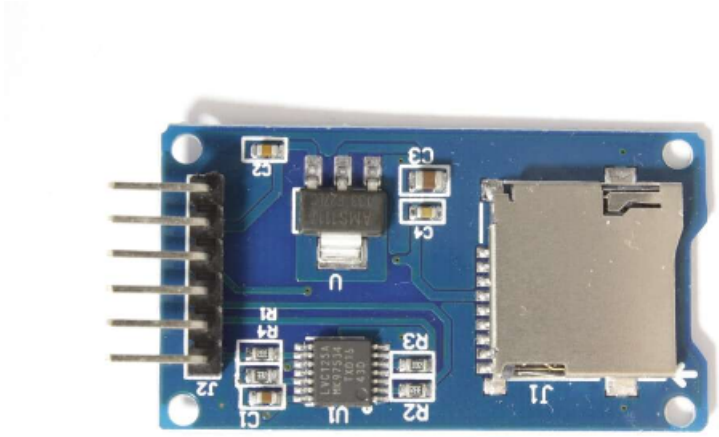
Graças à sua precisão e facilidade de uso, o GPS NEO-6M é amplamente utilizado em diversas aplicações, como rastreadores veiculares, sistemas de monitoramento de frota, dispositivos de geolocalização, drones, dispositivos de mapeamento e muitos outros projetos que exigem informações precisas de localização. Sua capacidade de receber sinais de satélites de diferentes constelações e sua eficiência energética o tornam uma opção popular para projetos de GPS.

4.5 MÓDULO MICROSD ADAPTER

O MicroSD Adapter, é um dispositivo utilizado para permitir o uso de cartões de memória MicroSD em dispositivos que não possuem um slot específico para esse tipo de cartão (OLIVEIRA, 2019). O cartão MicroSD é um formato de armazenamento de memória utilizado em muitos dispositivos eletrônicos, como smartphones, câmeras, tablets e dispositivos de áudio. Ele oferece uma forma compacta e portátil de armazenar e transferir dados, como fotos, vídeos, músicas e outros arquivos. A Figura 9 apresenta o MicroSD

Adapter.

Figura 9 – Módulo MicroSD Adapter



Fonte: (USINAINFO, 2023)

No entanto, nem todos os dispositivos possuem um slot dedicado para cartões MicroSD. Nesse contexto, surge o MicroSD Adapter. Trata-se de um pequeno dispositivo que possui um slot para inserção do cartão MicroSD em um lado e um conector compatível com outro tipo de interface no outro lado, como USB. A comunicação entre o módulo e o microcontrolador ocorre através de uma interface de comunicação SPI. (ARDUINOCIA, 2020).

4.6 MÓDULO INA219 I2C

O módulo INA219 é um dispositivo amplamente utilizado para a medição de corrente e tensão em sistemas eletrônicos. Projetado pela Texas Instruments, o INA219 é um integrado que oferece alta precisão e facilidade de uso em aplicações na faixa de 0 a 26 VDC (Volts em corrente contínua) que exigem monitoramento preciso de energia.

Um dos principais recursos do INA219 é sua capacidade de medir a corrente de forma não invasiva, para isso utiliza-se um resistor de shunt interno para criar uma queda de tensão proporcional à corrente que flui através do circuito. Isso permite a obtenção de leituras de corrente com precisão e sem a necessidade de interromper o circuito (AMARAL, 2017). Além disso, o INA219 é capaz de medir a tensão através de dois terminais, o que proporciona informações abrangentes sobre o consumo de energia. Sua interface de comunicação I2C facilita a integração com microcontroladores Arduino, ESP8266 e outros

dispositivos com interface I2C, o que permite uma leitura eficiente dos dados de corrente e tensão (ELETROGATE, 2023b).

O módulo INA219 é particularmente útil em projetos onde a eficiência energética é crucial, pois fornece informações detalhadas sobre o consumo de energia, o que possibilita realizar otimizações para maximizar a vida útil da bateria ou minimizar o desperdício de energia. Essa capacidade de monitorar precisamente o consumo elétrico faz do INA219 uma escolha popular em sistemas embarcados, projetos de IoT (Internet das Coisas) e outras aplicações eletrônicas que demandam controle e eficiência no uso de energia.

Nesse capítulo foi escrito sobre os componentes que estão presentes no projeto, e como eles funcionam. No capítulo seguinte falaremos sobre a metodologia adotada para realizar o projeto.

5 METODOLOGIA

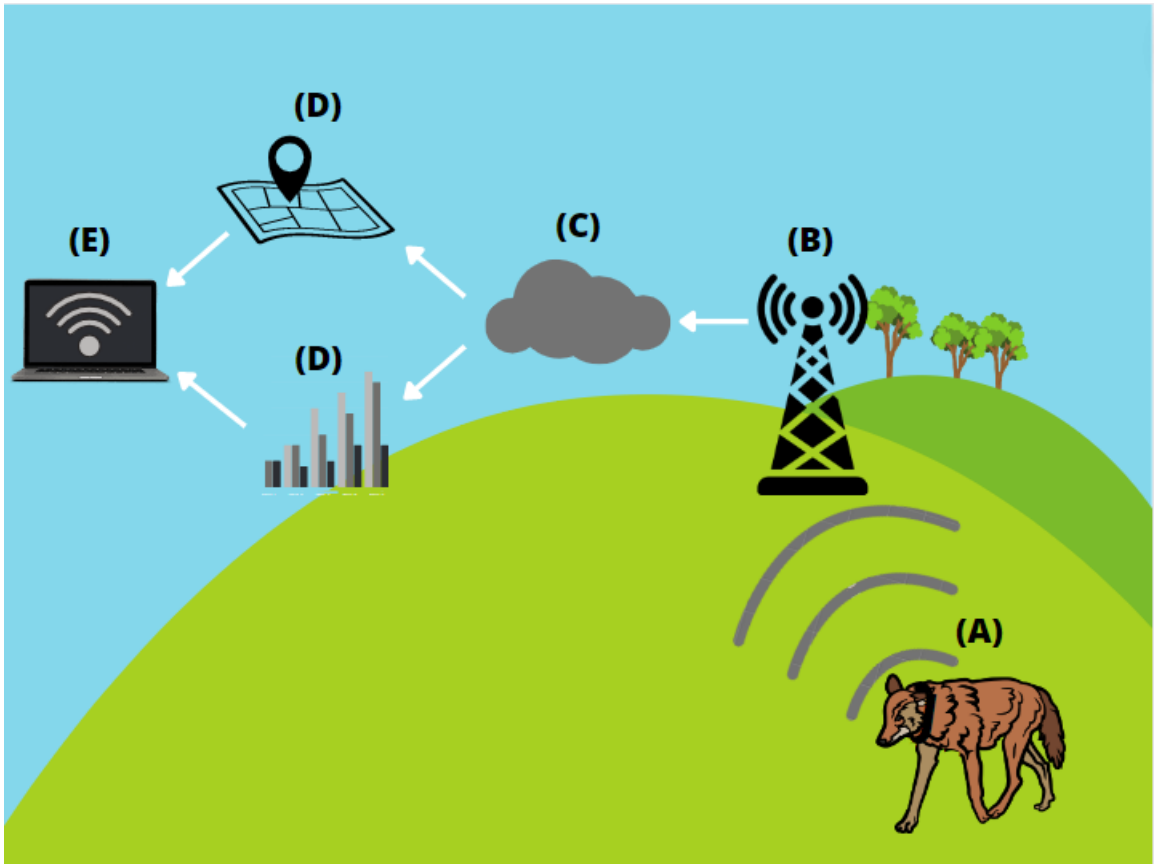
Este projeto propõe a integração sinérgica de tecnologias fundamentais, o que inclue localização via antena, radiocomunicação e geolocalização, com o objetivo de otimizar a precisão da rastreabilidade. Para a implementação desse conceito, é necessário a utilização estratégica de um módulo celular, responsável por estabelecer conexões com antenas de celular, juntamente com tecnologias de radiofrequência. A radiotelemetria possibilita a triangulação da área frequentada pelo animal monitorado. A comunicação com as antenas de celular será realizada por meio de um módulo GPRS (*General Packet Radio Service*)responsável pela transmissão dos dados obtidos pelo GPS em pacotes capazes de ampliar as taxas de transferência de redes GSM (*Global System for Mobile*).O dispositivo que utiliza essa tecnologia conta com um chip semelhante ao dos aparelhos celulares, com esse chip o responsável por receber os dados via satélite e transmiti-los para as centrais de atendimento, que utiliza de padrões digitais de comunicação de dados (RASTREAMENTO, 2021).

Para isso, esse trabalho inicialmente será utilizado o Arduino Uno. No entanto, à medida que o projeto avançar, será necessário o uso de componentes mais compactos, razão pela qual o Arduino Uno será substituído pelo Arduino Mega. O Arduino desempenhará o papel de cérebro do projeto, o que o torna o componente central do sistema de monitoramento. Por meio do Arduino, será possível implementar o código do sistema e coletar dados provenientes de outros módulos. A escolha do Arduino como plataforma baseia-se em sua facilidade de uso, flexibilidade, vasta comunidade de suporte e disponibilidade de recursos e exemplos de projetos relacionados.

O sistema deve funcionar conforme apresentado na Figura 10. O colar inicialmente é colocado no animal a ser monitorado e, após a soltura do animal em uma região com cobertura de antenas de celular. (Fig. 10a) a coleira de monitoramento do animal (Fig. 10b) envia de dados para antena de celular (Fig. 10c) que por sua vez envia para as nuvens (Fig. 10d). A partir desses dados na nuvem, são geradas representações gráficas e mapas (Fig. 10e) os dados podem ser acessados pelo aplicativo, ou por computador conectado, para estudo e monitoramento do comportamento do animal. A Figura 10 apresenta um diagrama que explica melhor como o sistema irá funcionar.

Com o propósito de conferir organização e funcionalidade ao projeto, uma abordagem metodológica foi adotada, que incluiu a divisão do desenvolvimento em etapas cruciais. O início do projeto deu-se na fase inicial de Revisão da Literatura, permeada por um diálogo construtivo com empresas e organizações parceiras. Durante esta fase, foram estudados artigos, livros, jornais e blogs pertinentes ao tema, com o intuito de criar

Figura 10 – Diagrama do Sistema. A coleira (a) estabelece conexão com a antena (b), responsável por transmitir os dados de localização do animal. Essas informações são enviadas para um serviço de armazenamento em nuvem (c). Posteriormente, a partir dos dados armazenados na nuvem, são geradas representações gráficas e mapas (d), os quais podem ser acessados tanto por um aplicativo quanto por um computador (e).



embasamentos teóricos sólidos que fundamentam e justificam o desenvolvimento do projeto.

Simultaneamente, estabeleceram-se conexões colaborativas com ONGs (Organizações Não Governamentais) como o Floreta Cheia, Órgãos Governamentais como o IBAMA (Instituto Brasileiro do Meio Ambiente) e o CETAS (Centro de Triagem de Animais Selvagens) e empresas especializadas em monitoramento e proteção animal. Esse processo permitiu uma imersão mais aprofundada nas nuances do problema em questão, o que ofereceu insights valiosos sobre as complexidades associadas ao monitoramento. Além disso, essa interação ajudou a obter um conhecimento mais aprofundado sobre o problema e as dificuldades encontradas no monitoramento, bem como adaptar-se à demanda do mercado.

Dentre as ONGs dedicadas a conservação ambiental, o *Floresta Cheia Instituto de Conservação Ambiental*, parceiro fundamental do projeto, foi fundado em 23 de outubro

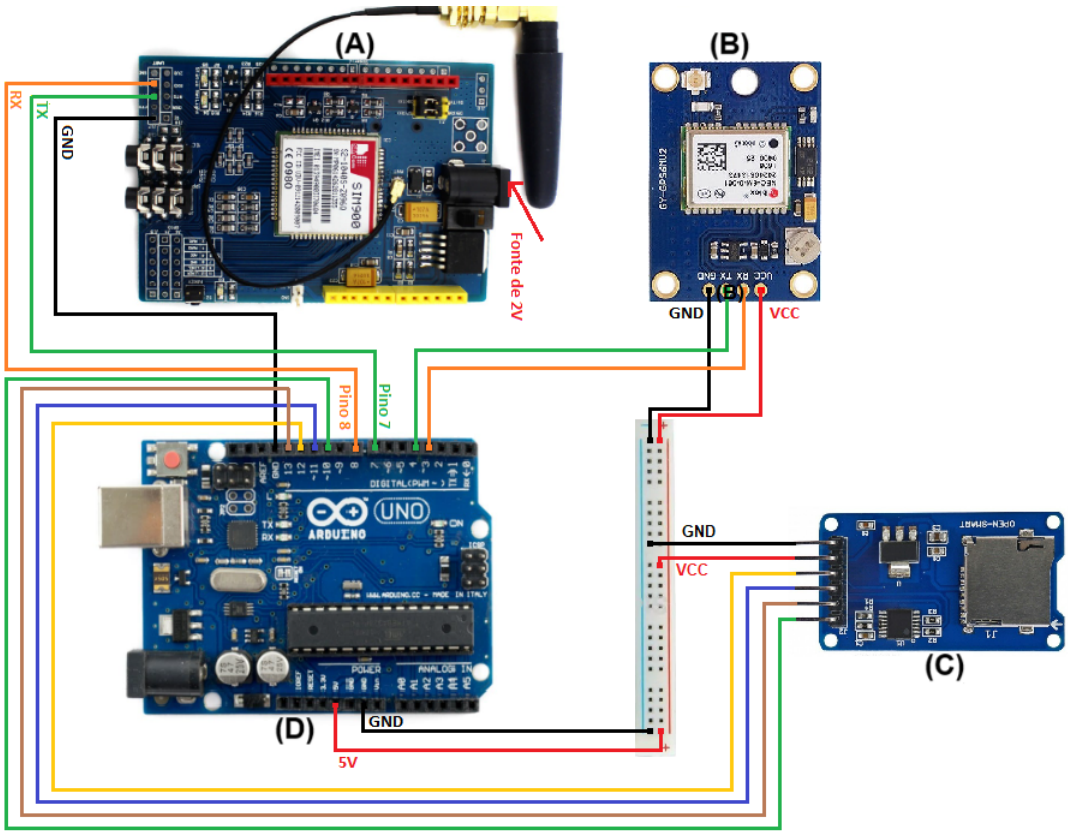
de 2019, na cidade de Goiânia, Goiás, este instituto foi concebido por biólogos engajados na conservação do meio ambiente e de todas as espécies que fazem parte dele. O Instituto tem como objetivo promover e executar ações voltadas para a conservação ambiental, com a finalidade de restaurar e proteger os serviços ecossistêmicos e ambientais. Para alcançar essa meta, o Instituto realiza um projeto de pesquisa em conjunto com o CETAS-GO (Centro de Triagem de Animais Selvagens de Goiâs), vinculado ao IBAMA, que busca auxiliar na reabilitação de animais silvestres recebidos, resgatados ou apreendidos, e depois devolvem ao seu ambiente natural, bem como avaliar o sucesso da sobrevivência dos animais soltos por esse órgão.

O projeto avançou com a elaboração do documento para registrar as informações e decisões. Nesse contexto, desenvolveu-se a Introdução como ponto inicial. Em seguida, a abordagem concentrou-se em aspectos de "precificação e prototipação". Durante essa fase, um MVP (Mínimo Produto Viável), foi definido, e foram conduzidos levantamentos de preços para os componentes do colar, com ênfase em elementos comerciais de baixo custo disponíveis no mercado nacional. Essa seleção estratégica foi crucial para identificar os componentes-chave do projeto, como o Arduino Uno, o módulo de comunicação GSM SIM900, o módulo de GPS NEO 6-M, um módulo MicroSD Card Adapter e baterias recarregáveis. É relevante observar que, mesmo que as dimensões e o peso da solução possam exceder as recomendações, tal abordagem possibilitará a condução de testes abrangentes relacionados ao desempenho da comunicação, monitoramento e exploração de técnicas de eficiência energética.

Com a seleção e definição dos componentes necessários para o projeto, procedeu-se à criação de um protótipo do sistema eletrônico por meio desses componentes. A prototipação é uma etapa crucial para validar a viabilidade técnica do projeto e realizar ajustes ou melhorias necessárias antes da implementação final. Após a conclusão do protótipo, o projeto foi montado, com todos os componentes conectados ao Arduino Uno e programados individualmente para permitir a integração do circuito. Inicialmente, os módulos GPS e GPRS foram integrados, o que permitiu que o sistema se conectasse à rede 2G e recebesse informações de latitude e longitude do satélite, as quais foram encaminhadas por meio de SMS para o celular. Nessa fase, foi novamente documentado o progresso do projeto e as informações obtidas.

A Figura 11 apresenta um diagrama do circuito de monitoramento a ser montado e testado, com os componentes utilizados: (Fig. 11a) Módulo SIM 900 GSM; (Fig. 11b) e Módulo GPS NEO 6-M; (Fig. 11c) módulo MicroSD Card Adapter (Fig. 11d) conectados ao controlador Arduino.

