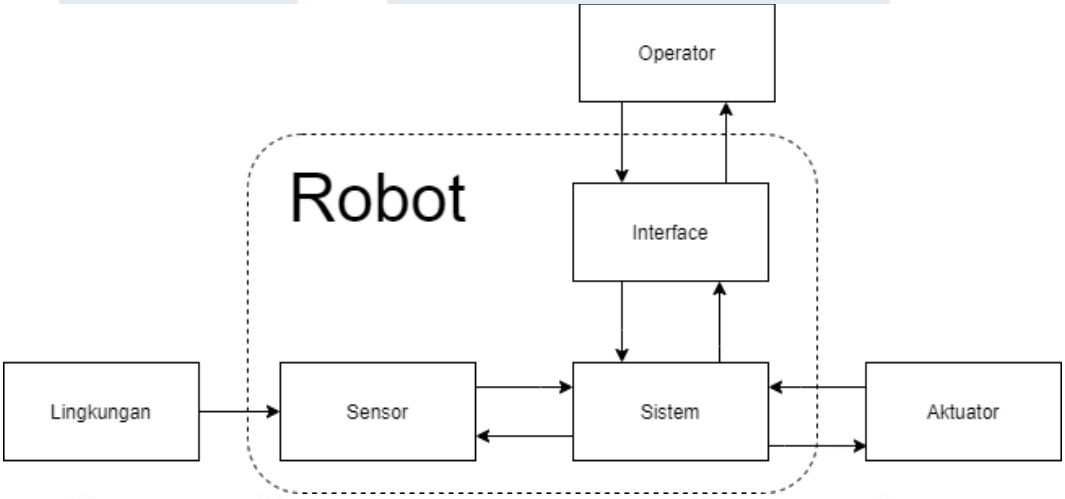
BAB III

PERANCANGAN DAN IMPLEMENTASI SISTEM

3.1 Tinjauan Desain Sistem

3.1.1 Desain Sistem Keseluruhan

Bagian ini membahas desain sistem secara keseluruhan. Desain ini menjelaskan mengenai hubungan antara sistem robot dengan lingkungan luar. Berikut merupakan *Data Flow Diagram* (DFD) *Level* 0 yang akan ditunjukkan pada Gambar 3.1



Gambar 3.1 - DFD Level 0 *Mobile Robot*

Tabel 3.1 - Penjelasan DFD Level 0 *Mobile Robot*

Parameter	Keterangan
Input	• Konfigurasi sistem dari pengguna. • Pembacaan sensor terhadap sistem pergerakan robot.
Output	• Pengendalian pergerakan aktuator.
Fungsi	• Mengatur pergerakan robot. • Mendapatkan data pembacaan sensor.

Berdasarkan tabel 3.1, *Mobile Robot* akan bergerak sesuai dengan *input* yang diberikan operator dan pembacaan sensor terhadap pergerakan dari robot. Serta memiliki fungsi untuk mengatur pergerakan robot berdasarkan konfigurasi yang diberikan oleh operator dan data sensor yang dibaca.

3.1.2 Desain Subsistem Lokomosi

Gambar 3.2 - DFD Level 1 *Mobile Robot*

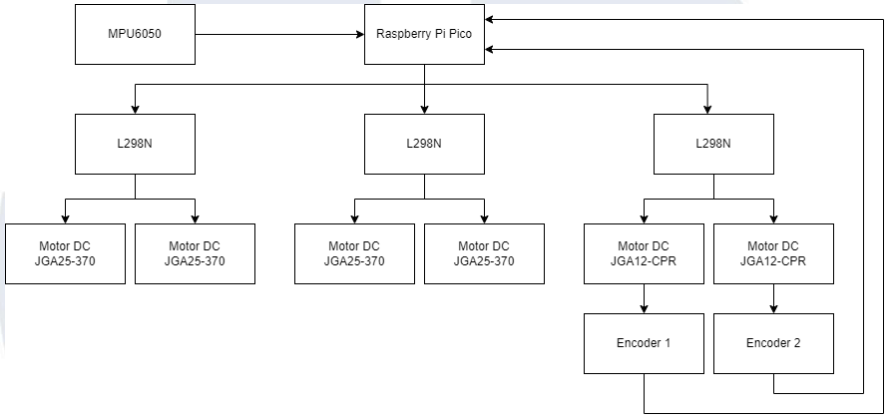
Tabel 3.2 - Penjelasan DFD Level 1 *Mobile Robot*

