



**REAL TIME SYSTEMS AND INTERNET OF THINGS FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA**

TomoStroke

GROUP 25

Muhammad Nadzhif Fikri	2306210102
Neyla Shakira	2306250655
Ryan Adidaru	2306266994
Salim	2306204604

PREFACE

Di era perkembangan teknologi Internet of Things yang sangat pesat, sistem tertanam telah berevolusi menjadi platform yang semakin kuat dan mampu mendukung interaksi real-time, integrasi sensor, dan perilaku cerdas. Proyek akhir untuk mata kuliah Real-Time Systems and Internet of Things ini menghadirkan **TomoStroke**, sebuah perangkat virtual pet berbasis IoT yang dirancang untuk menunjukkan penerapan nyata dari prinsip-prinsip desain sistem tertanam. Tujuan utama kami adalah membangun sistem interaktif dan responsif yang menunjukkan bagaimana mikrokontroler, sensor, dan perangkat lunak multitasking dapat digabungkan untuk menciptakan pengalaman pengguna yang dinamis.

TomoStroke mensimulasikan sekumpulan hewan digital yang mampu merespons interaksi pengguna, input dari lingkungan, serta perubahan kondisi internal seperti rasa lapar dan kebahagiaan. Sistem ini mengintegrasikan berbagai komponen, termasuk mikrokontroler ESP32, layar OLED, sensor getaran, buzzer, serta firmware berbasis task menggunakan FreeRTOS untuk memodelkan perilaku real-time pada perangkat keras yang memiliki keterbatasan sumber daya. Melalui proyek ini, kami menjembatani pengetahuan teoritis dengan implementasi praktis, menerjemahkan konsep-konsep sistem tertanam menjadi perangkat IoT fungsional dengan arsitektur perangkat lunak yang modular dan mudah dikembangkan.

Proyek ini memungkinkan kami untuk mengeksplorasi berbagai topik penting seperti penjadwalan real-time, pemrosesan berbasis event, penanganan sensor, komunikasi antar task menggunakan queue, serta proses rendering grafis pada mikrokontroler. Berbagai tantangan yang ditemui selama pengembangan memberikan wawasan berharga mengenai kompleksitas desain sistem IoT, mulai dari interfacing hardware, reliabilitas sinyal, prioritas task, hingga desain interaksi pengguna. Kami mengerjakan proyek akhir ini dengan antusiasme dan apresiasi, menyadari bahwa kesempatan ini merupakan wadah untuk menerapkan pengetahuan akademik dalam sebuah permasalahan rekayasa yang kreatif dan menantang secara teknis.

Depok, December 08, 2025

Group 25

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION.....	4
1.1 PROBLEM STATEMENT.....	4
1.2 PROPOSED SOLUTION.....	4
1.3 ACCEPTANCE CRITERIA.....	6
1.4 ROLES AND RESPONSIBILITIES.....	7
1.5 TIMELINE AND MILESTONES.....	8
 CHAPTER 2 IMPLEMENTATION.....	 11
2.1 HARDWARE DESIGN AND SCHEMATIC.....	11
2.2 SOFTWARE DEVELOPMENT.....	12
2.3 HARDWARE AND SOFTWARE INTEGRATION.....	17
 CHAPTER 3 TESTING AND EVALUATION.....	 18
3.1 TESTING.....	18
3.1.1 Unit Testing.....	18
3.1.2 Integration Testing.....	19
3.1.3 System Testing.....	19
3.1.4 Field Testing.....	20
3.2 RESULT.....	20
3.3 EVALUATION.....	21
 CHAPTER 4 CONCLUSION.....	 22

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Perkembangan sistem tertanam dan teknologi wireless telah memungkinkan lahirnya perangkat Internet of Things (IoT) yang semakin interaktif dan mampu merespons kondisi lingkungan secara real-time. Solusi IoT modern menggabungkan pemrosesan paralel, pembacaan sensor, dan interaksi pengguna untuk menghasilkan sistem yang responsif dan adaptif terhadap konteks.