Figura 11 – Diagrama do hardware. Composta por a placa celular (a), Módulo GPS (b), Módulo MicroSD Card Adapter (c) e Arduino Uno (d)



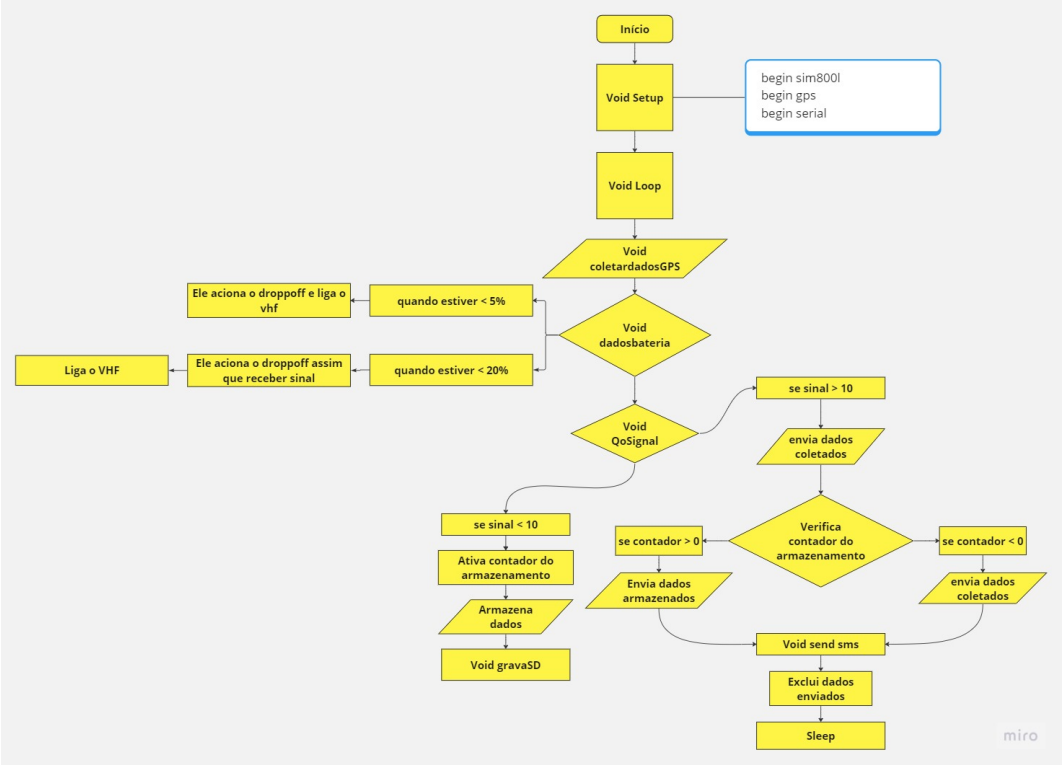
Após a integração inicial, o projeto foi submetido à avaliação da comissão examinadora. O objetivo foi relatar as atividades realizadas até o momento e destacar o potencial inerente ao projeto. A apresentação foi seguida pela revisão minuciosa da documentação, o que possibilita de realizar possíveis alterações e recomendações da comissão avaliadora. Isso foi crucial para orientar o desenvolvimento contínuo do projeto e sua transformação em realidade.

Dessa maneira, será iniciado o desenvolvimento do colar, e o primeiro protótipo passará por testes em animais domésticos e simulações conduzidas pelos parceiros durante atividades de campo. Posteriormente, os resultados provenientes do teste do primeiro protótipo serão meticulosamente documentados. A análise dessas informações permitirá a identificação de eventuais falhas no sistema, o que possibilita seu aprimoramento. Assim, o código inicial será refinado para ajustar o sistema, o que irá corrigir quaisquer deficiências apontadas pelo pessoal do Instituto Floresta Cheia ou pela banca avaliadora.

O funcionamento do protótipo será da seguinte maneira: utiliza-se o GPRS, a coleira tentará se comunicar com as antenas celulares. Caso seja estabelecida a comunicação,

o sistema obterá as informações de latitude e longitude do GPS e as enviará por SMS e para a nuvem. No entanto, caso a coleira não consiga se comunicar com a antena, as informações obtidas pelo GPS serão armazenadas em um microSD. Esses dados permanecerão salvos até que a coleira consiga estabelecer conexão com a rede celular. Após a comunicação com a antena ser restabelecida, o sistema irá obter novamente as informações de GPS e enviá-las por SMS e para a nuvem. Além disso a coleira precisa ter um sistema de "drop off", que faça com que a coleira antes de descarregar por completo, solte do animal, para que ela possa ser recuperada. O diagrama da Figura 12 mostra como o sistema deve funcionar no final do projeto.

Figura 12 – Fluxograma da lógica do sistema



Após a integração dos módulos, resta apenas a implementação de uma fonte de energia externa com durabilidade suficiente para assegurar a eficácia da coleira em campo, para conferir ao protótipo competitividade em relação às demais soluções comerciais. Nesse estágio, será cuidadosamente selecionada a bateria mais apropriada para atender às demandas do projeto.

Posteriormente, o objetivo é aprimorar a eficiência do sistema por meio da redução das dimensões, para isso vai-se optar por variantes mais compactas dos módulos e, quando possível, os consolidar em uma única placa capaz de desempenhar todas as funções. Essa

otimização resultará em um sistema mais atrativo, o que proporciona maior conforto aos animais, especialmente aos de menor porte.

Nesse capítulo foi abordado a metodologia utilizada para a organização e a execução do projeto, no capítulo posterior, começará a abordar o desenvolvimento de cada etapa do projeto.

6 DESENVOLVIMENTO

A primeira parte deste capítulo aborda detalhadamente os testes realizados nos componentes. Esta etapa é dedicada a explorar e analisar minuciosamente o funcionamento de cada peça individualmente. Serão apresentados os procedimentos adotados, as ferramentas utilizadas e os resultados obtidos durante os testes de cada componente. Essa fase desempenha um papel crucial na detecção de possíveis falhas ou componentes defeituosos. Isso possibilita a sua substituição, o que assegura um desempenho adequado do projeto em sua totalidade. Nesse estágio, optou-se por utilizar o microcontrolador Arduino UNO como a plataforma de teste, devido à sua programação acessível e eficiência na execução dos testes. Após a conclusão dos testes em cada componente, será realizado um teste que integrará todos os elementos, para que assim se viabilize, a execução do projeto final.

6.1 TESTE DO MÓDULO GPRS

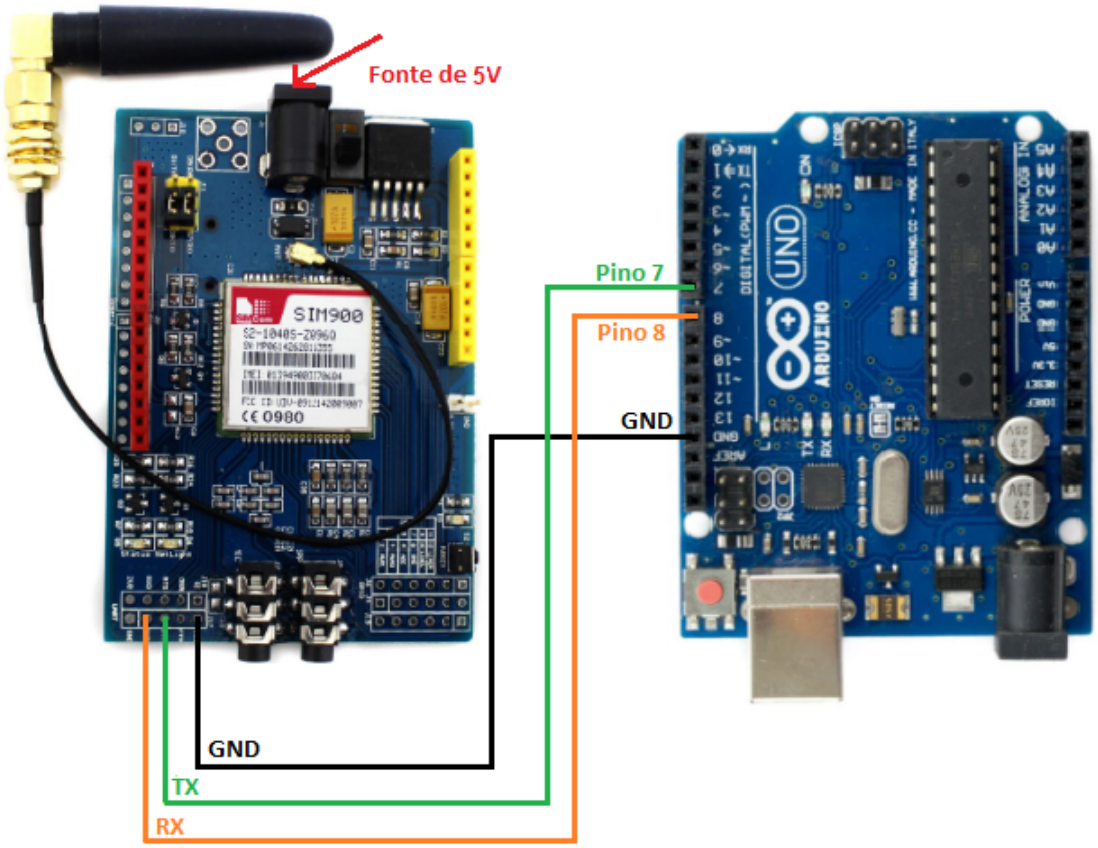
O primeiro teste realizado foi do módulo SIM900 GSM GPRS SHIELD, responsável pela comunicação entre a estação rádio-base mais próxima e a coleira. Para que essa comunicação seja estabelecida, é necessário que haja um chip tanto no SIM900 como no aparelho celular. No módulo, o chip é instalado na parte inferior, como mostra a Figura 5 que foi apresentado na fundamentação teórica.

Uma vez que os chips foram instalados, o hardware do sistema foi montado. O SHIELD GPRS SIM900 foi conectado junto com um Arduino UNO, no qual o TX do SIM900 é conectado ao RX do Arduino e vice-versa. Além disso o módulo estará conectado ao GROUND do Arduino para que possa receber 5 V(Volts), e também a uma fonte externa de 5 V para alimentar o SIM900. O desenho do esquemático desse projeto está demostrado na Figura 13.

Com a montagem do sistema eletrônico, procedeu-se à elaboração do código. A base do código foi obtida no site *Random Nerd Tutorials*, no qual foi feito um software que tem como objetivo que o SHIELD GPRS SIM900 envie um SMS (*Short Message Service*) para o número especificado, após o módulo entrar em contato com uma rede GPRS (RANDOMNERDTUTORIAL, 2017). Para que esse código funcione foi preciso incluir a biblioteca **<SoftwareSerial.h>** alem de especificar as portas que serão usadas, o número de celular ao qual o módulo deve enviar a mensagem que também foi especificado. Percebe-se que e um código bem pequeno, e de fácil entendimento.

O código começa por meio da inclusão da biblioteca **<SoftwareSerial.h>**. Essa biblioteca desempenha um papel crucial ao criar uma emulação de porta serial nos pinos

Figura 13 – Esquemático do módulo SIM900 GSM GPRS. Na esquerda tem o módulo SIM900 conectado em uma fonte externa e na direita um Arduino UNO.



7 e 8. O pino 7 é configurado como RX (Receptor), enquanto o pino 8 é definido como TX (Transmissor). Após a declaração das portas, observamos que o código possui três funções distintas. A primeira função é o *void setup()*, na qual o BAUDRATE do SIM900 é definido como 19200. Além disso, será chamado a função sendSMS.

Outra função presente no código é a *void loop()*. No entanto, uma vez que o objetivo do módulo é enviar o SMS apenas uma vez, essa função está vazia, sem nenhum conteúdo. Após o envio do SMS, o código não necessita de um loop contínuo de execução, pois a tarefa principal já foi executada.

A função *void sendSMS()* é responsável para que o módulo envie o SMS, no início dessa função foi usado o comando *AT+CMGF=1* ˚, que serve para colocar o módulo no modo de texto, para que o mesmo mande o SMS. Logo após o módulo entrar no modo texto, e usado o comando *AT+CMGS* que serve para enviar o SMS para o número especificado. Por fim, e escrito a string a ser enviada para o número do celular via SMS, além de definir um limite máximo de caracteres que essa string pode ter.

Dessa forma, o projeto deu início à sua fase inicial de codificação, o que marca o desenvolvimento do primeiro conjunto de instruções. Este marco inaugurou uma série de testes dos componentes, cujo código integral está registrado abaixo

```
1    #include <SoftwareSerial.h>
2    SoftwareSerial SIM900(7, 8);
3
4    void setup() {
5        SIM900.begin(19200);
6        delay(10000);
7        sendSMS();
8    }
9
10   void loop() {
11    }
12
13   void sendSMS() {
14       SIM900.print("AT+CMGF=1\r");
15       delay(100);
16
17
18       SIM900.println("AT+CMGS=\"+5562999856225\"");
19       delay(100);
20
21       SIM900.println("Bom Dia");
22       delay(100);
23
24       SIM900.println((char)26);
25       delay(100);
26       SIM900.println();
27       delay(10000);
28   }
29
```

Listing 6.1 – Código para testar o GPRS

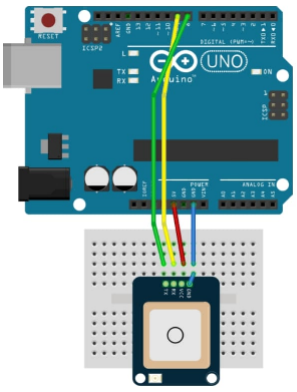
Após escrever o código no Arduino, é necessário aguardar a chegada do SMS. Nos testes realizados, verificou-se que a mensagem pode levar um pouco mais de um minuto para ser recebida no destino. No entanto, após investigações adicionais, constatou-se que o principal motivo para essa demora era o processamento do notebook no qual o projeto foi desenvolvido. Em outras palavras, o desempenho do notebook afetava o tempo necessário para que o SMS fosse entregue ao celular. A Figura 14 ilustra como o número especificado no código irá receber esse SMS.

Figura 14 – Mensagem SMS recebida.

6.2 TESTE DO MÓDULO GPS

Após testado o GPRS, foi testado o módulo GPS NEO-6M. Assim como o módulo SIM 900, a montagem do hardware do módulo GPS não é trabalhosa, requer alimentar o GPS com 5V (Volts), e conectar o RX do módulo ao TX do GPS, e vice-versa. E com o GPS que será realizada a comunicação com os satélites e verificada a posição geográfica que o módulo se encontra, através do terminal da IDE Arduino. A Figura 15 foi retirada do site *Vida de Silício* e apresenta como fica a montagem do teste.

Figura 15 – Esquemático do módulo GPS NEO-6M



Fonte: Vida de Silício

Montado o *hardware*, bastava programar o *firmware* que faria os testes, para isso inicialmente foi escrito um código para testar se os componentes estão à funcionar corretamente e se eles se conectam com os satélites, para isso o código imprime no terminal dados que utiliza do protocolo de NMEA. O Quadro 1 apresenta o que cada uma dessas

sentenças significa.

Quadro 1 – Quadro com os valores de NMEA.

Sentença NMEA	**Significado**
GPGGA	Dados globais da correção do sistema de posicionamento (tempo, posição, dados do tipo de reparo).
GPGLL	Posição geográfica, latitude, longitude
GPVTG	Informações de curso e velocidade relativas ao solo
GPRMC	Hora, data, posição, curso e velocidade
GPGSA	Modo de operação do receptor GPS, satélites usados na solução de posição e valores DOU
GPGSV	O número de satélites GPS em "View satellite ID numbers", elevação, azimute e valores SNR.
GPMSS	Relação sinal-ruído, intensidade do sinal, frequência e taxa de bits de um receptor de radiofarol.
GPTRF	Dados de correção de trânsito.
GPSTN	ID de dados múltiplos.
GPXTE	Erro transversal, medido.
GPZDA	Data e hora (mensagem de tempo PPS, sincronizada com PPS)
150	Enviar mensagem.

No entanto, essa etapa apresentou maiores desafios do que o inicialmente esperado. O módulo NEO-6M possui um LED que deveria piscar durante a comunicação, mas, após compilar o código, percebi que o LED permanecia apagado. Durante vários dias, presumi que o problema residia no código ou, talvez, nas condições climáticas adversas durante os testes realizados no mês de dezembro. Mesmo assim, continuei a modificar o código, na esperança de resolver a questão. No entanto, o GPS persistia em tentar se conectar ao satélite e apresentava apenas algumas informações no Terminal do Arduino, enquanto o LED permanecia sem piscar. Foi somente na virada do ano que comecei a suspeitar que o problema não estava relacionado ao código, mas sim ao próprio módulo NEO-6M.

Diante dessa constatação, optou-se por adquirir um novo módulo GPS com o propósito de conduzir uma série de ensaios. Para minha surpresa, esse recente componente logrou êxito em estabelecer uma comunicação eficaz com os satélites, após apresentar todas as informações previstas no Terminal, além de acionar o LED conforme as expectativas. O código empregado nesse teste inaugural foi derivado, com algumas modificações, da referência disponibilizada no site *Mundo Projetado*, com algumas alterações (GUIMARÃES, 2021).

No início desse código foi usado novamente a biblioteca **<SoftwareSerial.h>**,

nesse início também são definidos os valores dos pinod que serão usados como RX e TX, e o valor do Baudrate.

