MATRIX Robotics Guidebook - R4 Controller Edition

Guidebook Version: v3.0
Publication Date: 2025/12
Applicable Software Version: MATRIXblock (Mini R4) v1.0.8
Applicable Firmware Version: v6.0
Revision History:
v1.0	2025-07-03	Initial Release	Fang
v2.0	2025-12-26	Added MS-018 Line Tracer 10CH section and specs	Fang
v3.0	2025-12-26	Added IoT, DriveDC, Color Sensor ID	Anthony

1. Introduction

1.1. About MATRIX Robotics

MATRIX Robotics is a modular robotics system designed for educators, creators, and competition applications. This system combines sturdy metal structures, compatible electronic components, and an intuitive graphical programming platform. MATRIX allows users to gradually learn mechatronics integration and programming skills, from beginner to advanced levels. Whether in classroom settings, innovation workshops, or international robotics competitions, MATRIX offers stable and flexible support, making it an ideal tool for interdisciplinary learning.

1.2. Purpose of the R4 Controller Edition Guidebook

This manual provides a comprehensive introduction to the core controller in the MATRIX system: the MATRIX Mini R4 Controller. Whether you're new to MATRIX or looking to explore further component applications and advanced functions, this guidebook serves as a reliable resource for learning and development.

Contents include:

- Hardware features and connection methods of the R4 controller
- Usage of various sensors and motor modules
- Programming tutorials using MATRIXblock software
- Troubleshooting tips and system maintenance recommendations

1.3. How to Use This Guidebook

This guidebook is organized by functional modules, with real-world photos, wiring instructions, and programming examples to make it easy for readers to reference as needed. You can study the chapters in order or jump directly to the sections relevant to the components you're using. Suggested usage:

- Beginners: Start with Chapter 3 "MATRIXblock Programming" and Chapter 4 "Input Modules" to quickly understand the link between sensors and graphical programming.
- Teachers and Curriculum Designers: Combine various module chapters to flexibly design classroom teaching and themed activities.
- Advanced Users: Read Chapter 2 for insights into the hardware architecture and system integration, enabling high-level application development.

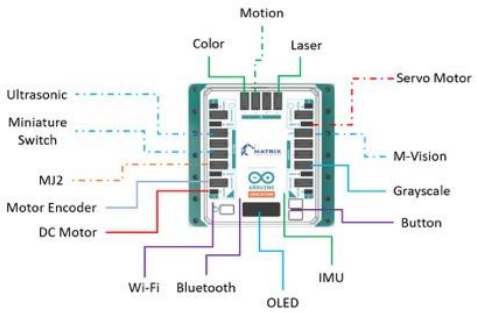
2. Overview of the MATRIX R4 Controller

2.1. Introduction

The MATRIX Mini R4 is a multi-function microcontroller designed for STEM education and innovative projects. The microcontroller adopts **dual-processor architecture. The Arduino UNO R4 WiFi is the core controller, responsible for the main program logic and user interaction, along with Wi-Fi and Bluetooth wireless communications. The STM32F103 microcontroller then acts as a co-processor to efficiently handle real-time tasks such as motor driving and reading sensor data**. This design not only maintains the ease of use and plentiful resources of the Arduino platform but also enhances the overall performance and stability of the system through the co-processor.

The Mini R4 has extensive modular expansion capabilities, supporting connections to various sensors and driver modules. It is an ideal central platform for STEM education, robotics competitions, and a wide range of creative projects. **Users can easily upload programs and provide basic power to the controller via its USB-C port. When high power components such as motors need to be driven, it is recommended to connect to an external power source ranging from 6V to 24V**. Additionally, it can wirelessly be connected with other devices, enabling more flexible applications.

By reading this chapter, you will gain a comprehensive understanding of the MATRIX Mini R4 controller's hardware structure, communication ports, power supply methods, corresponding software development tools, and common application scenarios. Thus, laying a foundation for subsequent development and teaching.



In this chapter, you will learn:

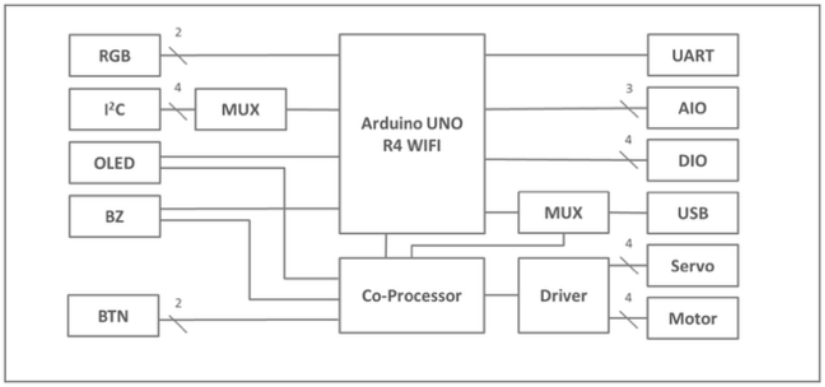
- The core architecture, basic functions, and application advantages of the Mini R4
- How to correctly connect peripheral modules and provide power to the system
- How to use the controller pins and various communication interfaces
- The meaning of the built-in LEDs and status indicators on the controller
- How to operate the software using MATRIXblock and Arduino C++

From hardware to software this chapter will help you fully master the operation and application of the MATRIX Mini R4 whether you are a novice controller user or a developer looking to integrate the control system into an innovative application.

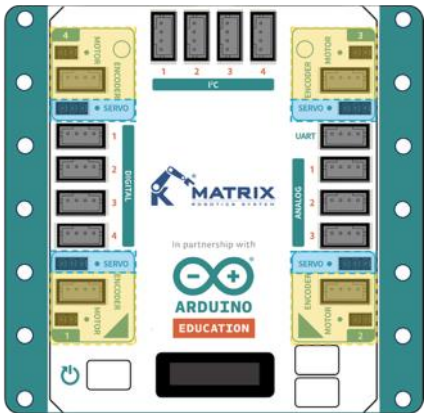
2.2. Features

With its unique dual-processor architecture and rich built-in features, the MATRIX Mini R4 controller provides a powerful and flexible central platform for STEM education, robotics competitions, and creative projects. The following are its main features:

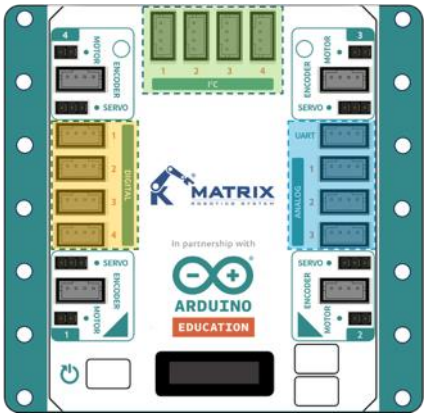
- **Efficient Dual-Core Processing Architecture**
  - **Main Control Core:** Equipped with the Arduino UNO R4 WiFi as the host controller, it is responsible for executing the main control logic, user programs, and providing wireless communication capabilities.
  - **Co-processor:** Features an STM32F103 microcontroller as the secondary controller, dedicated to tasks such as a precise motor control and real-time sensor data reading, effectively sharing the processing load from the main core.
  - **Architectural Advantage:** Combines the user-friendliness of Arduino with the high performance of STM32, enhancing overall system stability, responsiveness, and scalability, while ensuring the main controller retains sufficient processing power for complex algorithms.



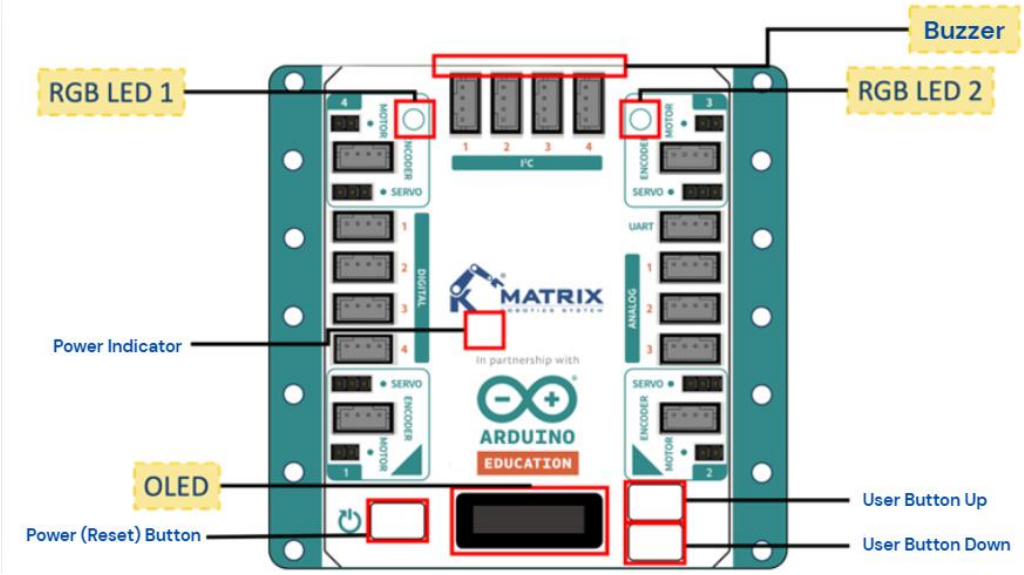
- **Convenient Wireless Communication**
  - **Built-in Wi-Fi Function:** Provided by the Arduino UNO R4 WiFi, it supports wireless network connectivity, making it convenient for remote data transmission, Internet of Things (loT) application development, or remote monitoring.
  - **Bluetooth® LE:** Also provided by the Arduino UNO R4 WiFi, it enables low-power wireless connections with smartphones, tablets, or other Bluetooth devices, suitable for scenarios such as remote-control data exchange.
- **Powerful and Precise Motor Control**
  - **DC Motor Ports:** Equipped with 4 DC motor output interfaces, driven by the STM32 co-processor.
  - **Servo Motor Ports:** Include 4 servo motor output interfaces, also controlled by the STM32 co-processor.
  - **Motor Encoder Ports:** Comes with 4 motor encoder interfaces, enabling closed-loop control when used with DC motors.



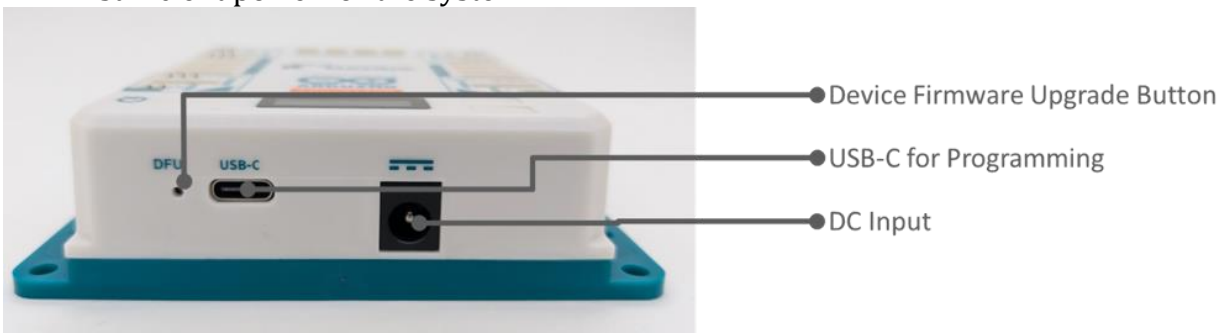
- **Rich and Diverse Expansion Interfaces**
  - **Digital Signal Ports:** Provides 4 digital signal interfaces (D1, D2, D3, D4), for connecting various digital sensors or modules.
  - **Analog Signal Ports:** Provides 3 analog signal interfaces (A1, A2, A3) for reading analog sensors values.
  - **I²C Ports:** Provides 4 12C interfaces, making it easy to connect multiple devices that support the I2C communication protocol.
  - **UART Serial Port:** Provides 1 UART interface for serial communication.
  - **SPI Port:** Supports the SPI communication protocol and can be accessed through the controller's D2 and D3 digital ports for connection and communication.
  - **CAN Bus:** Supports CAN bus communication and can be connected to the controller's D4 digital interface with a dedicated module.



