Electric Robot Design Workshop

Ferdinand Sellin

INTech

fsellin@telecom-sudparis.eu

3 October , 2024

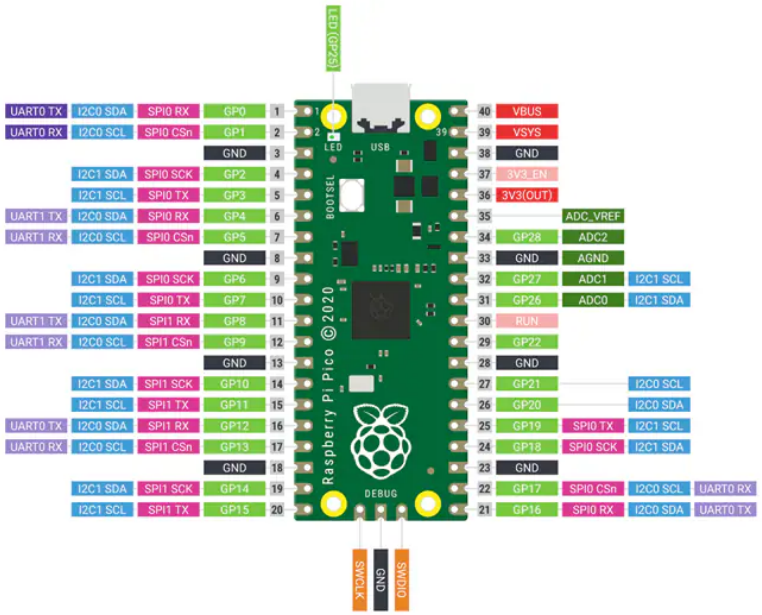
Outline

Find the files

GitHub Repository Link

Outline

Raspberry Hardware



(a) Raspberry Pico

(b) Raspberry Pi 5

Figure 1.1: Raspberry Hardware

PICO - RP2040

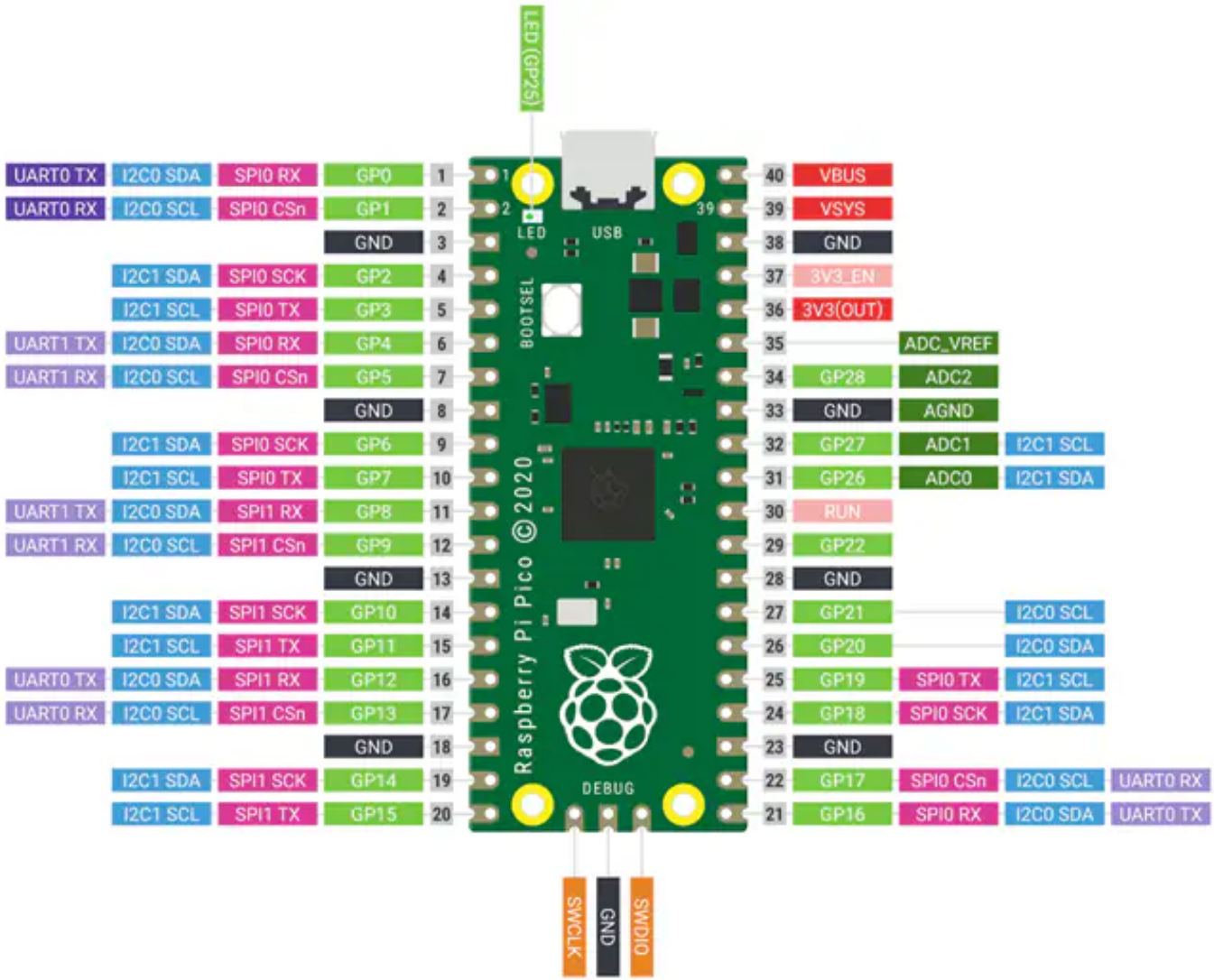


Figure 1.2: Raspberry Pico

Incremental Encoder

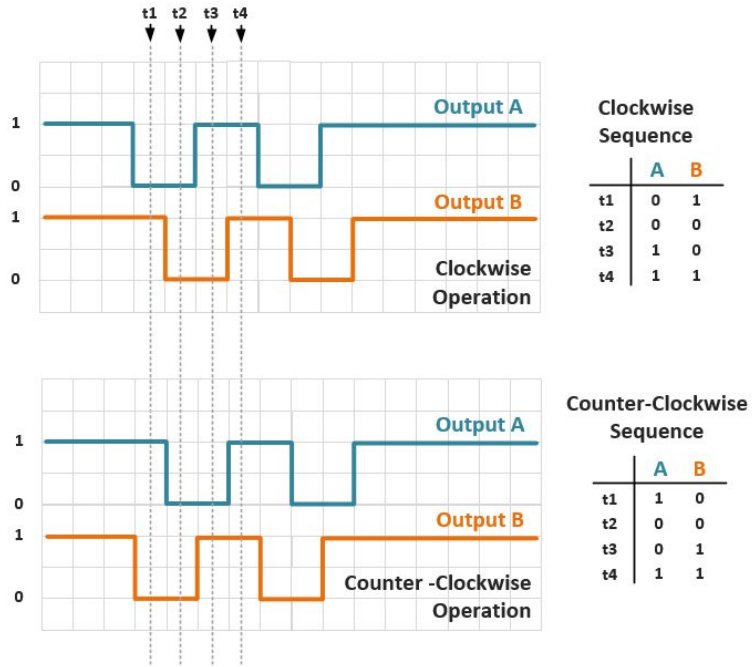


Figure 1.3: Incremental Encoder

Incremental Encoder

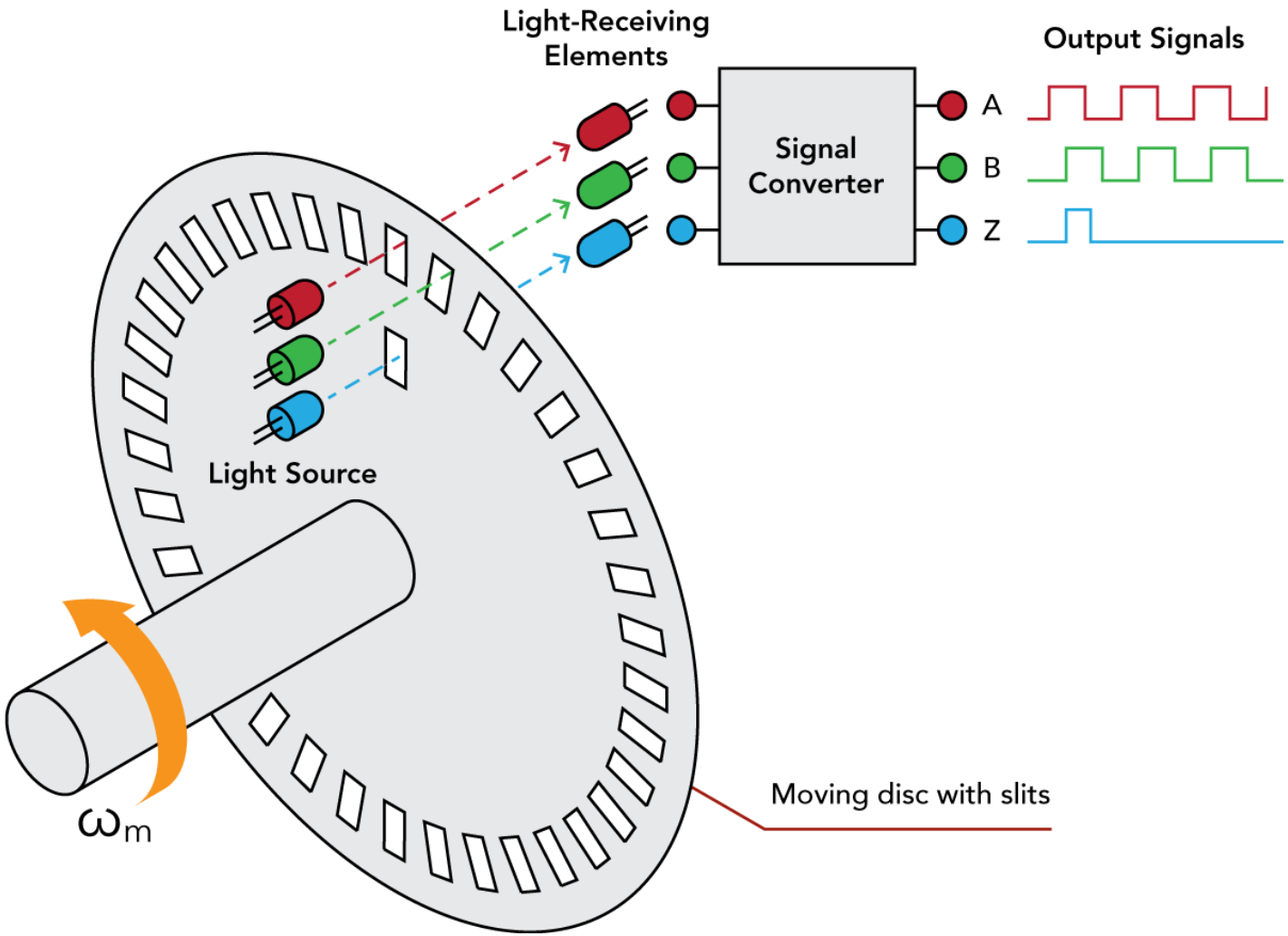


Figure 1.4: Incremental Encoder Mechanism

DC Motor

Figure 1.5: DC Motor

Figure 1.6: HBridge Driver

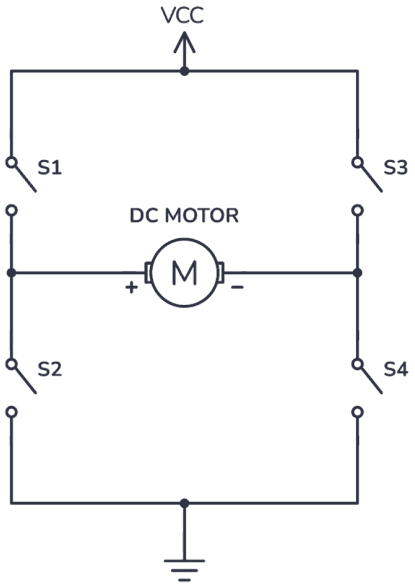


Figure 1.7: DC Motor

- **S1-S4 Closed :** Motor turn clockwise
- **S2-S3 Closed :** Motor turn counter clockwise

DC Motor Control PWM

Definition 1.1

Pulse Width Modulation (PWM) is a technique used to control the amount of power delivered to a device by modulating the width of the pulses in a signal. PWM achieves this by switching the signal between on and off states, and adjusting the amount of time the signal stays on within each period (duty cycle).

