DRISHTI
A Revolutionary Concept

MAKERNOVA 2.0

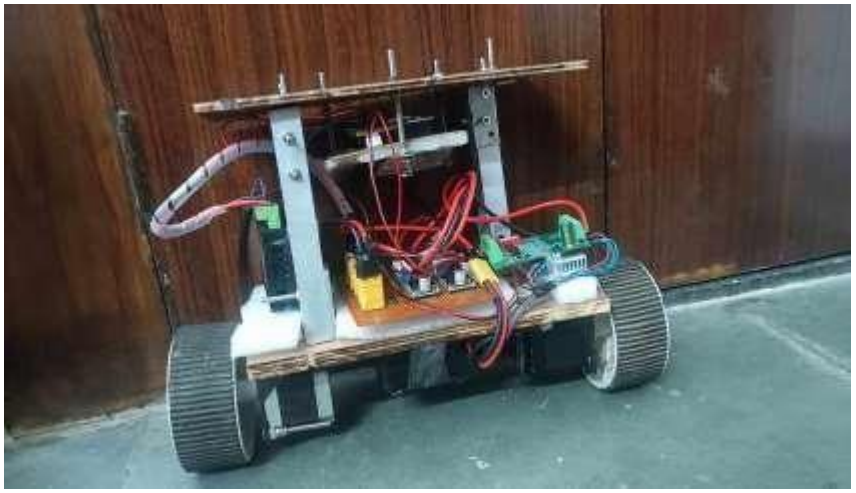
SELF BALANCING BOT

Team Members

Bhuvana Bolla
Chaitanya
Devaam Dalal
Jitesh Jain
Kriti Sati
Param Pandya
Parv Gupta
Sara Agalgaonkar
Sujay Bhati
Varshini
Vedika Tomar
Virendra Singh

Team Mentors

Dhruv Gandhi
Dipsi Hadiya
Lokesh Kumar



ACKNOWLEDGEMENT

Team Self balancing bot would like to thank Drishti for providing us with the platform where we can take our ideas from concepts to creations. We would also acknowledge the invaluable contributions of the mentors Mr. Dhruv Gandhi, Ms. Dipsi Hadiya and Mr. Lokesh in the development of self-balancing bot. Their expertise in the field of PID and algorithms played a crucial role in the making of the project. Their dedication, problem-solving skills, and innovative approach are very much appreciable.

INDEX

SR.NO	TITLE	PAGE NO.
1	Problem Statement	3
2	Abstract	4
3	Chassis Design	5
4	Components	6
5	Circuit Design	7
6	Hardware	8-13
7	Software	14
8	Communication Protocol	15-16
9	PID- controller	17-18
11	Flow Diagram	19
12	Problem Faced	20
13	Timeline	21
14	Bibliography	22

PROBLEM STATEMENT

Design and develop a self balancing robot capable of maintaining its upright position on two wheels without external support using PID Controller and other sensors.

ABSTRACT

This project focuses on the making of two wheeled Self-balancing bot using PID algorithms to maintain the balance by eliminating the possible errors in the direction of the motion. By utilizing advanced sensors particularly esp32 which comes with embedded Bluetooth and WIFI module and algorithms considering interrupt for accuracy, our bot is able to maintain its equilibrium and navigate various terrains. The bot balances itself autonomously. This project contributes to the idea of modern-day boost of transportation technology that is hoverboards and segways which uses the similar concepts and algorithms.

CHASSIS DESIGN

We decided to use a multilevel layout with all the components at the bottom, The IMU sensor in the middle and top floor is kept empty for extra load.

The middle floor, where the MPU6050 is located, would represent the bot's vertical center of gravity.

Course of action:

1) Using 2 pillars in a diagonal manner: we needed to decrease the weight of the bot.

2) Positioning of sensor: To get better measurements it was placed near center of mass.

3) Shortening of the bot: It was done to increase the stability and also decrease the torque needed.