Salah satu penerapan kreatif IoT terdapat pada digital companion system seperti virtual pet, yaitu sistem yang meniru perilaku makhluk hidup seperti rasa lapar, emosi, tingkat aktivitas, serta respons terhadap interaksi pengguna maupun lingkungan. Membangun sistem seperti ini pada perangkat dengan sumber daya terbatas merupakan tantangan tersendiri.

Proyek ini memperkenalkan **TomoStroke**, sebuah perangkat virtual pet berbasis ESP32 yang memanfaatkan OLED display, sensor getaran, tombol input, dan buzzer. Sistem ini mensimulasikan delapan digital pets dengan state independen, animasi, dan logika interaksi yang berjalan secara real-time menggunakan FreeRTOS. Selain itu, arsitektur perangkat dirancang fleksibel sehingga dapat diperluas untuk mendukung mesh networking, memungkinkan komunikasi antar-perangkat dan interaksi virtual pet lintas sistem.

1.2 PROPOSED SOLUTION

Solusi yang diusulkan dalam proyek ini adalah merancang dan membangun **TomoStroke**, sebuah sistem virtual pet berbasis Internet of Things (IoT) yang dikembangkan menggunakan mikrokontroler **ESP32** dan firmware berbasis real-time. Sistem ini mensimulasikan beberapa hewan digital, dimana masing-masing memiliki state mandiri seperti rasa lapar, kebahagiaan, tingkat aktivitas, dan ekspresi emosional. State tersebut berkembang dari waktu ke waktu dan berubah sebagai respons terhadap interaksi pengguna melalui berbagai sensor dan input fisik, sehingga menghasilkan pengalaman virtual pet yang dinamis dan menyerupai perilaku makhluk hidup.

TomoStroke mengintegrasikan sejumlah komponen utama IoT, yaitu:

- **OLED display**, sebagai media animasi dan tampilan status
- **vibration sensor**, sebagai mekanisme interaksi feeding berbasis gerakan
- **push button**, sebagai navigasi untuk memilih pet
- **buzzer**, sebagai penanda audio ekspresi suara pet

Pada sisi perangkat lunak, sistem memanfaatkan penjadwalan tugas menggunakan **FreeRTOS**, sehingga berbagai proses real-time seperti rendering tampilan, pembaruan state pet, pengolahan input sensor, dan pemutaran audio dapat berjalan secara paralel tanpa menyebabkan blocking ataupun penurunan performa. Arsitektur ini memungkinkan seluruh proses tetap responsif meskipun dijalankan secara bersamaan.

Proyek ini juga mencakup keseluruhan siklus pengembangan sistem IoT, mulai dari perancangan rangkaian hardware, implementasi firmware, pembuatan arsitektur berbasis task, kalibrasi sensor, pengujian real-time, hingga evaluasi performa. Selain itu, arsitektur TomoStroke dirancang agar modular dan mudah dikembangkan lebih lanjut, termasuk dukungan awal untuk **mesh networking**, sehingga perangkat dapat saling bertukar informasi status pet atau memicu interaksi antar perangkat di masa mendatang.

1.3 ACCEPTANCE CRITERIA

Proyek TomoStroke dinyatakan berhasil apabila memenuhi kriteria-kriteria berikut:

1. Sistem mampu mensimulasikan delapan virtual pet dengan state mandiri (hunger, happiness, blink, movement, dan voice), yang berubah secara dinamis dan responsif terhadap interaksi pengguna melalui sensor getaran dan push button.
2. FreeRTOS harus menjalankan multitasking tanpa blocking, memastikan rendering OLED, pembaruan state, pemrosesan input, dan pemutaran audio berjalan paralel dengan respons cepat dan tanpa lag.
3. OLED harus menampilkan animasi pet, ekspresi wajah, serta indikator hunger dan happiness secara mulus tanpa flicker, tearing, atau delay visual.
4. ESP32, OLED, buzzer, sensor getaran, dan push button harus berfungsi stabil tanpa menyebabkan reset, freeze, atau error selama operasi jangka panjang.