Key Characteristics:

- **Frequency:** The number of times the PWM signal repeats per second.
- **Duty Cycle:** The proportion of the signal period during which the signal is high (on), typically expressed as a percentage.
- **Applications:** PWM is used in controlling motors, dimming LEDs, and regulating power in electronic devices.

DC Motor Control PWM

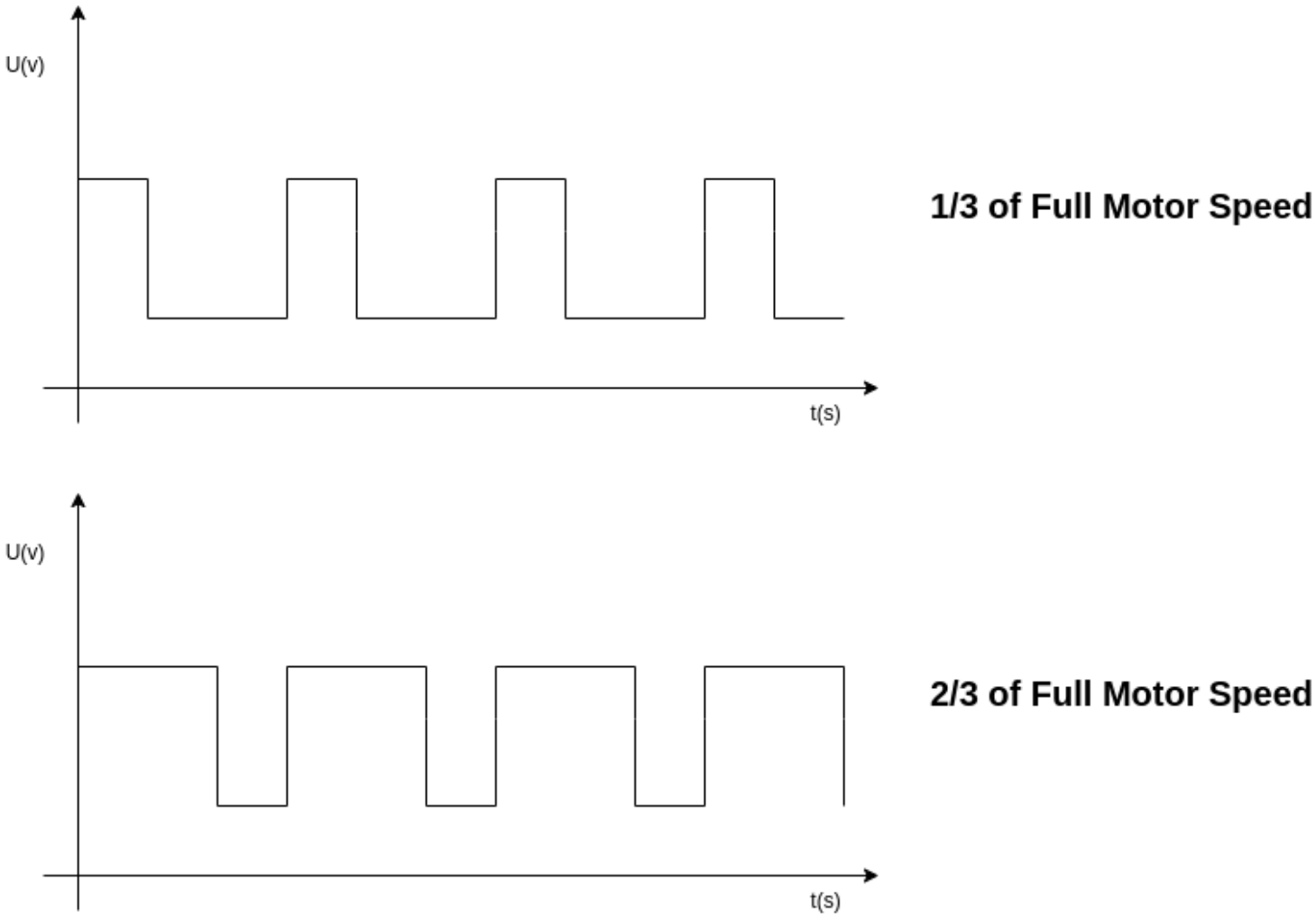


Figure 1.8: PWM signals examples

PWM Servo

Figure 1.9: 9g PWM Servo

- **Frequency:** 50Hz to 300Hz

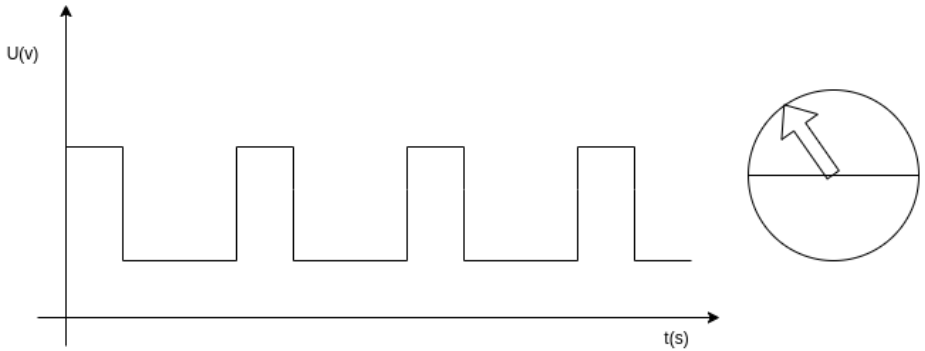


Figure 1.10: PWM signal for 60°

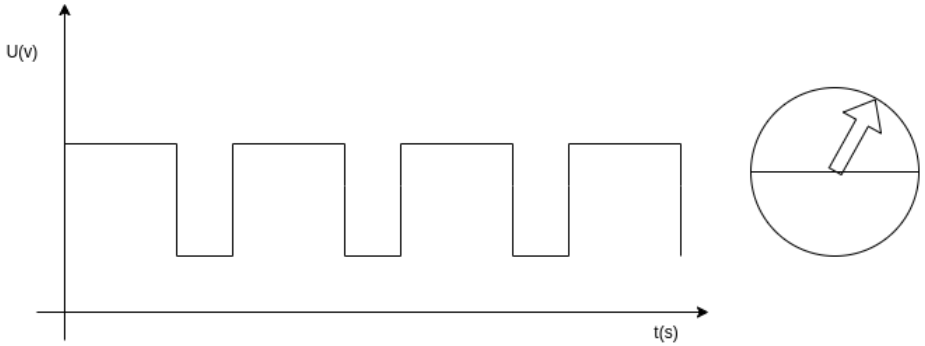


Figure 1.11: PWM signal for 120°

Dynamixels

- **Advanced Servos:** Precise control over movement and position.
- **Multiple Protocols:** TTL, RS-485, and USB communication.
- **Daisy Chain:** Easy connection of multiple units in series.
- **Torque Control:** Adjustable torque for varying force needs.
- **Position Feedback:** Accurate angular position tracking.
- **Multiple Modes:** Velocity, position, and torque control.

Figure 1.12: Dynamixel Range

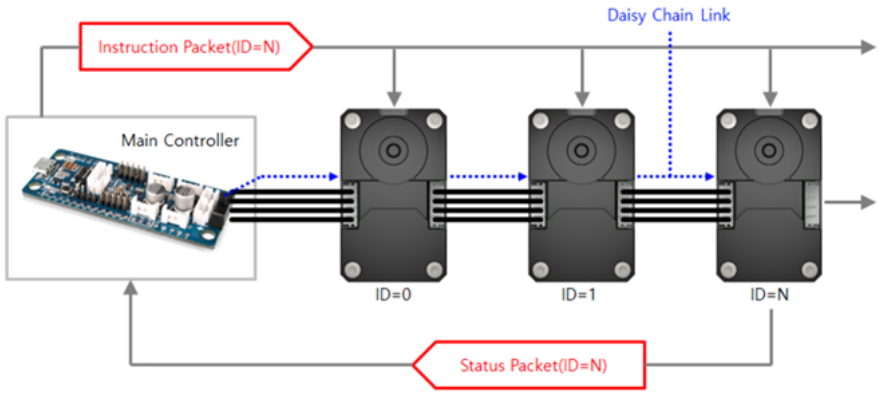


Figure 1.13: Dynamixel Serial

Stepper Motor

Figure 1.14: Stepper motor - NEMA17

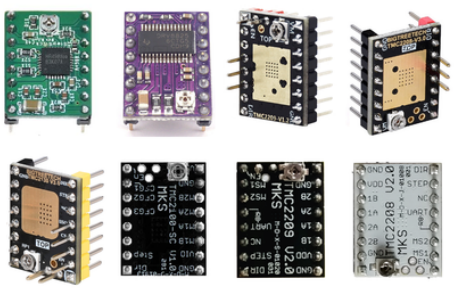


Figure 1.15: Steppers drivers

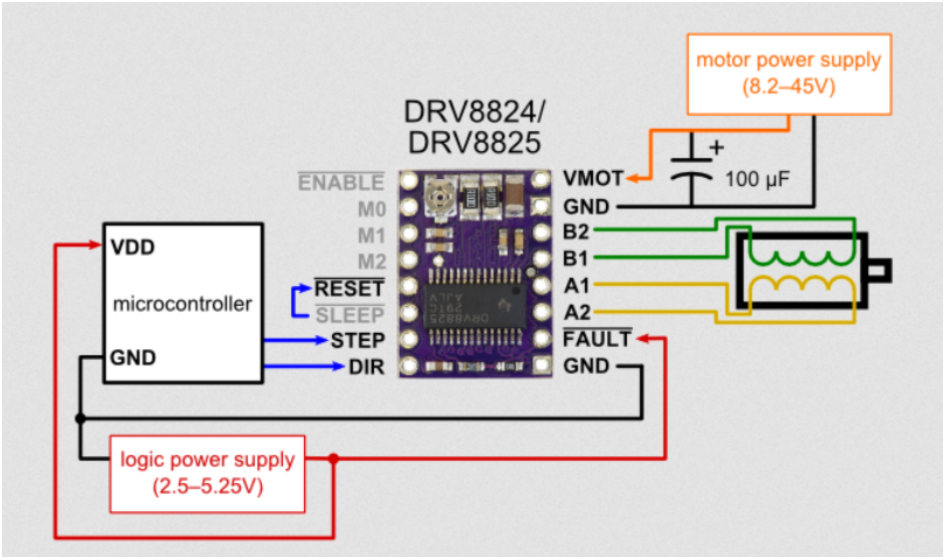
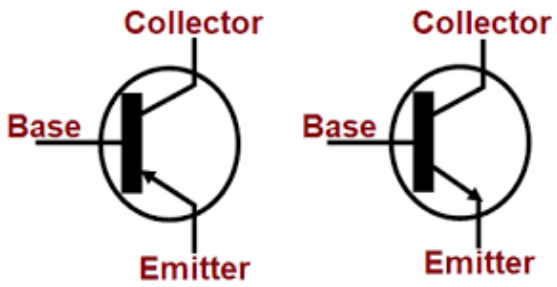