A função *void setup()* é responsável por realizar as configurações iniciais do programa antes de começar o loop principal. Dentro desta função, temos duas chamadas de função. A primeira linha **Serial.begin(115200)** inicia a comunicação serial padrão do Arduino com uma taxa de transmissão de 115200 baud. A segunda linha inicia a comunicação serial do objeto *gpsSerial*, que está conectado ao módulo GPS, com a taxa de transmissão de 9600, que é definida pela constante GPS_Serial_Baud.

E a função *void loop()* é a função responsável para que o módulo faça a saída dos dados para o terminal, a cada segundo.

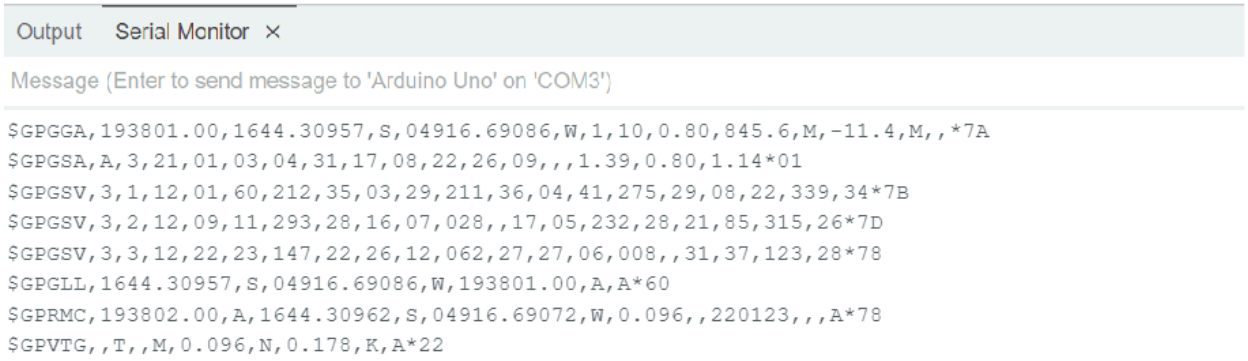
```
1     #define GPS_RX 4
2     #define GPS_TX 3
3     #define GPS_Serial_Baud 9600
4     #include <SoftwareSerial.h>
5     SoftwareSerial gpsSerial(GPS_RX, GPS_TX);
6
7     void setup(){
8        Serial.begin(115200);
9        gpsSerial.begin(GPS_Serial_Baud);
10 }
11    void loop(){
12      while (gpsSerial.available() > 0){
13        Serial.write(gpsSerial.read());
14   }
15 }
16
```

Listing 6.2 – Código do módulo GPS, sem tratamento dos dados

Após realizar a compilação do código no Arduino, observa-se o terminal da IDE do Arduino, onde a resposta será apresentada. É válido ressaltar que a resposta obtida estará sujeita a variações, as quais dependerão da quantidade de satélites detectados pelo GPS. A Figura 16 ilustra a resposta obtida por meio do GPS.

É evidente, com base no Quadro 1, que as sentenças de NMEA fornecem todas as informações necessárias para a leitura do GPS, o que é extremamente benéfico para a compreensão e manutenção do sistema. No entanto, no projeto final, não será necessário ter acesso a tantas informações. Portanto, a fim de evitar confusões durante a leitura, foi incluída a biblioteca **<TinyGPS.h>**, que se encarrega de processar esses dados.

Figura 16 – Resposta do GPS enquanto utiliza as sentenças NMEA. Os dados sem tratamento são de difícil leitura, logo e necessário que esses dados sejam tratados.



Para isso, foi desenvolvido um código que contém apenas as informações relevantes para o operador do sistema. O objetivo desse código é rastrear o módulo GPS NEO-6M e, após estabelecer a comunicação entre o módulo e o satélite, as informações de latitude e longitude, juntamente com um link da internet, serão exibidas no Terminal a cada segundo. Ao clicar no link, o usuário será redirecionado para o Google Maps, o que mostra a localização exibida no Terminal. Esse recurso simplifica ainda mais o rastreamento do módulo. Esse software foi inspirado no código usado no site *Vida de Silício*, com algumas alterações para que esse código se adapte melhor ao nosso sistema.

Ao se analisar o código usado para tratar os dados do GPS, percebe-se que a primeira coisa feita foi incluir as bibliotecas que seram necessarias para chamar as funções usadas no código. Logo depois de incluir as bibliotecas foi criado um objeto **SerialGPS(4,3)** que serve para informar as portas que serão usadas para fazerem a comunicação serial entre o GPS e o Arduino, onde a porta 4 é o RX (recepitor)e a porta 3 e o TX (Transmissor).

Logo após foram definidas as variáveis responsáveis por armazenar os dados obtidos no decorrer do código. As variáveis *lat, lon e vel* armazenaram as informações da latitude, longitude e Velocidade. As variáveis *data e hora* são responsáveis em armazenar a data e a hora ao qual estão à ser realizados os testes. E a variável *sat* será responsavel por armazenar a quantidade de satélites disponíveis.

Na função *void setup()* é informado o BAUDRATE tanto da porta serial, quanto para a comunicação no GPS. Para esse projeto o recomendado é que o valor sejá de 9600. A última linha dessa função terá como única responsabilidade imprimir no monitor a frase *Buscando satelites....*

Será na função *void loop()* que todos os dados do satélite seram armazenados no

GPS e tratados, o primeiro comando dessa função tem como objetivo identificar se existe algum dado disponível do GPS para o Arduino. Se houver, o código dentro do if() será executado.

É nesse bloco if() que os dados do satélite seram recebidos e informados. A primeira informação que a chave vai usar e os dados de data e hora. Como as duas variáveis foram declarados no início do código como *unsigned long* isso significa que as informações da data estarão no formato *ddmmaa* e o horário no formato *hhmmss*, então para tornar a leitura desses dados mais compreensível, foi necessário fazer algumas operações matemáticas nas variaveis **data** e **hora**. São feitas as seguintes divisões:

(hora / 1000000) - 3: Essa operação divide o valor da variável hora por 1000000 e subtrai 3. Isso é feito para ajustar a diferença de fuso horário, supondo que o fuso horário atual é 3 horas a menos em relação ao valor original de hora.

(hora % 1000000) / 10000: Essa operação pega o resto da divisão de hora por 1000000 e, em seguida, divide o resultado por 10000. Isso extrai a informação dos minutos do horário.

(hora % 10000) / 100: Essa operação pega o resto da divisão de hora por 10000 e, em seguida, divide o resultado por 100. Isso extrai a informação dos segundos do horário.

Depois das divisões, a data terá o formato data/mês/ano e o horário terá o formato horas/minutos/segundos.

As próximas informações obtidas dentro do bloco **if()** são a latitude e longitude, que representam a localização do sistema. Para obter essas informações, utiliza-se o comando **GPS.f_get_position(&lat, &lon)**, que armazena os valores da latitude na variável *lat* e os valores da longitude na variável *lon*. Para exibir esses valores no terminal, utiliza-se o comando **Serial.println(lat, 6)** e o comando **Serial.println(lon, 6)**. O número 6 especifica que deseja-se exibir até 6 casas decimais para a latitude e para a longitude. Isso garante uma melhor precisão na representação desses valores no monitor serial.

Depois, a próxima informação será a velocidade do sistema em relação ao satélite ao qual o módulo está à se comunicar. Para isso, utiliza-se o comando **GPS.f_speed_kmph()**, que armazena o valor da velocidade dentro da variável *vel* em quilometros por hora.

Em seguida, temos a parte responsável por gerar um link no Google Maps com as informações de latitude e longitude, o que possibilita uma visualização mais detalhada da

localização atual do sistema. Para isso, basta repetir a estrutura utilizada para exibir as informações de latitude e longitude, com a adição de apenas um comando para imprimir o URL necessário para gerar o link. Dessa forma, o usuário poderá acessar o link e ver a localização no Google Maps.

A última informação obtida será a dos satélites. Esses dados não serão exibidos no terminal, mas são interessantes de serem armazenados, pois ao longo do projeto pode ser necessário utilizá-los. Para obter essa informação, basta utilizar o comando **GPS.satellites()**, que irá armazenar a quantidade de satélites encontrados na variável *sat*. Em seguida, é escrita outra condição cujo objetivo é verificar se o valor de satélites armazenado é válido. Se essa condição for verdadeira, o valor será exibido no Monitor. A seguir, é apresentado o código utilizado nesta segunda etapa do teste (MOUNTAINBAJA, 2018).

```
1     #include<SoftwareSerial.h>
2     #include<TinyGPS.h>
3     SoftwareSerial SerialGPS(4, 3);
4     TinyGPS GPS;
5     float lat, lon, vel;
6     unsigned long data, hora;
7     unsigned short sat;
8
9     void setup() {
10        SerialGPS.begin(9600);
11        Serial.begin(9600);
12        Serial.println("Buscando satelites...");
13 }
14
15    void loop() {
16        while (SerialGPS.available()) {
17            if (GPS.encode(SerialGPS.read())) {
18                GPS.get_datetime(&data, &hora);
19                Serial.print("--");
20                Serial.print((hora / 1000000) - 3);
21                Serial.print(":");
22                Serial.print((hora % 1000000) / 10000);
23                Serial.print(":");
24                Serial.print((hora % 10000) / 100);
25                Serial.print(" -- ");
26                Serial.print(data / 10000);
27                Serial.print("/");
28                Serial.print((data % 10000) / 100);
29                Serial.print("/");
30                Serial.print(data % 100);
```

```
31                  Serial.println("--");
32                  GPS.f_get_position(&lat, &lon);
33                  Serial.print("Latitude: ");
34                  Serial.println(lat, 6);
35                  Serial.print("Longitude: ");
36                  Serial.println(lon, 6);
37                  vel = GPS.f_speed_kmph();
38                  Serial.print("Velocidade: ");
39                  Serial.println(vel);
40                  Serial.print("http://maps.google.com/maps?q=");
41                  Serial.print(lat, 6);
42                  Serial.print(",");
43                  Serial.print(lon, 6);
44                  Serial.println("");
45                  sat = GPS.satellites();
46                  if (sat != TinyGPS::GPS_INVALID_SATELLITES) {
47                  Serial.print("Satelites: ");
48                  Serial.println(sat);
49              }
50                  Serial.println("");
51          }
52      }
53  }
```

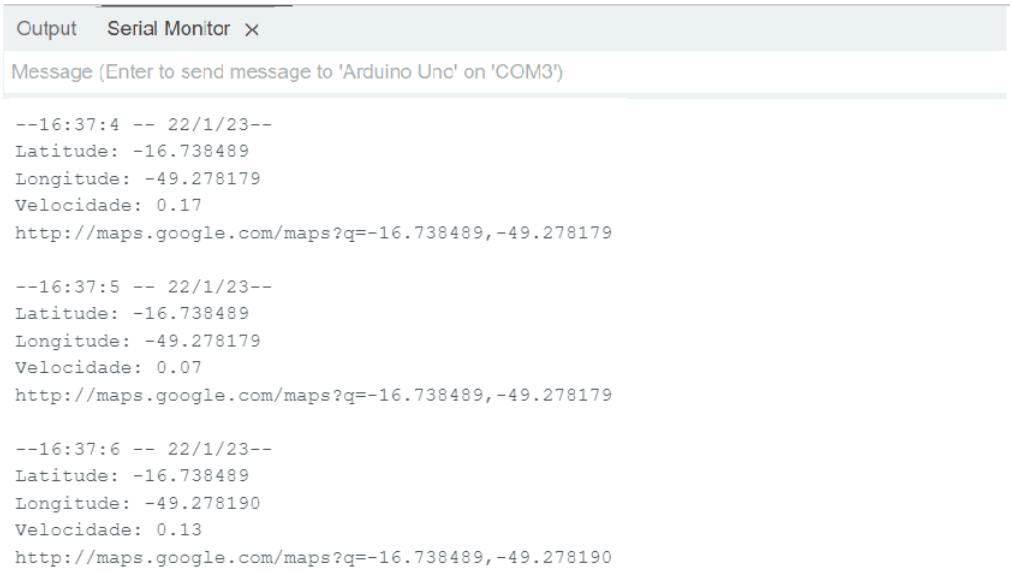
Listing 6.3 – Código do GPS, com os dados tratados

Finalmente, após compilar esse código no Arduino, basta verificar a resposta no terminal. Nota-se que a cada segundo o terminal exibe a informação obtida no rastreamento, além de disponibilizar um link para que o usuário possa ter uma informação visual de onde o módulo se encontra naquele momento, o que facilita ainda mais a leitura desses dados. A Figura 17 mostra os dados que apareceram no terminal.

6.3 INTEGRAÇÃO DOS MÓDULOS

Após desenvolver os códigos dos módulos, foi possivel entender melhor o funcionamento do GPS e do GPRS, o que ajuda na hora de integrar todos componentes em um único sistema, o que possibilita a criação do primeiro protótipo do projeto. O código começa por incluir as bibliotecas **<SoftwareSerial.h>** para realizar a comunicação das portas do arduino com os módulos. e **<TinyGPS.h>**, que serão utilizadas para a comunicação com o módulo GPS e a interpretação dos dados recebidos. Em seguida, são criados objetos que utiliza das classes *SoftwareSerial* que seram respónsavel para comunicar com o GPS e SIM900, e um objeto da classe *TinyGPS* para o que faz o processamento dos dados do GPS. Logo após foi declarado as variáveis lat, lon, vel, data, hora e sat, com essas variáveis responsaveis para armazenar a latitude, longitude, velocidade, data, hora e

Figura 17 – Dados tratados do GPS



quantidades de satélites disponíveis.

Na *função setup()*, é declarada a BAUDRATE tanto para os módulos GPRS SIM900 e GPS NEO-6M, quanto do monitor serial do Arduino . O que e responsável pela inicialização das comunicações *Serial* e *serial GPS*, além de exibir uma mensagem que indica que o código está em busca de conectar com os satélites.

Na *função loop()*, o código verifica se há dados disponíveis no módulo GPS ao utilizar a instrução **SerialGPS.available()**. Se houver, o código chama a instrução**GPS.encode()** para obter os dados e tratá-los, afim de exibir as informações de data, hora, latitude, longitude, velocidade e número de satélites disponiveis com a formatação adequada.

A função *sendSMS()* é responsável por configurar o módulo SIM900 para enviar mensagens SMS e envia as informações ao utilizar o comando **AT+CMGF=1**. Em seguida, os dados do SMS são preparados, com as informações de latitude e longitude coletadas pelo GPS, juntamente com um link para o Google Maps que direciona para a localização exata do dispositivo. E, por fim, para evitar o envio excessivo de mensagens, um atraso de 5 segundos é adicionado ao final da função sendSMS(). Esse intervalo de tempo garante que o envio de mensagens seja controlado e evita a sobrecarga do sistema de comunicação. A seguir, e apresentado o código utilizado nesta etapa do teste.

```
1 #include<SoftwareSerial.h>
2 #include<TinyGPS.h>
3
```

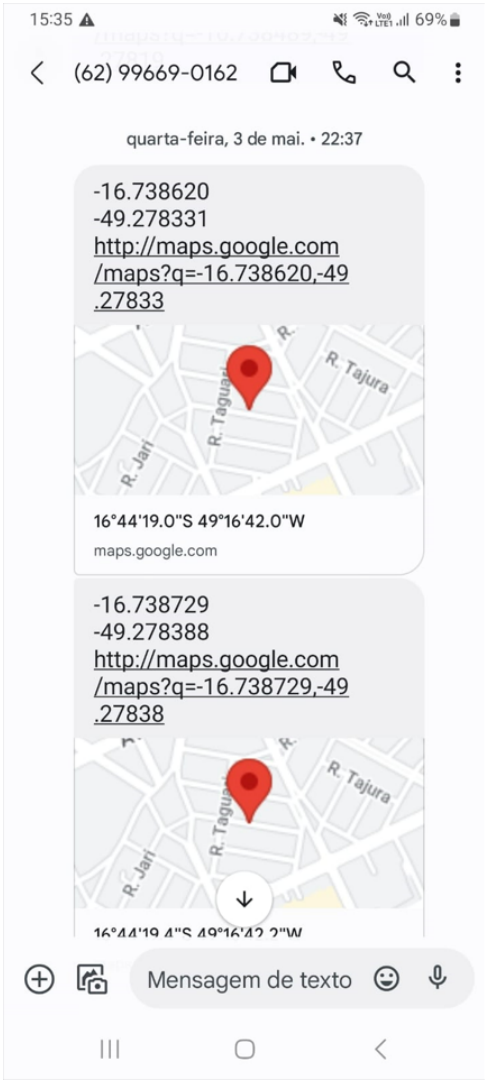
```
4 SoftwareSerial SerialGPS(4, 3);
5 SoftwareSerial SIM900(7, 8);
6
7 TinyGPS GPS;
8
9 float lat, lon, vel;
10 unsigned long data, hora;
11 unsigned short sat;
12
13 void setup() {
14   SIM900.begin(19200);
15   SerialGPS.begin(9600);
16   Serial.begin(9600);
17
18   Serial.println("Buscando satelites...");
19 }
20
21 void loop() {
22
23   while (SerialGPS.available()) {
24     if (GPS.encode(SerialGPS.read())) {
25
26       //Hora e data
27       GPS.get_datetime(&data, &hora);
28
29       Serial.print("--");
30       Serial.print((hora / 1000000) - 3);
31       Serial.print(":");
32       Serial.print((hora % 1000000) / 10000);
33       Serial.print(":");
34       Serial.print((hora % 10000) / 100);
35       Serial.print(" -- ");
36       Serial.print(data / 10000);
37       Serial.print("/");
38       Serial.print((data % 10000) / 100);
39       Serial.print("/");
40       Serial.print(data % 100);
41       Serial.println("--");
42
43       //latitude e longitude
44       GPS.f_get_position(&lat, &lon);
45
46       Serial.print("Latitude: ");
47       Serial.println(lat, 6);
48       Serial.print("Longitude: ");
49       Serial.println(lon, 6);
50
```

```
51        //velocidade
52        vel = GPS.f_speed_kmph();
53
54        Serial.print("Velocidade: ");
55        Serial.println(vel);
56        Serial.print("http://maps.google.com/maps?q=");
57        Serial.print(lat, 6);
58        Serial.print(",");
59        Serial.print(lon, 6);
60        Serial.println("");
61        delay(2000);
62        sendSMS();
63
64        //Satelites
65        sat = GPS.satellites();
66
67        if (sat != TinyGPS::GPS_INVALID_SATELLITES) {
68          Serial.print("Satelites: ");
69          Serial.println(sat);
70        }
71      }
72    }
73 }
74
75 void sendSMS() {
76    SIM900.print("AT+CMGF=1\r");
77    delay(1000);
78    SIM900.println("AT+CMGS=\"+5562999856225\"");
79    delay(1000);
80    SIM900.println(lat,6);
81    SIM900.println(lon,6);
82    SIM900.print("http://maps.google.com/maps?q=");
83
84    SIM900.print(lat, 6);
85    SIM900.print(",");
86    SIM900.print(lon, 6);
87    delay(1000);
88
89    SIM900.println((char)26);
90    delay(1000);
91    SIM900.println();
92    delay(5000);
93 }
```

Listing 6.4 – Código dos módulos integrados

Finalmente, depois de escrever o código, foi observado o envio do SMS. A Figura 18 ilustra como esse SMS é exibido no celular.