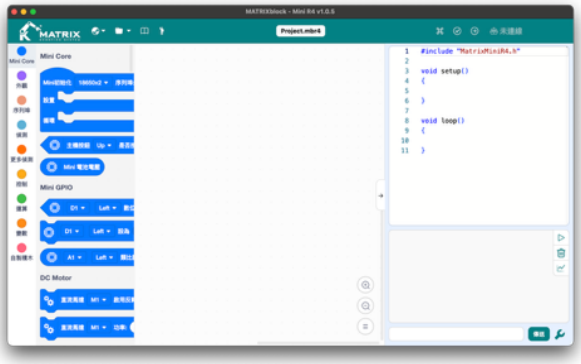
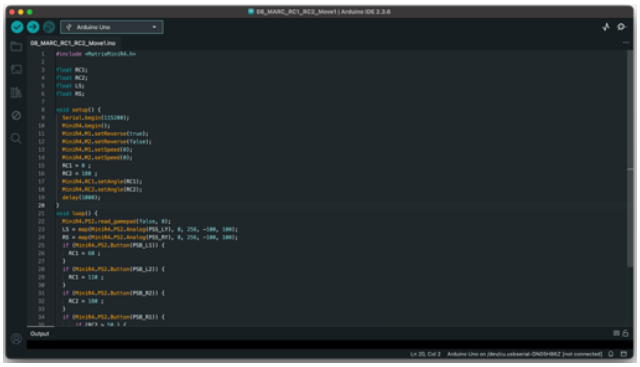
- **Integrated Practical Components**
  - **OLED Display:** Built-in 0.91-inch OLED screen for real-time information display.
  - **IMU:** Onboard IMU capable of detecting orientation and motion data.
  - **Programmable RGB LEDs:** Provides 2 individually controllable full-color RGB LEDs.
  - **Programmable Buttons:** Provides 2 user-programmable buttons for custom functions.
  - **Buzzer:** Onboard buzzer for sound prompts.
  - **Power and Reset Buttons:** Physical buttons for easy operation of power control and system reset.



- **Flexible Power Solutions**
  - **USB-C Interface Functions:** Allows program uploading through the USB-C port and provides basic power to the controller core (only for low-power modules and program testing).
  - **External Power Input:** Supports a wide input range of 6VDC to 24VDC, to provide sufficient power for the system.



- **User-Friendly Software Support**
  - **Multiple Programming Environments:** Supports both MATRIXblock visual programming and Arduino IDE (C++) text-based programming.
  - **Comprehensive Libraries and Examples:** Provides official libraries and application examples to accelerate the development process.
  - **Firmware Update Support:** The controller supports firmware update for ongoing improvements and new features.

	
MATRIXblock	Arduino IDE

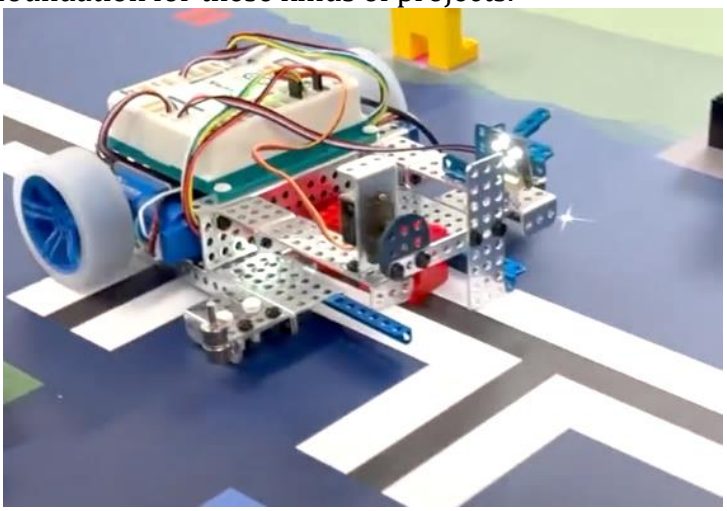
2.3. Applications

With its powerful combination of functions, flexible expandability, and user-friendly development environment, the MATRIX Mini R4 controller has become an ideal choice for a wide range of applications including STEM education, robotics competitions, and creative project development. The Mini R4 can help you whether you are a learner beginning to explore the world of automation or a developer ready to show your talents on the competitive stage.

A. Educational and Learning Projects

The Mini R4 is very suitable for classroom teaching and independent learning, helping students understand programming logic, hardware interaction, and system integration.

- **Basic Robotics Projects:** For example, line-following cars, obstacle-avoiding robots, and light-tracking robots, ect. The Mini R4's precise motor control and diverse sensor interfaces provide a solid foundation for these kinds of projects.



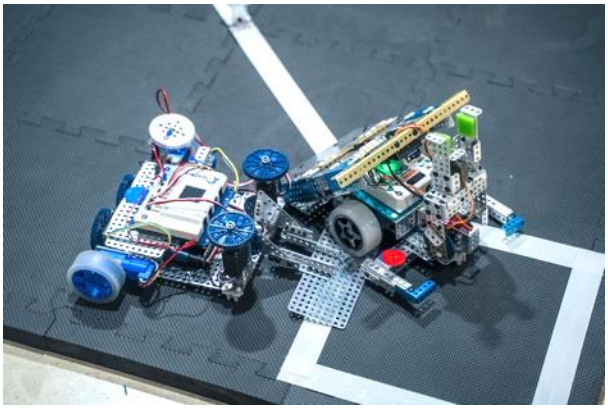
- **Interactive Art Installations**： Combine sensors (such as sound, light, or distance sensors) with output devices (such as LEDs, buzzers, or small motors) to create engaging interactive art pieces or smart toys.

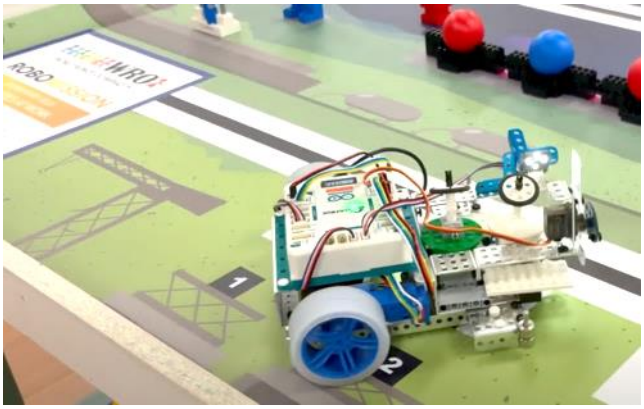
- **Sensor Data Collection and Analysis:** Use the Mini R4 to connect sensors such as temperature and humidity, gas sensors, ect. for environmental monitoring. The built-in wireless capabilities allow data to be transmitted to a computer or the cloud for further analysis.

B. Robotics Competition Platform

The Mini R4's stability, real-time responsiveness, and rich interfaces make it an ideal core controller for a variety of mainstream robotics competitions.

- **World Robot Olympiad (WRO):**
  - **RoboMission:** The Mini R4's multi-motor control, encoder feedback, and sensor integration capabilities make it ideal for building competition robots that perform complex path planning and precise operation tasks.
  - **Future Innovators:** Its built-in Wi-Fi and Bluetooth capabilities, along with expandability for connecting various sensors and modules, strongly supports participants in proposing innovative solutions to real-world problems and creating prototypes.
- **MARC Robot Series:**
  - **MARC 2V2 Alliance Competition:** The Mini R4 can serve as the core controller of robots in alliance competitions. Its stable communication and control performance helps teams gain an advantage in cooperative competitions.
- **Other Suitable Competition Types:**
  - In addition to the competitions mentioned above, the design of the Mini R4 also makes it a strong candidate for use in events such as the "Mobile Robotics" category of the WorldSkills competition, as well as certain divisions with the FIRST (For Inspiration and Recognition of Science and Technology) competition series. It provides participants with a reliable and feature-rich development platform





C. Internet of Things (IoT) and Wireless Interaction Applications

With built-in Wi-Fi and Bluetooth functionality, the Mini R4 can easily enable wireless connectivity and remote control of devices.

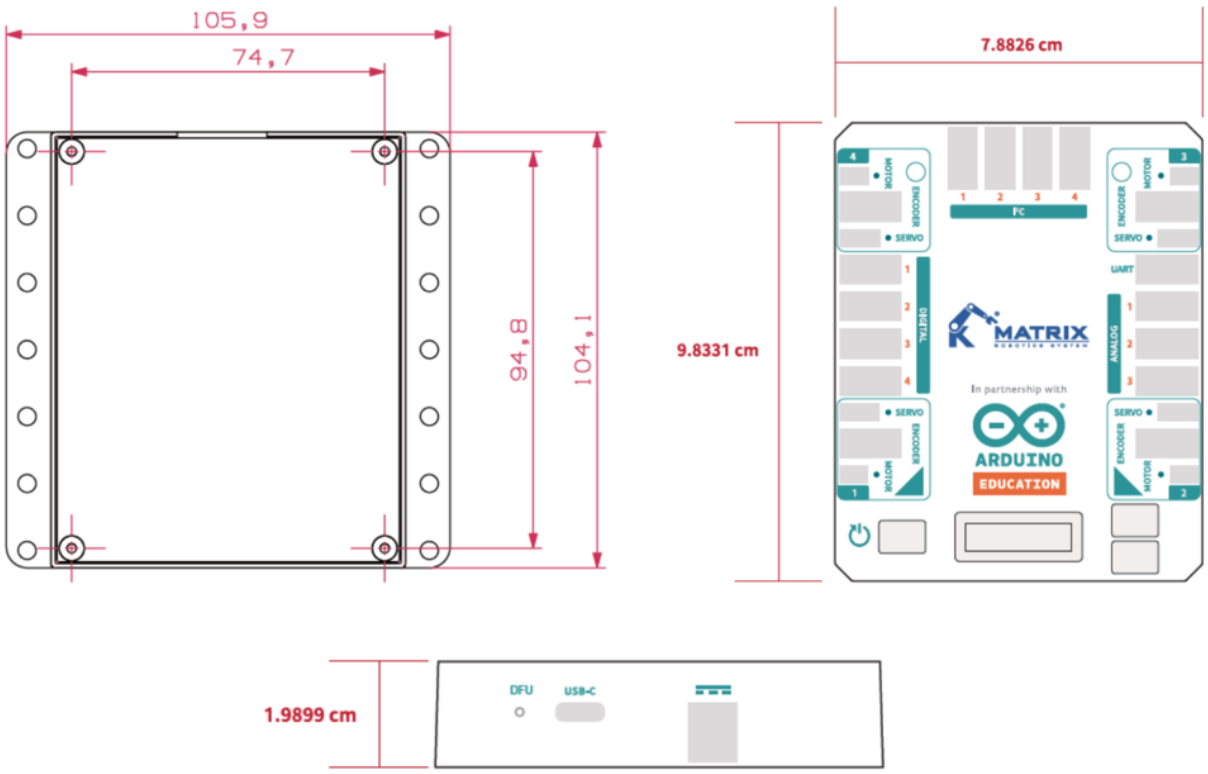
- **Smart Home Prototypes:** For example, remotely controlled lights and fans, or sensor devices that automatically report status.
- **Wireless Remote-Controlled Devices:** Wirelessly control robots or other devices by connecting to a mobile app or dedicated remote controllers (such as the Matrix Joystick 2) via Bluetooth.
- **Mini Weather Stations or Environmental Monitors:** Collect data and upload it to a cloud platform via Wi-Fi, allowing for remote monitoring.

D. Creative Projects and Advanced Development

For experienced developers or users looking to take on more complex projects, Mini R4's dual-core architecture and expandability offers extensive room for innovation and development.