Figure 1.16: Stepper driver wiring



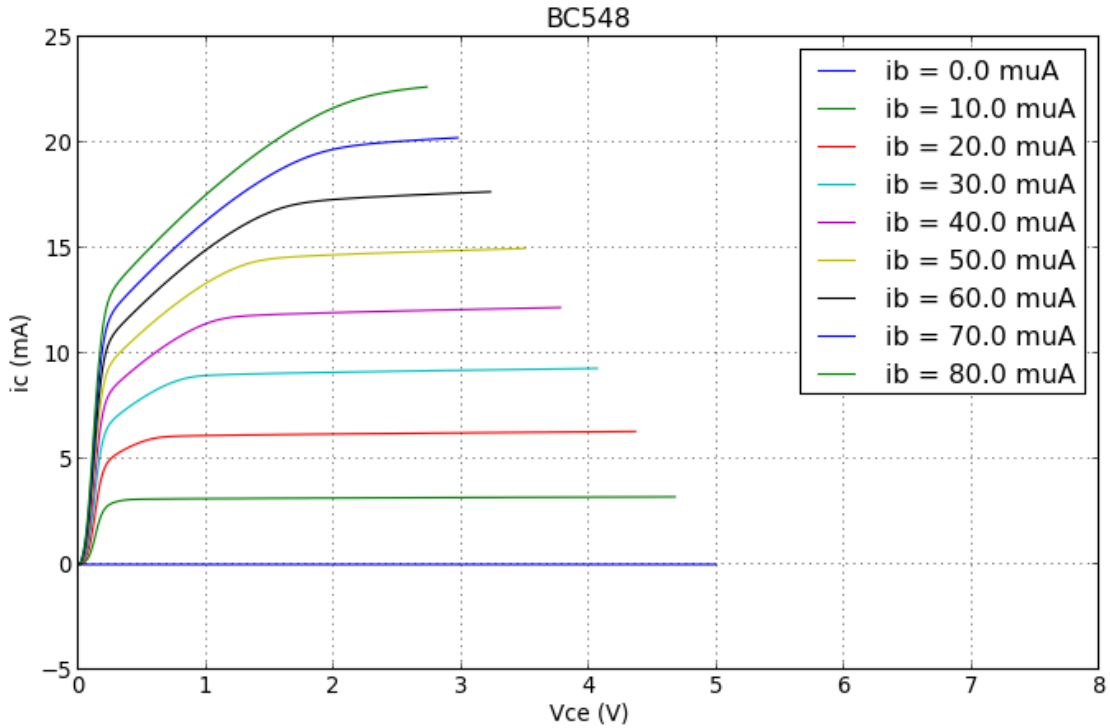


Figure 1.17: Characteristics of BJT

BJT as Switch

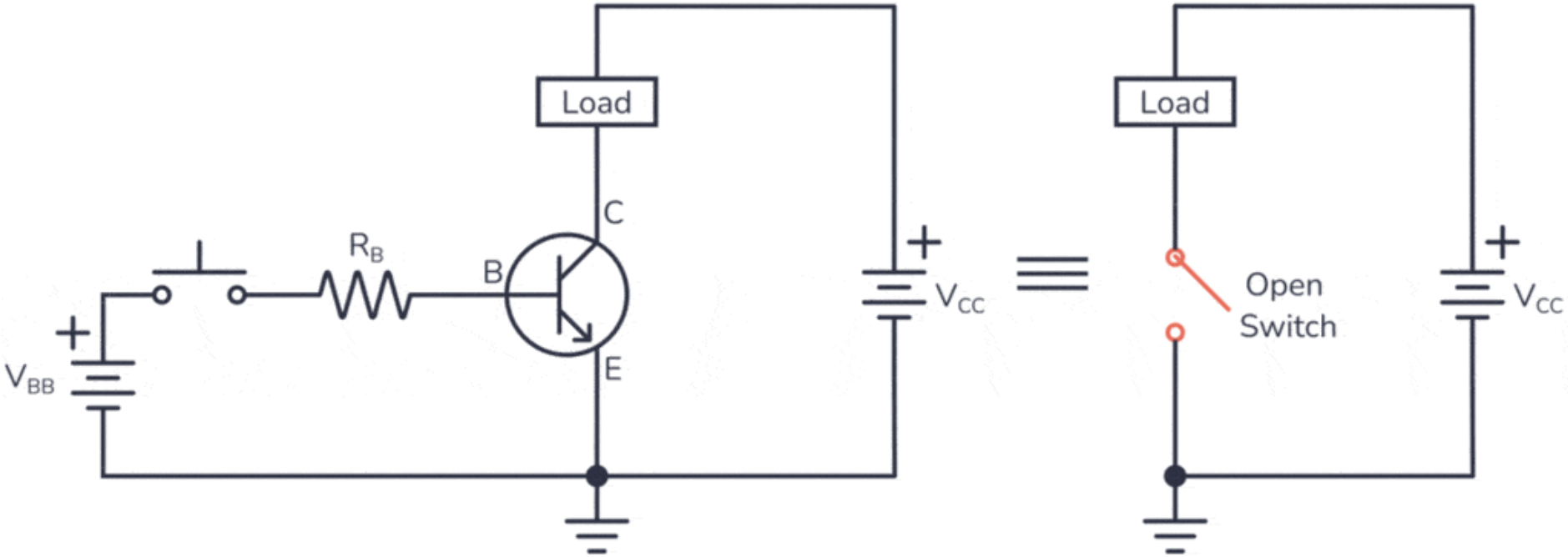


Figure 1.18: BJT used as Switch

$$I_C = \beta I_B \tag{1.1}$$

Load Position

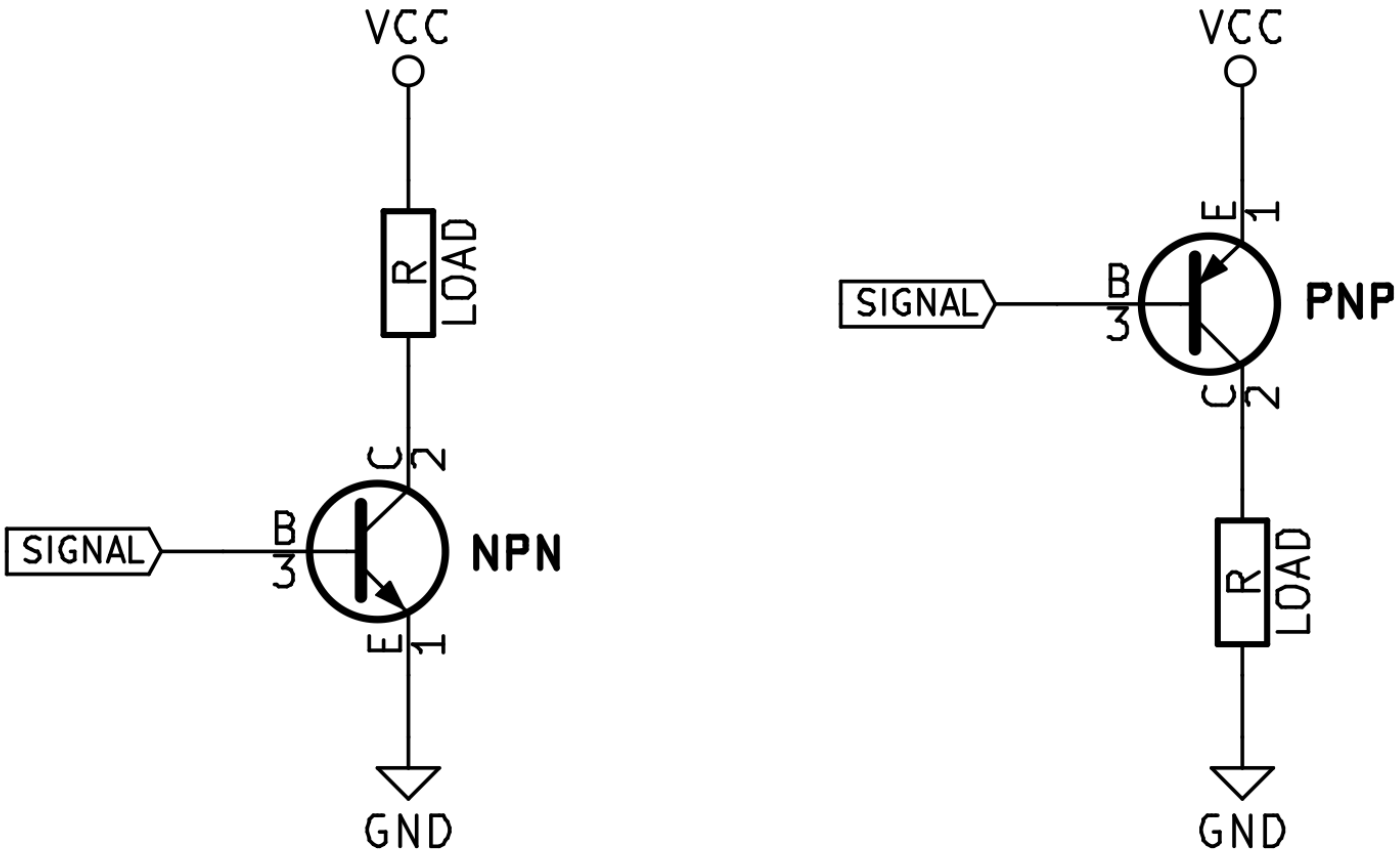


Figure 1.19: PNP/NPN Load Position

Mosfet

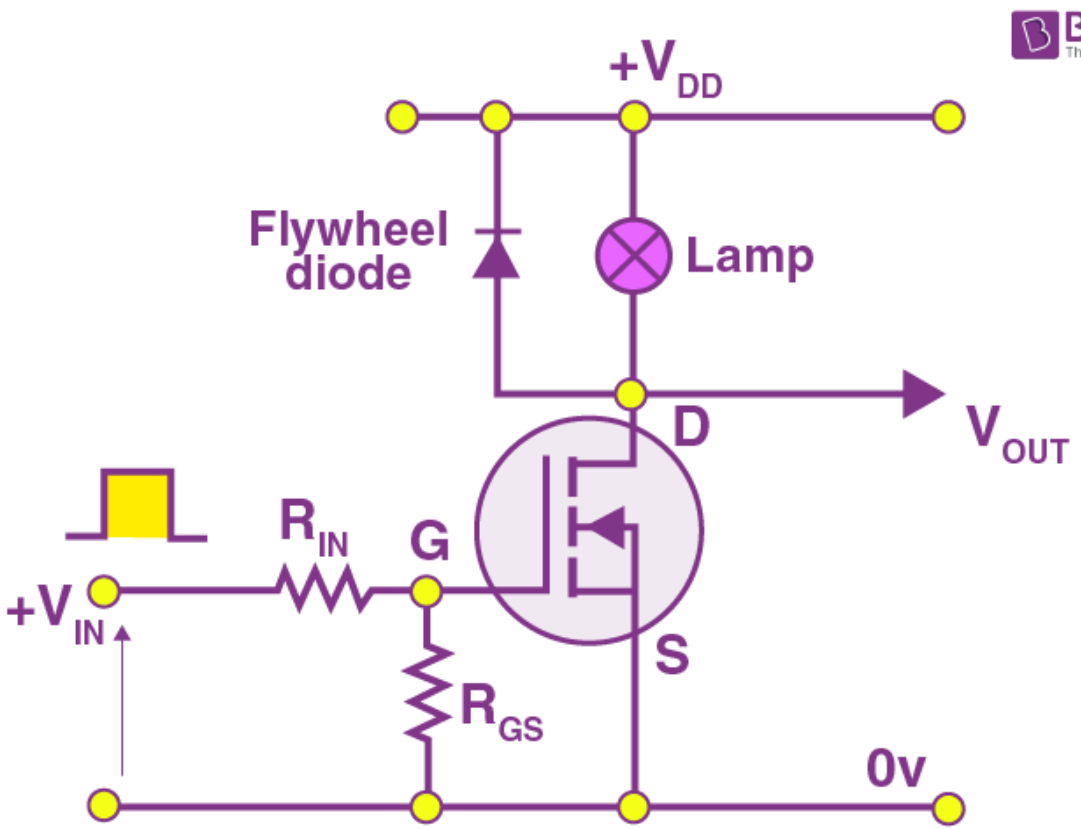


Figure 1.20: Mosfet as switch

Outline

CAN - Controller Area Network

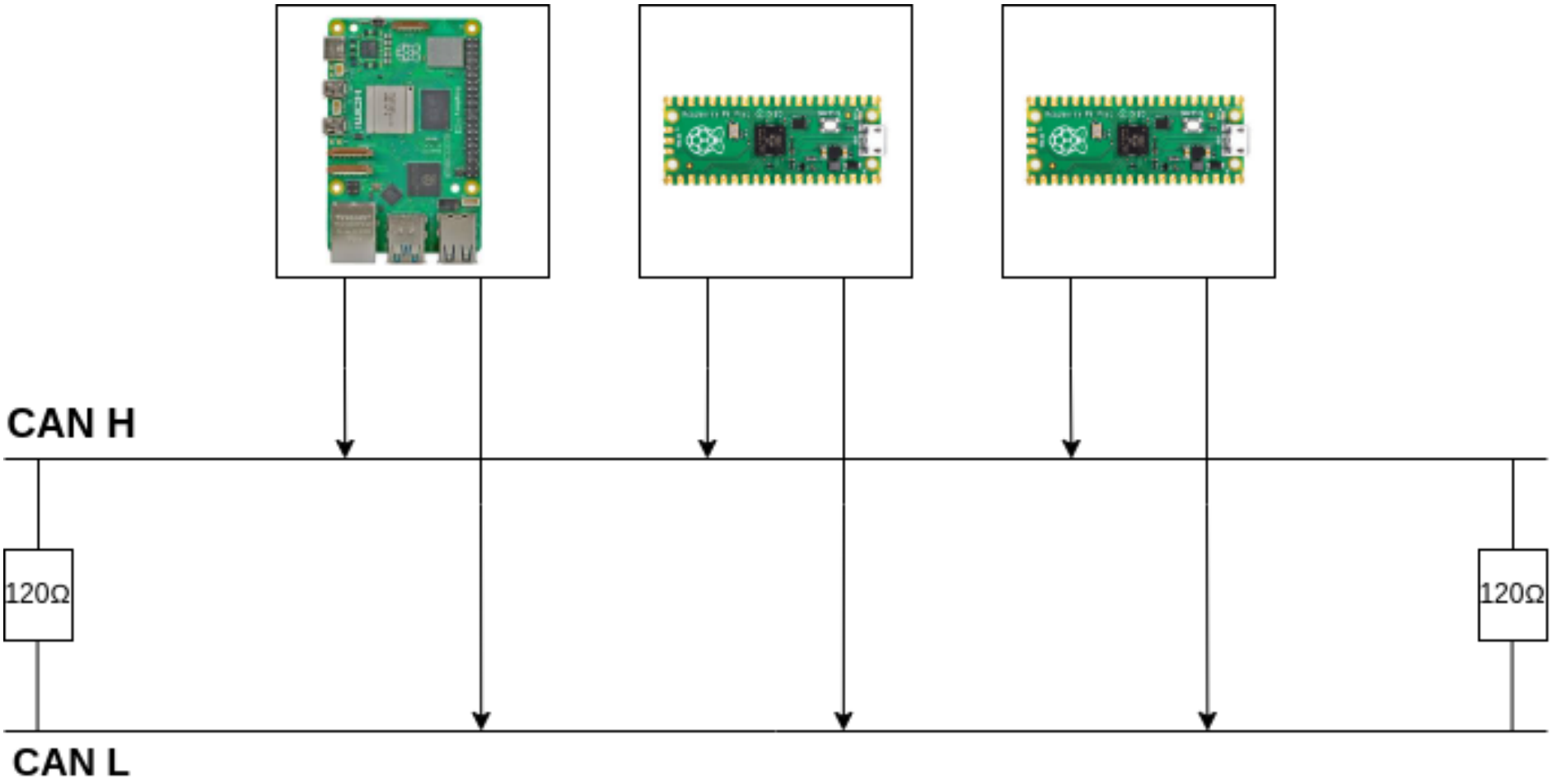


Figure 2.1: Can-Bus Architecture

CAN - Node Architecture

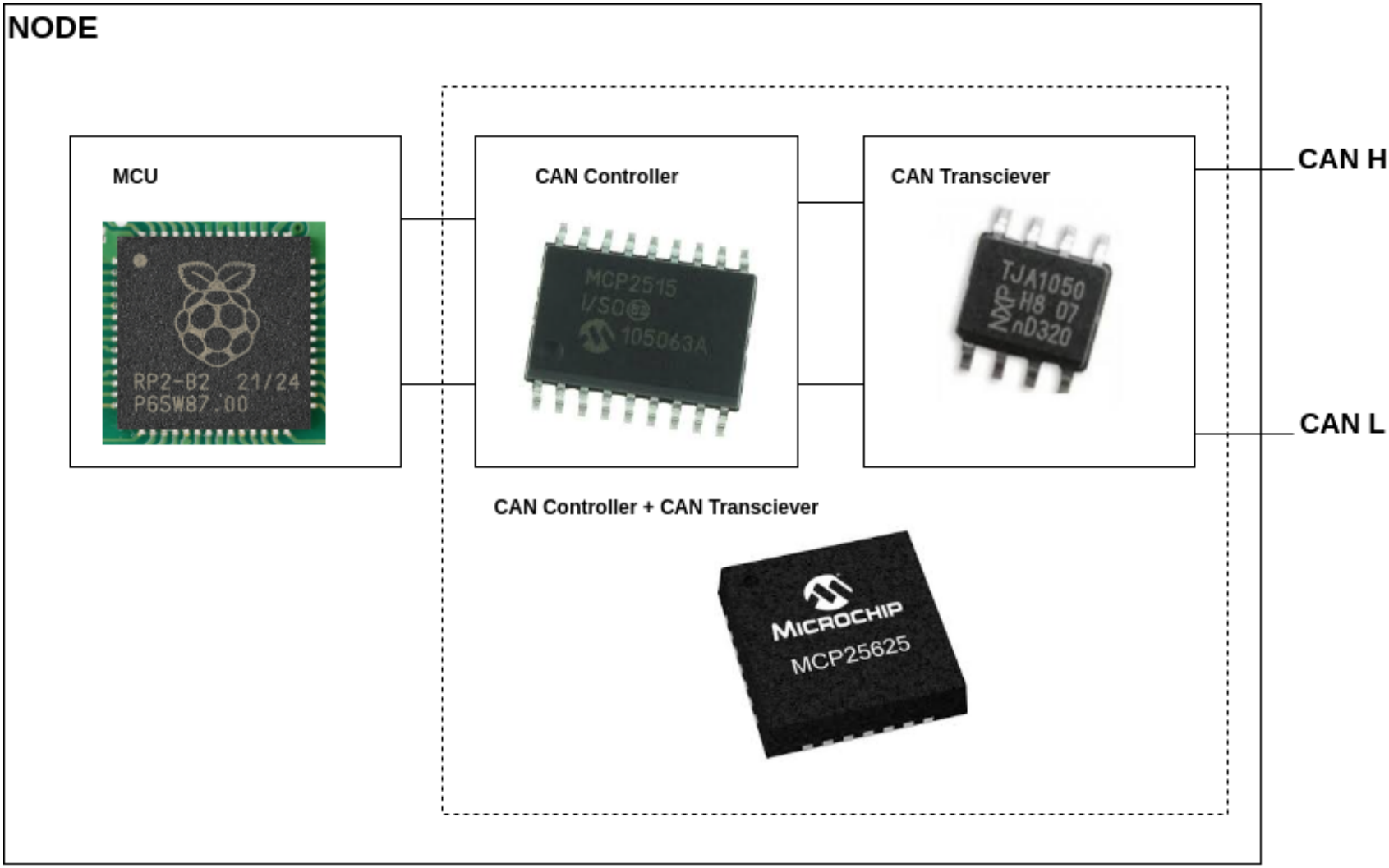


Figure 2.2: Can-Bus Architecture

CAN - Advantages of High-Speed CAN

- **Increased Data Transfer Rate**
  - Supports data rates up to 1 Mbps, allowing faster communication between nodes.
- **Efficient Error Handling**
  - Implements sophisticated error detection and handling mechanisms to ensure data integrity.
- **Robustness**
  - Highly resilient to electrical noise and interference, making it suitable for automotive and industrial applications.
- **Real-Time Performance**
  - Provides deterministic response times, which is crucial for time-sensitive applications.

CAN - Frame Structure

Start of Frame	Identifier	Control	Data	CRC	ACK	End of Frame
1 bit	11 bits (Standard)	6 bits	0-8 bytes	15 bits	2 bits	7 bits

Description des champs :

- **Start of Frame:** Indicates the beginning of the frame (1 bit).
- **Identifier:** Contains the message ID (11 bits for standard format).
- **Control:** Contains control bits, including the length of the data.
- **Data:** Data to be sent, between 0 and 8 bytes.
- **CRC:** Cyclic Redundancy Check code for error detection (15 bits).
- **ACK:** Acknowledgement field (2 bits).
- **End of Frame:** Indicates the end of the frame (7 bits).