Figure 1.1: front view of the bot

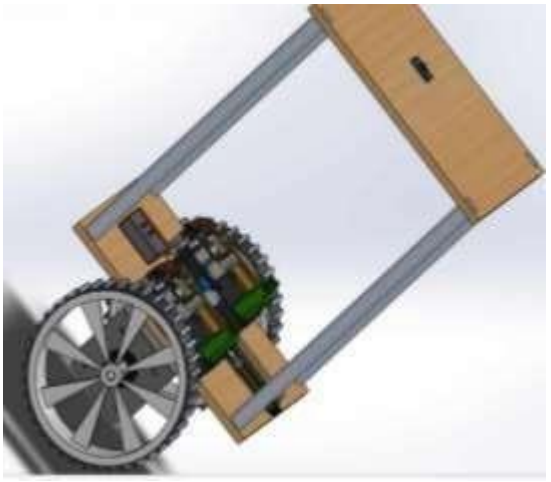


Figure1.2: side view of the bot

COMPONENTS

Sr.No.	Part Name	Description	Quantity
1	ESP-32 DEVKIT V1	ESP32 is a microcontroller, which is known for its robust features, including Wi-Fi and Bluetooth capabilities	1
2	TB-6600 STEPPER DRIVER	It enables independent control of the direction of each motor, crucial for the robot to rotate its wheels forward or backward to maintain balance.	2
3	IMU – MPU-6050	This sensor module typically combines an accelerometer and a gyroscope.	1
4	STEPPER Motors NEMA 23, 10Kg/cm	It is a brushless, synchronous electric motor that converts digital pulses into mechanical shaft rotation.	2
5	Wheels Shaft diameter 7 mm Wheel diameter 10 cm Wheel width 4cm	Two rubberized wheels are used to move robot frictionless and to avoid slipping.	2
6	BUCK CONVERTER LM 2596 S	DC-to-DC converter which decreases voltage, while increasing current, from its input to its output .	1
7	Battery (11.1 V) Lithium-ion 3S	It is used to give the power throughout the circuit.	1
8	Chassis	This is the main body or frame of the robot that holds all the other components	1

Table 1: Components

CIRCUIT DESIGNING

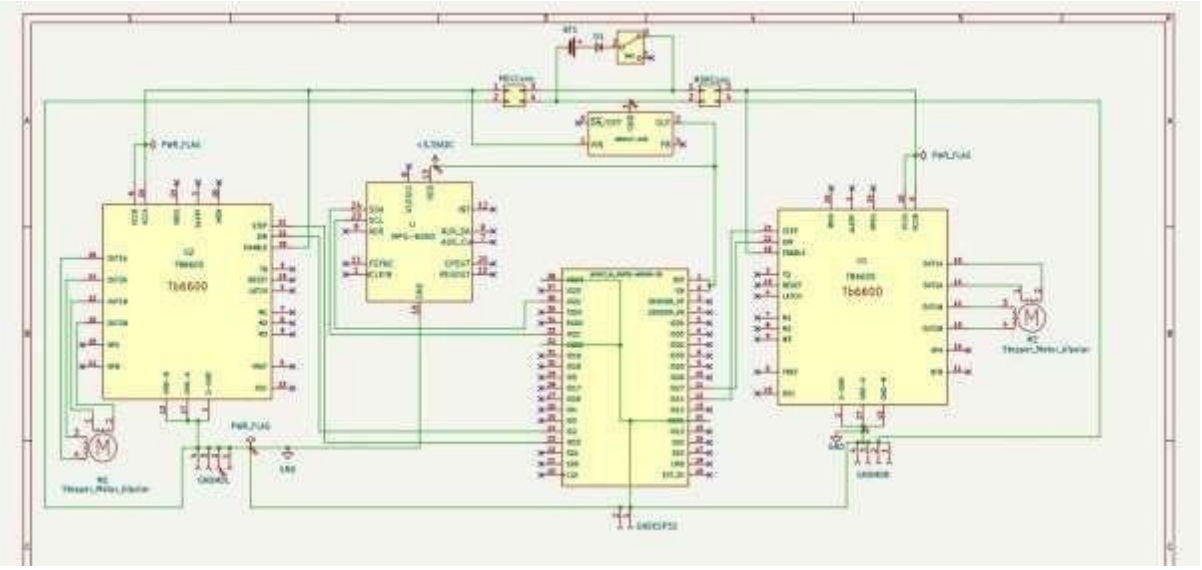


Figure 1.3: Circuit Diagram

The components present in this circuit are:

- A lithium ion battery which will provide power to the circuit.
- Three pairs of XT60 connectors to connect battery and motor driver.
- A diode to ensure the safety of the circuit if the polarity get changed.
- A Switch so we can manually turn OFF and ON the robot whenever we want.
- A buck converter which will convert 12V supply to 5V which later will be fed to ESP32 DEVKIT V1 and MPU6050.
- A ESP32 DEVKIT V1 microcontroller which will read sensor data and give command to motor driver.
- Two TB6600 Stepper Motor driver to control the direction of Stepper Motor with the help of microcontroller.
- Two Stepper Motor for high torque and high precision movement of robot.
- A MPU6050 which is a 6-axis gyroscopic sensor to measure the tilt angle of the robot.
- Two wheels which will be connected to the motor shaft and provide grip to the surface.

HARDWARE

ESP32 DEVKIT V1:

The ESP32 DevKit V1 is a popular development board based on the ESP32 microcontroller, which is known for its robust features, including Wi-Fi and Bluetooth capabilities. Developed by **Espressif Systems**, the ESP32 series is widely used for Internet of Things (IoT) projects, smart home automation, wearable electronics, and other embedded applications.