5. Sistem harus memiliki modul MeshController yang mampu membangun dan memproses pesan “BRD” dan “FED” secara konsisten, termasuk mekanisme TTL untuk menjaga daftar perangkat mesh tetap akurat.

1.4 ROLES AND RESPONSIBILITIES

Berikut adalah pembagian kerja dan tanggung jawab dari masing-masing member kelompok kami

Roles	Responsibilities	Person
Role 1	Membuat Pet + Membuat Sprites + Merancang Schematic	Muhammad Nadzhif Fikri
Role 2	Laporan + Merangkai Hardware	Neyla Shakira
Role 3	Laporan + Merangkai Hardware	Ryan Adidaru
Role 4	Membuat Orchestrator	Salim

Table 1. Roles and Responsibilities

1.5 TIMELINE AND MILESTONES

Berikut adalah timeline pembuatan proyek “TomoStroke” kami:

Tanggal	Tasks	Deskripsi
02/12/2025	Penyusunan Struktur & Pembagian Kerja	Menyusun struktur lengkap laporan, menetapkan pembagian tugas, serta menyiapkan template dokumen dan checklist komponen.
03/12/2025	Finalisasi Hardware & Dokumentasi	Merakit rangkaian final, mengambil foto dokumentasi, membuat

		schematic, serta memastikan semua komponen bekerja stabil.
04/12/2025	Integrasi & Penyempurnaan Software	Merapikan firmware, menguji task FreeRTOS, memperbaiki bug sensor/OLED, serta menyiapkan screenshot interface pet.
05/12/2025	Pengujian Lengkap (Unit, Integration, System, Field)	Melakukan seluruh rangkaian pengujian Pet Class, sensor, task, dan pengujian lapangan. Mendokumentasikan hasilnya.
06/12/2025	Penulisan Laporan Chapter 1–2	Menulis latar belakang, solusi, hardware, software architecture, serta memasukkan foto/gambar pendukung.
07/12/2025	Penulisan Laporan Chapter 3–4	Menyusun bagian Testing, Evaluation, dan Conclusion berdasarkan hasil uji sistem.
08/12/2025	Finalisasi, Formatting, & Submission	Melakukan quality check keseluruhan laporan, penyempurnaan format, pembuatan PDF, dan memastikan dokumen siap dikumpulkan.

CHAPTER 2

IMPLEMENTATION

2.1 HARDWARE DESIGN AND SCHEMATIC

Alat dan komponen yang digunakan dalam proyek ini adalah sebagai berikut:

1. ESP32 Development Board
2. SSD1306 128×64 OLED Display
3. Push Button (GPIO input)
4. Vibration Sensor (SW-420)
5. Piezo Buzzer
6. Jumper Wires & Breadboard
7. Arduino IDE / PlatformIO
8. FreeRTOS (ESP32 built-in)
9. Adafruit GFX & SSD1306 Libraries
10. Custom Class Modules (Pet, Bar, Icon, MeshController)

2.2 SOFTWARE DEVELOPMENT

Implementasi perangkat lunak TomoStroke melibatkan penggabungan antara integrasi perangkat keras dan pengembangan firmware real-time. ESP32 berfungsi sebagai pengendali utama yang menjalankan beberapa task secara paralel menggunakan FreeRTOS.

MainLoop Task bertanggung jawab untuk membaca input sensor, memperbarui state setiap pet, serta melakukan rendering tampilan OLED pada frame rate yang stabil. **VoiceTask** menangani proses generasi suara melalui buzzer dan dipicu dengan sistem antrian (queue) sehingga tidak menghambat task utama yang mengatur UI.

Setiap pet direpresentasikan sebagai sebuah objek yang menyimpan parameter seperti tingkat kelaparan, kebahagiaan, kondisi emosional, dan status highlight ketika dipilih oleh pengguna.