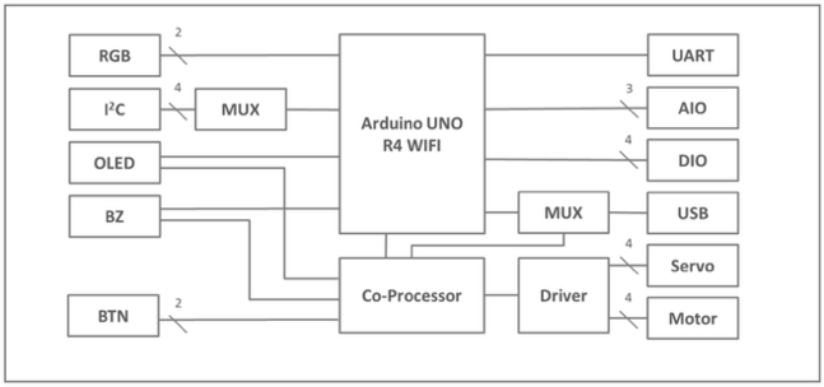
- **Projects with Machine Vision Capabilities:** Can be paired with visual sensors such as the HuskyLens AI camera to perform applications such as color recognition and object tracking.
- **Multi-Axis Robotic Arm Control:** Take advantage of its ability to control multiple servo motors to build and operate small multi-degree-of-freedom robots.
- **Custom Automated Equipment:** Design specialized automation tools or testing platforms tailored to specific needs.

2.4. Dimensions



2.5. Block Diagram

This functional block diagram is designed to help you understand the roles of the main components inside the MATRIX Mini R4 controller and how they interact with each other. Through this diagram, you can gain an initial understanding of the controller's system architecture.



Main Block Descriptions:

- **Arduino UNO R4 WiFi (Main Controller):**
  - This is the core computing unit of the MATRIX Mini R4. It executes the user-developed main program, handles primary control logic, and provides wireless communication features such as Wi-Fi and Bluetooth.
  - It directly manages and controls multiple onboard peripherals such as the OLED display, buzzer (BZ), RGB LED lights, and most common signal interfaces including UART (serial communication port), AIO (analog signal interface), and DIO (digital signal interface).
- **Co-Processor (STM32F103):**
  - To assist the main controller and improve the overall system performance, the Mini R4 is equipped with an STM32F103 co-processor.
  - This co-processor is responsible for handling tasks requiring high real-time performance and precision, such as reading the status of onboard buttons (BTN), and accurately controlling servo motors and DC motors through the motor driver below. This design shares some of the workload from the main controller, allowing it to focus more effectively on core application logic.
- **Driver (Motor Driver):**
  - The purpose of this driver circuit is to amplify the control signals from the co-processor, supplying sufficient current to drive the servo motors and DC motors, ensuring stable operation.
- **MUX (Multiplexer):**
  - A multiplexer (MUX) is an electronic switch that allows the main controller to selectively connect between multiple signal sources or target devices.
  - In the Mini R4, the MUX is used to expand the available interfaces of the main controller. For example, it enables multiple I^2C devices to share the same

communication lines and may also be used to manage multiple functions over the USB interface.

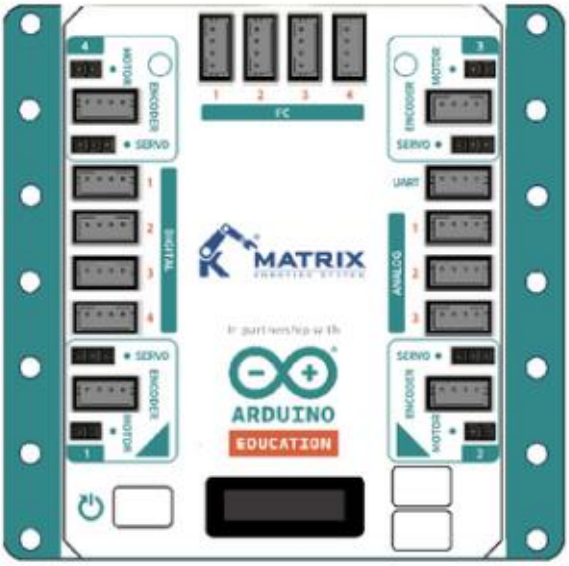
- **Connection Relationships Between Interfaces and Onboard Components:**
  - **RGB (Color LED), OLED (Display), BZ (Buzzer):** These components are directly controlled by the main controller (Arduino UNO R4 WiFi) and are used for status indication or user interaction.
  - **I^2C (Serial Communication Interface):** Connected to the main controller through the MUX, this interface is used to connect various sensors, displays, and other peripheral modules.
  - **BTN (Button):** Connected to the co-processor, it allows users to input commands or trigger events.
  - **UART (Serial Communication Port):** Managed by the main controller, often used for data exchange with other microcontrollers, computers or specific modules.
  - **AIO (Analog Signal Interface):** Managed by the main controller, this interface is used to read continuous signals from analog sensors such as photoresistors or variable resistors.
  - **DIO (Digital Signal Interface):** managed by the main controller, used for processing digital signals such as those from buttons or infrared obstacle avoidance sensors.
  - **USB (Universal Serial Bus Interface):** Primarily used to upload programs to the main controller, provide basic operating power, and serve as a data communication channel.
  - **Servo (Servo Motor Interface) / Motor (DC Motor Interface):** Controlled by the co-processor and motor driver, these interfaces provide power and control for robots or automation components.

In summary, the MATRIX Mini R4 achieves powerful and flexible control capabilities through the division of labor between the main controller (Arduino UNO R4 WiFi) and the co-processor (STM32F103), combined with a variety of interfaces and onboard components. This makes it well-suited to meet the diverse needs of education, competition, and project development.

2.6. Pinout

This section provides a detailed overview of the specific locations and functional definitions of each interface and pin on the MATRIX Mini R4 controller, helping you correctly connect various external modules and sensors.

I²C Pinout - I²C			
No.	Name	I/O	Description
1	SDA	Input/Output	Serial Data Line
2	SCL	Input/Output	Serial Clock Line
3	VCC	Output	Power Supply (3.3V-5V)
4	GND	-	Power Ground



UART Pinout - UART			
No.	Name	I/O	Description
1	TX	Output	Serial Transmit Line
2	RX	Input	Serial Receive Line
3	VCC	Output	Power Supply Output
4	GND	-	Power Ground

Analog Input Pinout - Analog In			
No.	Name	I/O	Description
1	AIN A	Input	Analog Input A
2	AIN B	Input	Analog Input B
3	A5V	Output	Analog 5V Power Out
4	GND	-	Power Ground

Digital I/O Pinout - Digital I/O			
No.	Name	I/O	Description
1	DIO A	I/O	Digital Input/Output Port A (GPIO A)
2	DIO B	I/O	Digital Input/Output Port B (GPIO B)
3	VCC	Output	Power Supply Output
4	GND	-	Power Ground

Encoder Pinout - Encoder			
No.	Name	I/O	Description
1	CH A	Input	Encoder Channel A Signal
2	CH B	Input	Encoder Channel B Signal
3	M5V	Output	Encoder 5V Power Supply
4	GND	-	Power Ground

Servo Output Pinout - Servo Out			
No.	Name	I/O	Description
1	GND	-	Servo Motor Power
2	5V	Output	Power Ground
3	PWM	Output	RC Servo Motor PWM Control Signal

Motor Output Pinout - Motor Out			
No.	Name	I/O	Description
1	M-	Output	H-Bridge Motor Output-Negative
2	M+	Output	H-Bridge Motor Output-Positive

The diagram above shows the main interface layout of the MATRIX Mini R4 controller. The table below details how the main interface pins correspond to the internal pins of the Arduino UNO R4 WiFi or are controlled by the co-processor.

MATRIX Mini R4		Sub-Pin Arduino Uno R4 WiFi Pin/Core	Notes/Related Peripherals
D1	D1A	3	-
	D1B	2	-
D2	D2A	5	-
	D2B	4	-
D3	D3A	12 (SPI_CIPO/MISO)	-
	D3B	11 (PWM, SPI_COPI/MOSI)	-
D4	D4A	13 (SPI_SCK)	-
	D4B	10 (PWM, SPI_CS)	-
A1	A1A	A1	-
	A1B	A0	-
A2	A2A	A3	-
	A2B	A2	-
A3	A3A	A4 (I^2C_SDA1)	-
	A3B	A5 (I^2C_SCL1)	-
UART	TX	1	-
	RX	0	-
I^2C	SDA	PCA9548A-SDA(0–3)	I^2C MUX
	SCL	PCA9548A-SCL(0–3)	I^2C MUX
Features	Buzzer	D6	Onboard
	RGB LED	D7	Onboard
RC Servo		Co-Processor	4 Ports
DC Motor		Co-Processor	4 Ports
Buttons (BTN)		Co-Processor	2 Ports

Important Notes:

- In the “Corresponding Arduino UNO R4 WiFi Pins” column of the table above, values such as D0, D1, A0-A5 refer to the standard Arduino pin names.
- Functions labeled as “Co-Processor” are directly controlled by the onboard STM32F103 co-processor
- The I^2C dedicated interface is connected to the I^2C bus of the main controller via the PCA9548A multiplexer chip.

For more detailed information on usage, electrical characteristics, and programming methods for each interface please refer to the relevant sections later in this guide, such as “2.8 Electrical Characteristics” and “2.8 Communication Interfaces.”

2.7. Electrical Characteristics

To ensure the MATRIX Mini R4 controller operates safely and reliably, understanding its key electrical specifications is essential. The table below lists the controller's electrical characteristic parameters in detail. The following sections will provide a more in-depth explanation of power requirements and I/O characteristics.

Parameter Name	Min	Typical	Max	Unit
Input Voltage	6	-	24	Volts (V)
I/O Pin Voltage	-0.3	5	6.5	Volts (V)
Digital I/O Pin Current	-	-	8	Milliamps (mA)
Analog Input Pin Current	-	-	8	Milliamps (mA)
Servo Output Voltage	-	5	-	Volts (V)
DC Motor Output Voltage	-	5	-	Volts (V)
Servo Output Current (per port)	-	-	1	Amps (A)
Motor Output Current (per port)	-	1.5	2	Amps (A)
UART Baud Rate	300	9600	115200	bit/s
I^2C Transmission Speed	100	-	400	kHz
I^2C Low-Level Input Voltage	-0.5	-	0.33×VCC	-
I^2C High-Level Input Voltage	0.7×VCC	-	VCC	-
LED Red Wavelength	620	-	625	Nanometers (nm)
LED Green Wavelength	522	-	525	Nanometers (nm)
LED Blue Wavelength	465	-	467	Nanometers (nm)
Operating Temperature	-40	25	85	°C

Note: VCC refers to the operating voltage of the controller, which is usually 5V. For detailed pin definitions, please refer to section "2.6 Pinout Definitions" and the corresponding interface description sections.

2.7.1. Power Supply Requirements

The MATRIX Mini R4 controller is designed with a flexible power supply solution to accommodate different application scenarios. Its primary power requirements are based on the "Input Voltage" and related output specifications listed in the table below. Detailed explanations are as follows:

- **External Main Power Input:**
  - **Input Voltage Range:** The controller supports a DC voltage input from 6V to 24V. Users should select a suitable battery or DC power supply within this range based on the specifications of the connected motors or other modules.
  - **Connector Type:** The external power input uses a standard DC power jack with a 5.5mm outer diameter and 2.1mm inner diameter, and the center pin is positive (Center Positive).
  -

    - **Recommended Current Capacity:** Considering that the controller can drive multiple motors simultaneously (each DC motor output typically is 1.5A, with a maximum output current of 2A; each servo maximum output is 1A), it is recommended that the external power supply be capable of providing at least 3A of current. The specific requirement should be evaluated based on the total connected load, especially noting that motors may generate higher inrush currents at startup.
- **USB-C Power Supply:**
    - **Voltage:** When powering the MATRIX Mini R4 through the onboard USB-C port, the input voltage is the standard 5V DC.
    - **Primary Use & Limitations:** This power method is mainly used for uploading programs to the main controller and supplying basic operating power to the controller's core logic circuits. It is suitable for use with low-power modules or during programming and testing only (limited to low-power modules and program testing).
    - **Important Note:** When driving motors, servo motors, or other high-power peripherals, it is strongly recommended to use the above mentioned "External Main Power Input" method instead. Using USB-C alone may result in system instability or damage due to insufficient power supply capacity.
- **Onboard Logic Operating Voltage:**
    - The typical operating voltage for the main digital logic circuits (such as I/O pins) inside the MATRIX Mini R4 controller is 5V.