- The ESP32 has dual-core processor that runs up to 240MHz, providing significance processing power.
- It has clock cycle of 80MHz to 240MHz.
- It has flash memory of 4MB and SRAM of 520KB.
- It has total of 30 PIN out of which are
    i. 25 GPIO pins
    ii. 15 analog pins
    iii. a pair of I2C pins
    iv. a pair of GND
    v. two pair of UART pins
    vi. one Vin
    vii. one 3.3V pin
    viii. one enable.

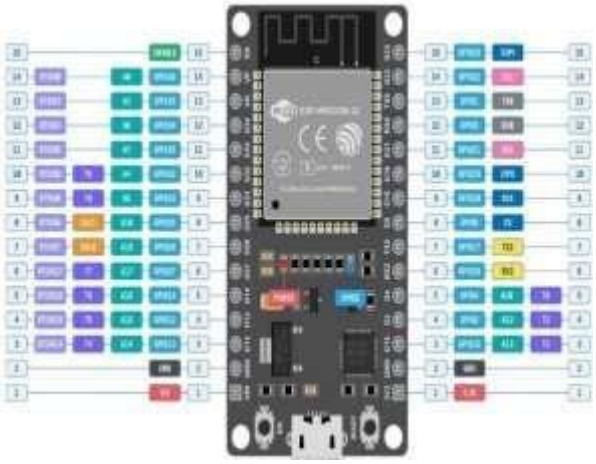


Figure 1.4: ESP-32 DEVKIT V1

Arduino Mega 2560:

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 .

It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove.

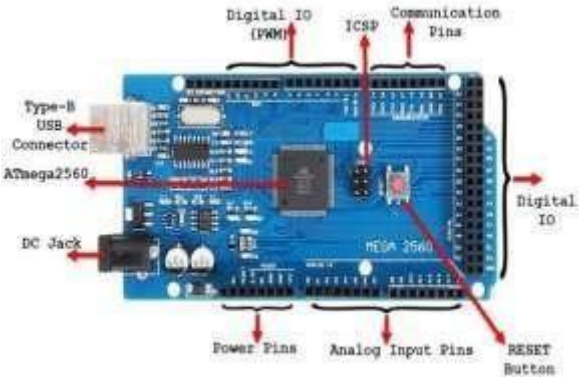


Figure 1.5: Arduino Mega 2560

Arduino Uno R3:

The Arduino UNO R3 is the perfect board to get familiar with electronics and coding. This versatile development board is equipped with the well-known ATmega328P and the ATMega 16U2 Processor. This board will give you a great first experience within the world of Arduino.

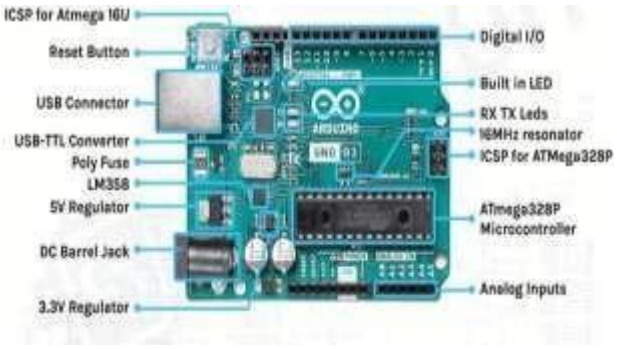


Figure 1.6: Arduino Uno R3

STM32 Microcontroller:

STM32 microcontroller is a 32 bit microcontroller developed by STMicroelectronics, it is based on ARM Cortex-M series of processor including (Cortex-M0,M0+,M3,M4 and M7). These microcontrollers are widely used in embedded system due to their high performance, low power consumption and extensive range of features.

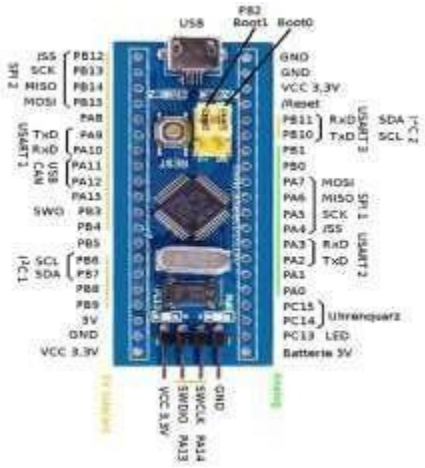


Figure 1.7: STM 32

- These microcontrollers have clock speed range from 24MHz to 480MHz.
- STM32 microcontrollers come in both single-core and dual-core configuration, depending on the specific series and model.
- STM32 microcontrollers have a variety of integrated peripherals, such as ADCs, DACs, timer, UARTs, I2C, SPI, CAN, USB, Ethernet, and more, to make them versatile for various embedded application.
- The flash memory size of an STM32 microcontrollers varies by model, and can range up to 16KB to 2MB. And SRAM can vary up to 4KB to 96KB.

We chose ESP32 over other microcontroller:

- It is cost efficient.
- It has embedded Bluetooth and WIFI module.
- It has more pins than of other three microcontroller.
- It has low power consumption.
- It has multiple I/O interfaces.