Sensor getaran digunakan sebagai mekanisme interaksi berbasis gerakan yang memungkinkan pengguna memberikan “feeding” pada pet yang sedang dipilih. Sementara itu, push button digunakan untuk navigasi antar-pet. Animasi sprite, ikon, dan progress bar

dirender pada OLED menggunakan library GFX yang mendukung tampilan grafis ringan pada mikrokontroler.

Struktur firmware disusun secara modular, memungkinkan penambahan fitur seperti komunikasi mesh melalui *MeshController.hpp*. Dengan arsitektur yang fleksibel ini, TomoStroke dapat diperluas untuk mendukung interaksi multi-perangkat di masa mendatang.

2.3 HARDWARE AND SOFTWARE INTEGRATION

Integrasi perangkat keras dan perangkat lunak pada TomoStroke dilakukan untuk memastikan seluruh komponen sistem bekerja secara selaras dalam lingkungan real-time. ESP32 berfungsi sebagai pusat pemrosesan yang menghubungkan OLED display, sensor getaran, tombol input, dan buzzer dengan logika perangkat lunak berbasis FreeRTOS. MainLoop Task mengelola pembaruan state pet serta membaca input tombol dan sensor, sementara VoiceTask menangani pemutaran audio melalui *queue* agar tidak mengganggu proses utama.

OLED ditangani melalui library Adafruit GFX, yang menerima data sprite dan indikator dari objek pet untuk divisualisasikan secara konsisten pada layar. Sensor getaran diintegrasikan untuk mendeteksi aksi feeding, dengan kalibrasi sensitivitas agar hanya gerakan yang disengaja yang diproses sebagai input valid. Tombol navigasi terhubung ke logika pemilihan pet dan diproses melalui mekanisme debounce.

Selain itu, sistem mesh networking disiapkan melalui modul Mesh Controller, yang mengelola pertukaran informasi antar-perangkat menggunakan format pesan ringan. Integrasi antara rendering, input sensor, audio, dan komunikasi mesh memastikan TomoStroke dapat beroperasi secara stabil, responsif, dan siap dikembangkan menjadi sistem multi-perangkat. Dengan arsitektur ini, perangkat keras dan perangkat lunak dapat bekerja bersama secara efisien untuk mendukung interaksi virtual pet secara real-time.

CHAPTER 3

TESTING AND ANALYSIS

3.1 TESTING

Pengujian dilakukan untuk memverifikasi kebenaran fungsi, kestabilan, dan tingkat responsivitas sistem TomoStroke selama beroperasi. Proses pengujian mencakup beberapa aspek utama, mulai dari input tombol untuk memastikan pemilihan pet berjalan dengan benar dan mekanisme debounce bekerja stabil, hingga pengujian sensor getaran untuk memastikan aksi feeding hanya terjadi ketika getaran yang disengaja terdeteksi. Tampilan OLED diuji untuk memastikan proses rendering berjalan mulus pada sekitar 25 FPS tanpa flicker atau artefak visual. Selain itu, pengujian dilakukan pada penjadwalan task FreeRTOS untuk memastikan tidak terjadi blocking dan seluruh task dapat berjalan secara paralel dengan baik. Perilaku state pet seperti perubahan hunger, happiness, dan ekspresi wajah diamati untuk memastikan transisi state berlangsung sesuai logika internal. Terakhir, buzzer diuji untuk

memastikan pola suara dapat diputar dengan benar tanpa mengganggu performa sistem lainnya.

3.1.1 Unit Testing

Unit testing dilakukan pada setiap komponen perangkat lunak untuk memastikan bahwa fungsi dasar sistem bekerja sesuai spesifikasi sebelum seluruh modul diintegrasikan. Pengujian dimulai dari **Pet class**, di mana setiap atribut seperti *hunger*, *happiness*, *blink interval*, dan *emotion state* diuji secara individual untuk memastikan perubahan state mengikuti logika internal yang telah ditetapkan. Fungsi-fungsi seperti `feed()`, `update()`, dan `setHighlight()` diuji dengan berbagai input untuk melihat apakah keluaran yang dihasilkan sesuai dengan perilaku yang diharapkan.