2.7.2. I/O Voltage and Current Ratings

To protect the MATRIX Mini R4 controller from damage due to mismatched electrical specifications, and to ensure that externally connected sensors and modules function properly, it is important to understand the voltage and current limits of its input/output (I/O) pins.

- **I/O Pin Operating Voltage:**
    - The typical operating voltage for all digital and analog I/O pins on the controller is 5V.
    - Under no circumstances should the voltage applied to any I/O pin exceed its specified range: minimum -0.3V, maximum 6.5V. Voltages exceeding this range may cause permanent damage to the controller.
- **Digital I/O Pin Current Limit:**

    - Each digital I/O pin (whether configured as input or output) can source or withstand a maximum current of 8mA.
    - When using digital pins to drive external components (e.g., LEDs), ensure that the current drawn does not exceed this limit. For components requiring higher current, use external driver circuits (such as transistors, relays, or dedicated driver ICs) instead of driving directly from the I/O pins.
- **Analog Input Pin Current Limit:**
    - Each analog input pin can withstand a maximum input current of 8mA.
    - Although the analog input pins are primarily used for reading voltage signals, in special cases (such as improper circuit design or external component failure), this current limit should still be checked.

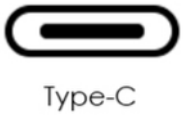
Important Safety Notes:

- Before connecting any external sensor, module, or circuit to the Mini R4's I/O pins, always confirm that the signal voltage levels of the external device are compatible with the Mini R4's 5V logic level.
- For loads requiring a larger drive current, never drive them directly from I/O pins, always add an appropriate drive circuit.
- Improper electrical connections or operation beyond the specified limits may result in malfunction or permanent damage to the controller.

2.8. Control and Communication Interfaces

2.8.1. USB-C

The USB-C port on the MATRIX Mini R4 serves as the primary way to connect to a computer for program development and basic interaction. This port is connected to the STM32F103 co-processor on the board, which helps in handle communication with the computer.



Main Functions:

- **Program Upload:** With the assistance of the STM32F103 co-processor, the USB-C port allows code developed using tools such as the Arduino IDE or MATRIXblock to be uploaded to the core Arduino UNO R4 WiFi microcontroller.
- **Basic Power Supply:** The USB-C port can provide 5VDC power to the core logic circuits of the controller. However, this power source is intended only for program uploading, connecting low-power sensor modules, and conducting initial logic testing.
- **Serial Communication:** Through forwarding by the STM32, the USB-C port enables serial data exchange between the computer and the Arduino UNO R4 WiFi microcontroller. This is particularly useful during program development and debugging. It can be used with the "Serial Monitor" in the Arduino IDE or the "Infor Window" in MATRIXblock to view data or send commands.

Usage Tips and Power Supply Precautions:

- It is recommended to use a high-quality USB-C cable to ensure stable data transmission and power delivery.
- **Important Power Notice:** When powered only through the USB-C port, the controller will not be able to supply sufficient power to drive motors (DC and servo motors). If your project requires driving any motors or other high-power peripherals, be sure to connect and use the controller's external main power input (6V-24VDC) to ensure full functionality and system stability.

2.8.2. Bluetooth

The MATRIX Mini R4, though its Arduino UNO R4 WiFi main controller, has built-in Bluetooth® Low Energy (BLE) functionality. This technology is designed for short-range, low-power wireless communication, allowing the Mini R4 to easily interact with other BLE-enabled devices, such as smartphones, tablets, and more.

Main Applications:

- **Wireless Data Transmission:** Enables small-scale wireless data exchange between the Mini R4 and a smartphone or tablet, such as sending sensor readings or receiving simple commands.
- **Wireless Control and Monitoring:** Users can develop mobile applications (apps) that connect to the Mini R4 via Bluetooth LE for wireless robot control, real-time sensor monitoring, or adjusting operational parameters. For example. Using a mobile app to control an autonomous car or display temperature and humidity readings.
- **Other Potential Applications:** Bluetooth LE technology is also suitable for data synchronization in wearable devices or as the basis for proximity-sensing beacons like iBeacon.

Usage Tips:

- **Development Method:** The Bluetooth LE functionality on the Mini R4 is managed by the Arduino UNO R4 WiFi main control main chip and is usually programmed using relevant Bluetooth libraries available in the Arduino IDE.
- **Bluetooth Gamepad Connection:** To control the Mini R4 using a specific Bluetooth game controller, first ensure that the controller is compatible with the built-in Bluetooth LE protocol. In some cases, an additional Bluetooth adapter or receiver module may be required for successful connection.

2.8.3. Wi-Fi

The MATRIX Mini R4 controller has built-in Wi-Fi connectivity through its core microcontroller, the Arduino UNO R4 WiFi. This allows the Mini R4 to easily connect to existing wireless local area networks (WLANs) and access the internet, unlocking a wide range of possibilities for your projects.

Main Applications and Features:

- **Network Connectivity and Data Exchange:**
    - **Internet Access:** The Mini R4 can access online resources and exchange data with cloud service platforms such as Arduino Cloud, AWS loT, Google Cloud loT, and more. It can also be used to send notifications to your email or messaging apps.
    - **Remote Data Monitoring and Control:** You can develop a web interface or mobile application to remotely monitor sensor data collected by the Mini R4 or to control devices connected to the Mini R4 over the internet.

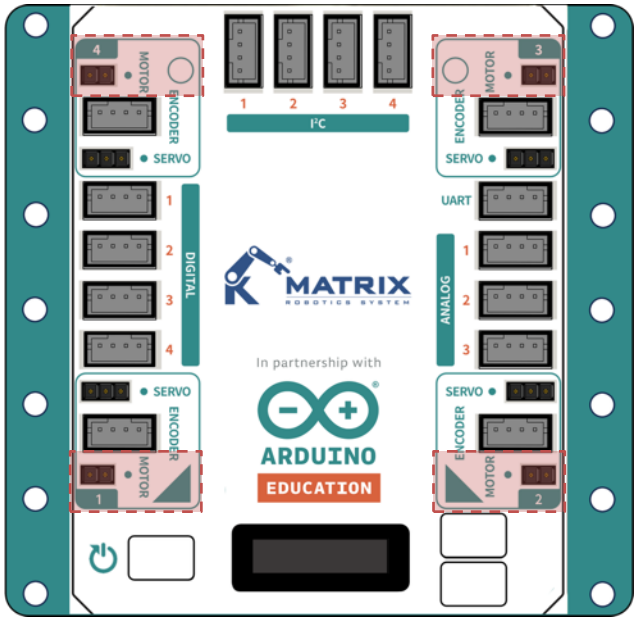
- **Internet of Things (loT) Applications:** Wi-Fi is a key technology for implementing loT projects. You can use the Mini R4 to build various loT devices, such as smart environmental sensors, remote appliance controllers, automated data logging systems, and more.
- **Local Network Service Setup (AP Mode):** In certain applications, the Mini R4 can be configured as a small wireless access point (AP), allowing other Wi-Fi devices to connect directly to the local network created by the Mini R4 for communication. *(Note: Currently, this feature is primarily programmed through the Arduino IDE, and MATRIXblock does not yet support direct AP mode configuration.)*

Usage Tips:

- **Software Development:** In the Arduino IDE, Wi-Fi functionality is typically programmed using specific Wi-Fi libraries, such as the commonly used WiFiS3 library for the Arduino UNO R4 WiFi. These libraries support tasks like scanning available networks, connecting to a designated access point (AP), and sending or receiving data over the network.
- **Network Security:** When connecting the Mini R4 to a Wi-Fi network, ensure that a secure connection is used (e.g., WPA2/WPA3 encryption). Safeguard any credentials, such as passwords, required for network access to protect your device and data.

2.8.4. DC Motor Interface

A DC motor is a basic component commonly used to provide power in robotics projects. The MATRIX Mini R4 controller is equipped with dedicated DC motor interfaces, making it easy for users to connect and control the motor's speed and direction.



Interface Configuration and Pin Functions:

- **Number of Interfaces:** The MATRIX Mini R4 provides 4 DC motor control interfaces.
- **Drive Core:** These interfaces are driven by the onboard STM32 co-processor, which provides stable and efficient control signals
- **Interface Pins (DC MOTOR PORT):**
  - Each DC motor interface is usually a 2-pin design consisting of:
    - **M+**：Motor positive terminal.
    - **M-**：Motor negative terminal.
  - **Important Note:** When connecting the motor, be sure to align the motor's negative wire with the "●" (black dot) marking printed on the controller's interface housing. This black dot usually corresponds to the M- terminal.
- **Output Specifications:** Each DC motor interface provides an output of 5V/1.5A and can support up to 2A current output for short durations, accommodating various types of motors.

Control Principles and Methods:

- **Driving Principle:** The controller uses a built-in motor driver chip (controlled by the STM32) to change the polarity of the voltage applied across the motor terminals, thereby controlling the motor's rotation direction (forward/reverse).

- **Speed Control via PWM:** Motor speed is usually controlled using PWM (Pulse Width Modulation) signals. By adjusting the duty cycle of the PWM signal, the average voltage supplied to the motor can be modified, allowing control over its speed. A higher duty cycle corresponds to a higher average voltage, which generally results in a faster motor speed.
- **Direction Control:** In addition to speed, the rotation direction of the motor can be controlled through software. This is usually done by setting specific control pins on the motor driver chip.

Compatible Motor Types:

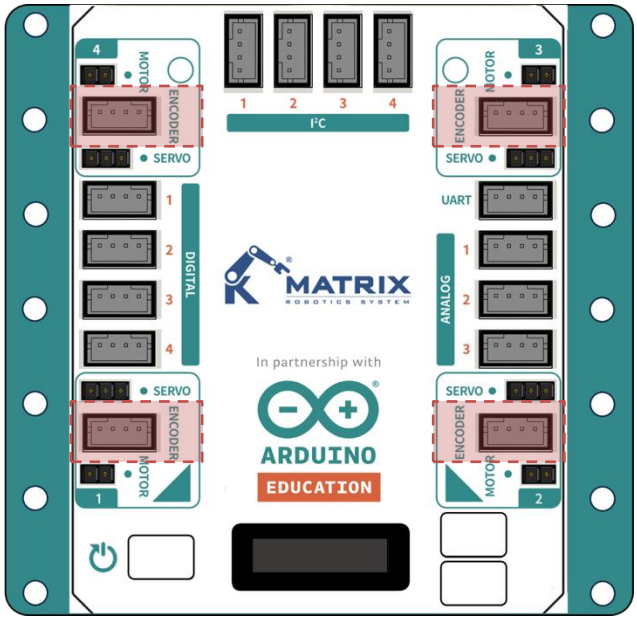
- Suitable for connecting common small DC motors with gearboxes, such as toy motors or TT motors, ect. These motors should operate at around 5V, with both their normal operating current and stall current falling within the interface's output specifications (1.5A typical, 2A peak)
- It is **not recommended** to directly connect high-power motors that exceed these voltage or current specifications, as this may damage the controller.

Usage Tips:

- **Always Use an External Power Supply:** Relying solely on USB-C power is insufficient for driving motors. Since DC motors consume relatively high current, the MATRIX Mini R4 must be powered through an external main power source (6V-24VDC) to ensure proper motor operation and overall system stability.
- **Verify Motor Specifications:** Before connecting third-party DC motors, always confirm that their operating voltage and current draw are within the output limits of the Mini R4's motor interface. This helps ensure stable operation and protects your equipment.
- **Software Control:** Both MATRIXblock and the Arduino IDE provide corresponding commands or libraries to control motor speed and direction through code.

2.8.5. Encoder Interface

A rotary encoder is a sensor used to detect a motor's rotation angle, speed, and direction. It is commonly used alongside DC motors to enable more precise closed-loop control. The MATRIX Mini R4 controller provides dedicated encoder interfaces, to facilitate the connection and integrate of these sensors.