Figura 18 – SMS do sistema integrado. O SMS também recebe um link do maps que é clicável.

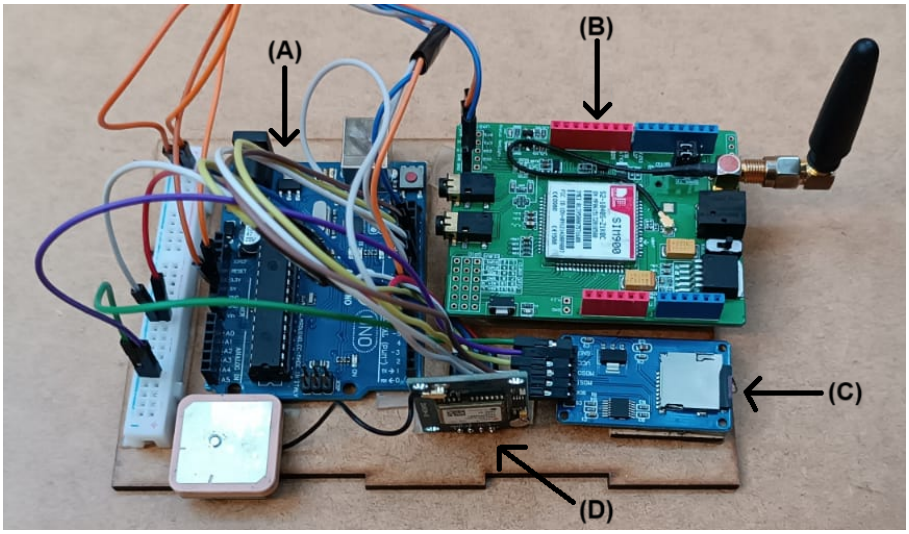


6.4 APRIMORAMENTO DO PROTÓTIPO INICIAL

A primeira versão do protótipo da coleira está pronta e operacional. No entanto, durante o desenvolvimento, percebeu-se que o circuito eletrônico funcionava perfeitamente devido aos testes realizados em áreas urbanas com acesso abundante a redes celulares. No entanto, nos locais de soltura de animais e em muitos dos lugares onde nossos parceiros atuam, a conectividade celular é mais frágil ou praticamente inexistente. O que acarretou em um novo desafio: como a coleira deveria se comportar quando não houvesse sinal de rede disponível? Para resolver essa questão, foram dedicados vários dias em busca de uma solução adequada. Foi decidido que o código precisaria verificar se a coleira está conectada

às antenas de celular. Caso não haja conexão, as informações serão salvas no cartão de memória Flash, MicroSD. A Figura 19 apresenta o circuito com todos os componentes necessários para que esse ajuste funcione, o que oferece uma abordagem adaptada para situações sem conectividade.

Figura 19 – O circuito montado com todos os componentes. Composta por a Arduino Uno (a), Módulo GPRS SIM900A (b), Módulo MicroSD Card Adapter (c) e Módulo GPS NEO-6M (d)



Além disso, também foi percebido que o sistema estava muito grande, o que traria limitações para a coleira. Durante as discussões com os representantes do CETAS e do Floresta Cheia, ficou estabelecido que a coleira deveria pesar aproximadamente 3% do peso do animal a ser monitorado. Ao considerar que a coleira atual pesa cerca de 280 gramas, ela seria adequada apenas para animais de grande porte. Para solucionar esse problema, foi proposta a criação de uma coleira de tamanho reduzido, que substituiria muitos dos componentes atuais por versões menores.

Dessa forma, foi planejada a substituição do Arduino Uno pelo Arduino Mega, o que exigiria algumas adaptações no software, porém nada de difícil implementação. Além disso, seria feita a troca do módulo SIM900 pelo SIM7020e. Essa substituição representaria uma mudança drástica, uma vez que o novo componente utiliza o pacote NB-IOT em vez do GPRS utilizado pelo SIM900. Vale ressaltar que o SIM7020e é um componente recente no mercado e, devido a isso, não foram encontrados tutoriais em vídeo disponíveis para utilizá-lo. Portanto, essa transição representa um grande desafio para o projeto.

6.5 CÓDIGO PARA VERIFICAR O SINAL DO GPRS

Depois de definida essa adaptação ao projeto o código inicial deve ser aprimorado para algo que atenda a demanda atual. O primeiro passo e criar um código que será responsável por verificar se o GPRS consegue comunicar com a rede, além de mostrar a qualidade desse sinal.

Assim como todos os outros códigos, ele se inicia com a declaração da biblioteca **<SoftwareSerial.h>** que é usada para realizar transformar uma porta analogica do arduino em uma porta serial. Então foi declarado um objeto **SoftwareSerial SIM900(7, 8)** para estabelecer os pinos 07 e 08 do Arduino como os pinos de comunicação RX e TX, respectivamente. Já Na função *void setup()*, é declarada a taxa de transmissão do monitor serial e do GPRS, o que define a velocidade de comunicação entre o serial e o SIM900.

Na função *void loop()* Vai ser verificado a cada 10 segundos a força do sinal. Caso o sinal tenha uma força maior que 10, significa que e possível realizar a comunicação entre o módulo e a antena. Agora caso a força do sinal seja menor do que 10, significa que a comunicação não está boa, a função exibirá “sem sinal” e aguardará 5 segundos antes de continuar.

Por fim, e a função *void checkSignal()* que verifica a força do sinal de rede do SIM900. Ela envia o comando **AT+CSQ** para o módulo e aguarda uma resposta. Então essa resposta é analisada e o valor da força do sinal é extraído, para então retornar um valor entre 0 e 31 (valores recomendados pelo próprio componente). Se o valor não estiver nessa faixa numérica, a função retorna -1"para indicar um valor inválido. O código que foi escrito nessa etapa do desenvolvimento está mostrado de forma completa a seguir.

```
1 #include <SoftwareSerial.h>
2 SoftwareSerial SIM900(7, 8);
3
4 void setup(){
5   Serial.begin(9600);
6   SIM900.begin(19200);
7   delay(2000); // Aguarda 2 segundos para o SIM900 inicializar
8   int signalStrength = checkSignal();
9   Serial.print("Forca do sinal: ");
10  Serial.println(signalStrength);
11 }
12
13 void loop(){
14  // Verificar sinal de rede a cada 10 segundos
15  delay(10000);
16  int signalStrength = checkSignal();
```

```
17   Serial.print("Forca do sinal: ");
18   Serial.println(signalStrength);
19   while(checkSignal() >= 10){
20     Serial.println("sinal ok");
21   }
22   if (!checkSignal()){
23     Serial.println("sem sinal");
24     delay(5000);
25     }
26 }
27 int checkSignal(){
28   SIM900.println("AT+CSQ");
29   delay(1000);
30   String response = "";
31   while (SIM900.available()){
32     response += char(SIM900.read());
33 }
34   if (response.indexOf("+CSQ: ") >= 0){
35     int signalValue = response.substring(response.indexOf("+CSQ: ") + 6,
       response.indexOf("+CSQ: ") + 8).toInt();
36     if (signalValue >= 0 && signalValue <= 31){
37       return signalValue;
38       }
39   }
40   return -1; // Valor de sinal inv lido
41 }
```

Listing 6.5 – Código para verificação de sinal

Após a implementação do código, foi possível verificar que o mesmo responde conforme o esperado. A Figura 20 exibe o resultado no terminal quando o módulo GPRS estabelece comunicação com a rede celular, enquanto a Figura 21 apresenta a saída correspondente quando a comunicação falha.

Figura 20 – O GPRS ao se comunicar com a antena celular.

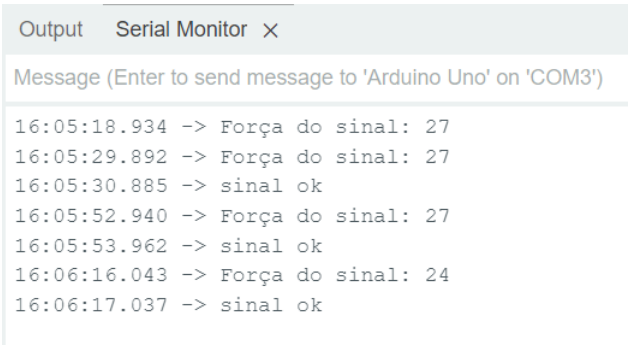
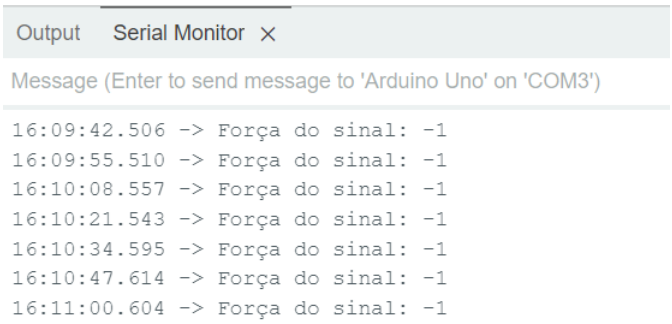


Figura 21 – O GPRS tendo falha na comunicação com a antena celular.



6.6 PROXIMOS PASSOS DO DESENVOLVIMENTO

Embora o progresso do projeto tenha sido significativo, uma série de tarefas ainda aguarda conclusão. Dentre essas pendências, destaca-se a integração do MicroSD Adapter no âmbito do projeto. Este procedimento implicará no desenvolvimento de um código com a finalidade de efetuar a leitura do cartão SD. O objetivo primordial é viabilizar a leitura do cartão sem a necessidade de retirá-lo do sistema, ou seja, o próprio Arduino será capacitado para realizar a leitura e apresentar as respostas no hiperterminal. Esse avanço propiciará a interpretação dos dados sem requerer a conexão com um computador por meio de cabos. Subsequentemente à implementação deste código de leitura, um novo conjunto de códigos será elaborado, para permitir o armazenamento das informações provenientes do GPS no cartão SD. Isso facultará a preservação das informações adquiridas durante os períodos em que a coleira estiver sem sinal de rede.

Após a redação dos códigos para o MicroSD, os componentes serão conectados às baterias para a realização de testes, o que visa determinar a durabilidade destas em alimentar a coleira. Os resultados obtidos destes testes proporcionarão insights sobre a adequação das baterias atuais ou a necessidade de investir em baterias de maior qualidade.

Na fase subsequente, procedeu-se à integração de todos os componentes no sistema, que orquestra a comunicação entre os módulos e assegura que executem suas funções de forma coordenada, conforme delineado no projeto.

Finalizado, a etapa derradeira consistirá na redução dos componentes, nessa parte contempla-se a substituição dos módulos em uso por alternativas de dimensões reduzidas. Este procedimento visa conferir ao sistema maior compacticidade e eficiência econômica.

6.7 A EVOLUÇÃO DA COLEIRA

Após a conclusão da fase de verificação de sinal, procedeu-se à etapa de integração do sistema com o SD. Inicialmente, iniciou-se o desenvolvimento do código para essa finalidade; contudo, enfrentou-se desafios significativos na implementação. Durante várias semanas, o foco do projeto esteve direcionado à resolução dessa etapa, porém, o código, por razões não identificadas, não se comportava conforme o esperado. Diante dessa problemática, a estratégia adotada consistiu em realizar uma revisão dos componentes, a fim de identificar possíveis falhas em algum deles. Foi constatado que a biblioteca SoftwareSerial, por razões específicas, deixou de funcionar, após apresentar uma limitação ao projeto, pois a simulação de portas seriais tornou-se impraticável. Assim, optou-se por utilizar as portas seriais RX e TX nativas do microcontrolador, o que torna esse um ajuste necessário, embora tenha gerado um novo desafio, uma vez que tanto o módulo GPRS quanto o módulo GPS requerem portas seriais RX e TX para a comunicação, e a MCU do Arduino UNO (ATmega328P) dispõe de apenas uma porta serial nativa.

Diante da necessidade de múltiplas portas lógicas, a transição para o Arduino MEGA tornou-se imperativa, pois este microcontrolador possui quatro portas seriais e uma capacidade de memória superior, o que possibilita a execução do código completo sem a demanda de dois microcontroladores no mesmo projeto.

Adicionalmente, algumas alterações foram nos componentes foram efetuadas, com o objetivo de reduzir o tamanho e peso do protótipo, o que confere à coleira dimensões mínimas. Como foi o caso do módulo SIM900, que foi substituído pelo SIM800L, uma versão mais compacta e versátil, o que mantem as funcionalidades necessárias.

Essas modificações conferiram à coleira uma forma mais compacta e atraente em comparação à sua versão inicial, além de proporcionar um aumento significativo na capacidade de memória e poder de processamento, o que confere maior eficácia e desempenho.

6.8 CÓDIGO DOS MÓDULOS INTEGRADOS AO SD

Feito os testes, integrado os módulos, e adicionado o SD finalmente se tornou possível desenvolver o código final do protótipo. O qual será responsável por fazer o sistema funcionar conforme o previsto nos diagramas.

A função *void setup()* o código começa com a inicialização da comunicação serial, os módulos GPS e SIM800L, e o cartão SD. Ele define a taxa de transmissão de dados para os módulos GPS e SIM800L e verifica se o cartão SD funciona corretamente.

Na função *void loop()* começa com a declaração de três variáveis de string, latitude, longitude e hora, são inicializadas com o valor "0". Em seguida, a função verifica se há dados disponíveis na porta serial do módulo GPS, ao utilizar um **loop while**.

O código verifica se o módulo GPS obteve uma atualização de localização e se a localização é válida. Se sim, ele extrai os valores da latitude e longitude com precisão de 6 casas decimais e também obtém a hora do sinal do GPS. O valor da hora é ajustado para o horário local do Brasil (UTC-3).

Com os dados de latitude, longitude e hora disponíveis, o código monta uma mensagem que contem esses dados, formatados como uma URL do Google Maps. Em seguida, essa mensagem é gravada no arquivo **test.txt** no cartão SD.

O código envia o comando **AT+CSQ** para o módulo SIM900, para verificar a intensidade do sinal da rede celular. Ele lê a resposta do módulo SIM900 e verifica se a qualidade do sinal é forte o suficiente **(>= 10)**. Se for, ele lê o conteúdo do arquivo **test.txt** e envia as mensagens contidas para o número de telefone especificado.

Após o envio bem-sucedido de todas as mensagens, o código move as entradas enviadas para um arquivo temporário, exclui o arquivo original e cria um novo arquivo com as entradas não enviadas. O código repete esse processo indefinidamente.