Pada **input button**, pengujian dilakukan dengan membaca nilai HIGH dan LOW dari GPIO untuk memastikan mekanisme debounce berjalan dengan benar dan tidak terjadi input ganda. **Sensor getaran** diuji dengan berbagai tingkat getaran untuk memastikan bahwa sensor hanya memberikan output valid ketika getaran yang cukup kuat terdeteksi, sehingga tidak sensitif terhadap noise kecil.

Untuk **buzzer dan VoiceTask**, fungsi `tone()` diuji dengan berbagai frekuensi dan durasi. Selain itu, pengiriman perintah suara melalui FreeRTOS queue diuji untuk memastikan bahwa sistem antrian bekerja tanpa kehilangan pesan. **OLED rendering** diuji dengan memastikan bahwa fungsi-fungsi dasar seperti menggambar sprite, ikon, progress bar, dan melakukan refresh layar dapat bekerja tanpa error.

3.1.2 Integration Testing

Integration testing dilakukan untuk memastikan setiap modul dalam sistem dapat bekerja dengan baik ketika dihubungkan satu sama lain. Pengujian dimulai dengan mengintegrasikan *Pet class* dengan OLED display, memastikan bahwa perubahan state seperti *hunger*, *happiness*, dan ekspresi wajah langsung muncul pada tampilan tanpa keterlambatan. Sensor getaran dan tombol navigasi kemudian diuji bersama MainLoop Task untuk memastikan feeding dan perpindahan antar-pet dapat dilakukan dengan stabil selama sistem berjalan.

Selanjutnya, integrasi VoiceTask dengan sistem antrian (queue) diuji untuk memastikan pemutaran suara pada buzzer tidak mengganggu proses rendering maupun

pembaruan state pet. Pengujian multitasking FreeRTOS menunjukkan bahwa seluruh task seperti rendering, input handling, dan audio processing dapat berjalan secara paralel tanpa blocking. Secara keseluruhan, seluruh modul perangkat keras dan perangkat lunak berhasil berfungsi dengan baik ketika diintegrasikan.

3.1.3 System Testing

System testing dilakukan untuk memastikan bahwa seluruh sistem TomoStroke dapat beroperasi dengan stabil saat dijalankan dalam kondisi normal. Pengujian mencakup evaluasi performa tampilan OLED, yang menunjukkan bahwa rendering animasi dapat berjalan pada sekitar 25 FPS tanpa flicker atau penurunan kualitas. Responsivitas sistem juga diuji dengan melakukan navigasi antar-pet serta feeding menggunakan sensor getaran, dan seluruh fungsi berjalan lancar tanpa delay.

Selain itu, sistem multitasking FreeRTOS diuji untuk memastikan tidak terjadi blocking antar-task. MainLoop Task, VoiceTask, dan pembaruan state pet dapat berjalan secara paralel tanpa mengganggu satu sama lain. Perubahan state pet seperti hunger, happiness, dan ekspresi wajah juga diamati dalam jangka waktu tertentu untuk memastikan perilaku sistem sesuai logika yang telah ditentukan. Secara keseluruhan, hasil system testing menunjukkan bahwa perangkat berfungsi stabil, responsif, dan konsisten selama operasi.

3.1.4 Field Testing

Tahap akhir pengujian dilakukan di lingkungan nyata untuk menilai performa TomoStroke saat digunakan langsung oleh pengguna. Fokus utama pengujian lapangan ini adalah memastikan bahwa seluruh interaksi fisik terutama feeding melalui sensor getaran dapat bekerja sesuai kondisi penggunaan sehari-hari. Sensitivitas sensor getaran dikalibrasi secara iteratif untuk menemukan nilai yang paling stabil, sehingga feeding hanya terpicu oleh gerakan yang disengaja dan tidak dipengaruhi oleh getaran kecil atau noise dari lingkungan sekitar.