IMU Sensor (Inertial Measurement Unit):

IMU sensors, short for Inertial Measurement Units, are devices that combine multiple sensors—typically accelerometers, gyroscopes, and sometimes magnetometers—to measure a system's specific force, angular rate, and sometimes orientation. These sensors are crucial in various applications, including robotics, navigation, motion tracking, and stabilization systems.

IMU Sensor contains:

1. **Accelerometer:** Measures linear acceleration in multiple axes (usually 3). This data can be used to determine orientation relative to the Earth's gravitational field.
2. **Gyroscope:** Measures angular velocity (rotation rate) around each axis. This helps in tracking orientation changes over time.
3. **Magnetometer (In some IMU):** Measures the magnetic field strength in each direction, often used to correct the orientation drift in gyroscope data and provide absolute heading relative to the Earth's magnetic field.

IMU Sensors we Researched on:

1. **BNO055 Sensor :-** The BNO055 is a 9-axis sensor that integrates an accelerometer, gyroscope, and magnetometer with an onboard microcontroller that performs sensor fusion, providing direct orientation data like Euler angles and quaternions.
2. **MPU6050 Sensor :-** The MPU6050 is a 6-axis IMU sensor that combines a 3-axis accelerometer and a 3-axis gyroscope, commonly used for motion tracking and orientation detection in applications like drones and robotics.

BNO055:

The BNO055 is a sophisticated 9-axis Sensor from Bosch Sensortech that integrates an accelerometer, gyroscope, and magnetometer along with a 32-bit microcontroller to perform advanced sensor fusion internally.

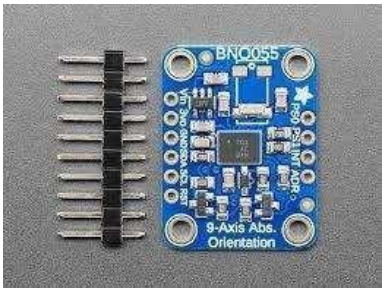


Figure 1.8: BNO055

General Specifications:

1. Power Supply Voltage: 2.4V to 3.6V.
2. Operating Current:
   - Normal mode: 12 mA (typical).
   - Low power mode: 4.7 mA.
   - Suspend mode: 0.5 mA.
3. Communication Interfaces: I2C (up to 400kHz), SPI (up to 10MHz).

MPU6050:

The MPU6050 is a motion tracking device that combines a 3-axis gyroscope and a 3-axis accelerometer to detect changes in motion, acceleration, and rotation. It is based on Micro Electronics Mechanical System (MEMS) technology.

Figure 1.9: MPU6050

- Power Supply Voltage: 2.375V to 3.46V.
- Operating Current: 3.9 mA (typical).
- Standby Current: 5 µA.
- Communication Interface: I2C (up to 400kHz), SPI (up to 1MHz).
- In order to use MPU6050 multiple things needs to be setup first.
- **PINOUT configuration**:
    - Vcc
    - GND
    - SCL
    - SDA

Here's how you should connect the MPU6050 to the ESP32:

VCC: Connect to the 3.3V pin on the ESP32.

GND: Connect to the GND pin on the ESP32.

SCL: Connect to the I2C Clock (SCL) pin on the ESP32 at GPIO 22.

SDA: Connect to the I2C Data (SDA) pin on the ESP32 at GPIO 21.

We are using MPU6050 instead of BNO055 in the Self Balancing Bot

- The MPU6050 provides raw accelerometer and gyroscope data, which allows for direct control over the sensor fusion process. This is crucial in self-balancing bots where fine-tuning the sensor fusion algorithm (such as a Kalman filter) can lead to better performance tailored to the specific dynamics of the bot.
- With the MPU6050, you can implement your own sensor fusion algorithm, giving you the flexibility to optimize the response time and filtering techniques according to the needs and requirements of the Self Balancing bot.
- The MPU6050 is generally less expensive than the BNO055, making it a more economical choice for projects like self-balancing bot.
- For the self-balancing bot, we primarily need fast and accurate accelerometer and gyroscope data, which the MPU6050 provides efficiently without the need for the additional features (like magnetometer) that the BNO055 offers.
- The MPU6050 is widely used in the robotics community, especially in self-balancing bots.

Stepper Motor:

The stepper motor provides precision, in our project this is our main concern, hence we're using it. It is supplied with dc electricity in controlled sequence to cause rotation. Rotation can be in forward direction, reverse or even in steps, it can hold its position too.

Figure 1.10: Stepper motor

Main components:

- Shaft - attached with rotor, rotates with it, 2 bearings are attached at either end to provide smoothness.
- Rotor - rotor is made of 2 permanent magnets with polarity north and south. Teeth are carved in each, teeth of one magnet aligns with gap of the other.
- Stator - Surrounds rotor and remains stationary, it consists of windings which are not connected together but in 2 groups of 4. There are teeth surrounding inner parameter of the stator, it provides alignment.