No *void sendSMS()* temos a função responsável por enviar o comando **AT+CMGF** que define o modo de mensagem para o formato de texto. E envia essa mensagem via SMS para o número de telefone especificado.

```
1  # include <SPI.h>
2  # include <SD.h>
3  # include <TinyGPS++.h>
4
5  File myFile;
6  TinyGPSPlus gps;
7
8  void setup() {
9    Serial.begin(9600);
10   Serial1.begin(9600);
11   Serial2.begin(19200);
12   delay(1000);
13
14   Serial.print("Inicializando o cart o SD...");
15   if (!SD.begin(4)) {
```

```
16      Serial.println("Falha na inicializa  o!");
17      while (1);
18    }
19    Serial.println("Inicializa  o conclu da.");
20  }
21
22  void loop() {
23    String latitude = "0";
24    String longitude = "0";
25
26    while (Serial1.available() > 0){
27      gps.encode(Serial1.read());
28    }
29
30    if (gps.location.isUpdated()){
31      if (gps.location.isValid()){
32        latitude = String(gps.location.lat(), 6);
33        longitude = String(gps.location.lng(), 6);
34      }
35
36      // Montar a mensagem com os dados de latitude e longitude
37      String mensagem = "http://maps.google.com/maps?q=" + latitude + ","
     + longitude;
38
39      // Gravar os dados no SD
40      myFile = SD.open("test.txt", FILE_WRITE);
41      if (myFile) {
42        myFile.println(mensagem);
43        myFile.close();
44      } else {
45        Serial.println("Erro ao abrir test.txt");
46      }
47
48      Serial2.println("AT+CSQ"); // Envia o comando AT+CSQ para o m dulo
     SIM900
49      delay(1000);
50
51      while (Serial2.available()) {
52        String resposta = Serial2.readString(); // L   a resposta do
     m dulo SIM900 como uma string
53        int inicio = resposta.indexOf(": "); // Encontra a posi   o do
     caractere ": " na resposta
54        int fim = resposta.indexOf(","); // Encontra a posi   o do
     caractere "," na resposta
55        if (inicio != -1 && fim != -1) { // Se ambos os caracteres foram
     encontrados
56          String qualidade = resposta.substring(inicio + 2, fim); //
```

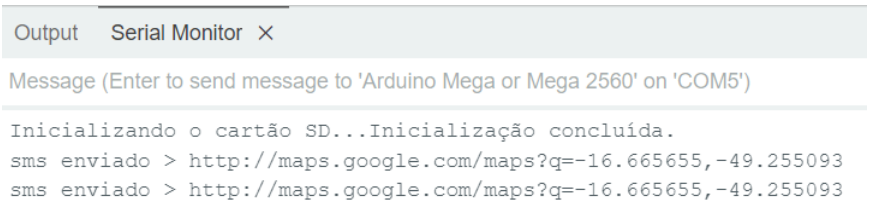
```
    Extrai a substring entre os caracteres
57        int valor = qualidade.toInt(); // Converte a substring em um
    valor inteiro
58
59        if (valor >= 10) { // Se o valor for maior ou igual a 10 enviar
    dados pelo sms
60          File readFile = SD.open("test.txt");
61          while(readFile.available()) {
62            String line = readFile.readStringUntil('\n');
63            sendSMS(line);
64            delay(5000); // Aguardar 30 segundos antes de enviar outro
    SMS
65
66            // Ap s o envio bem-sucedido, mover as entradas n o
    enviadas para um arquivo tempor rio
67            File tempFile = SD.open("temp.txt", FILE_WRITE);
68            if (tempFile) {
69              tempFile.println(line);
70              tempFile.close();
71            } else {
72              Serial.println("Erro ao abrir temp.txt");
73            }
74          }
75          readFile.close();
76
77          // Ap s o envio de todas as entradas, excluir o arquivo
    original e criar um novo arquivo com as entradas n o enviadas
78          SD.remove("test.txt");
79
80          File oldFile = SD.open("temp.txt");
81          File newFile = SD.open("test.txt", FILE_WRITE);
82
83          while(oldFile.available()) {
84            newFile.write(oldFile.read());
85          }
86
87          oldFile.close();
88          newFile.close();
89
90          SD.remove("temp.txt");
91        }
92      }
93    }
94  }
95 }
96
97 void sendSMS(String message) {
```

```
98    Serial.print("sms enviado > ");
99    Serial.println(message);
100   Serial2.println("AT");
101   delay(1000);
102   Serial2.println("AT+CMGF=1");
103   delay(100);
104   Serial2.print("AT+CMGS=\"+5562999856225\"\r");
105   delay(100);
106
107   Serial2.print(message);
108   delay(100);
109
110   Serial2.write(26);
111 }
```

Listing 6.6 – Código dos módulos integrados com o SD

Com a implementação do código foi possível perceber que o projeto foi um sucesso, consequentemente quando a coleira tem sinal, ela envia as informações da latitude e longitude via SMS para o celular. A Figura 22 mostra a resposta obtida no Monitor Serial.

Figura 22 – Resposta do Monitor Serial do código integrado



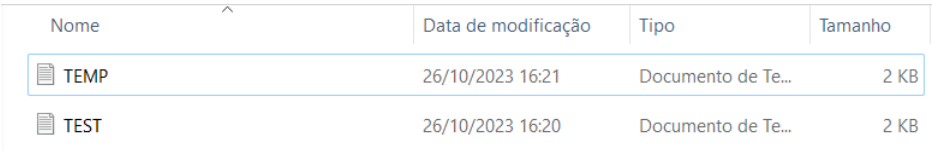
Caso a coleira não consiga se comunicar com a rede 2G ela vai gravar esses dados no SD. Para isso vai ser criado 2 pastas, uma delas é a foi nomeada de "TEMP"e a outra é de "TEST". As duas iram gravar os valores obtidos quando o sistema não consegue se comunicar com as antenas celulares, porem elas tem uma diferença entre si.

- Pasta TEST: grava os dados obtidos enquanto não fosse possivel ter comunicação, e assim que a comunicação seja recuperada, ele envia esses dados via SMS e deleta as informações da pasta. O motivo de ela deletar esse valor e para que essa pasta fique apenas com os valores que foram obtidos na ultima vez que a coleira estava sem sinal, o que evita que no proximo SMS ele não envie os valores que foram enviados anteriormente.

- Pasta TEMP: grava todos os dados obtidos enquanto não ocorreu a comunicação, e não os deleta em momento algum, o que permite que eventualmente esses dados possam ser usados como backup.

A Figura 23 demonstra esses pastas criadas quando a coleira não conseguia se comunicar.

Figura 23 – Pastas criadas para gravar os dados quando a coleira não consegue se comunicar com a rede celular



As figuras 24 e 25 mostra como os valores ficam escrito dentro das pastas, note que as pastas não possuem nenhuma diferença inicial, uma vez que elas so vão se diferenciar de fato depois de um tempo, uma vez que a pasta TEST vai ficar vazia.

Figura 24 – Dados gravados na Pasta TEST

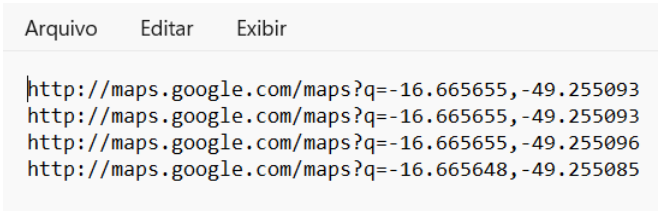


Figura 25 – Dados gravados na Pasta TEMP

6.9 ANÁLISE DO CONSUMO DE CORRENTE E SELEÇÃO DA BATERIA

Depois de escrito o código, pode-se ir para o componente central do sistema de monitoramento, que é o seu sistema de alimentação por bateria, uma vez que a coleira não será alimentada por fonte externa, já que ela vai operar de forma contínua em ambientes remotos.

Mas para que fosse possível saber qual a bateria que devería ser usada no projeto, primeiro era necessário saber o consumo de energia do sistema. Para isso inicialmente o sistema foi conectado a uma fonte cujo ela transmitia 9V para o microcontrolador. Com essa tensão foi mostrado pela fonte que o consumo de energia da coleira era de aproximadamente 30 mA. Um consumo razoável e que durante muito tempo foi esse valor que foi usado como verdadeiro.

Porem antes de definir de fato a bateria que seria usada na prototipação, foi feito uma revisão dos módulos que estavam presente na coleira, e com essa revisão se percebeu que a maioria dos componentes tinhas capacitores embarcados. Então o valor gerado pela fonte poderia ter uma pequena variação, uma vez que o valor de corrente da saída (apresentado pela fonte) e o valor de corrente do sistema interno tinha uma variação. Então para descobrir o consumo real da coleira, o circuito teve uma adição do INA219, que é um sensor de corrente. Além disso foi necessário desenvolver um software que lia os valores da corrente a cada 10 ms (Milissegundos) e plotava um gráfico com a média dessas variações.

Basicamente esse código usa as bibliotecas **EEPROM** para ler e escrever valores na memória EEPROM, **Wire** para comunicação I2C e **Adafruit_INA219** para se comunicar com o sensor INA219.

Logo após isso o código vai fazer a iniciação do INA219, aonde um objeto **ina219_B** da classe **Adafruit_INA219** é criado com o endereço I2C 0x040. Nas linhas abaixo ocorre a inicialização da capacidade da bateria, além de definir a capacidade restante da bateria.

Na função *void setup()* A comunicação serial é iniciada, com uma taxa de baud de 115200 bits por segundo. Logo após isso a estrutura de repetição **while (!Serial) delay(1);** aguarda até que a comunicação serial esteja pronta. Isso é útil para garantir que a comunicação esteja estabelecida antes de continuar com o restante do código. Logo após a linha **if (!ina219_B.begin())** verifica se a inicialização do sensor INA219 foi bem-sucedida. Se não for, imprime uma mensagem de erro e entra em um loop infinito.

Após isso ocorre a leitura da capacidade restante da bateria armazenada na posição de memória 0 da EEPROM e atribui esse valor à variável **batteryRemainingCapacity_mAh**, e por fim a função imprime uma mensagem informativa no Monitor Serial que indica que o programa conseguia medir a tensão e corrente com o sensor INA219.

Já na função *void loop()* ocorre as medições de tensão, corrente e potência. no barramento. Além disso o **float hoursLeft** é declarado como a capacidade restante da bateria dividida pela corrente atual. Isso representa uma estimativa do tempo de operação

restante em horas. Logo após isso a capacidade da bateria e atualizado, o que subtrai a corrente multiplicada pelo tempo assumido para a execução de um ciclo (9 segundos) do total da capacidade da bateria. E finalmente a função que realiza o loop Interno para realizar medições e imprimir amostras de corrente. O loop interno é executado enquanto i for menor que 50, e durante cada iteração do loop, há cálculos e leituras adicionais do sensor INA219. O código usado para plotar essa média será apresentado abaixo:

```
# include <EEPROM.h>
# include <Wire.h>
# include <Adafruit_INA219.h>

Adafruit_INA219 ina219_B(0x040);

// Capacidade total da bateria em mAh
float batteryFullCapacity_mAh = 9800;
// Capacidade restante da bateria em mAh (inicialmente cheia)
float batteryRemainingCapacity_mAh = batteryFullCapacity_mAh;

void setup(void){
  Serial.begin(115200);
  while (!Serial) {
      delay(1);
  }

  if (! ina219_B.begin()) {
    Serial.println("FALHA AO TENTAR LOCALIZAR O CHIP INA219");
    while (1) { delay(10); }
  }

  // Ler a capacidade restante da bateria da EEPROM
  EEPROM.get(0, batteryRemainingCapacity_mAh);

  Serial.println("MEDINDO TENS O E CORRENTE COM O INA219 ...");
}

void loop(void){
  float shuntvoltage = 0;
  float busvoltage = 0;
  float current_mA = 0;
  float loadvoltage = 0;
  float power_mW = 0;

  shuntvoltage = ina219_B.getShuntVoltage_mV();
  busvoltage = ina219_B.getBusVoltage_V();
  current_mA = ina219_B.getCurrent_mA();
  power_mW = ina219_B.getPower_mW();
```

```
40   loadvoltage = busvoltage + (shuntvoltage / 1000);
41
42   // C lculo da dura  o estimada da bateria
43   float hoursLeft = batteryRemainingCapacity_mAh / current_mA;
44
45   // Atualizar a capacidade restante da bateria
46   batteryRemainingCapacity_mAh -= current_mA * (9 / 3600.0); // supondo
      que o loop demore cerca de 2 segundos
47
48    // Armazenar a capacidade restante da bateria na EEPROM
49    EEPROM.put(0, batteryRemainingCapacity_mAh);
50
51    // C lculo da porcentagem de bateria restante
52    float batteryPercentageLeft = (batteryRemainingCapacity_mAh /
      batteryFullCapacity_mAh) * 100;
53
54     int i=0;
55     while(i<50){
56       float soma_amostras, amostra = 0;
57       int quant_amostras = 5;
58
59       for(int aux=0;aux<quant_amostras;aux++){
60           soma_amostras += ina219_B.getCurrent_mA();
61           delay(1);
62       }
63       amostra = soma_amostras/quant_amostras;
64       Serial.println(amostra);
65       soma_amostras = 0;
66       i++;
67     }
68     delay(100000);
69   }
```

Listing 6.7 – Código para descobrir o consumo de corrente do sistema

Depois de escrito o código basta verificar o gráfico que será plotado no hiperterminal para que seja possivel dar continuidade na escolha da bateria. E assim, as figuras abaixo mostraram a variação da corrente no sistema a cada 10 ms(Milissegundos) em um intervalo de 2,45 segundos. A Figura 26 representa o menor valor e a Figura 27 o maior valor encontrado.

6.9.1 Seleção da Bateria

A escolha da bateria representou uma fase crucial no projeto, dada a importância de garantir uma fonte de energia sustentada e a necessidade inerente de mobilidade. A análise dos gráficos de consumo de corrente proporcionou uma previsão mais precisa das

Figura 26 – Variação média da corrente (Valor mais baixo)

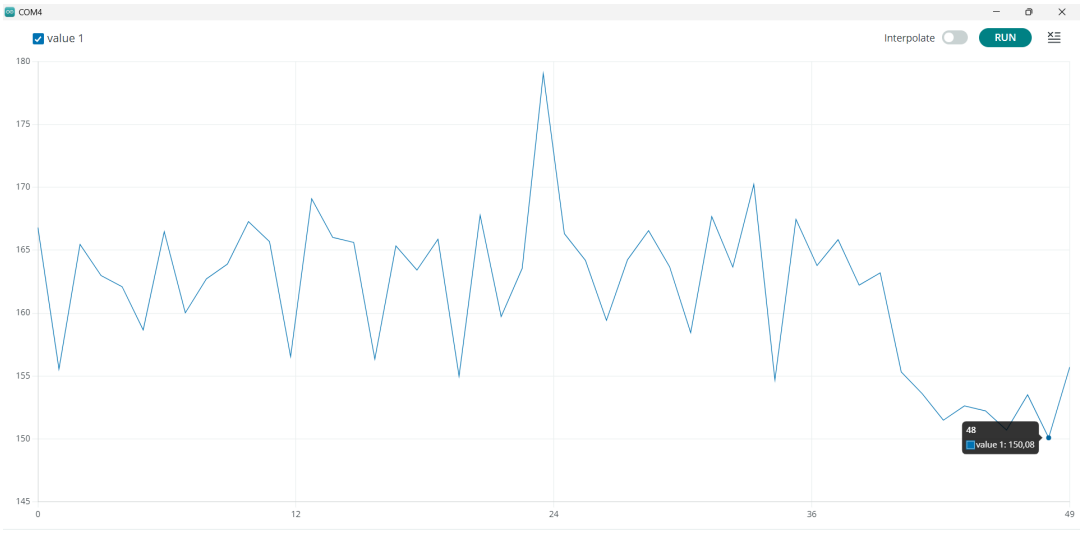
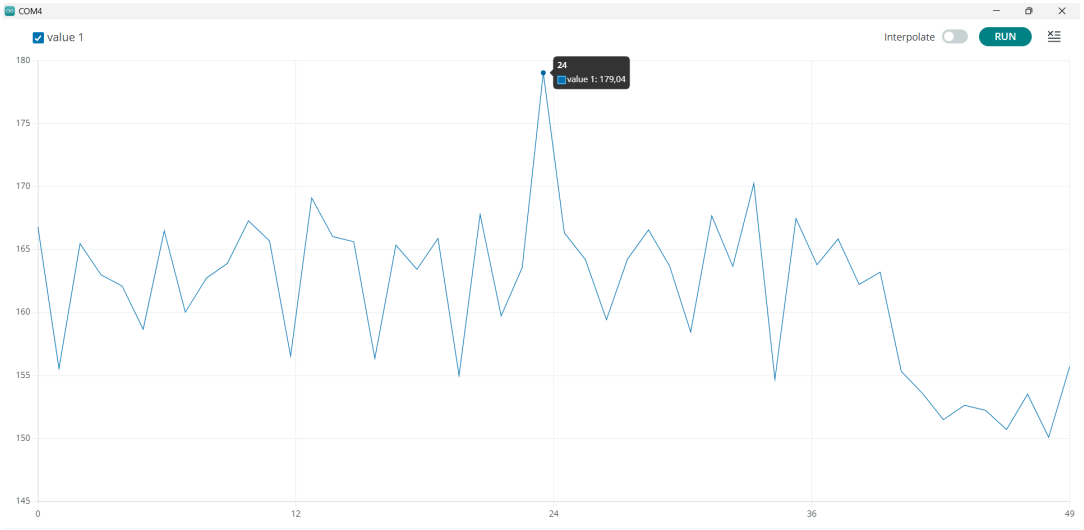


Figura 27 – Variação média da corrente (Valor mais alto)



exigências do sistema, o que facilitou os cálculos necessários para a seleção da bateria ideal. Optamos por uma bateria de íon de lítio, com tensão de 3,7V e capacidade de $9800mAh$ (miliAmpère-hora), similar àquelas utilizadas em lanternas táticas. Essa escolha se fundamenta na alta densidade de energia, na leveza e na longa vida útil dessas baterias. A capacidade da bateria foi ajustada para satisfazer os requisitos operacionais do sistema, com ênfase em períodos prolongados sem a necessidade de recarga, além de considerar possíveis variações nas condições ambientais.