Selain itu, respons navigasi pet melalui tombol diuji dalam berbagai kondisi untuk memastikan perpindahan antar-pet tetap akurat tanpa input ganda. Tampilan OLED juga diamati dalam kondisi pencahayaan berbeda untuk memastikan sprite, ikon, dan indikator bar tetap jelas dan mudah dibaca. Pengujian lapangan ini memperlihatkan bagaimana task-task FreeRTOS bekerja secara real-time selama perangkat dijalankan dalam sesi penggunaan yang

lebih panjang, sehingga dapat diamati konsistensi pembaruan state hunger dan happiness serta stabilitas rendering tampilan. Secara keseluruhan, hasil pengujian lapangan menunjukkan bahwa TomoStroke mampu berfungsi dengan responsif dan stabil dalam kondisi penggunaan sebenarnya.

3.2 RESULT

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi ornare accumsan nisl sit amet sodales. Suspendisse sed dictum velit, in suscipit sem. Vestibulum egestas neque vel velit tristique, id venenatis nunc fringilla. Mauris condimentum diam consequat egestas tincidunt. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Vivamus semper pharetra commodo. Integer hendrerit ultricies lacus. Nullam id magna sed risus placerat luctus sed at mauris. Curabitur ligula urna, pharetra eget mi sit amet, sagittis feugiat magna. Curabitur ex nisl, eleifend et mattis sit amet, condimentum nonnisi.

Donec at iaculis leo. Integer congue sed lacus suscipit iaculis. Nulla a augue ut sapien rutrum consectetur. Sed ac dignissim lorem. Maecenas hendrerit nisl a metus posuere, vel vehicula metus eleifend. Mauris blandit, dolor nec malesuada tempor, purus nibh aliquet nibh, at faucibus leo felis a nisi. Donec pharetra leo risus, in vestibulum dui laoreet in. Nulla facilisi. Etiam nec consequat justo. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam erat volutpat. Etiam pharetra eleifend hendrerit.

3.3 EVALUATION

Evaluasi sistem menunjukkan bahwa TomoStroke telah mampu memenuhi sebagian besar kriteria keberhasilan yang ditetapkan. Sistem berjalan stabil pada ESP32, dengan FreeRTOS yang efektif menangani beberapa task secara paralel tanpa menimbulkan blocking atau penurunan performa. Mekanisme antrian untuk pengiriman suara juga terbukti mampu mencegah konflik eksekusi antara tampilan OLED, input sensor, dan pemutar audio.

Dari sisi interaksi, respons tombol cukup konsisten setelah penerapan debounce, sementara sensor getaran dapat mendeteksi aksi feeding dengan akurat meskipun masih sensitif terhadap getaran eksternal. Hal ini menunjukkan perlunya perbaikan lebih lanjut pada kalibrasi atau penggunaan filter untuk meningkatkan akurasi input.

Visualisasi pada OLED berjalan baik, namun keterbatasan ukuran layar mengharuskan desain UI yang efisien agar informasi tetap mudah dibaca. Animasi sprite dan

indikator status dapat tampil dengan lancar, tetapi refresh rate turun sedikit ketika banyak objek di gambar secara bersamaan.

Secara keseluruhan, sistem telah menunjukkan performa yang baik dan sesuai dengan tujuan utama, namun masih terdapat ruang pengembangan seperti peningkatan stabilitas sensor, optimalisasi rendering, serta integrasi fitur mesh networking yang sudah disiapkan dalam struktur program.