SWD - Single Wire Debug

Definition 2.1 (SWD)

The SWD protocol operates using a **request-response** *structure. The host (usually a debugger or programming tool) sends a request to the target device (the microcontroller), and the target responds accordingly.*

The protocol supports different operations like:

- reading/writing memory,
- accessing peripheral registers,
- halting/resuming execution, and
- managing breakpoints.

SWD - Architecture

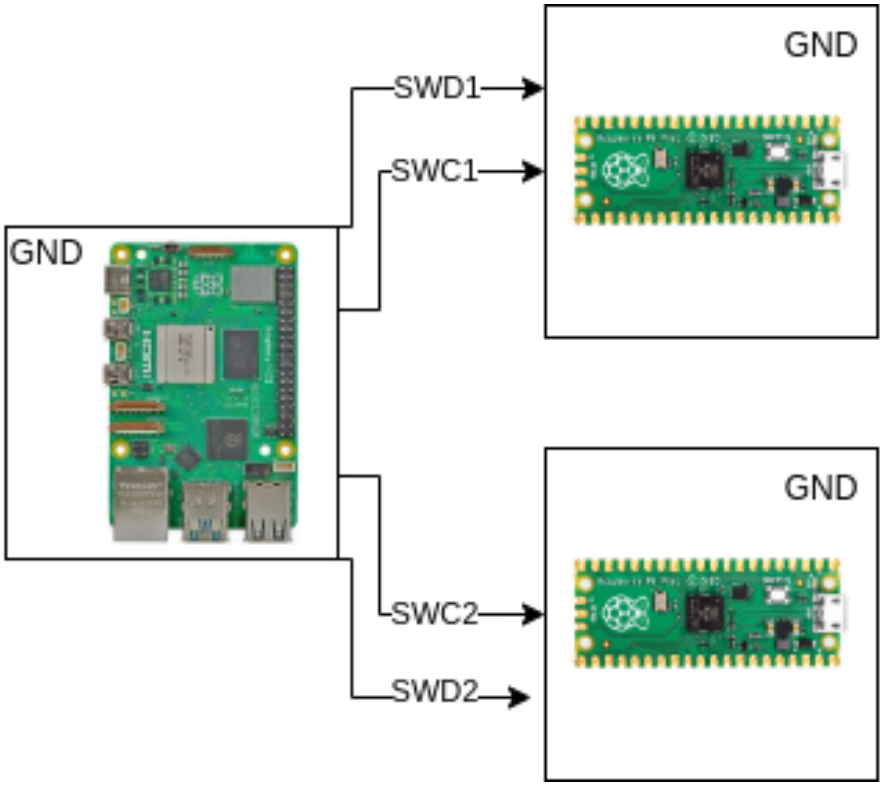


Figure 2.3: Can-Bus Architecture

- **SWDIO:** The main data line used for bidirectional data transfer. It carries both command and response data between the debugger and the target.
- **SWCLK:** The clock signal that synchronizes data transmission. It is used by the debugger to time the sending and receiving of data on the SWDIO line.