6.9.2 Cálculo do consumo da bateria enquanto a coleira está ligada

Conforme evidenciado pelos gráficos apresentados nas Figuras 26 e 27,percebemos que a corrente do sistema oscila entre 150,08 mA e 179,04 mA (miliAmpère).Essa variação indica um consumo relativamente moderado, o que sugere a viabilidade do uso de uma bateria de menor capacidade. Nesse contexto, a escolha de uma bateria de 9800 mAh (miliAmpère-hora) se mostra adequada. A seguir, foi realizado os cálculos, após considerar uma corrente de consumo de 200 mA para simplificar as análises. Para isso basta dividir a Capacidade da bateria (em mAh) pelo consumo de corrente (em mA). Abaixo é mostrado o resultado dessa conta.

$$\frac{9800mAh}{200mA} = 49 \text{ h} \qquad (6.1)$$

Este valor de consumo é calculado após se considerar que a bateria permaneça ligada durante todo o dia. Em outras palavras, se a coleira permanecer constantemente ativa, a bateria durará apenas 49 horas. No entanto, não é necessário que a coleira permaneça ligada o tempo todo. Precisamos que ela esteja operacional apenas no momento em que se comunica com a antena e envia o SMS; durante o restante do tempo, pode entrar no modo de economia de energia, conhecido como modo *sleep*, similar ao comportamento de outras coleiras. Portanto, é necessário recalcula o tempo de duração da bateria após considerar que a coleira ligue apenas 2 vezes ao dia, e permanece ativa por 5 minutos em cada ligação, para que tenha tempo o suficiente de que a coleira consiga obter as informações necessárias nos poucos momentos que ela ficará ligada.

Assim, teremos 2 amostras por dia, e cada amostra deve ter uma duração de 5 minutos. Então para que seja possui saber quantas horas por dia essa coleira permanecerá ligada devemos multiplicar o numero de amostras pelo tempo (em segundos) que ela ficará ligada. A fórmula abaixo mostra essa multiplicação.

$$2 \cdot (\frac{5}{60}) = 0{,}167 \text{ h/dia} \qquad (6.2)$$

Com isso basta multiplicar a quantidade de horas que o sistema ficará ligado com o consumo da coleira. E assim descobriremos o quanto de corrente o sistema ira gastar durante esse tempo.

$$200mA \cdot 0{,}167h = 33{,}4 \text{ mAh} \qquad (6.3)$$

Com esses cálculos, se torna possível determinar a duração da bateria durante o período em que ela permanecerá ligada. Para achar esse valor, basta dividir a Capacidade da bateria (em mAh) pelo consumo de corrente (em mA).

$$\frac{9800mAh/dia}{33{,}4mAh} = 293{,}41 \text{ dias} \qquad (6.4)$$

Esse é o tempo que a bateria dura após considerar apenas o consumo nas 2 amostras. No entanto, é importante destacar que mesmo durante o modo sleep, o módulo ainda consome energia. Então e necessário calcular o consumo da corrente nesse modo, para isso foi escrito um código para colocar os módulos no modo sleep e ver qual será o valor de corrente consumida.

6.9.3 Cálculo do consumo da bateria no modo sleep

O código começa com a declaração das bibliotecas que precisará ser usada. Após isso, declarasse uma variável chamada **myFile** do tipo **File**. Esta variável será usada para interagir com o cartão SD, no Arduino. Depois disso, Declara uma variável chamada **gps** do tipo **TinyGPSPlus**. Essa é uma biblioteca para a manipulação de dados de GPS. Essa variável será usada para interagir com o módulo GPS no Arduino. Depois disso, declara uma constante chamada **buttonPin** e atribui a ela o valor 3. Esta constante representa o pino ao qual está conectado o botão que será utilizado para acordar o Arduino do modo de economia de energia. Após isso temos a função *void wakeUp()* Esta função é chamada quando o botão é acionado.

Na função *void setup()* Define-se o modo do **buttonPin**,o pino associado ao botão, como entrada com pull-up interno. Após isso a função anexa uma interrupção ao pino do botão para acionar a função *wakeUp()* quando há uma mudança no estado do botão.

A função *void loop()* Verifica se o botão está pressionado, o que indica a necessidade de colocar o Arduino, o SIM800L e o GPS em modo de economia de energia. Caso o botão esteja pressionado, coloca os dispositivos em modo de economia de energia com o comando **SLEEP_MODE_PWR_DOWN**. Caso não esteja acionado o programa executa o restante do código. O trecho de código do *else* é executado quando o botão não está à ser pressionado. Com isso, o programa inicializa as comunicações com os módulos GPS e SIM800L, lê dados de localização do GPS, grava esses dados em um cartão SD e realiza outras operações, como o envio de SMS.

Após a execução do modo de economia de energia, a função continua a partir desse ponto. Primeiro o código vai desativar o modo de economia de energia e despertar os dispositivos. Depois inicializa-se as comunicações, configurações e bibliotecas necessárias para que os módulos funcionem. Finalmente a função lê dados do módulo GPS, e se as informações de localização do GPS são válidas, cria uma mensagem com os dados de hora,

latitude e longitude. Após isso a função grava a mensagem em um arquivo no cartão SD, e por fim, realiza uma série de ações relacionadas à comunicação com o módulo SIM800, que inclue o envio de mensagens SMS e manipulação de arquivos.

```
1 #include <avr/sleep.h>
2 #include <SPI.h>
3 #include <SD.h>
4 #include <TinyGPS++.h>
5
6 File myFile;
7 TinyGPSPlus gps;
8 const int buttonPin = 3; // O pino do bot o que ser  usado para
      acordar o Arduino
9
10 void wakeUp(){
11   // Esta fun  o ser  chamada quando a interrup  o for acionada.
12 }
13
14 void setup(){
15   pinMode(buttonPin, INPUT_PULLUP);
16   attachInterrupt(digitalPinToInterrupt(buttonPin), wakeUp, CHANGE);
17 }
18
19 void loop(){
20   if (digitalRead(buttonPin) == LOW) {
21     // Coloca o Arduino, SIM800L e GPS em modo sleep.
22      Serial.println("AT+CSCLK=1"); // Coloca o SIM800L em modo sleep
23     set_sleep_mode(SLEEP_MODE_PWR_DOWN);
24     sleep_enable();
25     attachInterrupt(digitalPinToInterrupt(buttonPin), wakeUp, CHANGE);
26     sleep_mode();
27
28     // O programa continua daqui depois de acordar.
29     sleep_disable();
30     detachInterrupt(digitalPinToInterrupt(buttonPin));
31
32
33     Serial.println("AT+CSCLK=0"); // Acorda o SIM800L
34   } else {
35     // O restante do seu c digo vai aqui.
36     Serial.begin(9600);
37     Serial1.begin(115200); // Comunica  o com o m dulo GPS (Serial1 =
      RX1 e TX1 no Mega)
38     Serial2.begin(19200); // Comunica  o com o m dulo SIM900
39     delay(1000);
40
```

```
41      Serial.print("Inicializando o cart o SD...");
42      if (!SD.begin(4)) {
43        Serial.println("Falha na inicializa  o!");
44        while (1);
45      }
46      Serial.println("Inicializa  o conclu da.");
47      String latitude = "0";
48      String longitude = "0";
49      String hora = "0";
50
51      while (Serial1.available() > 0){
52        gps.encode(Serial1.read());
53      }
54
55      if (gps.location.isUpdated()){
56        if (gps.location.isValid()){
57          latitude = String(gps.location.lat(), 6);
58          longitude = String(gps.location.lng(), 6);
59          hora = String(gps.time.hour()) + ":" + String(gps.time.minute())
      + ":" + String(gps.time.second());
60        }
61
62        String mensagem = "Hora: " + hora + " ||http://maps.google.com/
     maps?q=" + latitude + "," + longitude;
63
64        // Gravar os dados no SD
65        myFile = SD.open("test.txt", FILE_WRITE);
66        if (myFile) {
67          myFile.println(mensagem);
68          myFile.close();
69        } else {
70          Serial.println("Erro ao abrir test.txt");
71        }
72
73        Serial2.println("AT+CSQ"); // Envia o comando AT+CSQ para o
     m dulo SIM900
74        delay(1000);
75
76        while (Serial2.available()) {
77          String resposta = Serial2.readString(); // L   a resposta do
     m dulo SIM900 como uma string
78          int inicio = resposta.indexOf(": "); // Encontra a posi   o do
     caractere ": " na resposta
79          int fim = resposta.indexOf(","); // Encontra a posi   o do
     caractere "," na resposta
80          if (inicio != -1 && fim != -1) { // Se ambos os caracteres foram
      encontrados
```

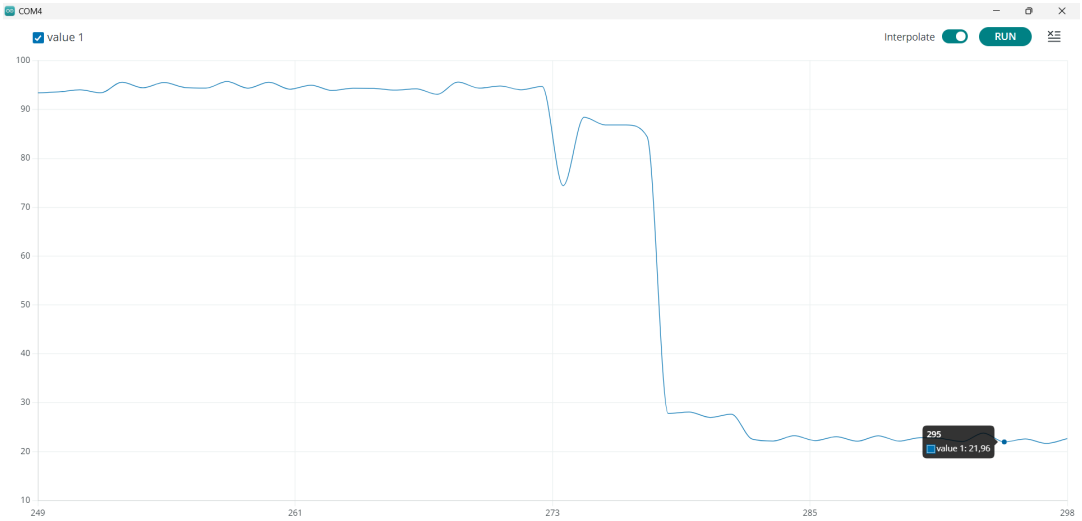
```
81            String qualidade = resposta.substring(inicio + 2, fim); //
    Extrai a substring entre os caracteres
82            int valor = qualidade.toInt(); // Converte a substring em um
    valor inteiro
83
84            if (valor >= 10) { // Se o valor for maior ou igual a 10
    enviar dados pelo sms
85              sendSMS(mensagem);
86              delay(5000); // Aguardar 30 segundos antes de enviar outro
    SMS
87
88              // Ap s o envio bem-sucedido, mover as entradas enviadas
    para um arquivo tempor rio
89              File tempFile = SD.open("temp.txt", FILE_WRITE);
90              if (tempFile) {
91                tempFile.println(mensagem);
92                tempFile.close();
93              } else {
94                Serial.println("Erro ao abrir temp.txt");
95              }
96            }
97          }
98        }
99
100       // Ap s o envio de todas as entradas, excluir o arquivo original
    e criar um novo arquivo com as entradas n o enviadas
101       SD.remove("test.txt");
102
103       File oldFile = SD.open("temp.txt");
104       File newFile = SD.open("test.txt", FILE_WRITE);
105
106       while(oldFile.available()) {
107         newFile.write(oldFile.read());
108       }
109
110       oldFile.close();
111       newFile.close();
112
113       SD.remove("temp.txt");
114     }
115   }
116 }
117
118
119 void sendSMS(String message) {
120   Serial2.println("AT");
121   delay(1000);
```

```
122   Serial2.println("AT+CMGF=1");
123   delay(100);
124   Serial2.print("AT+CMGS=\"+5562999292346\"\r");
125   delay(100);
126
127   Serial2.print(message);
128   delay(100);
129
130   Serial2.write(26);
131 }
```

Listing 6.8 – Código que coloca o sistema no modo sleep, e mostra o gráfico da corrente

Com este código pronto, colocou-se os módulos para dormir e, com isso, finalmente foi possível medir o quanto de corrente eles consumiam no modo sleep. Então foi plotado um gráfico que mostra a queda de consumo de quando o módulo esta à funcionar normalmente, e passa a funcionar no modo sleep. A Figura 28 apresenta esse gráfico.

Figura 28 – Gráfico do consumo da corrente, enquanto os módulos estão no modo sleep



Após analisar o gráfico, é possível observar que, no modo sleep, a coleira consome aproximadamente 20 mA. Em outras palavras, o consumo de energia é 10 vezes menor do que quando o sistema está à operar normalmente. Logo, durante o período em que a coleira permanece no modo de repouso, ocorre uma significativa economia no consumo, o que resulta em uma notável melhoria na durabilidade da bateria. Com base nessas informações, é factível calcular o consumo da bateria enquanto o sistema está nesse estado de repouso. Para realizar esse cálculo, inicialmente, subtrai-se o tempo que a coleira fica ativa do total de minutos em um dia, uma vez que um dia possui 1440 minutos. Assim

como mostra a equação 6.6.

$$1440 - (2 \cdot \ 5) = 1430 \text{ minutos} \qquad (6.5)$$

Uma vez que é conhecido quantos minutos a coleira ficará no modo sleep por dia. Precisamos saber o consumo por hora do sistema no modo repouso, Então para que seja possui calcular esse consumo, precisa-se fazer igual foi feito na equação 6.3. Multiplicar o consumo de carga (20mA) com o tempo que a coleira deve ficar ligado em horas. .

$$20mA \cdot (\frac{1430}{60}) = 476{,}67\text{mAh} \qquad (6.6)$$

6.9.4 Cálculo definitivo do gasto da bateria

Foi identificado tanto o valor do consumo durante as amostras quanto o valor do consumo durante o modo sleep. Com esses valores agora e possível terminar a conta para descobrir quantos dias de duração terá a bateria. Para isso primeiro precisamos somar esses dois valores, para achar o consumo (em mA) que será gasto por dia.

$$33{,}4\text{mAh} + 476{,}67\text{mAh} = 510{,}07\text{mAh} \qquad (6.7)$$

Então a coleira consome 1200 mA por dia. Com esse valor podemos concluir a estimativa de quantos dias a bateria vai durar.

$$\frac{9800mAh/dia}{510{,}07mAh} = 19{,}21 \text{ dias} \qquad (6.8)$$

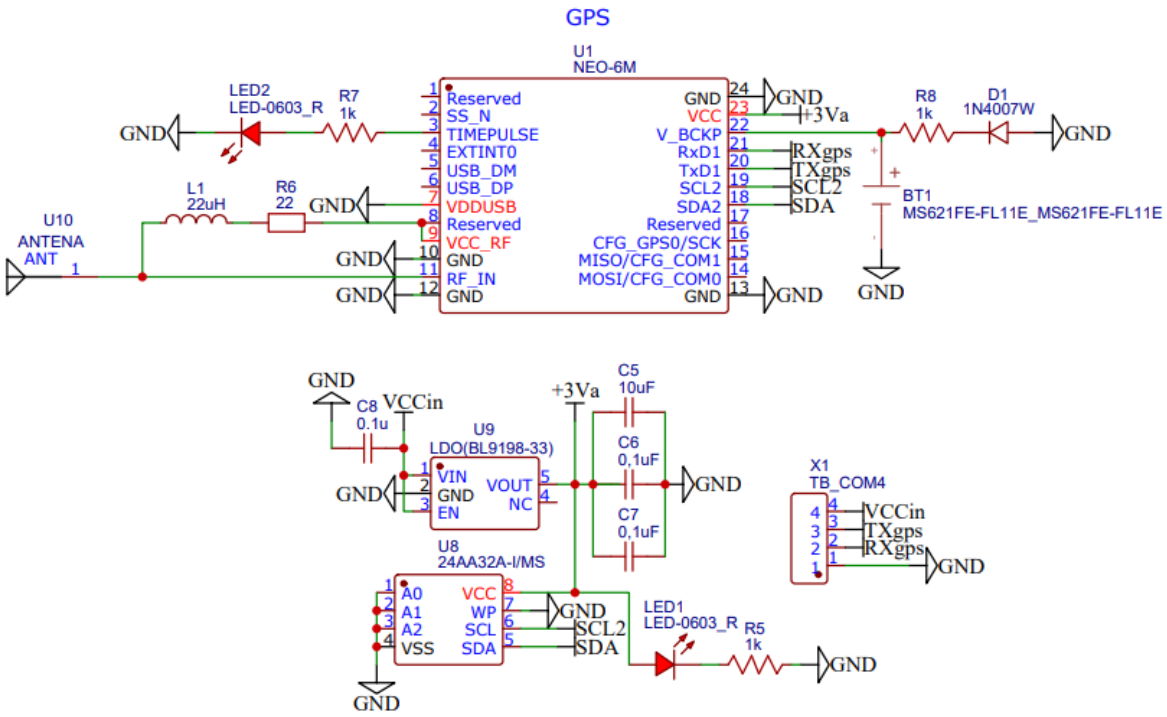
Dessa forma, a bateria apresenta uma autonomia de aproximadamente 8 dias. Esse resultado revela que, mesmo com uma bateria de capacidade considerável, é necessário realizar uma otimização na coleira. Essa otimização deve ser feita para assegurar que a durabilidade atenda plenamente às nossas expectativas.