Parameter	Keterangan
Input	● Pembacaan data pergerakan oleh *encoder*.
Output	● Sistem kendali pergerakan actuator.
Fungsi	● Mengendalikan parameter pergerakan actuator. ● Memperbaiki arah maju. ● Mempertahankan kecepatan di permukaan yang berbeda. ● Mengatur robot untuk bergerak ke lokasi yang ditentukan dengan parameter yang diberikan.

Pada gambar 3.2 dan tabel 3.2, dapat diketahui bahwa sistem ini mencakup pembacaan data pergerakan oleh *encoder* dan data *yaw* serta *pitch* yang dibaca oleh MPU6050. Kedua data ini akan diolah oleh Raspberry Pi Pico yang merupakan *microcontroller* yang digunakan pada robot ini. Subsistem perbaikan arah maju memiliki fungsi untuk memperbaiki arah robot sehingga robot dapat berjalan dengan lurus dengan menggunakan data *yaw* dan pembacaan *encoder* sebagai acuan. Subsistem mempertahankan kecepatan memiliki fungsi untuk mempertahankan kecepatan ketika robot berjalan di permukaan yang tidak rata, sehingga pada permukaan yang landai, tanjakan dan

turunan, robot dapat menghasilkan kecepatan yang sama. Hubungan dari keseluruhan subsistem dapat dilihat pada Gambar 3.2 DFD level 1 *mobile robot*.

3.1.3 Diagram Subsitem Perbaikan Arah Maju dan Subsistem Mempertahankan Kecepatan



Gambar 3.3 - DFD Level 2 Subsistem Perbaikan Arah Maju dan Subsistem Mempertahankan Kecepatan

Tabel 3.3 - Penjelasan DFD Level 2 Subsistem Perbaikan Arah Maju dan Subsistem Mempertahankan Kecepatan

Parameter	Keterangan
Input	• Pembacaan data pergerakan motor DC menggunakan *encoder*. • Pembacaan data *yaw* dan *pitch* menggunakan MPU6050
Output	• Sistem kendali pergerakan *actuator*.
Fungsi	• Mengendalikan parameter pergerakan *actuator*. • Memperbaiki arah maju. • Mempertahankan kecepatan di permukaan yang berbeda. • Mengatur robot untuk bergerak ke lokasi yang ditentukan dengan parameter yang diberikan.