UART/SPI/I2C

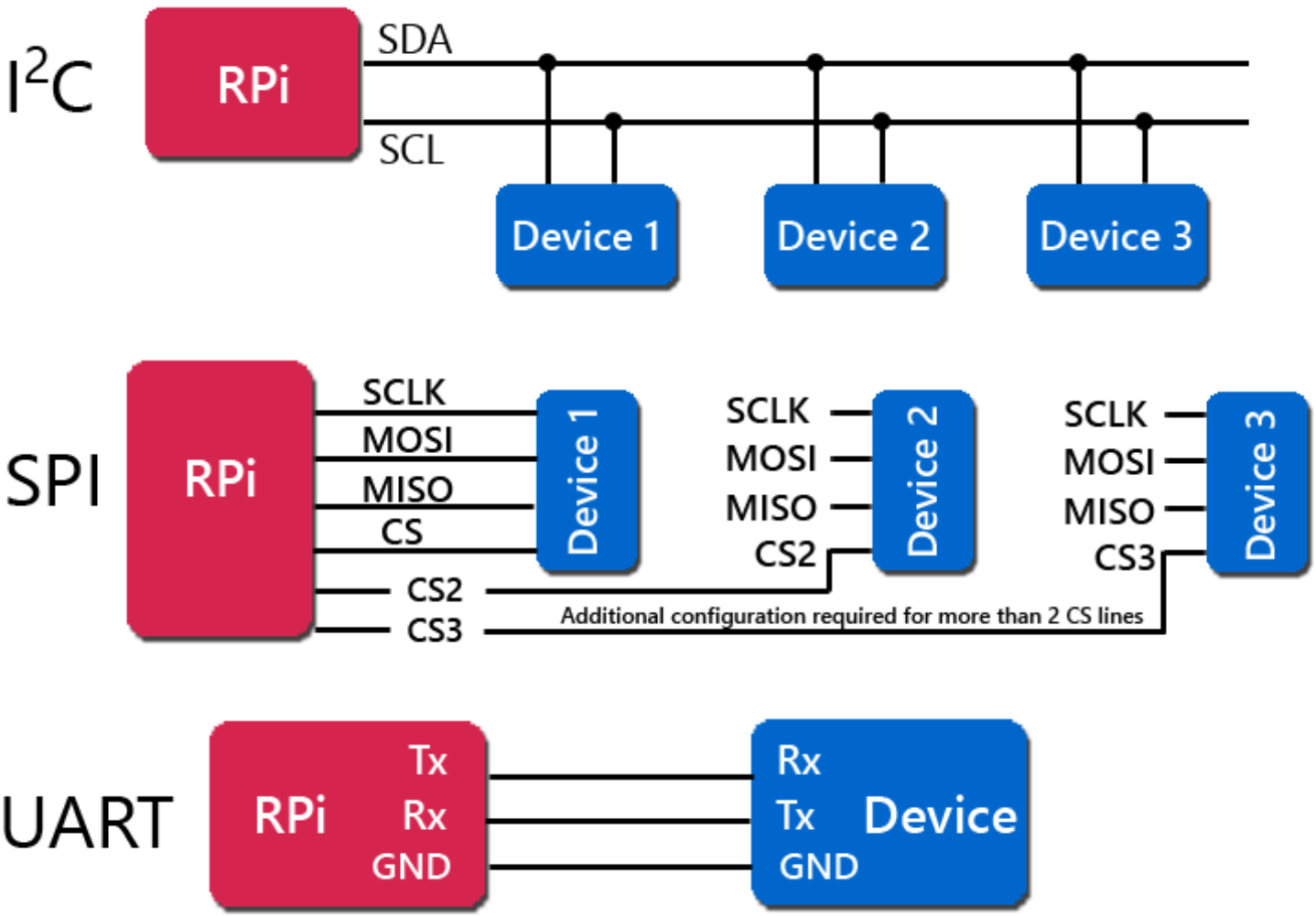


Figure 2.4: UART SPI I2C

Outline

Global Architecture

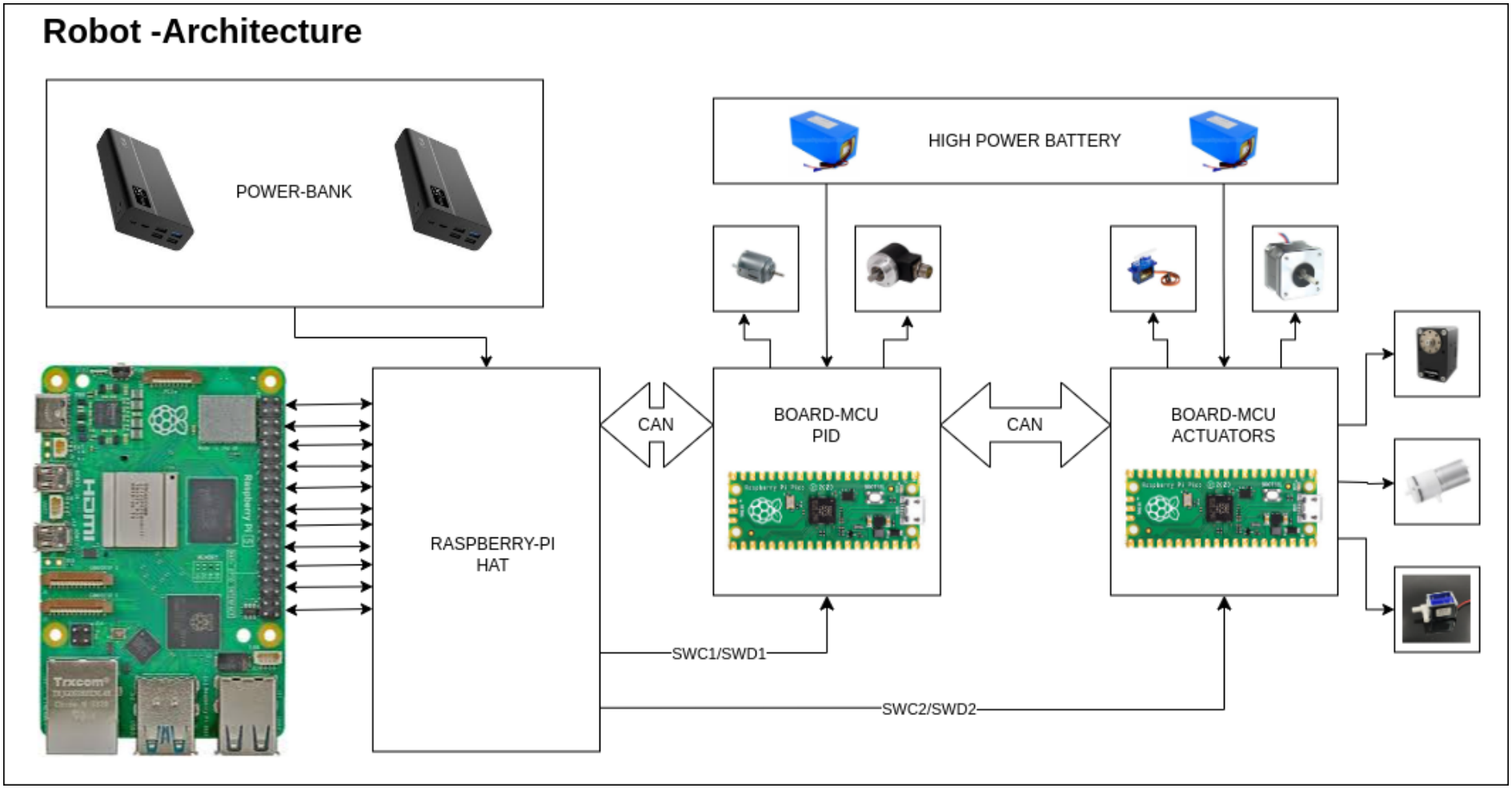


Figure 3.1: Global architecture

Global Architecture

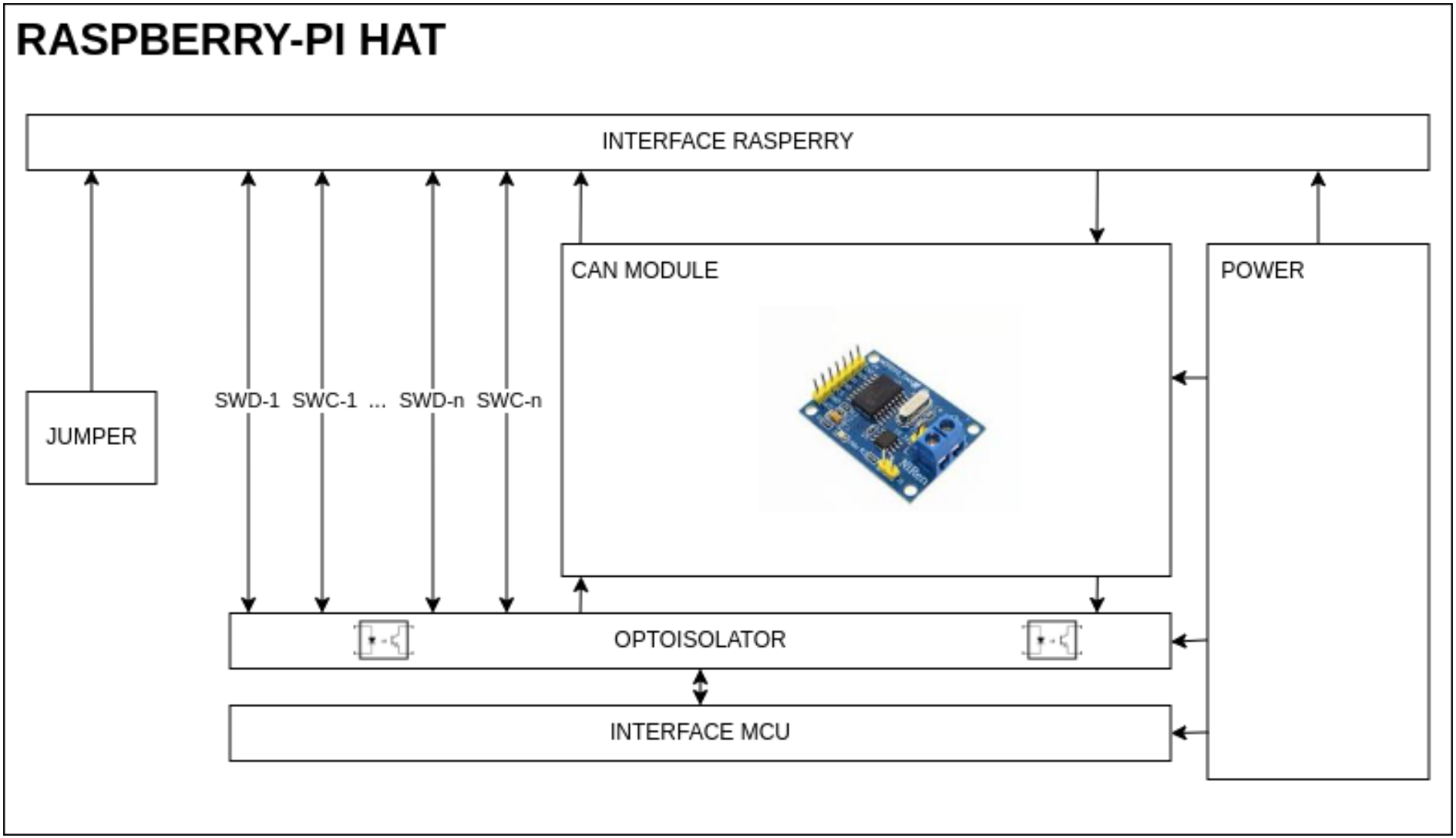


Figure 3.2: Raspberry-Pi Hat

PID Board

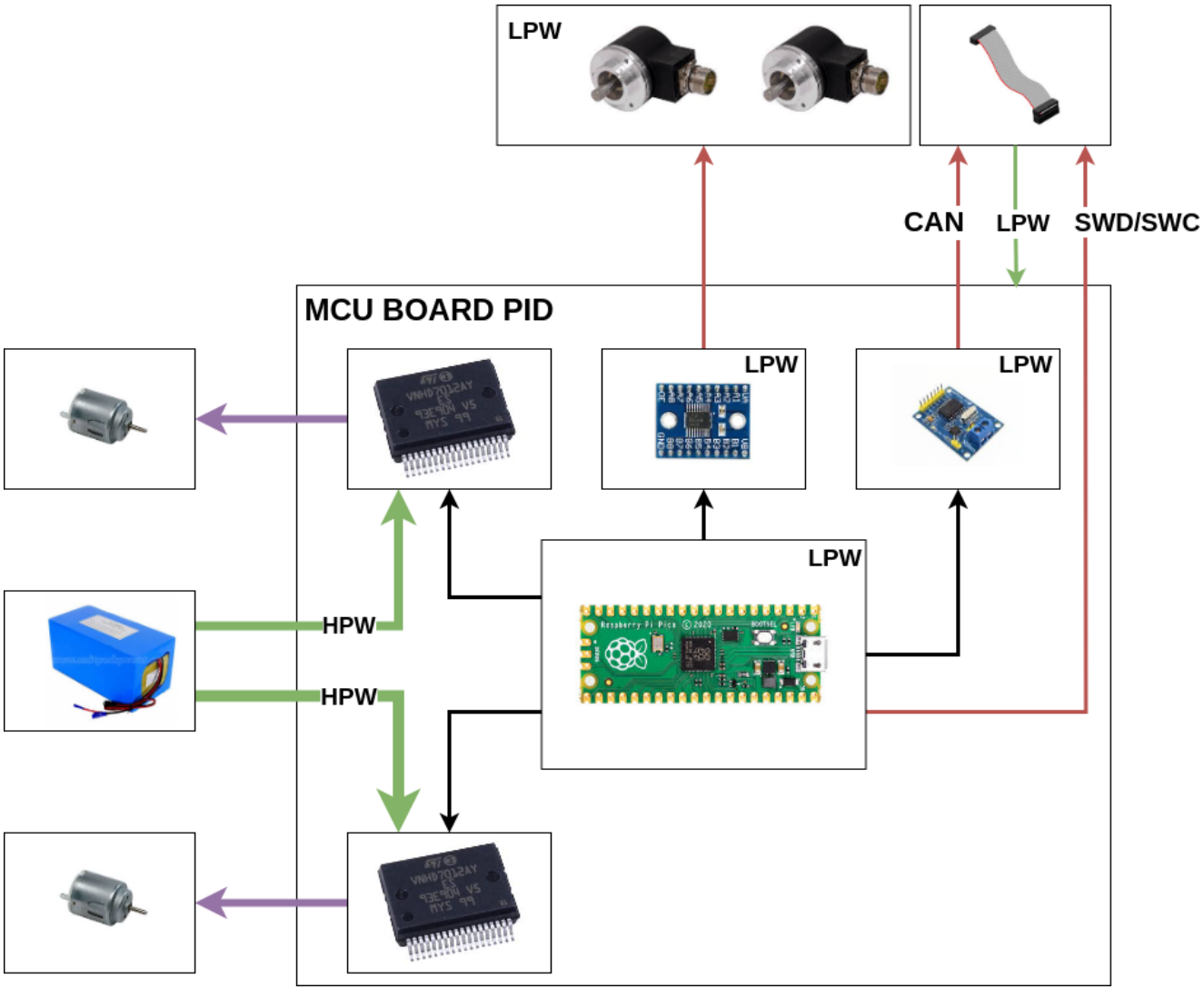


Figure 3.3: PID MCU Board

Actuators Board

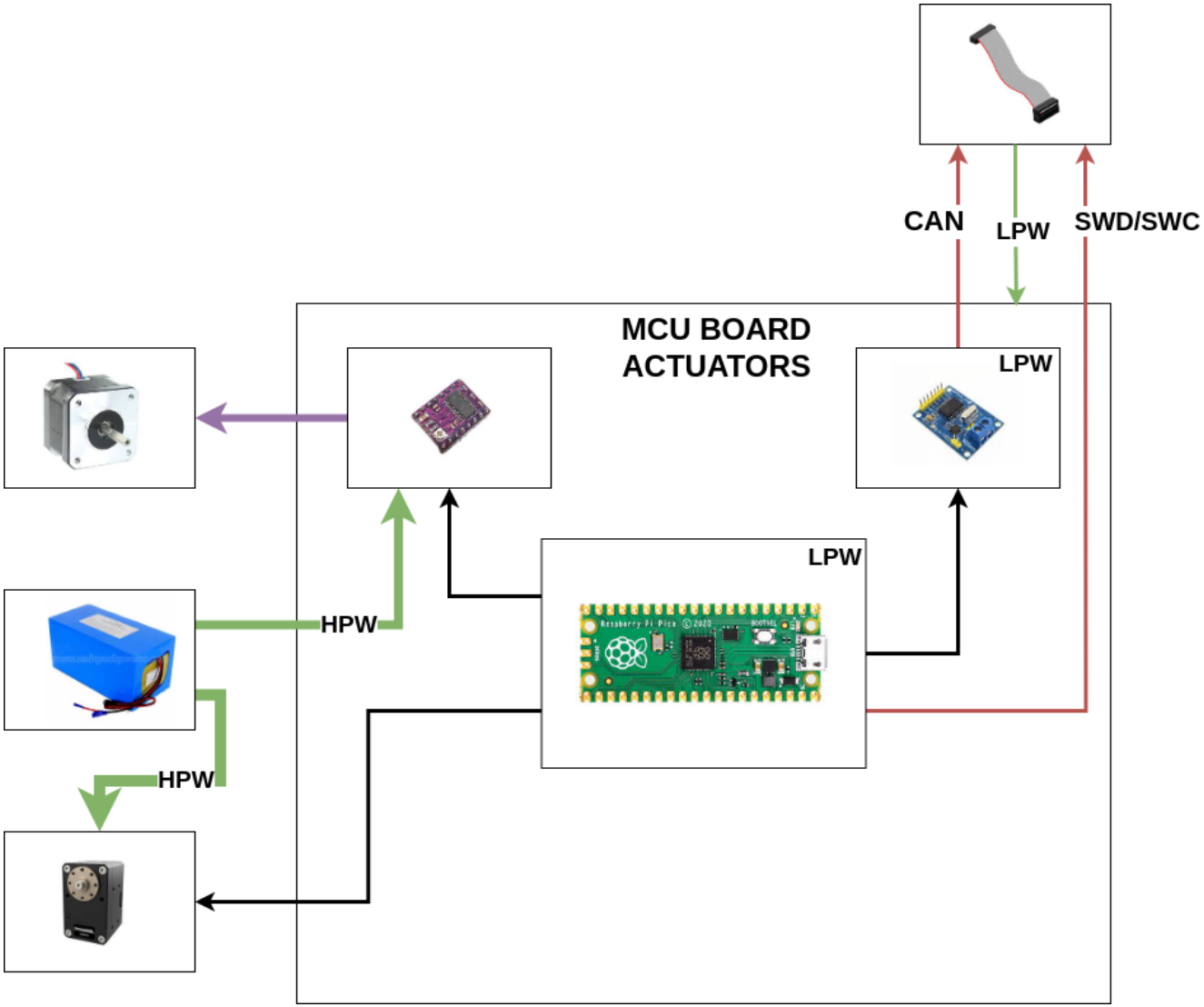