The motor is connected to motor driver which is basically a high speed switching system which is connected to microcontroller which controls this switching action. Switches allow electricity to flow in pulses through coil. The speed of switching action and the order determines rotational speed and direction.

Motor Driver:

A stepper motor driver is a circuit that controls the operation of a stepper motor, managing the current and voltage supplied to the motor's coils to ensure precise movement. The driver translates low-power control signals (from a microcontroller, for example) into the higher power required by the motor.

TB6600:

- **Current Capacity**: Up to 4.5A per coil.
- **Micro stepping**: Up to 1/32nd micro stepping.
- **Supply Voltage**: 9V to 42V.

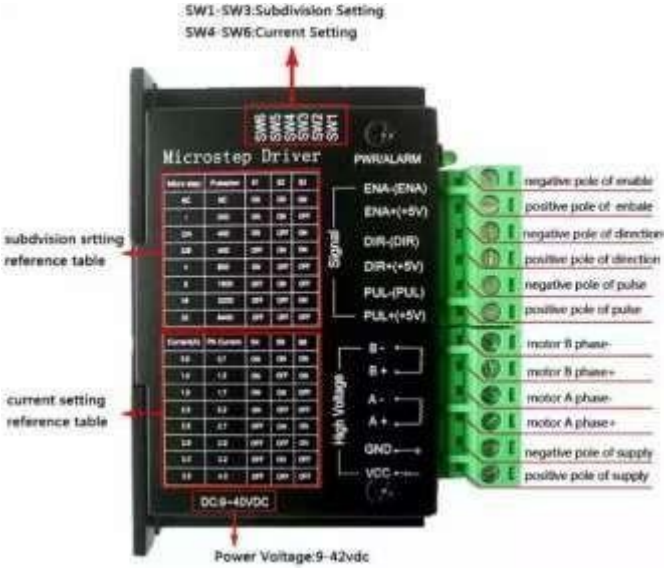


Figure 1.11: TB-6600 Stepper Driver

We used TB6600 stepper motor driver for our self balancing bot, because of high current capability, smooth microstepping, stable operation and many more features which collectively contribute to better performance in maintaining the balance and control of the bot.

Pinout configuration:

ENA-(ENA) - Enable pin(-)
ENA+(5V) - Enable(+5V)
DIR-(DIR) - Direction(-)
DIR+(+5V) - Direction(+5V)
PUL-(PUL) - Pulse(-)
PUL+(+5V) - Pulse(+5V)

B- - Stepper motor 1 coil wire
B+ - Stepper motor 1 coil wire
A- - Stepper motor 2 coil wire
A+ - Stepper motor 2 coil wire
GND - Ground
VCC - Input Voltage (9-40V)

Advantages:

Precise position control: Stepper motors divide a full rotation into a large number of equal steps, which gives precise control to the angle of rotation. This makes them ideal for applications requiring accurate positioning.

Open loop control: Unlike servo motors, which require feedback sensors, stepper motors can typically operate in an open- loop system, where you can control them without needing a position sensor.

Torque at low speeds: Stepper motors gives high torque at low speeds, making them ideal for applications requiring precise control and holding torque without needing high rotational speeds.

Micro-stepping capability: Modern stepper drivers gives micro-stepping, where the motor is moved in fraction of a step.

Lithium – ion Battery:

It is a type of rechargeable battery commonly used in electronics items, electric vehicles, and many other applications due to high energy density, light weight, and long cycle life.

Figure 1.12: Lithium – ion Battery

Advantages

- High efficiency and fast charging.
- Longer lifespan compared to other rechargeable batteries.
- High voltage output.
- Lightweight and compact.

Disadvantages:

- Sensitive to high temperature.
- Risk of Fire and Explosion.
- They are more expensive than some other battery types due to the cost of material .

SOFTWARE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

For programming the microcontroller there are multiple IDEs (Integrated Development Environment) which depends on the microcontroller used. In this project we will be using Arduino IDE for programming the ESP32 microcontroller.

ESP can also be programmed through its own IDE that is ESP IDF developed by espressif.

But better and easy way to program ESP is to use Arduino IDE as there is a wealth of tutorials, code examples, and community support available, making it easier to get started and troubleshoot issues.

For updating code in ESP refer the following steps:

1. Connect the ESP32 to the system.
2. Select your ESP32 board from: Tools > Board menu>Esp32 Dev Module.
3. Select the COM from: Tools >Port (you can check the COM from: device manager>ports.

In this project we would be using several libraries:

MPU6050 Library:

1. Go to git hub and search for MPU6050 library.
2. Download it as a zip file.
3. Sketch->include libraries->Add .zip file.

For downloading Stepper Standard library:

Standard libraries are already present in the IDE package but are needed to be installed to use Sketch->include libraries->manage libraries->search AccelStepper-> download AccelStepper.