CHAPTER 4

CONCLUSION

Sistem TomoStroke bekerja melalui koordinasi beberapa task FreeRTOS yang berjalan secara paralel untuk menangani logika pet, tampilan, input sensor, audio, dan komunikasi antar-perangkat. **MainLoop Task** menjadi pusat operasi dengan membaca input tombol dan sensor getaran, memperbarui state setiap pet (hunger, happiness, blink, movement), serta merender sprite dan indikator ke OLED display secara real-time. Task ini dirancang non-blocking sehingga tampilan tetap stabil pada frame rate yang konsisten.

Selain itu, **VoiceTask** memproses permintaan suara yang dikirim melalui *voiceQueue*, menghasilkan pola nada unik untuk setiap pet tanpa mengganggu proses utama. Sensor getaran digunakan sebagai input feeding, dan ambang sensitivitasnya disesuaikan agar hanya getaran yang disengaja yang memicu aksi tersebut. Tombol navigasi mengatur perpindahan antar-pet dan diproses dengan mekanisme debounce untuk menjaga responsivitas.

Sistem juga dilengkapi **Mesh Controller** yang menangani broadcast informasi pet dan pesan feeding antar-perangkat. Modul ini membaca dan memperbarui daftar pet eksternal

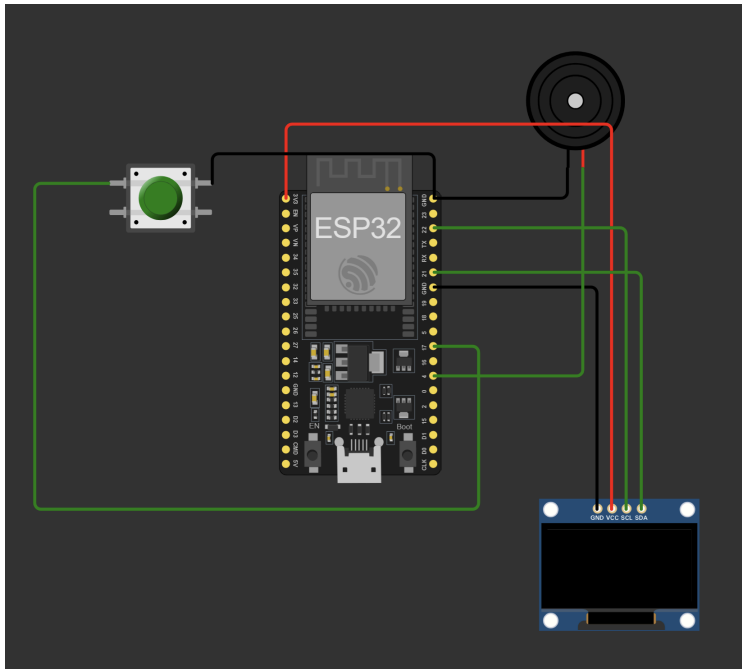
berdasarkan MAC address dan TTL, serta memastikan interaksi jarak jauh dapat dilakukan secara ringan dan efisien. Dengan pembagian tugas ini, seluruh komponen mulai dari rendering, input, audio, hingga komunikasi dapat berjalan bersamaan dengan stabil, sehingga menghasilkan sistem virtual pet yang responsif dan siap dikembangkan lebih lanjut.

REFERENCES

- [1] “Internet of Things | Digilab UI,” *Digilabdte.com*, 2018.
<https://learn.digilabdte.com/books/internet-of-things>
- [2] “Technical Documents | Espressif Systems,” *www.espressif.com*.
<https://www.espressif.com/en/support/documents/technical-documents>
- [3] “Free RTOS Book and Reference Manual,” *FreeRTOS*.
https://www.freertos.org/Documentation/RTOS_book.html
- [4] Adafruit, “Adafruit GFX Library,” *GitHub*, Oct. 14, 2021.
<https://github.com/adafruit/Adafruit-GFX-Library>
- [5] “ESP-NOW - ESP32 - — ESP-IDF Programming Guide latest documentation,”
docs.espressif.com.
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_now.html

APPENDICES

Appendix A: Project Schematic



Appendix B: Documentation