Figure 3.4: Actuators MCU Board

Outline

Hbridge Schematics

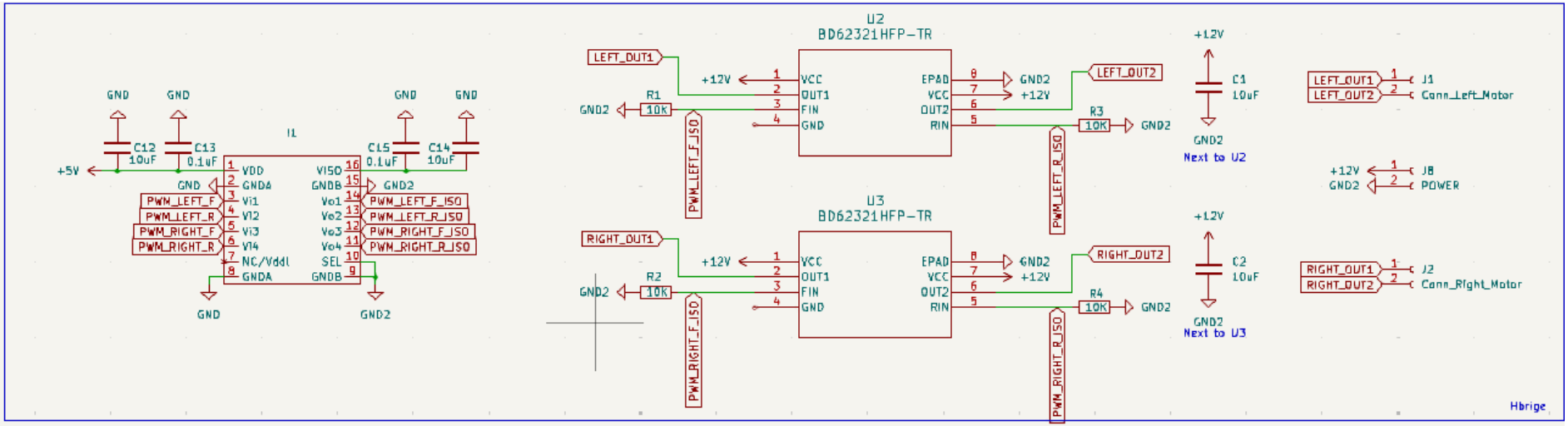


Figure 4.1: Kicad Schematics HBridge

Encoder Layout

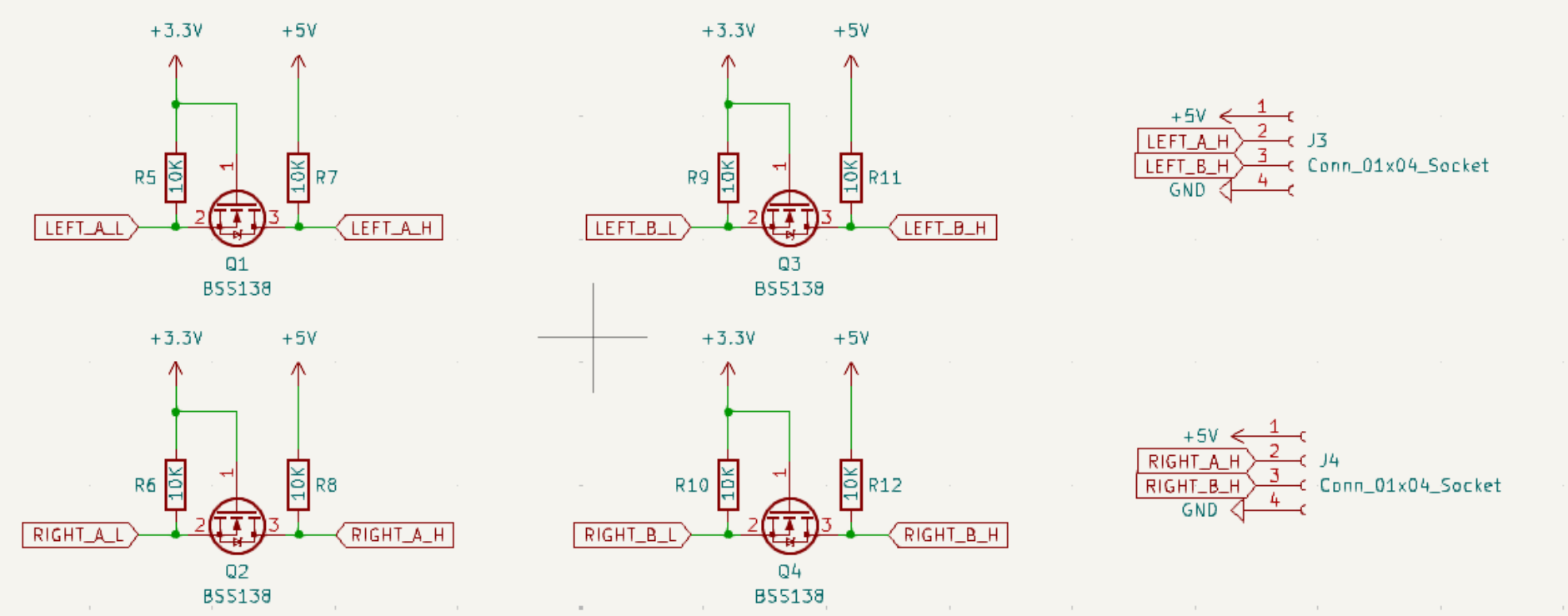


Figure 4.2: Kicad Schematics Encoder

Can Layout

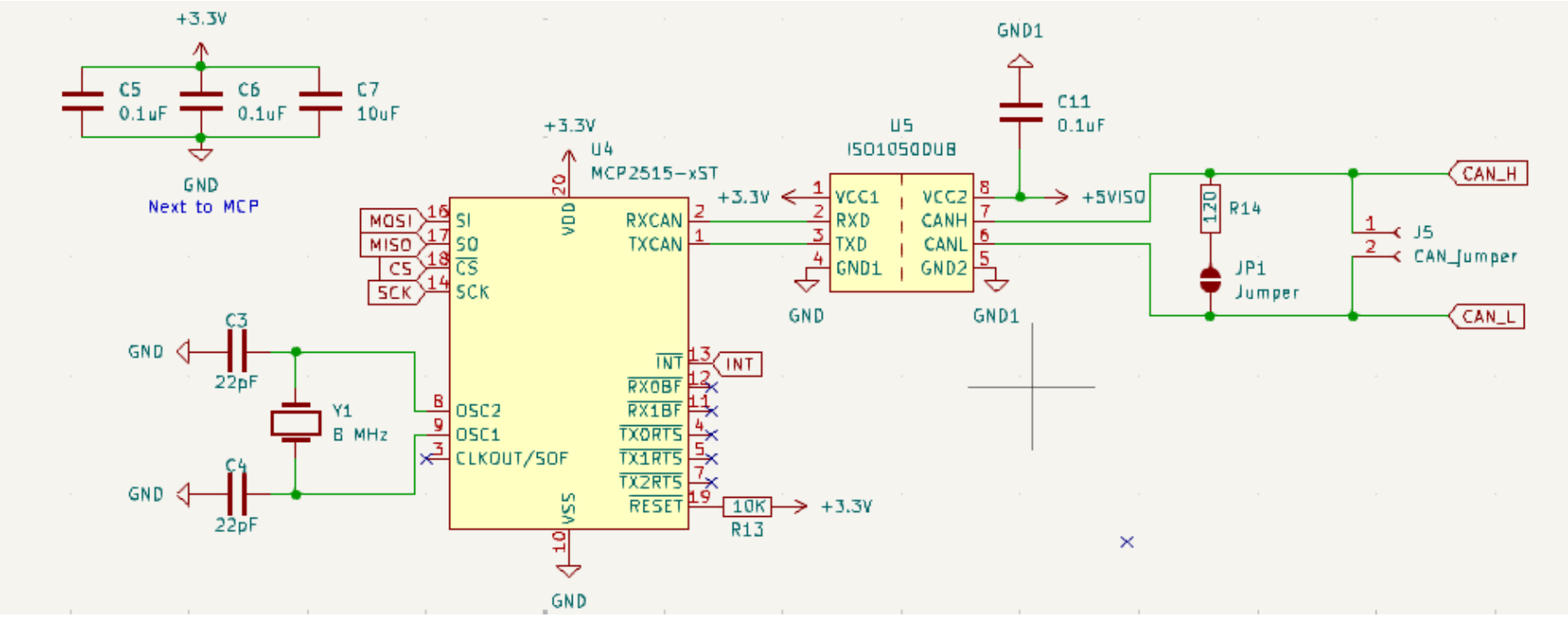


Figure 4.3: Kicad Schematics CAN

Kicad Schematics MCU

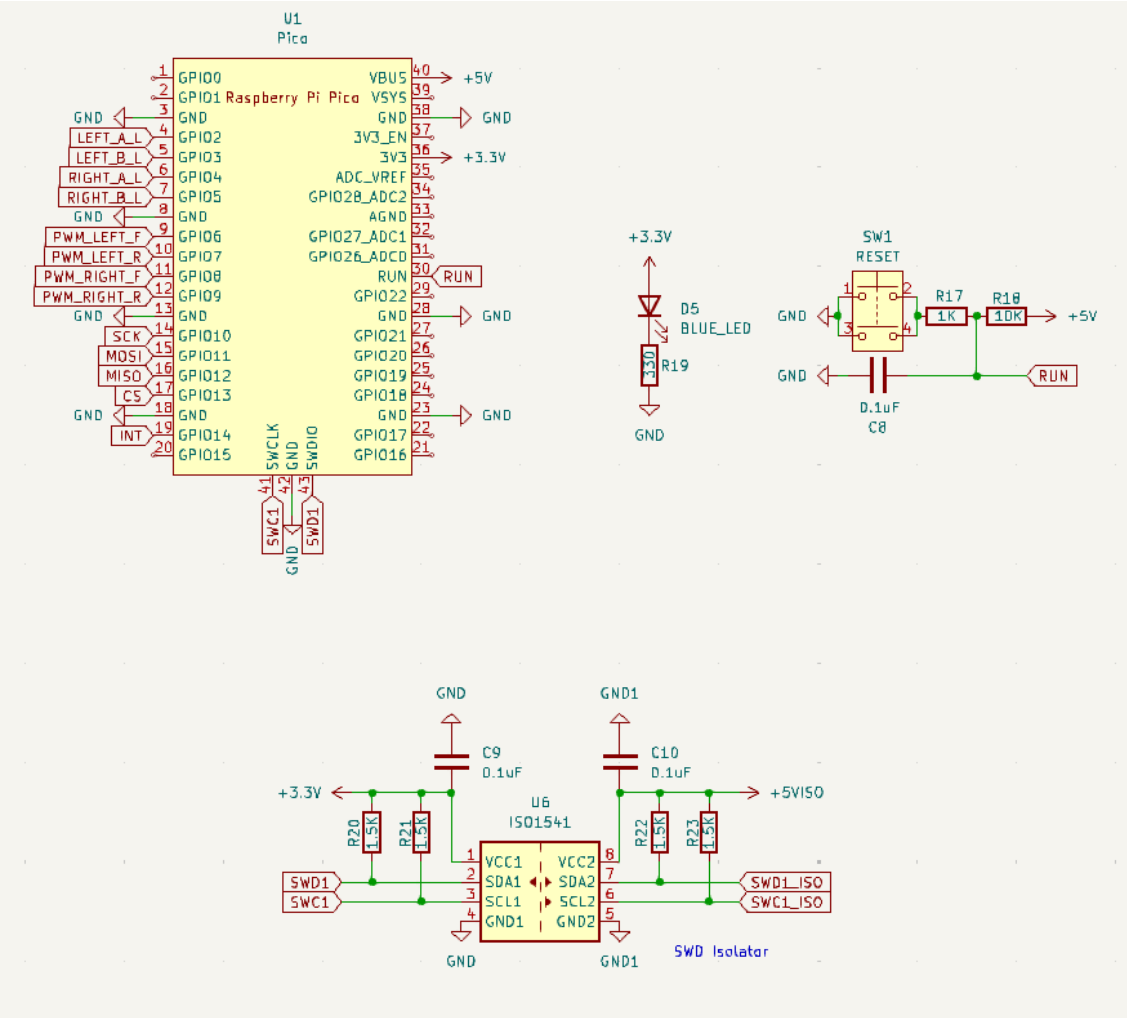