COMMUNICATION PROTOCOL

I2C protocol:

Inter Integrated circuit bus protocol or TWI (Two wire interface). It is a multi-master multi-slave configuration type of communication protocol that means we can connected multiple master and slave. Supports up to 128 Devices.

It works in half duplex mode ,a half-duplex system is a communication system that allows data transmission in both directions from master to slave or slave to master, but not at the same time. In other words, a half-duplex system can either send or receive data at any given time.

Full duplex mode allows for simultaneous data transmission in both directions. This is similar to a two-lane road, where vehicles can move in both directions simultaneously without waiting.

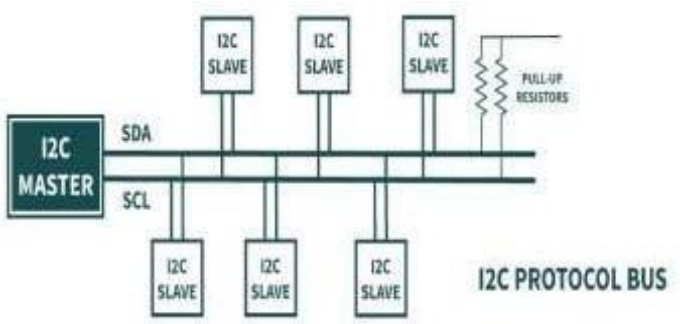


Figure 1.13: I2C protocol communication

It requires only 2 lines to interface peripherals and devices:

1. SDA –for transmitting data.
2. SCL- for clock pulse.

SPI Communication protocol:

SPI (Serial Peripheral Interface) is a synchronous serial communication protocol used for short-distance communication, primarily in embedded systems. It enables communication between a master device and one or more slave devices using separate data and clock lines.

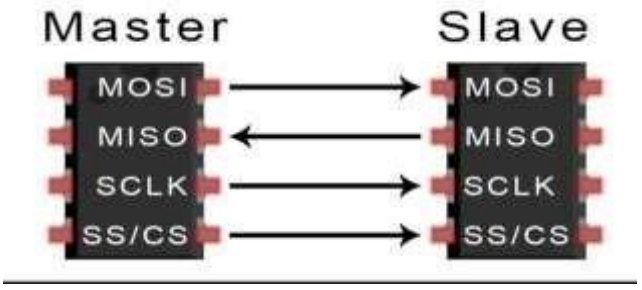


Figure 1.14: SPI protocol

Overview:

- **Master and Slave:** SPI operates with a master-slave architecture. The master device controls the communication, while the slave devices respond to the master's commands.
- **Clock:** SPI uses a clock signal (SCK) generated by the master device to synchronize-data transmission.
- **Data Lines:** There are typically four lines:
- **PIN OUT SUMMARY:**
    - **MOSI**
    - **MISO**
    - **SCK**
    - **SS/CS**

We use I2C over SPI:

I2C protocol has less wiring compared to SPI protocol without complicating the circuit and I2C protocol has the flexibility to connect multiple devices on the same bus without separate chip selection i.e. Multi master and Multi slave capability with unique addressing. And I2C has an advantage of Clock synchronization that required for bot. I2C Protocol contains error detection mechanism. I2C supports different speed modes. I2C has more Compatibility. And also for sensors like MPU6050 with low-power and limited number of GPIO pins I2C is more preferred.

PID CONTROLLER

The PID Controller mainly used to maintain balance, controlling motor speed, and for smooth and stable operation of the bot. The three controllers i.e.- proportional(P), integral(I), derivative(D) are used to detect the error and to correct the error. We use proportional control to measure the angle of tilt of the robot and produce the corrective force proportional to the error. Integral components sum up past errors over time. The derivative part predicts the future trend of the error by measuring its rate of change. The Pid controller adjusts the speed of the motors driving the wheels to keep the bot balanced.

Proportional term:

The proportional term attempts to drive the position error to zero by contributing to the control signal proportionally to the current position error.

U(t)=Kp*e(t); Kp=proportional gain

Transfer Function, C(s)= Kp

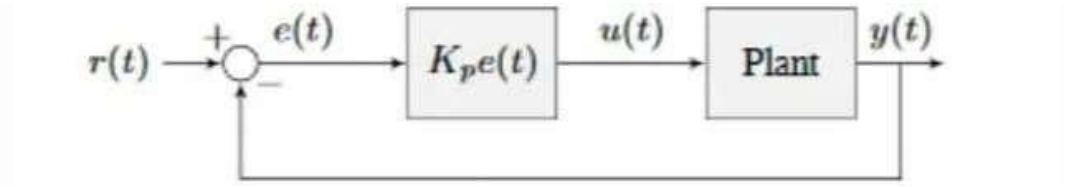


Figure 1.15: P control block diagram

Proportional gain acts like a “software-defined spring” that pulls the system toward a desired position. It tries to move the output towards the reference.