6.10 PROJETO DA PLACA DE CIRCUITO IMPRESSO

Depois de tudo o que foi desenvolvido até então deu-se início a ultima etapa do projeto, o desenvolvimento é a montagem do protótipo da PCB (Placa de circuito impresso). Para isso foi desenvolvido um conjunto de esquemático que utiliza o easyEDA. Os esquemáticos foram desenvolvidos em folhas separadas, para facilitar na hora de criar uma PCB. A Figura 29 ilustra o esquematico do GPS, A Figura 30 apresenta o esquemático

do SD, a Figura 31 demonstra o esquemático do GPRS e a Figura 32 exibe o esquemático do ATMega2560.

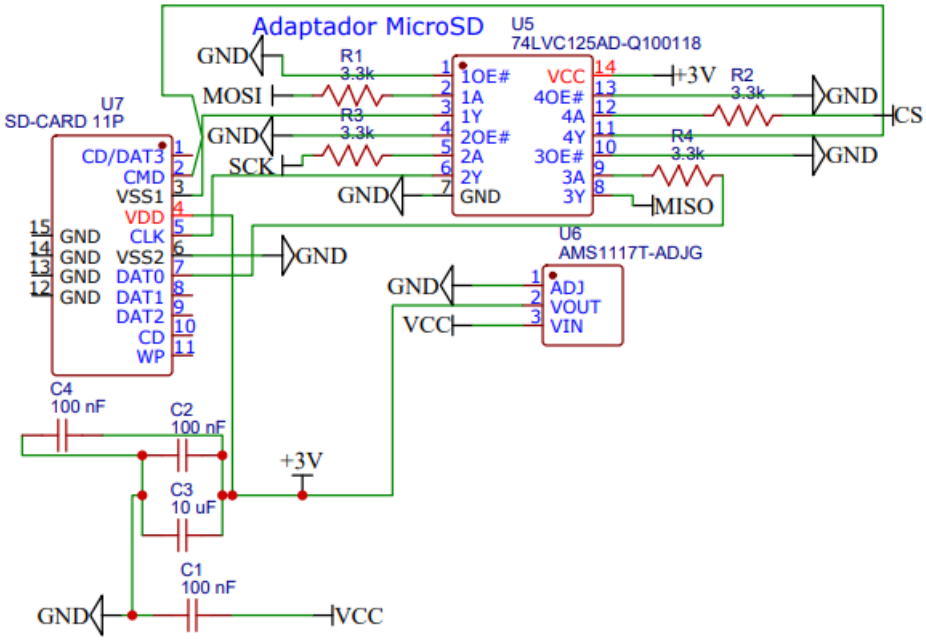
Figura 29 – Esquemático do GPS feita no EasyEDA



O esquemático do GPS evidencia a presença de três componentes. O primeiro deles é o U1, identificado como NEO-6M, esse é um chip de receptor GPS integrado que incorpora todos os quatro blocos mencionados acima. O chip NEO-6M é um chip de baixo custo e alta eficiência, o que o torna uma boa opção para aplicações de receptores GPS de baixo custo. Já o componente U8, designado como 24AA32A-I/MS, configura-se como um chip de memória flash. Sua atribuição central é armazenar dados oriundos do receptor GPS, o que faz com que ele obtenha informações vitais como posição geográfica, data e hora. Complementarmente, o chip U9, referenciado como LDO (BL9198-33), exerce a função de regulador de tensão, o que garante uma alimentação estável de 3,3 V indispensável para o desempenho eficiente do NEO-6M. Esses elementos desempenham papéis fundamentais no sistema, o que contribui para a robustez e eficácia do módulo GPS.

O esquemático do adaptador microSD, é composto por dois principais blocos. O primeiro é um conversor de nível lógico e o outro é um circuito de proteção. O conversor de nível lógico converte os sinais de tensão de 3,3 V do microSD para os sinais de tensão de 1,8 V do microcontrolador. O circuito de proteção protege o microSD contra sobrecargas

Figura 30 – Esquemático do adaptador MicroSD feita no EasyEDA



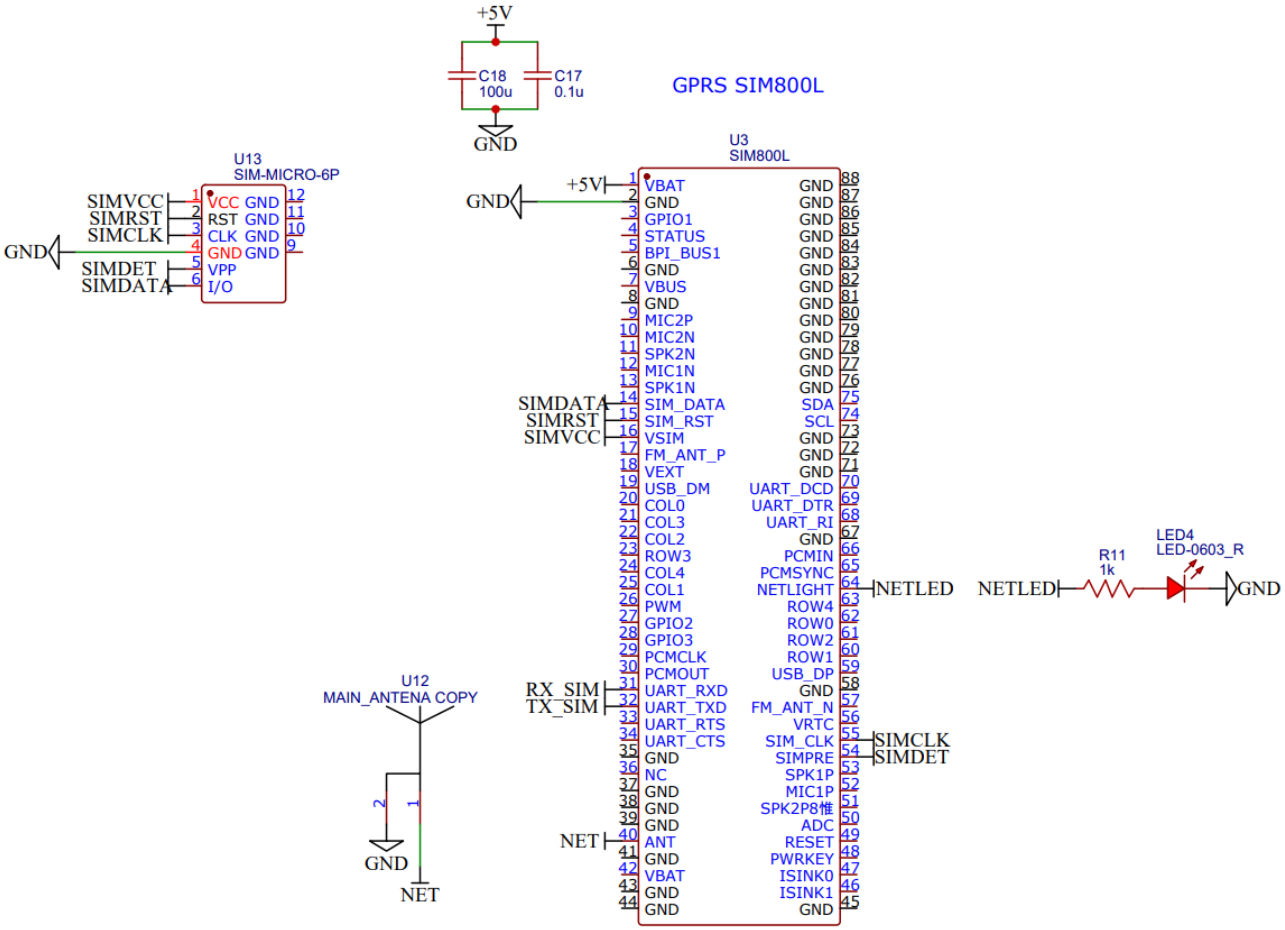
de corrente e sobretensões.

O esquemático do GPRS SIM800L incorpora uma antena de chip de 17 cm, caracterizada por sua eficácia em receptores GPS de custo acessível e natureza omnidirecional, capaz de receber sinais de rádio de qualquer direção. Para filtrar interferências, emprega-se um filtro passa-banda de 1,57542 GHz, sintonizado à frequência de transmissão dos satélites GPRS. O receptor de rádio adotado é do tipo superheterodino, que faz a converção da frequência dos sinais recebidos para uma mais baixa, o que facilita o subsequente processamento pelo microcontrolador de baixo custo e alta eficiência. Este microcontrolador, crucial para aplicações de receptores GPS e GPRS acessíveis, processa os sinais digitais da antena, é calcula posição e tempo. Para Completar o conjunto, o módulo de comunicação GPRS de baixo custo encarrega-se da transmissão e recepção de dados pela rede GPRS. Esses componentes combinados proporcionam um sistema robusto e eficiente para a finalidade proposta.

E por fim tem o esquemático do ATMEGA2560, que de todos, é o mais simples, uma vez que esse componente se trata de um microcontrolador. Por esse motivo esse esquemático tem apenas o essencial para que o ATMEGA funcione da melhor forma possível, e com isso ele possa atuar corretamente no restante do projeto.

Ao utilizar esses esquemáticos, finalmente foi possivel ter as informações necessárias

Figura 31 – Esquemático do GPRS feita no EasyEDA



para desenvolver a parte mais importante do projeto: O primeiro protótipo da PCB (Placa de circuito impresso). Com a montagem desse protótipo representa uma fase crucial para a validação prática do design, pois, ao ter um circuito próprio, completo com todos os componentes utilizados ao longo do desenvolvimento, torna-se possível verificar a integração desses componentes e avaliar do desempenho do sistema em condições reais. Com esse protótipo, também é possível obter uma noção das dimensões e do peso da coleira. A Figura 33 apresenta a PCB desenvolvida, sendo que as trilhas vermelhas são as trilhas superiores, as em azul são as trilhas inferiores, e o amarelo são os componentes. Vale ressaltar, que os resultados obtidos durante essa etapa podem ser discutidos e documentados em futuras atualizações deste trabalho, o que pode proporcionar uma visão abrangente do desenvolvimento do projeto.

Figura 32 – Esquemático do ATMega2560 feita no EasyEDA

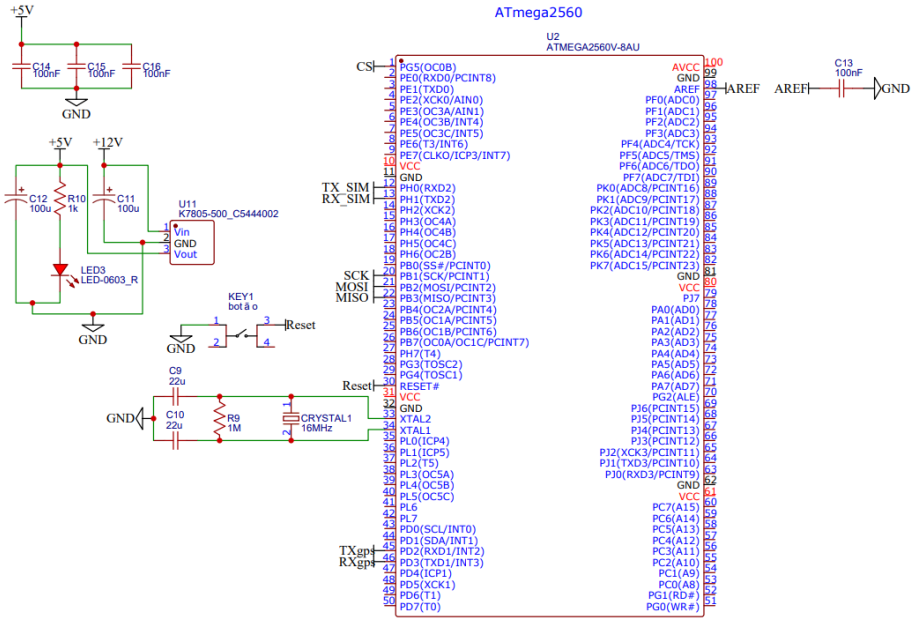
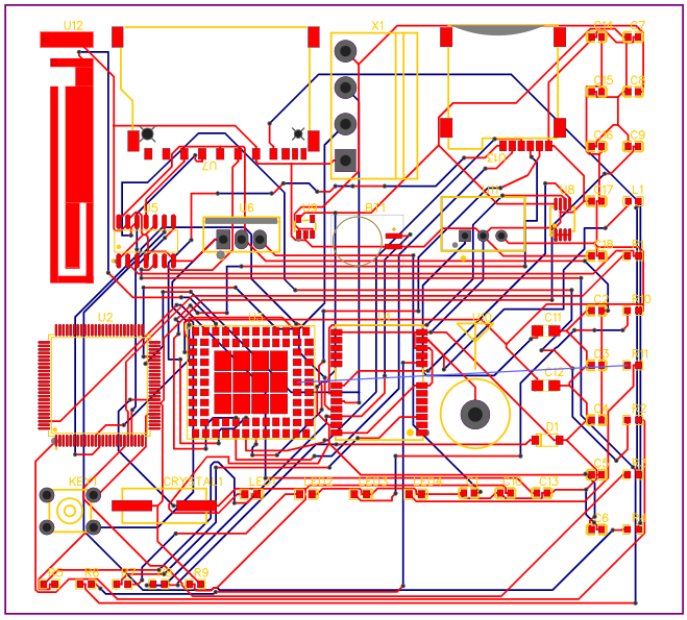


Figura 33 – Protótipo da primeira placa de circuito impresso



6.11 CUSTO PARA DESENVOLVER A COLEIRA

Com todas essas informações disponíveis, e possível entender qual vai ser o custo de produção do circuito. Assim o custo para desenvolver essa tecnologia, sem contar com o custo da mão de obra é de R$ 648,00 (seiscentos e quarenta e oito reais), sujeito a variações de acordo com a flutuação dos preços dos componentes ou possíveis substituições. A

seleção desses componentes proporcionará um equipamento com preço acessível em relação a outras soluções disponíveis no mercado, o que possibilitará uma ótima margem de aprimoramento tecnológico e viabilizará sua aplicação em instituições com recursos limitados para o monitoramento ambiental. A Tabela 7 apresenta o preço médio dos componentes utilizados.

Tabela 7 – Tabela com o preço dos componentes, em lojas de eletrônica em Novembro de 2023.

Componente	**Preço**
Arduino Mega	R$ 150,00
GPRS SIM900	R$ 120,00
GPS NEO-6M	R$ 57,00
MicroSD Adapter	R$ 13,00
TinyRTC	R$ 13,00
Bateria 3,7 V 9800 mA (x2)	R$ 44,00
Bateria de Lítio 3 V	R$ 2,00
Coleira de couro	R$ 70,00
Caixa Protetora	R$ 80,00
Cartão SD 128 GB	R$ 70,00
Chip celular	R$ 12,00
TOTAL	**R$ 648,00**

7 DESAFIOS ENFRENTADOS

Neste capítulo, serão abordados os desafios enfrentados durante a implementação e desenvolvimento do projeto. Serão abordados as dificuldades técnicas, limitações e obstáculos encontrados ao longo do processo, assim como as estratégias adotadas para superá-los.

7.1 LIMITAÇÕES DE RECURSOS

O primeiro desafio significativo foi o orçamento restrito. Uma vez que a coleira tem como premissa ser desenvolvido com um baixo orçamento, de forma que ela fosse acessível, para que ela fosse competitiva e conseguisse concorrer com os outros produtos disponíveis no mercado. Esse fato que impactou na escolha dos componentes, que deveriam ter um custo baixo. Isso levou a compromissos na qualidade e na capacidade do sistema, o que exigiu a busca por alternativas viáveis que atendessem aos requisitos mínimos do projeto.

Além disso como a disponibilidade desses componentes eram limitada, e com a perca de alguns deles durante o desenvolvimento, além da falta de conhecimento sobre o funcionamento de alguns equipamentos de laboratório como os analisadores de espectro, que poderia ajudar a entender a frequência de rádio de algumas coleiras usados pelo CETAS que usavam de VHF, esses fatores restringiu a capacidade de realizar experimentos e verificações extensivas. Ressalta-se que, à medida que o projeto avançava no desenvolvimento e os módulos precisaram ser trocados, os recursos financeiros aumentaram um pouco. Nesse momento foi possível adquirir os componentes para substituir aqueles que haviam estragados ou que precisavam ser trocados.

7.2 DESAFIOS TÉCNICOS

Durante a fase inicial do projeto, uma das principais dificuldades encontradas envolveu a integração dos diferentes módulos de hardware, o que incluiu o GPRS SIM800L, o GPS NEO-6M e o MicroSD Adapter. A complexidade da comunicação entre esses dispositivos resultou em problemas de sincronização e interferência de sinal, o que dificulta a obtenção de dados precisos e confiáveis.

Além disso a parte de software dos módulos integrados apresentou desafios significativos, uma vez que, frequentemente, as diversas bibliotecas tiveram conflitos de compatibilidade, além de que durante o final do projeto algumas atualizações súditas fizeram com que fosse necessário mudar o hardware para um com mais portas seriais, além de ser mais potente. O que fez com que fosse consumido um tempo considerável, que

atrasou o progresso.