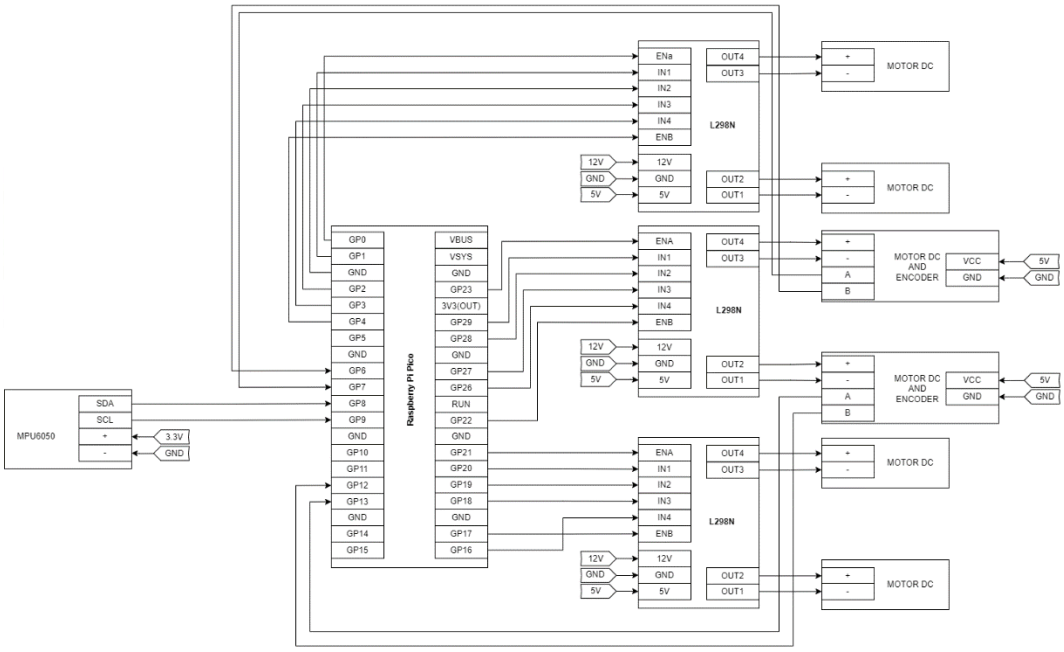
Dari tabel 3.3, dapat diketahui bahwa subsistem memiliki fungsi untuk mengatur dan mengendalikan pergerakan motor DC. Dengan menggunakan *encoder* sebagai *input* untuk membaca pergerakan motor DC, kecepatan motor

DC dapat diatur sehingga pergerakan yang dihasilkan oleh motor DC dapat membuat robot berjalan dengan stabil dan lurus.

Dari Gambar 3.3 dapat diketahui bahwa pada DFD ini terdapat dua jenis subsistem dengan menggunakan komponen yang sama. Serta dari tabel 3.3 dapat diketahui bahwa untuk memperbaiki arah robot dan membuat robot tersebut berjalan dengan lurus, perlu menggunakan dua sensor yaitu *encoder* dan MPU6050 (IMU). *Encoder* akan membaca pergerakan kedua motor sehingga dapat diketahui bahwa kecepatan kedua motor adalah sama, dan data *yaw* yang didapat menggunakan MPU6050 digunakan untuk mengetahui pembacaan sudut dari perjalanan robot, apabila MPU6050 memberikan data bahwa robot terlalu ke kanan, kecepatan motor di sisi kanan akan ditambah begitu pula sebaliknya. *Encoder* juga memiliki fungsi untuk mengetahui jarak yang di *set* oleh operator. Pada subsistem kedua, sama seperti subsistem sebelumnya yaitu dengan menggunakan MPU6050 sebagai sensor untuk acuan data. Namun pada subsistem ini, MPU6050 tidak akan membaca *yaw* melainkan akan membaca data *pitch* dari MPU6050, sehingga sistem dapat mengetahui apabila robot sedang berada di tanjakan, kecepatan motor akan ditambahkan, apabila robot berada di turunan, kecepatan motor akan diturunkan dan ketika robot berada di jalanan yang landai, robot akan mempertahankan kecepatan utamanya.