Derivative term:

The Derivative term attempts to drive the derivative of the error to zero by contributing to the control signal proportionally to the derivative of the error.

U(t)= Kp*e(t) +Kd*de/dt; Kd=derivative gain

Transfer Function, C(s)= Kp +Kd.s

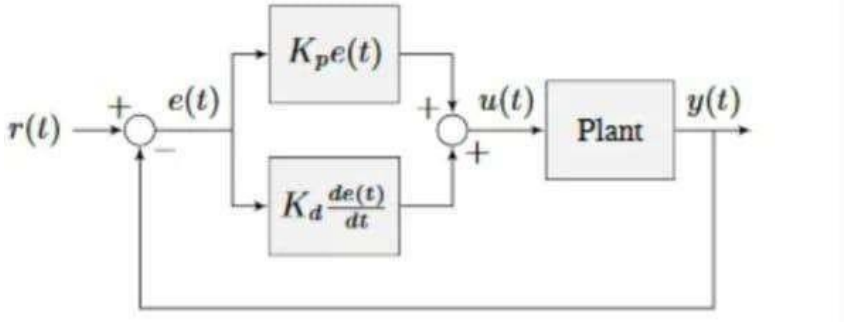


Figure 1.16: PD Control Block Diagram

It tries to make the output move at the same rate as the reference. The PID controller remains a fundamental tool in control system due to its simplicity and effectiveness. By combining proportional integral and derivative actions, PID controllers achieve robust performance across various application. Despite their effectiveness,PID controller face challenges such as handling nonlinearities PID Controller.

Implementation:

PID controllers can be implemented in various ways:

- Analog Implementation: Analog PID controllers use electronic components such as operational amplifiers to achieve PID control.
- Digital Implementation: Digital PID controllers use microcontrollers or digital signal processors (DSPs) to compute the PID algorithm. They provide greater flexibility and ease of adjustment through software and are widely used in modern control systems.
- Software Implementation: In software, PID algorithms are implemented using discrete-time approximations of the continuous-time equations.
- A system driven by a PID controller generally has three types of responses:-

  1) underdamped.

  2) critically damped.

  3) over-damped.

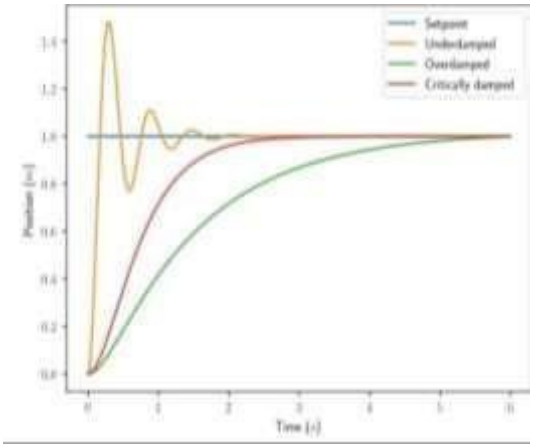


Figure 1.17: PID damping

Advantages:

1) Simplicity and Ease of implementation.
2) straightforward Design.
3) Wide Adaptation.
4) Versatility.
5) Broad Range of applications.
6) Adaptability.
7) Effective Control Performance.
8) Error Reduction.
9) Improved stability.
10) Real-Time Operation.
11) Ease of Tuning.