Function and Working Principle of Encoders:

- **Functions:**
  - **Measuring Rotation Speed:** By counting the number of pulses outputed by the encoder within a specific time period, the actual rotation speed of the motor can be calculated.
  - **Measuring Rotation Angle/Distance:** Each pulse corresponds to small rotation of the motor shaft. By accumulating pulse counts, the total rotation angle or the corresponding travel distance can be determined.
  - **Determining Rotation Direction:** Most incremental rotary encoders output two pulse signals (Channel A and Channel B) with a phase difference of 90 degrees. By comparing the sequence in which these signals occur, the motor's rotation direction (forward or reverse) can be determined.
- **A/B Phase Signal Principle (Quadrature Encoding):**
  - Inside the encoder, there is usually a disk with evenly spaced markings. As the motor rotates, a photosensor detects the passing or blocking of light through these markings, generating pulse signals.
  - Channels A and B are two such pulse signals. Their waveforms are offset by one-quarter of a pulse cycle in time (i.e., a 90-degree phase difference). When the motor rotates forwards, Channel A leads Channel B, when rotating in reverse Channel B

leads Channel A. By monitoring the phase relationship between these two signals, the controller can determine the direction of rotation.

Interface Configuration and Main Signals:

- **Number of interfaces:** The MATRIX Mini R4 is equipped with 4 encoder interfaces.
- **Connector Type:** These interfaces use 4-pin JST connectors, making it easy to connect DC motors with built-in encoders or standalone encoder modules.
- **Main Signals:** Each interface provides inputs for the encoder's two pulse signals, Channel A (CH A) and Channel B (CH B), as well as 5V power (M5V) and Ground (GND). For the exact pin order and detailed definitions of each pin, please refer to section **2.6 Pinout Definitions** under the encoder interface description.
- **Connection Convenience:** The use of JST connectors and dedicated cables helps ensure correct connections and reduces the risk of wiring errors.

Implementing Closed-Loop Control:

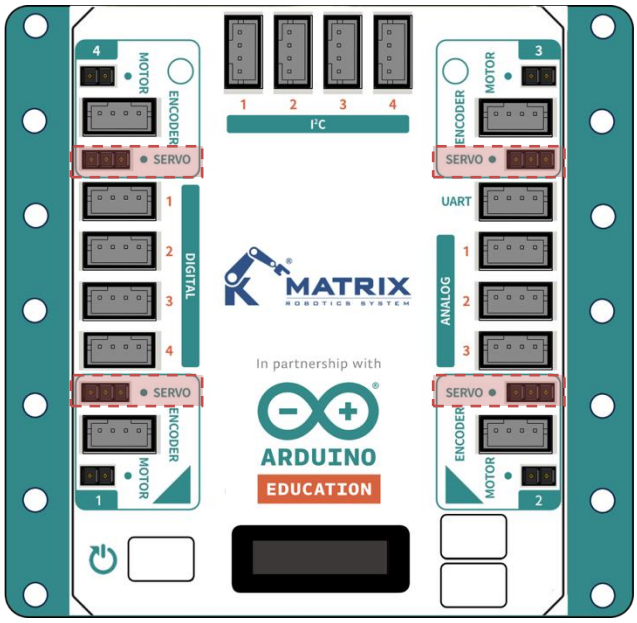
- By reading the encoder's returned A/B channel signals, the controller can instantly determine the motor's actual operating status -- speed, direction, and position.
- By comparing the actual status with the desired target values, the controller can dynamically adjust the motor's drive signals (e.g. by modifying the PWM output) to reduce errors. This real-time feedback and adjustment process is the core concept of **closed-loop control**, enabling more precise and stable motor operation.

Usage Tips:

- **Signal Reading:** The A and B channel signals are typically connected to microcontroller pins that support external interrupts or counting functions, allowing precise pulse detection and calculation. On the Mini R4, these signals are handled by the **STM32 co-processor**.
- **Software Support:** When programming, specific libraries or custom logic are required to decode A/B signals in order to calculate motor speed, direction, and pulse count.
- **Use with Motor Interface:** Encoder interfaces are typically used in conjunction with the DC motor interfaces described in **section 2.8.4**, one for controlling motor power, and the other for reading motor status,

2.8.6. Servo Motor Interface

A servo motor, especially the commonly used RC (Radio Control) servo motor in model-making, is a type of actuator capable of precisely rotating its output shaft to a specified angle. The MATRIX Mini R4 controller provides multiple dedicated servo motor interfaces, making it easy for users to drive mechanisms that require accurate angular control, such as robotic arms, joints, or steering systems.



Basic Principles and PWM Control of Servo Motors:

- **Basic Structure:** An RC servo motor usually contains a small DC motor, a gear reduction set, a position sensor (usually a potentiometer), and a control circuit board.
- **Built-in Closed-Loop Control:** The internal control circuit compares the external control signal (desired angle) with feedback from the position sensor (actual angle). It then drives the internal DC motor until the output shaft reaches the target angle. This is a built-in form of closed-loop control.
- **PWM-Based Angle Control:** The angle of an RC servo motor is determined by the with of the incoming PWM (Pulse Width Modulation) signal.
    - The typical PWM signal has a 20 ms period (i.e. a 50 Hz frequency).
    - **PWM Control Angle:** The angle of an RC servo motor is determined by the pulse width of the incoming PWM (Pulse Width Modulation) signal. Typically, the PWM signal has a 20 ms period (50 Hz frequency), and the output angle is controlled by varying the pulse width (usually from 0.5 ms to 2.5 ms) with different widths corresponding to different target angles.

Interface Configuration and Main Signals:

- **Number of Interfaces:** The MATRIX Mini R4 provides 4 PWM control interfaces for servo motors.
- **Drive Core:** These interfaces are controlled by the onboard STM32 co-processor, which generates precise PWM signals required for accurate servo positioning.
- **Connector Type and Main Signals (RC SERVO PORT):** Each servo interface typically uses a 3-pin connector, which includes:
    - **PWM:** Signal pin that carries the PWM control signal.
    - **M5V:** 5V power supply for the servo motor.
    - **GND:** Ground connection.
    - **Connector Type and Main Signals (RC SERVO PORT):** Each servo motor interface uses a 3-pin design, providing the control signal line (PWM), 5V power (M5V), and ground (GND) required for servo motor operation. For the specific pin order, definitions, and important connection orientation indicators (such as how the ground wire aligns with markings on the connector), please refer to the description of the servo motor interface in the "2.6 Pinout Definitions" section.
- **Output Specifications:** Each servo motor interface supports an output of 5V/1A, sufficient to meet the voltage and current requirements of most standard RC servo motors.

Suitable Servo Motor Types and Specifications:

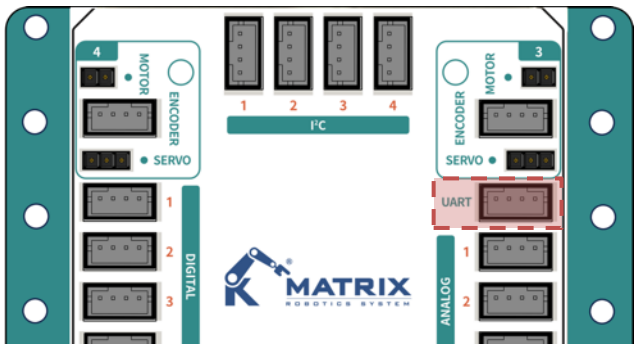
- Primarily designed for use with common RC (Radio Control) standard servo motors, such as SG90, MG995, MG996R, and other small to medium-sized servo motors.
- When purchasing third-party servo motors, users are advised to first confirm that the operating voltage is 5V, and that both the operating current and stall current fall with the output capability of the interface (maximum 1A), to ensure stable system operation.

Usage Tips:

- **Important Power Notice:** Powering the system through USB-C alone is not sufficient to drive servo motors. When operating servo motors, especially multiple motors simultaneously, the total current consumption can be significant. Therefore, it is essential to use an external main power supply (6V-24VDC) to power the MATRIX Mini R4 in order to ensure proper servo operation and overall system stability.
- **Carefully Check Wiring Orientation:** Although servo motor connectors are integrated, it does not have a fool-proof design. Careful attention is needed when connecting. Always verify the position of the signal wire, power wire (usually red), and ground wire (usually black or brown). Make sure that the ground wire is aligned with the "●" (black dot) marking printed on the controller's connector housing. Incorrect wiring may damage the servo motor or the controller.

2.8.7. UART (Serial Communication Port)

UART (Universal Asynchronous Receiver-Transmitter) is a common and fundamental hardware-based serial communication method that allows the MATRIX Mini R4 to exchange data point-to-point with other devices that also support UART.



Main Features and Applications:

- **Basic Principle:** UART communication operates using two primary signal lines: one for receiving data (RX – Receive) and one for transmitting data (TX – Transmit). Data is transmitted asynchronously as a stream of bits, meaning no additional clock signal is required to synchronize the sender and receiver.
- **Connection Pins:** The MATRIX Mini R4 controller provides one UART interface, mapped to D0 (RX0) and D1 (TX0) pins on the Arduino UNO R4 WiFi main controller.
- **Full-Duplex Communication:** This UART interface supports full-duplex communication, allowing the controller to send and receive data simultaneously.
- **Common Applications:**
    - **Communication with other microcontrollers or developmental boards:** For example, enabling the Mini R4 to exchange data with another Arduino board, a Raspberry Pi, or other brands of microcontrollers.
    - **Connecting specific sensors or modules:** Some GPS modules, fingerprint scanners, and certain types of sensors or displays use UART interfaces for data transmission.
    - **Connecting MATRIX-specific modules and other external devices:** For instance, the MATRIX M-Vision smart camera can be connected via UART to transmit visual recognition data, or the MATRIX Motor Controller can exchange commands and status information. It also supports custom hardware or modules with UART communication capabilities.
    - **Debugging and monitoring:** Although the USB-C serial connection is commonly used for debugging during development, in some cases, the dedicated UART interface can also be used for runtime logging or system monitoring in certain cases.

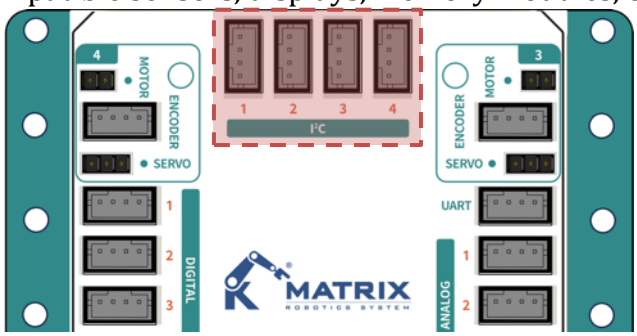
Usage Tips:

- **Voltage Level:** When connecting external UART devices, always confirm that their signal voltage levels are compatible with the MATRIX Mini R4 (typically 5V logic level) to avoid damage.
- **Baud Rate:** Both devices involved in UART communication must be set to the dame baud rate (i.e, data transmission speed, measured in bits per second) in order to properly decode messages. Mini R4 supports a variety of common baud rates (e.g., 9600, 115200, ect.).
- **RX/TX Cross Connection:** When connecting two UART devices, it's usually necessary to connect the TX pin of one device to the RX pin of the other, and vice versa.
- **Relationship with USB Serial Port:** On the Arduino UNO R4 WiFi, pins D0 (RX0) and D1 (TX0) are also connected to the onboard USB-to-Serial chip. This means that when you're

using the Serial Monitor via USB-C, you cannot connect external devices to D0/D1 for UART communication at the same time, as this may cause conflicts.

2.8.8. I²C (Internal Integrated Circuit Interface)

I²C (Inter-Integrated Circuit, also commonly written as I2C or IIC) is a widely used serial communication protocol that requires only two main signal lines -- SDA (Serial Data Line) and SCL (Serial Clock Line) -- to enable communication between a master device and subordinate devices. The MATRIX Mini R4 provides as I²C provides as convenient I²C interface, allowing users to easily connect a variety of I²C compatible sensors, displays, memory modules, or other devices.