3.1.4 Diagram Subsitem Perbaikan Arah Maju dan Subsistem Mempertahankan Kecepatan

Gambar 3.4 merupakan *wiring diagram* untuk sistem secara keseluruhan



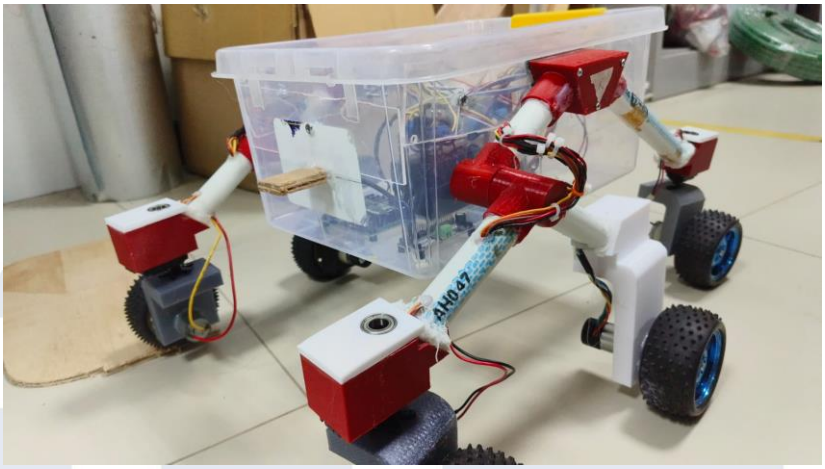
Gambar 3.4 - *Wiring Diagram*

3.2 Implementasi Sistem

3.2.1 Hasil Implementasi

a. Hasil Implementasi Desain Fisik

Mobile robot ini memiliki tiga bagian utama, yaitu badan utama, suspensi *Rocker-Bogie* dan *Link*. Sebagian besar komponen elektronik akan diletakkan di badan utama. Komponen yang diletakkan didalam badan utama, antara lain baterai lipo, *buck converter* 12V, 7V dan 5V, 3 driver L298N dan Raspberry Pi Pico yang sudah dipasang pada PCB. Di Bagian luar robot terdapat 6 motor DC, 2 diantaranya memiliki *encoder* yang terletak di tengah kanan dan tengah kiri robot, serta terdapat 4 *servo* yang diletakkan di depan dan dibelakang robot. *Servo* sendiri memiliki *support* yang memiliki fungsi untuk membantu perputaran roda menjadi lebih kuat. Tampak depan dari *mobile robot* dapat dilihat pada Gambar 3.5.



Gambar 3.5 - Tampak Depan *Mobile Robot*

Badan utama robot yang memiliki bentuk persegi panjang terbuat dari box plastik yang memiliki ketebalan 3mm. Box ini memiliki dimensi 20 cm x 32,5 cm x 20 cm. Bagian samping kanan dan samping kiri badan akan dipasangkan *mount* suspensi *Rocker-Bogie*, yang juga merupakan kabel-kabel listrik diletakkan. *Mount* akan dipasangkan menggunakan 3 buah baut M3 x 60 mm dan mur serta dilekatkan dengan menggunakan lem *adhesive* sehingga robot menjadi lebih rekat. Pada bagian *Rocker* yang merupakan bagian suspensi yang memiliki fleksibilitas akan dipasangkan satu sama lain dan dikuatkan menggunakan baut M3 x 60 mm yang akan diletakkan didalam *Rocker*. Bagian kaki-kaki terdapat mount *servo* untuk bagian depan dan belakang serta *mount* motor DC yang terletak di tengah dan yang terhubung dengan *servo* dibuat dengan menggunakan 3D *printer* dengan bahan PETG dan PLA+. Serta terdapat link dengan menggunakan pipa PVC sebagai bahan utamanya.

b. Hasil Implementasi Subsistem Perbaikan Arah Maju dan Subsistem Mempertahankan Kecepatan

Subsistem perbaikan arah maju memiliki tugas utama untuk menyelaraskan seluruh motor DC untuk memiliki kecepatan yang sama, dan membaca data sudut *yaw*, sehingga robot dapat berjalan dengan lurus dan stabil. Subsistem mempertahankan kecepatan hanya membaca data sudut *pitch*. Kedua subsistem ini dikendalikan oleh Raspberry Pi Pico yang diprogram menggunakan aplikasi Thonny IDE.