FLOW DIAGRAM

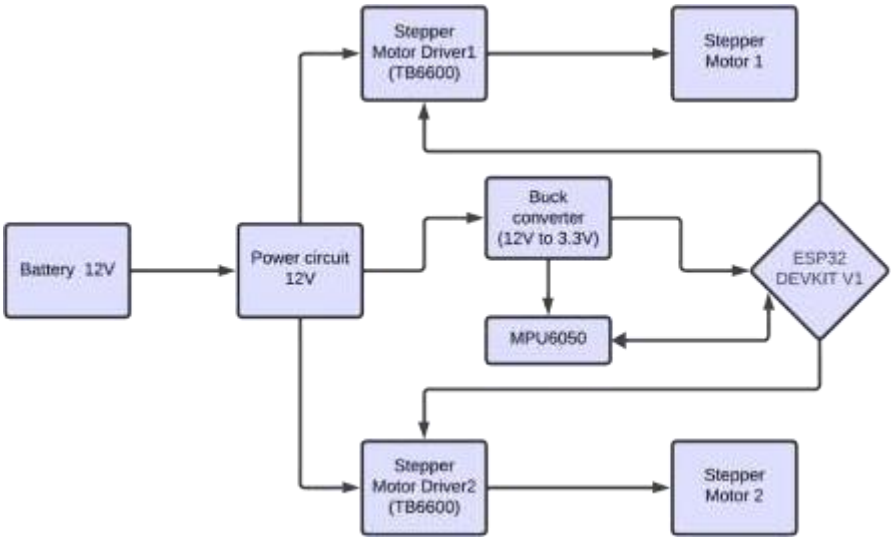


Figure 1.18: Overall Flow Diagram

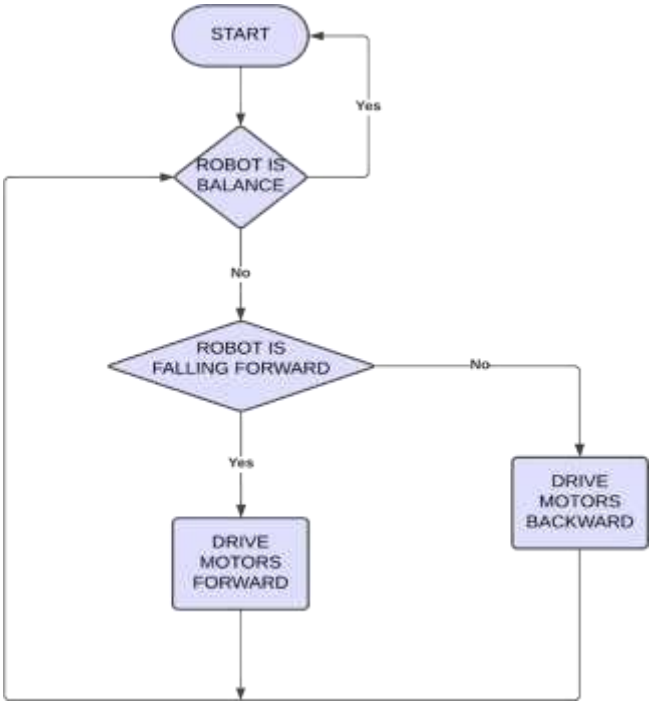


Figure 1.19: Balancing flow chart

PROBLEM FACED

Problem: After downloading the IDE and connecting the esp32 board to laptops , the port wasn't being detected.
Solution: We found out regarding the driver and installation of the driver solved the issue.

Steps to install driver:

1. Install CP210x Universal Windows Driver
2. Go to downloads and from CP210x file extract all items
3. Connect esp32 to laptop
4. Go to Device Manager and you'll find the installed driver under Ports
5. Right click it and update driver, Browse to downloads -- CP210x folder and select it
6. Update driver

Problem: Motors were not responding so a misconception was all 5kg motors were defective.
Solution: Analysed and observed we mistakenly used the wrong coils in the 6 pin steppermotor. More specifically 1-3,4-6 instead of 1-4,3-6.

Problem: ESP32 was not working correctly on 3.3V.
Solution: We increased the input voltage from 3.3V to 5V using Buck Converter LM-2956-S.

Problem: Our PD was not tuning properly and the mass was not align of chassis.
Solution: We did some modificatons in chassis like we shortend the height of the bot andreduce the width of base**.**

TIMELINE

(1Aug to 10Aug)

Research on components

We have researched various components and selected the most suitable one

(11Aug to 15Aug)

Chassis design

we have decided measurement of chassis to arranged components such that centre of gravity is maintained

(15Aug to 16Aug)

Power circuit

we have connected mpu6-50, motor driver, esp32 using i2c protocol based on supply provided

(17Aug to 19Aug)

libraries &code

Researched on different libraries, and implemented the code according to circuit connections

(20Aug to 2Sep)

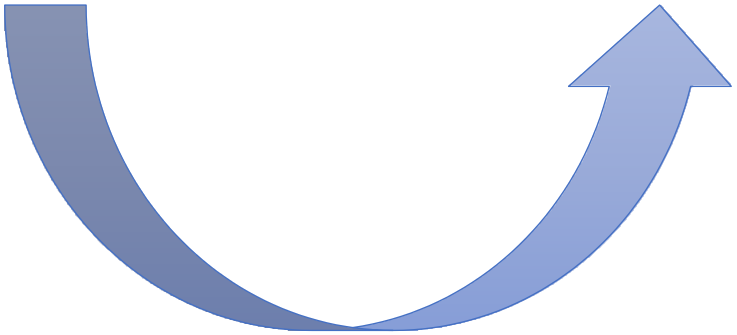
Bot Operation

we have created bot ,and tested the bot by implementing the code

(3Aug to 14Sep)

Remodify & Simplification

we have decreased the height and width of chassis to make balance, and we made few corrections



BIBLIOGRAPHY

For compiling the project, we have taken help from the following websites:

- **For MPU6050:** components101.com.
- **For BNO055:** cdn-learn.adafruit.com.
- **For ESP32:** circuitstate.com.
- **For Self balancing bot:** flyrobo.in, Circuitdigest.com.
- **For Stepper driver:** handsontec.com.
- **For Arduino uno and Mega:** arduino.cc
- **For STM32:** theengineeringprojects.com

Name of articles we referred:

- Instructable – Arduino Self Balancing Robot
- PLOS ONE – self- Balancing Robot
- Research paper – KTH Two Wheeled Self Balancing Bot