Main Features and Applications:

- **Two-Wire Communication:** The most notable feature of I²C is that it requires only two signal lines (SDA and SCL) to operate. Along with power (VCC) and ground (GND), it is typically provided as a 4-pin interface.
- **Multiple Device Support:** Multiple subordinate devices can be connected on the same I²C bus. Each device has a unique address, and the master controller communicates with a specific device by calling its address.
- **Onboard I²C Interface:**
  - The MATRIX Mini R4 controller is equipped with 4 dedicated I²C ports on the top (labeled I²C Port 1 to 4). These ports are provided through and onboard I²C multiplexer (PCA9548A), allowing users to stably connect and manage multiple I²C devices simultaneously.
  - Additionally, the native I²C interface of the Arduino UNO R4 WiFi's main controller chip is also retained. Users can access it through the analog interface A3 on the controller, specifically using the A3A pin as SDA1 and the A3B pin as SCL1.
- **Common Applications:**
  - **Sensor Connections:** Many environmental sensors (such as temperature and humidity sensors, barometers, light sensors), motion sensors (such as accelerometers, gyroscopes), and distance sensors (for example certain laser rangefinders) all use the I²C interface.
  - **Display Modules:** Common OLED or LCD character/graphic display modules also often communicate with the main controller via I²C.
  - **Expansion Modules:** For example, RTC (Real-Time Clock modules), EEPROM (memory modules), and more.

Tips for Use:

- **Pull-up Resistors:** The SDA and SCL lines in I²C communication typically require pull-up resistors to function properly. The dedicated I²C interface on the MATRIX Mini R4 includes

built-in pull-up resistors. However, when using the native I^2C interface on the A3 pins or connecting multiple devices, you may need to verify or add external pull-up resistors depending on the situation.

- **Device Address:** Before connecting an I^2C device, make sure to check its address to avoid conflicts with other devices on the same bus.
- **Transmission Speed:** I^2C supports different data rates, such as Standard Mode (100 kHz) and Fast Mode (400 kHz). The Mini R4's I^2C interface supports transmission speeds from 100 kHz to 400 kHz (refer to the "2.7 Electrical Characteristics" table for details).
- **Software Development:** In the Arduino IDE, the built-in Wire.h library is used for programming and communicating with I^2C devices.

2.8.9. SPI (Serial Peripheral Interface)

SPI (Serial Peripheral Interface) is a high-speed, full-duplex, synchronous serial communication protocol. It is commonly used for short-distance communication between microcontrollers and various peripheral chips such as memory, sensors, and displays.

Main Features:

- **Typical Four-Wire Communication:** Standard SPI communication primarily uses four signal lines: SCLK (Clock), MOSI (Master Out Subordinate In), MISO (Master In Subordinate Out), and CS (Chip Select).
- **High-Speed Transmission:** Compared to I^2C, SPI usually achieves higher data transmission rates.
- **Full-Duplex Communication:** Allows data to be sent and received simultaneously.

Connection and Applications:

- **Connection Pins:** The MATRIX Mini R4 provides SPI functionality through its digital interfaces. Users can connect SPI devices using specific pins on the D2 and D3 digital interfaces. These pins correspond to the standard SPI communication pins on the Arduino UNO R4 WiFi's main controller chip (D10/SS, D11/MOSI, D12/MISO, D13/SCK). For detailed pin mapping, please refer to section "2.6 Pin Definitions (Pinout)".

- **Primary Application Example – Connecting the MATRIX Joystick (MJ2) Receiver Adapter Board:**
  - Currently, the most common application of the SPI interface on the Mini R4 is to connect the receiver adapter board of the MATRIX Joystick (MJ2) wireless controller. Through this connection, users can conveniently use the MJ2 joystick to wirelessly control robots or other projects.
- **Other Potential Applications:**
  - In theory, the SPI interface can also be used to connect other standard SPI peripheral devices, such as external Flash memory, certain types of high-speed ADC/DAC modules, some high-resolution sensors (like specific IMUs), SD card reader/writer modules, or specific display screens.
  - *(Please note: These additional applications require users to have more advanced hardware knowledge and programming skills, and to confirm the electrical compatibility of the related modules with the Mini R4, as well as the availability of relevant supporting software libraries.)*

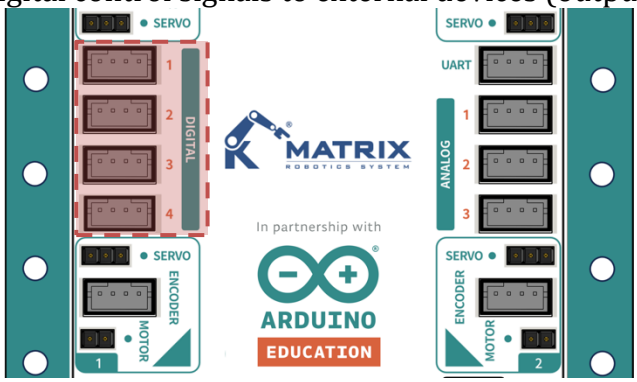
Usage Tips:

- **Master-Subordinate Architecture:** SPI communication is based on a master-subordinate architecture with the MATRIX Mini R4 typically acting as the master device.
- **CS/SS Pin:** When connecting SPI devices, the CS (Chip Select) pin is used to select the subordinate device for communication.

- **SPI Modules:** SPI communication supports multiple operating modes (Mode 0, 1, 2, 3), determined by Clock Polarity (CPOL) and Clock Phase (CPHA). When connecting, ensure that both the master and subordinate devices are consistent and set to the same SPI mode.
- **Software Development:** In the Arduino IDE, SPI device programming and communication are typically done using Arduino's standard SPI.h library.

2.8.10.Digital Signal Interface (Digital I/O)

The MATRIX Mini R4 provides multiple digital signal interfaces (Digital I/O) for handling binary signals such as ON/OFF states (also referred to as high/low voltage levels). These interfaces are versatile and can be configured either to read digital signals from external sources (input mode) or to send digital control signals to external devices (output mode).



Interface Overview and Main Functions:

- **Interface Configuration:** The controller is equipped with 4 sets of digital signal interfaces, labeled D1, D2, D3, and D4. Each set usually includes two independently usable digital signal pins, along with a 5V power supply and (GND), making it convenient for connecting sensors or modules.
- **Core Connection:** The signal pins of these digital interfaces are connected to the digital pins of the Arduino UNO R4 WiFi main controller chip. For detailed pin mapping, please refer to section "2.6 Pin Definitions (Pinout)".
- **Basic Operations:**
  - **Digital Input:** Used to detect digital states such as whether a button is pressed or whether an infrared sensor has detected an object.
  - **Digital Output:** Used to control digital devices such as turning LEDs lights, activating relays, and more.
- **Pulse Width Modulation (PWM):**
  - Certain digital pins support PWM functionality, which is a technique that simulates analog voltage output by quickly switching a digital signal on and off at different duty cycles.
  - This function is commonly used to control LED brightness, adjust the speed of DC motors (when used with a motor driver), or generate simple tones. For details on which pins support PWM, please refer to section "2.6 Pin Definitions (Pinout)".

Common Application Examples:

- Connecting buttons, micro switches, ect., as user input.
- Driving LEDs for status indication.

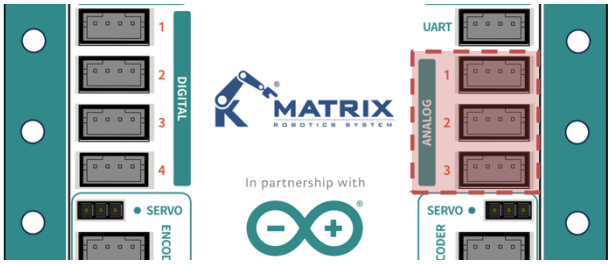
- Connecting infrared obstacle avoidance sensors or ultrasonic distance sensors.
- Using PWM functionality to create a breathing LED light effect.

Usage Tips:

- **Voltage Levels and Current:** The logic high level of the digital interfaces is 5V. When connecting external devices, be sure to confirm signal level compatibility and pay attention to the current limits of each pin (see section "2.7.2 I/O Voltage and Current Limits"). If you need to drive devices requiring higher current, use an external driver circuit.
- **Program Configuration:** In the program, pins must first be set to INPUT or OUTPUT mode. If using PWM functionality, appropriate control must also be implemented in the code.

2.8.11. Analog Signal Interface (Analog I/O)

Analog signal interfaces (Analog I/O) are mainly used to read signals that change continuously in value – for example, brightness changes from a photoresistor, rotation angle from a potentiometer, or voltage rates returned by certain temperature sensors. Through these interfaces, the MATRIX Mini R4 can sense subtle analog variations in the environment.



Interface Overview and Main Functions:

- **Interface Configuration:** The controller is equipped with 3 sets of analog signal interfaces, labeled A1, A2, and A3. Each analog interface typically includes two independently usable analog input pins, along with 5V power and ground (GND) pins, making it convenient for connecting sensor modules.
- **Core Connection:** The signal pins of these analog interfaces are connected to the Analog-to-Digital Converter (ADC) input pins of the Arduino UNO R4 WiFi main controller chip. They correspond to the A0 to A5 analog pins on the Arduino UNO R4 WiFi. For detailed pin mapping, please refer to section "2.6 Pin Definitions (Pinout)".
- **Analog Input:**
  - The main function is to read the voltage value on the input pin and convert it into a digital value (usually from 0 to 1023, representing a voltage range from 0V to 5V; the exact resolution depends on the ADC configuration).
  - By reading these values, the program can determine factors such as light intensity, knob position, ambient temperature, and more.
- **I^2C Function Multiplexing (A3 Interface):** It is important to note that the A3 analog interface's A3A and A3B pins also serve as the native I^2C interface of the Arduino UNO R4 WiFi main controller chip (corresponding to A4/SDA1 and A5/SCL1, respectively).

Therefore, when the A3 interface is used for I^2C communication, these pins cannot be used simultaneously as pure analog inputs.

Common Application Examples:

- Connecting a photoresistor to detect ambient light intensity.
- Connecting a potentiometer (knob) to use as a parameter adjustment or control input.
- Connecting a microphone sound sensor to detect sound levels.
- Connecting temperature or humidity sensors that output analog voltage.

Usage Tips:

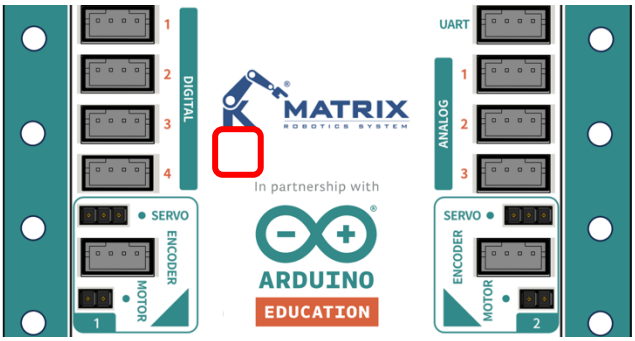
- **Input Voltage Range:** Analog input pins are typically designed to read voltages between 0V and 5V. Applying voltages outside this range may damage the controller.
- **Resolution:** Arduino's ADC usually has a 10-bit resolution, meaning it can convert an input analog voltage into a digital value between 0 and 1023.
- **Signal Stability:** Analog signals are prone to noise interference. In precision measurement applications, you may need to pay attention circuit layout or by adding filtering measures.

2.9. Indicator Lights and Status Signals

The MATRIX Mini R4 controller is equipped with several indicator lights that help you quickly understand the controller's basic power status, operating mode, and provide customizable visual feedback functionality.

Power Indicator LED

- **Location:**



- **Status Description:**
  - **Solid green light:** Indicates that the controller is correctly powered and the power supply is normal.
  - **Flashing red light:** Indicates insufficient voltage or abnormal power supply is detected. Please check whether your power connection and supply meet the specifications outlined in section "2.7.1 Power Requirements."