Untuk melakukan implementasi pergerakan motor DC, diperlukan 3 *driver* motor yang akan terhubung dengan motor DC. *Driver* motor yang digunakan adalah L298N yang akan dikendalikan oleh Raspberry Pi Pico. Pada Gambar 3.5, dijelaskan bahwa *driver* motor L298N memiliki 2 input tegangan, yaitu *input* 5V untuk memberi daya pada logika internal *driver* motor dan *input* 12V untuk memberi daya pada motor yang dapat diatur tegangan yang akan masuk ke dalam motor. Terdapat 2 tipe motor yang terpasang pada robot, yaitu JGA25-370 yang diletakkan didepan dan dibelakang robot dan JGA25-12CPR yang diletakkan di tengah robot. JGA25-12CPR dilengkapi dengan *encoder* yang memiliki input tegangan 5V. Pada *driver* motor, terdapat pin ENA dan ENB yang merupakan *input* PWM yang difungsikan untuk mengatur kecepatan motor, dan pin IN1 hingga IN4 merupakan pin untuk mengatur arah berputarnya motor. Serta pada implementasi ini akan menggunakan MPU6050 dengan tegangan masuk sebesar 3.3V. MPU6050 memiliki pin SDA dan SCL yang akan memberikan data pembacaan ke *microcontroller*. MPU6050 digunakan sebagai sensor yang mendeteksi sudut dengan menggunakan data *yaw* sehingga robot dapat memperbaiki arah robot. *Encoder* motor dan sensor MPU6050 akan dihubungkan langsung dengan Raspberry Pi Pico. Untuk implementasi subsistem perbaikan arah, *input* yang akan dipakai adalah data pembacaan *encoder* dan data *yaw* dari MPU6050, sedangkan pada subsistem

mempertahankan kecepatan, hanya memakai MPU6050 sebagai sensor dan mengambil data *pitch* untuk mengatur kecepatan pada motor.

3.2.2 Hambatan dan Solusi Implementasi

a. Hambatan dan Solusi Implementasi Desain Fisik

Hambatan yang terdapat pada implementasi desain fisik yaitu pada tahap ketika robot berjalan. *Badan* utama yang berbahan plastik membuat suspensi dan roda menjadi menyimpang atau terjadinya *chamber* sehingga menghasilkan gerakan robot yang tidak lurus. Sehingga ditambahkan layer akrilik didalam *badan* utama dan ditambahkan *support* kayu ditengah robot, sehingga *mount* suspensi tidak menyimpang dan menghasilkan robot yang dapat berjalan dengan lurus. Pada *badan* utama juga terdapat permasalahan pada saat pemasangan kabel, yang mana kabel yang terdapat didalam *badan* utama sangatlah banyak dan menjadi berantakan. Sehingga seluruh kabel tersebut dirapikan menggunakan alat *cable ties* dan *cable organizer* sehingga kabel di dalam robot menjadi lebih rapi dan terorganisir secara rapi.

b. Hambatan dan Solusi Implementasi Subsistem Perbaikan Arah Maju dan Subsistem Mempertahankan Kecepatan

Pada implementasi subsistem pergerakan motor, terdapat permasalahan yang ditemukan. Jenis motor yang dipakai pada robot ini memiliki spesifikasi yang berbeda-beda, yaitu 4 motor JGA25-370 dan 2 motor JGA25-12CPR. Hal ini menghasilkan dengan *input* PWM yang sama dapat memberikan perbedaan kecepatan yang sangat signifikan. Lalu dari 6 motor yang terletak pada subsistem ini, hanya 2 motor saja yang memiliki *encoder* sehingga pembacaan kecepatan antara motor yang berada di kiri dan kanan tidak lah sesuai. Meskipun hanya terdapat dua motor yang dilengkapi dengan *encoder*, pelaksanaan PID masih dapat dilakukan dengan menempatkan kedua motor tersebut di kedua sisi yang berbeda, sehingga informasi mengenai kecepatan sisi kanan dan kiri dapat diperoleh. Untuk mencapai kecepatan yang sama antar motor, dilakukan *trial and error* dengan menghitung berapa banyak putaran yang dihasilkan oleh roda dalam

jangka waktu tertentu. Hal ini diimplementasikan ke semua motor terlebih dahulu sebagai acuan dasar untuk mengetahui kecepatan yang diinginkan. Terdapat masalah lainnya yaitu *driver* motor ini menggunakan terlalu banyak pin pada Raspberry Pi Pico, sehingga mustahil untuk menerapkan PID pergerakan motor dengan PID perputaran *servo* dengan kurangnya pin yang terdapat pada Raspberry Pi Pico. Oleh karena itu implementasi PID untuk kedua subsistem dilakukan secara terpisah.