Figure 4.4: Kicad Schematics MCU

Kicad Schematics Connector

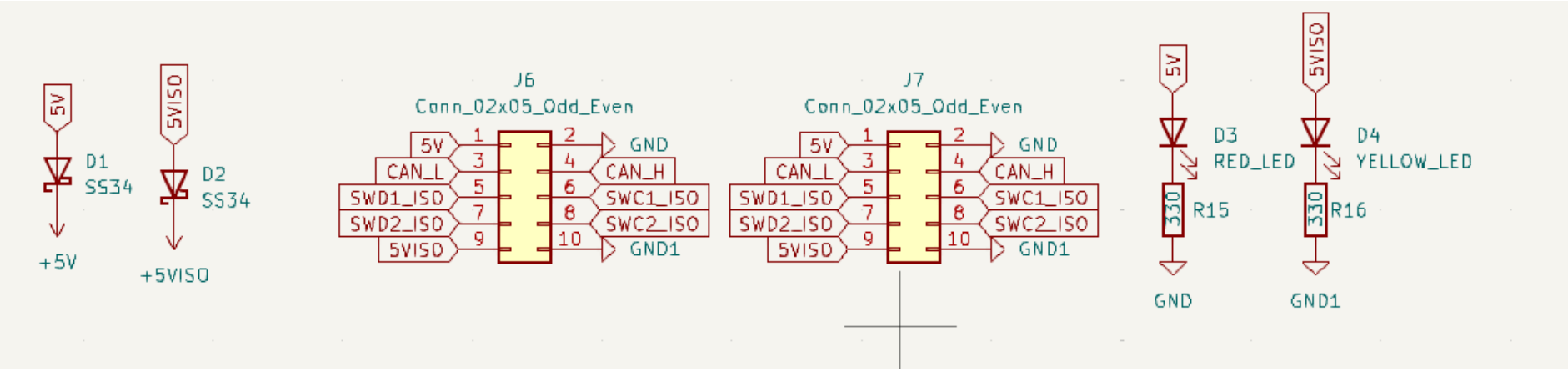


Figure 4.5: Kicad Schematics MCU

Outline

What is a PCB ?

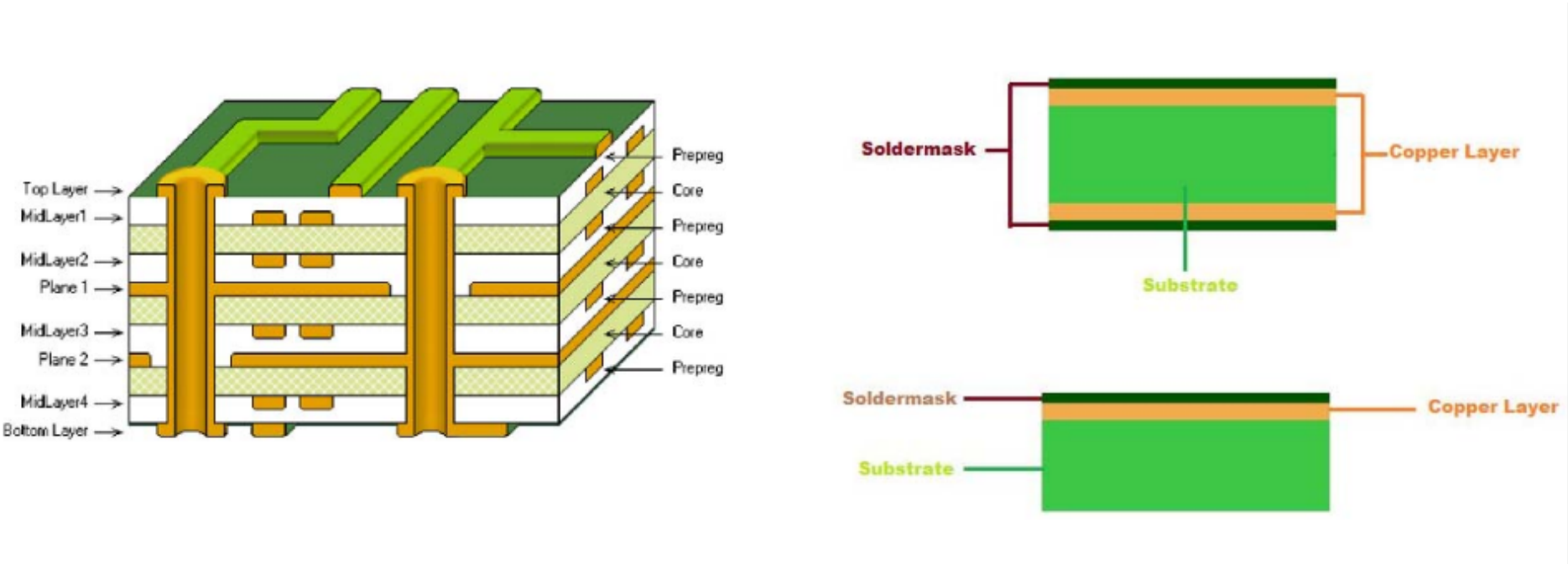


Figure 5.1: PCB

PCB Design Workflow

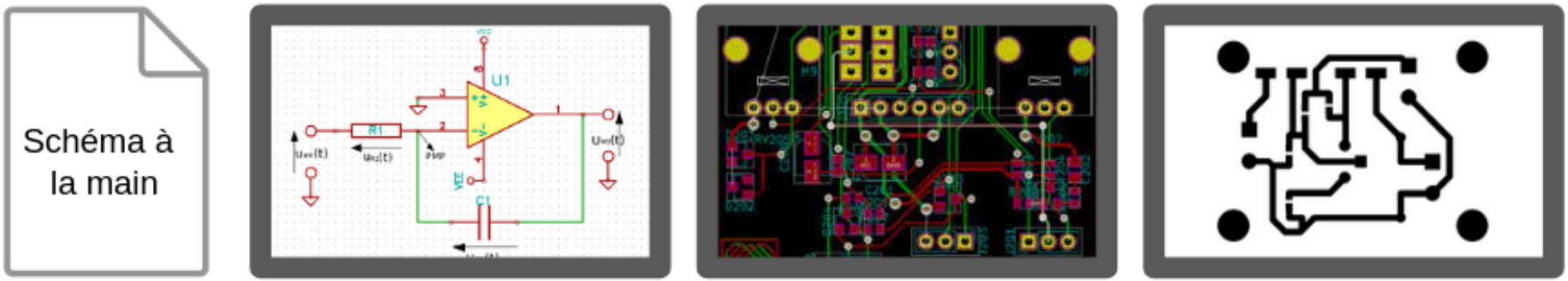


Figure 5.2: PCB Design Workflow

Gerber File

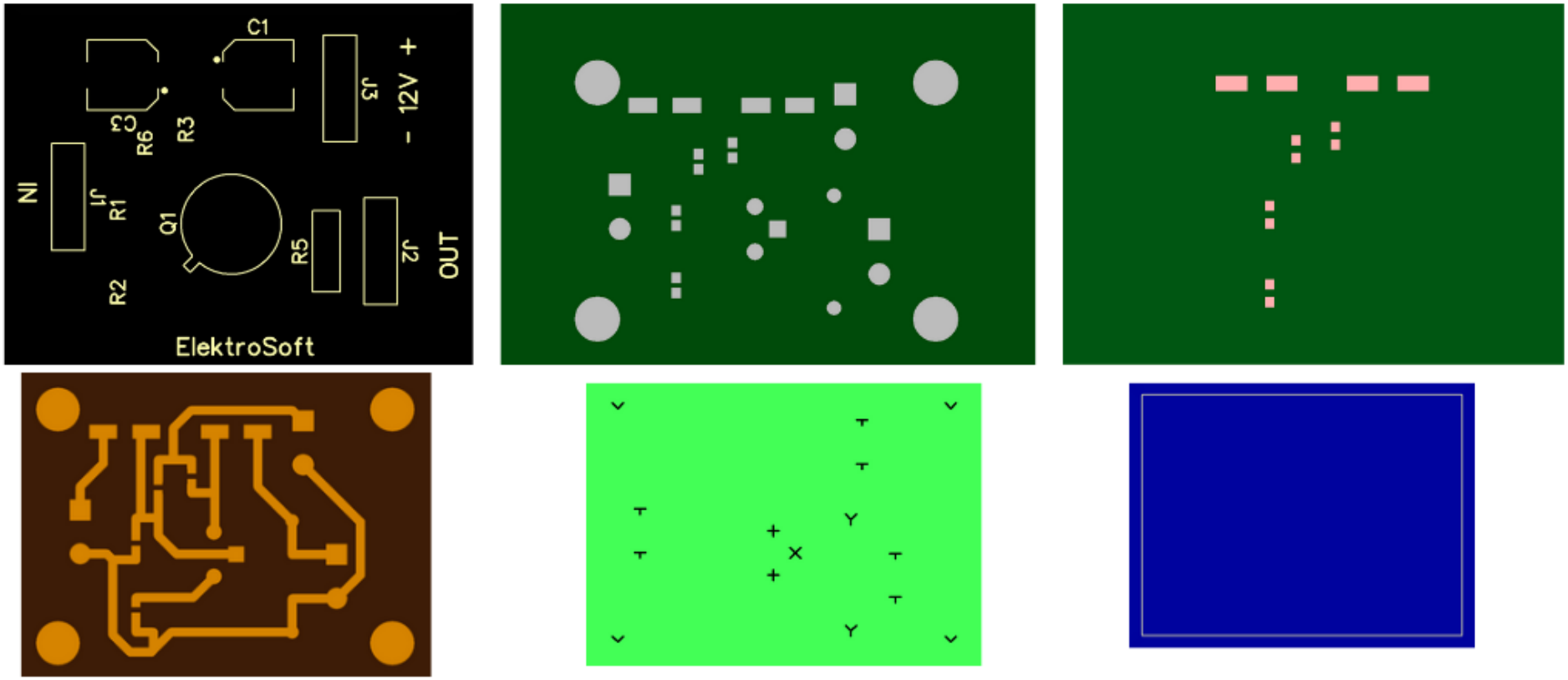


Figure 5.3: Gerber File

To Your Stations