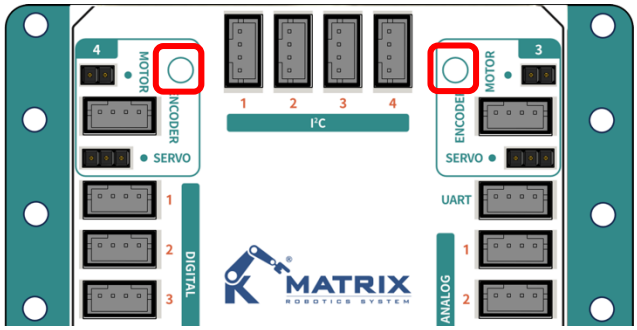
DFU Mode Indicator Light

- **Location:**

- **Status Description:**
  - **Flashing blue light:** Indicates that the controller has entered DFU (Device Firmware Update) mode. This mode is usually used for firmware updates or system recovery operations. For detailed steps on how to enter DFU mode and perform a firmware update, please refer to section "3.4 Firmware Update" of this guide.

Programmable RGB LEDs

- **Quality and Location:** The MATRIX Mini R4 is equipped with two onboard programmable RGB LEDs.



- **Function:** These two RGB LEDs are fully customizable and controlled by the user through programming, with no predefined indicator functions.
- **Suggested Applications:**
    - **Custom Status Display:** You can write programs to display different colors or blinking patterns based on various operating states (for example: Wi-Fi connected, Bluetooth paired, specific task completed).
    - **Visual Effects:** Used in projects to enhance interactivity or aesthetics, such as creating breathing lights, flowing light effects, and more.
    - **Debugging Aid:** During program development, you can use the RGB LED color changes to indicate that the program has reached certain stages or when variables meet specific conditions, serving as a simple visual debugging tool.
- **Control Method:** These two RGB LEDs correspond to specific digital pins on the Arduino UNO R4 WiFi main controller chip. Users can control the color and brightness via graphical commands in the MATRIXblock or by using relevant libraries in the Arduino IDE. For detailed pin assignments, please refer to section “2.6 Pin Definitions (Pinout).”

2.10. Hardware Usage Guide

Proper and safe operation and connection of the MATRIX Mini R4 controller are key to ensuring its normal operation and extending its lifespan. This section provides some basic hardware usage guidelines, especially suitable for first-time users of the controller.

A. Initial Use and Basic Connections

- **Unboxing Inspection:** After receiving the controller, first check its appearance for any obvious physical damage.
- **Power Connection:**
  - **Prefer External Power Supply:** If your project involves motors, multiple servos, or other high-power consumption modules, be sure to use an external power source that meets the specifications in section “2.7.1 Power Requirements” (6V-24V DC, connected via DC Jack).
  - **USB-C Power Limitations:** The USB-C port is primarily used for program uploads and basic power supply to the controller core. When powered only via USB-C, motors cannot be driven.
- **Controller Power On/Off and Reset:**
  - On the bottom left of the controller's front face, there is a button marked with the power symbol "⏻". This button serves dual functions for power on/off and reset.
  - **Power On/Off: Press and hold this button for 3 seconds** to turn the controller on or off.
  - **Reset: A short press of this button** will reset the controller, causing it to restart and run the program currently stored in its memory.
- **Connecting to a Computer:** Use a high-quality USB-C cable to connect the controller to a computer. This allows for program uploads and serial communication.

B. General Principles for Connecting External Modules and Sensors

- **JST Connectors and Dedicated Cables:** Many sensor and encoder interfaces on the MATRIX Mini R4 use JST connectors. It is recommended to use official MATRIX cables or compatible dedicated connection cables. This design includes foolproof features that significantly reduce the risk of damage caused by incorrect pin connections.
- **Verify Voltage Compatibility:** When connecting modules that do not use JST cables (e.g., using Dupont wires directly to I/O pins), make sure that the module’s operating voltage and signal levels are compatible with the Mini R4. The typical operating voltage for Mini R4’s I/O pins is 5V.
- **Double-check Pinout:** When wiring manually, carefully verify the VCC (power), GND (ground), and Signal pins to ensure correct connections. Incorrect wiring may damage the controller of the external module.
- **Avoid Hot-Plugging (Recommended):** To best protect electronic components, it is recommended to power off the controller before connecting or disconnecting external modules and cables.

- **Electrostatic Discharge (ESD) Protection:** In dry environments or situations where electricity is likely, it is advised to take proper ESD precautions, such as touching a metal object to discharge static from your body, before handling the controller.

C. Special Notes on Motor Connections

- **TT DC Motors and RC Servo Motors:**
  - When connecting a TT DC motor to the labeled "Motor" port or an RC servo motor to the labeled "Servo" port, pay close attention to the plug orientation.
  - **Important Tip:** Be sure to align the M- (motor negative) or GND (servo motor ground / black or brown wire) with the ● (black dot) marking printed on the controller's port casing. Correct alignment is crucial for proper motor operation and to prevent damage.
- **Encoder Motors:** If using a DC motor with an encoder, connect the motor power wires to the "Motor" port and connect the encoder signal wires to the corresponding "Encoder" port. The Encoder port also uses a JST connector, which helps prevent incorrect connections.
- **Always Use an External Power Source:** To reiterate when driving any motors (DC motors or servo motors), you must use an external main power supply to power the Mini R4.

D. Safety Operating Instructions

- **Avoid Liquids and Conductive Materials:** Ensure the controller is kept away from liquids to prevent any spillage onto the circuit board. Also, prevent foreign objects such as metal shavings or conductive dust from contacting the circuit board to avoid short circuits.
- **Operating Environment:** Use the controller in a dry, well-ventilated environment with a suitable temperature. Avoid prolonged exposure to high temperatures, high humidity, or dusty conditions.
- **Gentle Handling:** When connecting or disconnecting cables or modules, handle them gently to avoid applying excessive force to the connectors.

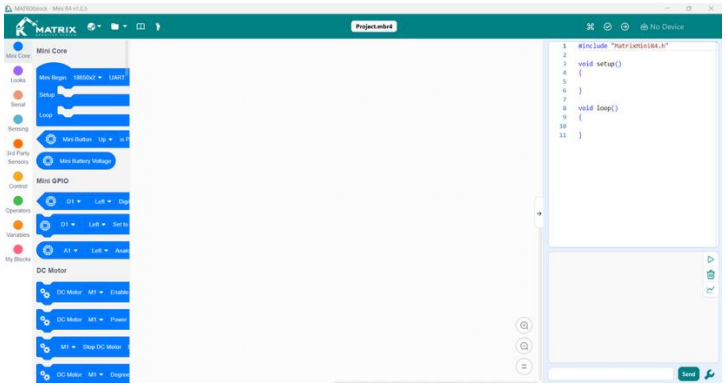
E. Basic Troubleshooting Tips

- **Controller Not Responding:** First, check whether the power connection is firmly secured and the power supply is functioning normally (refer to section "2.9 Indicator Lights and Status Signals" for the power indicator light status).
- **Module Not Working:** Check whether the module's wiring is correct and firmly secured, and verify that the program logic is correct.

2.11. Programming & API Overview

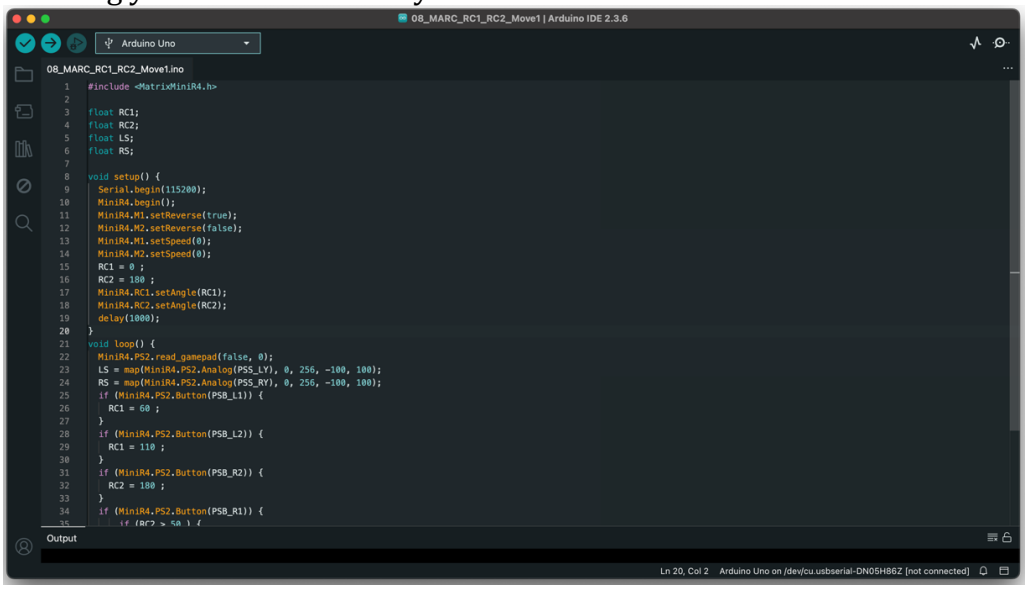
The full potential of hardware is realized through software. The powerful capabilities of the MATRIX Mini R4 controller ultimately rely on programming to enable specific applications. To accommodate users with varying levels of experience, from intuitive graphical interfaces to more flexible text-based programming, the Mini R4 offers multiple development options.
This section will introduce you to two primary software development tools:

- **MATRIXblock:** A graphical programming software designed specifically for beginners and educational settings. It allows you to easily control the Mini R4 by dragging and dropping visual blocks.



- **Arduino IDE:** A popular integrated development environment among makers and developers worldwide, it supports programming in C/C++ for more in-depth and low-level development. It also allows full utilization of Arduino's rich ecosystem of libraries.

The following subsections will introduce the basic operations and key features of these two tools, helping you choose the method that best suits your needs for interacting with the MATRIX Mini R4, and turning your ideas into reality.

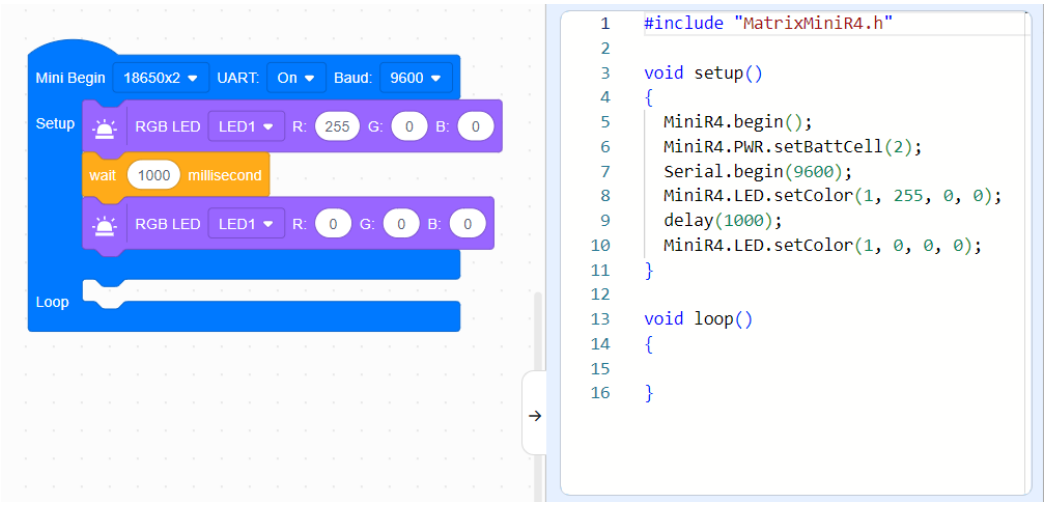


2.11.1.Using MATRIXblock

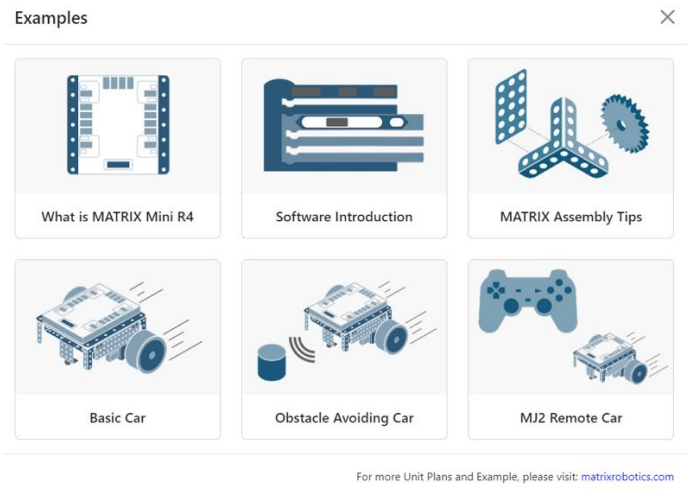
MATRIXblock is graphical programming software specifically designed for the MATRIX series controllers (including the Mini R4). It was developed based on Scratch and provides a programming environment that is friendly and familiar to both students and teachers.