7.3 DESAFIO DA CARGA

Uma parte fundamental para o projeto foi a escolha da bateria. Uma vez que ela deve possuir a carga necessária para alimentar a coleira, além de que ela deve ter alguns meses de carga (o objetivo inicial era que ela devia durar 6 meses), já que ela não pode ser recarregada, e precisa ser leve para atingir o objetivo de peso definido pelo CETAS (Centro de Triagem de Animais Selvagens).

Durante o desenvolvimento do projeto várias baterias foram usadas, o que resultou em algumas que não conseguiam alimentar a coleira, outras eram muitos grandes, algumas possuíam uma tensão tão alta que se usasse errado poderia queimar o circuito, mas nenhuma delas durava o suficiente para atingirmos o nosso objetivo principal. A bateria que escolhemos pode durar entre 2 e 3 meses, mas e bastante pesada, é muitas vezes nas etapas de testes ela ficava instável, o que resultou na necessidade usar uma fonte externa para alimentar o circuito.

8 CONCLUSÃO

Ao longo deste trabalho, foram explicados os fundamentos teóricos e práticos relacionados ao desenvolvimento e implementação do sistema de monitoramento de animais selvagens, além da importância desse monitoramento para a preservação ambiental. Através da integração cuidadosa de tecnologias como o Arduino Uno, o módulo GPRS SIM800L, o GPS NEO-6M e o MicroSD Adapter, foi possível criar um sistema eficaz e eficiente para rastreamento e coleta de dados.

Embora houvesse vários desafios ao longo do processo de pesquisa e desenvolvimento do projeto, os mesmos foram enfrentados com determinação e criatividade. Acompanhado de um planejamento cuidadoso, abordagens proativas, e o uso inteligente dos recursos disponíveis, foi possível superar obstáculos e alcançar resultados substanciais. A compreensão e a resolução desses desafios contribuíram significativamente para o aprimoramento do projeto e proporcionaram valiosas lições de aprendizado para futuras iniciativas de desenvolvimento tecnológico.

É importante ressaltar que este estudo representa um passo inicial na implementação de soluções acessíveis e de baixo custo para o monitoramento de animais selvagens. A partir das descobertas e experiências adquiridas neste projeto, abre-se um campo vasto para pesquisas futuras e desenvolvimentos adicionais, que visa aprimorar ainda mais a precisão, eficiência e aplicabilidade prática do sistema.

Uma vez compreendido a importância da conservação da vida selvagem e da preservação dos ecossistemas, esperou-se que este trabalho possa contribuir de maneira significativa para a conscientização e a proteção de espécies em risco, além de inspirar novas iniciativas e avanços tecnológicos nessa área crucial.

REFERÊNCIAS

ABREU, EF. *et al.* **Lista de Mamíferos do Brasil (2022-1) [Data set].Zenodo**. 2022. Disponível em: https://doi.org/10.5281/zenodo.7469767. Acesso em: 13 abr. 2023.

ALBUQUERQUE, Yure. **Conhecendo o Arduino Mega 2560**. Disponível em: https://blog.smartkits.com.br/conhecendo-o-arduino-mega-2560/#:~:text=O%20que%20%C3%A9%20Arduino%20Mega,flash%20e%208KB%20de%20SRAM.. Acesso em: 17 nov. 2023.

AMARAL, Haroldo. **Medindo corrente e tensão com o módulo INA219**. 2017. Disponível em: https://www.makerhero.com/blog/medindo-corrente-e-tensao-modulo-ina219/. Acesso em: 25 nov. 2023.

AMBPLUS. **O que é Monitoramento da Fauna Silvestre?** Disponível em: https://ambplus.com.br/o-que-e-o-monitoramento-da-fauna-silvestre/. Acesso em: 3 mar. 2023.

ARDUCORE. **Módulo GPS NEO 6M V2 + Aerial Antena para Arduino e Raspberry PI**. 2023. Disponível em: https://www.arducore.com.br/modulo-gps-antena-para-arduino-e-raspberry-pi. Acesso em: 25 jul. 2023.

ARDUINO. **Arduino® MEGA 2560 Rev3**. [*S.l.*], 2023. P. 17. Disponível em: https://docs.arduino.cc/resources/datasheets/A000067-datasheet.pdf. Acesso em: 12 nov. 2023.

ARDUINO. **Arduino® UNO R3**. [*S.l.*], 2023. P. 14. Disponível em: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf. Acesso em: 12 jun. 2023.

ARDUINO. **Arduino® UNO R3**. [*S.l.*], 2023. P. 13. Disponível em: https://docs.arduino.cc/hardware/uno-rev3. Acesso em: 12 jun. 2023.

ARDUINO. **Arduino® UNO R3**. [*S.l.*], 2023. P. 13. Disponível em: https://docs.arduino.cc/static/ff47e587a416bb4f20341dd1f7ed6152/A000066-datasheet.pdf. Acesso em: 12 jun. 2023.

ARDUINOCIA. **Como usar módulo cartão micro SD Arduino**. 2020. Disponível em: https://www.arduinoecia.com.br/como-usar-modulo-cartao-micro-sd-arduino/. Acesso em: 17 jun. 2023.

ARGOS. **Argos User's Manual**. [*S.l.*], 2007-2016. P. 64. Disponível em: https://www.argos-system.org/wp-content/uploads/2016/08/r363_9_argos_users_manual-v1.6.6.pdf. Acesso em: 12 mar. 2023.

ARRUDA, Moacir Bueno; NOGUEIRA DE SÁ, Luís Fernando s. **Corredores Ecológicos**: Uma abordagem integradora de ecossistemas no Brasil. 2003. Disponível em: http://www.ibama.gov.br/sophia/cnia/livros/corredoresecologicosdigital.pdf. Acesso em: 13 mai. 2023.

BATEMAN, P. W.; FLEMING, P. A. Big city life: Carnivores in urban environments. **Journal Zoology**, v. 283, p. 1–23, 2012.

BUCHERONI, Giulia. **Brasil 'ganha' mais espécies de aves e reforça o título de país megadiverso**. 30 jul. 2021. Disponível em: https://g1.globo.com/sp/campinas-regiao/terra-da-gente/noticia/2021/07/30/brasil-ganha-mais-especies-de-aves-e-reforca-o-titulo-de-pais-megadiverso.ghtml. Acesso em: 11 mar. 2023.

COMPONENTS101. **SIM800L GSM Module**. 30 set. 2021. Disponível em: https://components101.com/wireless/sim800l-gsm-module-pinout-datasheet-equivalent-circuit-specs. Acesso em: 25 jun. 2023.

DELTA S, Engenharia. **Monitoramento de Fauna e sua importância prática**. 13 ago. 2021. Disponível em: https://www.deltas.eng.br/post/monitoramento-de-fauna. Acesso em: 13 mar. 2023.

DICHOFF, Nicoli. **Pesquisadores desenvolvem modelo brasileiro de colares para monitoramento animal**. 25 abr. 2017. Disponível em: https://www.embrapa.br/pantanal/busca-de-noticias?p_p_id=buscanoticia_WAR_pcebusca6_1portlet&p_p_lifecycle=0&p_p_state=pop_up%5C&p_p_mode=view&p_p_col_id=column-1%5C&p_p_col_pos=1%5C&p_p_col_count=2%5C&_buscanoticia_WAR_pcebusca6_1portlet_groupId=1354999&_buscanoticia_WAR_pcebusca6_1portlet_

articleId=21629309&_buscanoticia_WAR_pcebusca6_1portlet_viewMode=print. Acesso em: 13 mar. 2023.

DONCASTER, C. P.; DICKMAN, C. R.; D.W., MACDONALD. Feeding ecology of red foxes (Vulpes vulpes) in the city of Oxford, EnglandJ. Mamm. 71, p. 188–194, 1990.

ECO. **O QUE É Monitoramento Remoto de Animais.** 30 mai. 2015. Disponível em: https://oeco.org.br/dicionario-ambiental/29026-o-que-e-monitoramento-remoto-de-animais/. Acesso em: 3 mar. 2023.

ELETROGATE. **Arduino**. Disponível em: https://www.eletrogate.com/arduino. Acesso em: 4 jun. 2023.

ELETROGATE. **Arduino Shield - GSM GPRS SIM900 Com Antena Quad Band**. Disponível em: https://www.eletrogate.com/arduino-shield-gsm-gprs-sim900-com-antena-quad-band. Acesso em: 17 jun. 2023.

ELETROGATE. **Sensor de Corrente DC INA219 I2C**. 2023. Disponível em: https://www.eletrogate.com/sensor-de-corrente-dc-ina219-i2c. Acesso em: 25 nov. 2023.

FORMAN, R. T.; SPERLING, D.; BISSONETTE, J. A.; CLEVENGER, A. P.; CUTSHALL, C. D.; DALE, V. H.; WINTER, T.C. Road ecology: science and solutions. **Island Press.**, p. 3–4, 2002.

FRANÇA, Valéria. **Inteligência Artificial: veja como a tecnologia está salvando as baleias jubarte**. 25 fev. 2022. Disponível em: https://istoe.com.br/inteligencia-artificial-salva-animais/. Acesso em: 19 mar. 2023.

FURLONGER, C. L.; DEWAR, H. J.; M.B., FENTON. Habitat use by foraging insectivorous bats. Can. J. Zool. 65, p. 284–288. 1987.

GALETTI, M.; ALVES-COSTAS, C. P.; CAZETTA, E. Effects of forest fragmentation, anthropogenic edges and fruit colour on the consumption of ornithocoric fruits. **Biological Conservation**, v. 111, p. 269–273, 2013.

GEHRT, S. D.; RILEY, S.P.D. Coyotes (Canis latrans). In: Urban Carnivores: Ecology, Conflict, and Conservation. **Johns Hopkins University Press**, p. 79–98, 2010.

GUIMARÃES, Fábio. **Módulo GPS NEO-6M com Arduino**. 5 jan. 2021. Disponível em: https://mundoprojetado.com.br/modulo-gps-neo-6m/. Acesso em: 4 jun. 2023.

KLINK, C. A.; MACHADO, R. B. Conservation of the Brazilian Cerrado. 2005. Disponível em: https://doi.org/10.1111/j.1523-1739.2005.00702.x.. Acesso em: 13 abr. 2023.

LIMA CARLOS, Marcos de. **História do Arduino (MIC571)**. 16 jun. 2021. Disponível em: https://www.newtoncbraga.com.br/index.php/microcontroladores/138-atmel/18422-historia-do-arduino-mic571.html. Acesso em: 4 jun. 2023.

MAGLE, S. B.; THEOBALD, D.T; CROOKS, K. R. Comparing isolation metrics predicting the distribution of a highly interactive species across an urban gradient. **Landscape Ecology 24**, p. 267–280, 2009.

MCKINNEY, M. L. Urbanization as a major cause of biotic homogenization. **Biological Conservation 127**, p. 247–260, 2006.

MCROBERTS, Michael. **Arduino Básico**. 2. ed. [*S.l.*]: Novatec, 2015.

MOTA, Allan. **O que é Arduino e como funciona?** 14 jul. 2021. Disponível em: https://portal.vidadesilicio.com.br/o-que-e-arduino-e-como-funciona/. Acesso em: 15 jun. 2023.

MOUNTAINBAJA. **Módulo GPS NEO-6M com Arduino**. 18 abr. 2018. Disponível em: https://mundoprojetado.com.br/modulo-gps-neo-6m/. Acesso em: 4 jun. 2023.

MYERS, N. Mittermeier, R. A., Mittermeier, C. G., da Fonseca, G. A. B., Kent, J. Biodiversity hotspots for conservation priorities. 2000. Disponível em: https://doi.org/10.1038/35002501. Acesso em: 13 abr. 2023.

OLIVEIRA, Euler. **Como usar com Arduino – Módulo Leitor de Micro SD Card**. 2019. Disponível em: https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-modulo-leitor-de-micro-sd-card. Acesso em: 17 jun. 2023.

ONÇAFARI. **Monitoramento: saiba a diferença entre os colares VHF e GPS**. 8 jan. 2020. Disponível em: https://oncafari.org/2020/01/08/monitoramento-saiba-a-diferenca-entre-os-colares-vhf-e-gps/. Acesso em: 3 abr. 2023.

PALMINTERI, Sue. **Fatos e curiosidades sobre o rastreamento de vida selvagem por satélite**. 2 jun. 2019. Disponível em: https://oeco.org.br/an%C3%A1lises/fatos-e-curiosidades-sobre-o-rastreamento-de-vida-selvagem-por-satelite/. Acesso em: 11 mar. 2023.

PIMENTA, Thais. **Brasil tem 1.249 animais ameaçados; anfíbio entra na lista dos extintos**. 8 jun. 2022. Disponível em: https://g1.globo.com/sp/campinas-regiao/terra-da-gente/noticia/2022/06/08/brasil-tem-1249-animais-ameacados-anfibio-entra-na-lista-dos-extintos.ghtml. Acesso em: 4 mar. 2023.

RANDOMNERDTUTORIAL. **Guide to SIM900 GSM GPRS Shield with Arduino**. 2017. Disponível em: https://randomnerdtutorials.com/sim900-gsm-gprs-shield-arduino/. Acesso em: 28 mai. 2023.

RASTREAMENTO, Sim. **Qual a diferença entre GPRS e GPS**. 24 out. 2021. Disponível em: https://www.simtrack.com.br/qual-a-diferenca-entre-gprs-e-gps/#:~:text=Principais%5C%20diferen%5C%C3%5C%A7as%5C%20entre%5C%20GPRS%5C%20e,termos%5C%20de%5C%20latitude%5C%20e%5C%20longitude. Acesso em: 13 abr. 2023.

RIBEIRO, J. F.; WALTER, B. M. T. As Principais Fitofisionomias do Bioma Cerrado, In: SANO, S. M; ALMEIDA, S. P.; RIBEIRO, J. F (Eds), Cerrado: Ecologia e Flora. **Embrapa Informação Tecnológica**, v. 1, p. 406, 2008.

SANO, E. E; ROSA, R.; BRITO, J. L. S.; FERREIRA, L. G. Mapeamento semidetalhado do uso da terra do Bioma Cerrado. **Pesquisa Agropecuária Brasileira**, v. 43, n. 1, p. 153–158, 2008.

SOUZA, Fábio. **Arduino MEGA 2560**. 2014. Disponível em: https://embarcados.com.br/arduino-mega-2560/. Acesso em: 17 nov. 2023.

SOUZA, Fábio. **Usando os pinos digitais do Arduino**. 2013. Disponível em: https://embarcados.com.br/pinos-digitais-do-arduino/. Acesso em: 15 jun. 2023.

STATHAM, M.; STATHAM, H. L. Movements and habits of brushtail possums (Trichosurus vulpecula Kerr) in an urban area. **Wildl. Res. 24**, p. 715–726, 1997.

STRASSBURG, B. B. N. Brooks, T., Feltran-Barbieri, R., Iribarrem, A., Crouzeilles, R., Loyola, R., Latawiec, A. E., Oliveira Filho, F. J. B., De Scaramuzza, C. A. M., Scarano, F. R., Soares-Filho, B., Balmford, A.. Moment of truth for the Cerrado hotspot. 2017. Disponível em: https://doi.org/10.1038/s41559-017-0099. Acesso em: 13 mai. 2023.

TERRACLASS. Mapeamento do Uso e Cobertura da Terra do Cerrado, projeto Terraclass Cerrado 2013. **Empresa Brasileira de Pesquisa Agropecuária (EMBRAPA) e Instituto Nacional de Pesquisas Espaciais (INPE)**, Brasília-DF, 2013.

U-BLOX. **NEO-6 u-blox 6 GPS Modules Data Sheet**. [*S.l.*], 2011. P. 25. Disponível em: https://content.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%5C%28GPS.G6-HW-09005%29.pdf. Acesso em: 12 jun. 2023.

URBANEK, R. E.; NIELSEN, C. K.; WILSON, S. E. Survival of unexploited raccoons on a rural refuge in Southern Illinois. **Trans. Ill. State Acad.**, p. 3–4, 2009.

USER, Super. **Guia para uso do Shield GPRS GSM SIM900 com o Arduino**. 12 jun. 2020. Disponível em: https://www.arduinoeautomacao.com.br/all-blogs/domotica/guia-para-uso-do-shield-gprs-gsm-sim900-com-o-arduino. Acesso em: 17 jun. 2023.

USINAINFO. **SD Card Arduino / Leitor Micro SD Card**. 2023. Disponível em: https://www.usinainfo.com.br/outros-modulos-arduino/sd-card-arduino-leitor-micro-sd-card-2637.html. Acesso em: 25 jul. 2023.

WAKKA, Wagner. **Inteligência Artificial está sendo usada para combater extinção de pinguins**. 20 jan. 2020. Disponível em: https://canaltech.com.br/inteligencia-artificial/inteligencia-artificial-esta-sendo-usada-para-combater-extincao-de-pinguins-159275/. Acesso em: 19 abr. 2023.

WILCOX, Christie. **Extinções provocadas pelo homem fazem os outros mamíferos perderem milhões de anos**. 26 ago. 2018. Disponível em: https://www.nationalgeographicbrasil.com/animais/2018/10/extincoes-provocadas-pelo-homem-fazem-os-outros-mamiferos-perderem-milhoes-de-anos. Acesso em: 2 mai. 2023.