Core Features and Advantages:

- **Block-Based Programming:** Users can build program logic by dragging and connecting different functional blocks, significantly lowering the barrier to learning programming.
- **Scratch-Based, Hardware-Optimized:** For teachers and students experienced in Scratch, it is easy to get started quickly. MATRIXblock also offers dedicated blocks that are tightly integrated with MATRIX hardware.
- **Real-Time C++ Code Display:** A major feature of MATRIXblock is its ability to display the corresponding C++ code in real time as you drag and arrange graphical blocks. This makes MATRIXblock an ideal transitional and entry-level tool for moving from graphical programming to text-based languages like C++.



- **Built-in Serial Monitor:** The software has a built-in serial monitor, even including a graph view for real-time tracking and visual representation of sensor data.
- **Multi-Language Support:** Supports multiple language interfaces, allowing students to read and annotate programming blocks in their native language.
- **Built-in Tutorial and Resources:** The software includes guided tutorials to help users get familiar with basic operations, and the official team plans to provide additional learning resources.

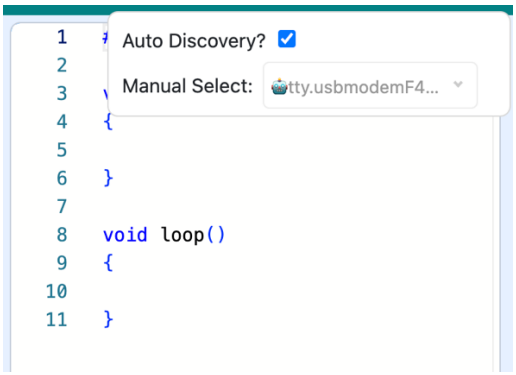


Software Download and Basic Setup:

A. **Download the Software:**
   - You can download the MATRIXblock software from the official MATRIX Robotics website download page (URL: https://www.matrixrobotics.com/matrixblock-software).

   - When downloading, be sure to select the version compatible with the MATRIX Mini R4, not the version for the older Mini (R3) controller, to ensure full compatibility with your hardware.
   - The software supports Windows (Windows 10, 11 x64) and macOS (Monterey or later, supporting both Arm and Intel CPUs) operating systems.

B. **Installation and Connection:**
   - Install the software according to the standard procedures for your operating system.
   - Connect the MATRIX Mini R4 controller to your computer using a USB-C cable.
   - After launching the MATRIXblock software, it will usually automatically detect and attempt to connect to the connected MATRIX Mini R4 controller. You can also check or manually select the corresponding serial port (COM Port) in the software interface to ensure successful communication.



C. **Writing and Uploading Programs:**
   - Drag the required functional blocks from the blocks area into the program editing area to build your program.
   - MATRIXblock provides block specifically designed for the MATRIX Mini R4, for controlling motors, reading sensors, operating RGB LEDs, OLED displays, and more.
   - **Import Tip: Before clicking the upload button, ensure your MATRIX Mini R4 controller is properly connected and powered on.**
   - After completing your program, click the "Compile and Upload" button to upload the program to the Mini R4 controller for execution.

Further Learning:

This section provides only a brief introduction to MATRIXblock. For detailed installation steps, a complete interference guide, and more comprehensive example programs, please refer to Chapter 3, "Programming with MATRIXblock," of this manual or other available learning resources.

2.11.2.Arduino Programming & Library Overview

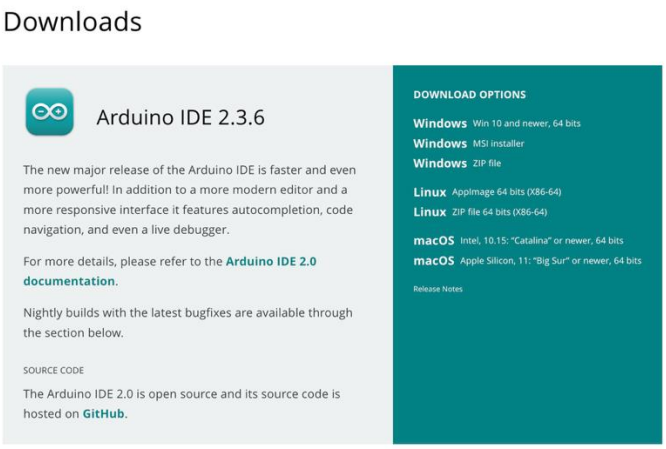
For users seeking lower-level, more flexible control or developers with a C/C++ programming background, the Arduino Integrated Development Environment (IDE) is a powerful and resource-rich choice. The MATRIX Mini R4's core controller is based on the Arduino UNO R4 WiFi, allowing full use of the Arduino ecosystem.

Introduction to Arduino IDE:

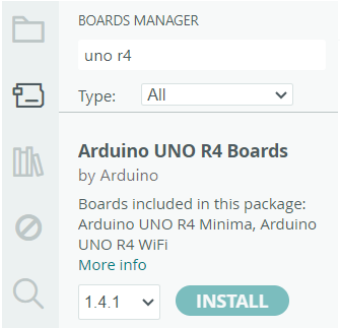
- **Open-source Development Platform:** Arduino IDE is a free, open-source, cross-platform application that provides a complete environment for writing, compiling code, and uploading it to Arduino-compatible hardware such as the MATRIX Mini R4.
- **Main Components:** It includes a simple code editor, a message output area, a serial monitor window (for debugging and data communication), and tools to easily manage boards and libraries.
- **Why it is Popular:** Thanks to its easy-to-learn programming language (based on C/C++), a large user community, and abundant educational resources and reusable libraries, Arduino has become one of the world's top platforms for makers, students, and professionals working on interactive projects.

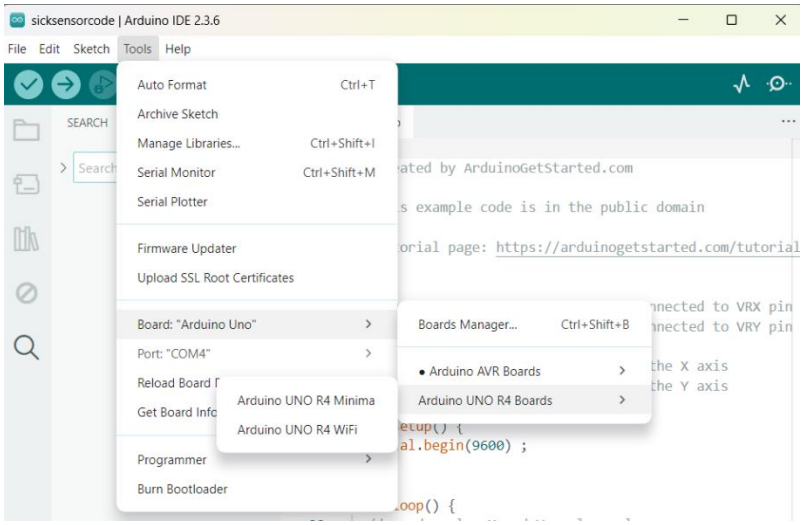
Basic Workflow for Developing MATRIX Mini R4 Programs Using Arduino IDE:

A. **Install Arduino IDE:** Download and install the latest version of the Arduino IDE from the official Arduino website (arduino.cc/en/software).

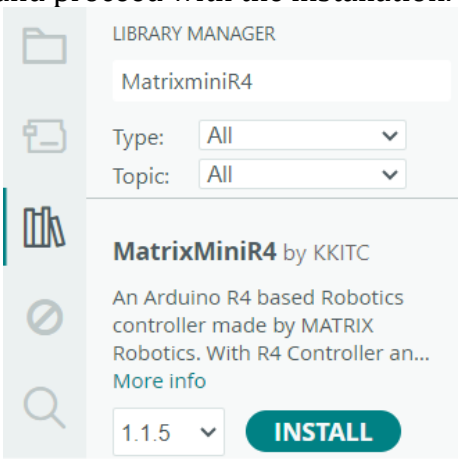


B. **Setting Up the Board:**
   - Since the MATRIX Mini R4's core controller is the Arduino UNO R4 WiFi, you need to ensure that the "Arduino UNO R4 WiFi" support package is installed or selected in the Arduino IDE's boards Manager.
   - Connect the MATRIX Mini R4 to your computer using a USB-C cable.
   - In the Arduino IDE's **Tools** menu, select the correct **Board** type (Arduino UNO R4 WiFi) and the appropriate **Port** to ensure proper communication.





C. **Installing the MATRIX Mini R4 Library:**
   - To facilitate control of the MATRIX Mini R4's unique hardware features, MATRIX provides a dedicated library.
   - **Library Name:** MatrixminiR4
   - **Installation Method:** You can install it via the Arduino IDE's built-in **Library Manager** (go to **Tools** > **Manage Libraries**...), search for the keyword "**MatrixminiR4**", and proceed with the installation.



D. **Writing, Compiling, and Uploading Programs:**
   - Write your Arduino program (commonly called a "sketch") in the code editor.
   - Click the "**Verify/Compile**" button to check for syntax errors in your code.
   - Once the Mini R4 is powered on and properly connected, click the "**Upload**" button to transfer the code to the controller for execution.

Overview of the MATRIX Mini R4 Dedicated Library (MatrixminiR4):

- **Purpose and Function:** The MatrixminiR4 library is designed to simplify the operation of the Mini R4's built-in hardware features and expansion interfaces, allowing users to fully unleash the controller's capabilities within the Arduino C++ environment.
- **Main Control Scope and Supported Hardware:** According to the official sample programs, this library supports and simplifies control of the following key hardware components:
   - **Motor Control:** DC motor speed and direction, servo motor angle.

    - **Onboard Interactive Components:** Onboard buttons, RGB LED, buzzer, and OLED display screen.
    - **Onboard Sensors:** IMU (Inertial Measurement Unit).
    - **General I/O Reading:** Analog signal input, digital signal input.
    - **MATRIX-Specific Sensor Modules (through I^2C):** Such as MX series color sensors, gesture sensors, laser distance sensors, and more.
    - **Other Common Sensors/Modules:** Such as DHT temperature and humidity sensors, and 1-wire digital temperature sensors.
    - **Smart Vision Modules:** Such as MATRIX M-Vision.
    - **External Controllers:** May include support for PS2 game controllers.
- **Usage Example:** For example, to control a servo motor connected to the RC1 port and set it to a specific angle, you can use the following command: MiniR4.RC1.setAngle(angleValue);

Using Standard Arduino Libraries:

In addition to the dedicated library, because the Mini R4's core controller is compatible with the Arduino UNO R4 WiFi, many standard Arduino libraries and third-party libraries can also be used with the Mini R4. Examples include standard functions for basic digital/analog I/O operations, as well as libraries like Wire.h (for I^2C communication) and SPI.h for (SPI communication), provided these interface functions are not completely taken over by the MatrixminiR4 functional library or have special configurations.

Learning Resources and Advanced Topics:

- **API Documentation:** For detailed descriptions of classes, methods, and parameters in the MatrixminiR4 library, refer to the official MATRIX Robotics API documentation: https://matrix-robotics.github.io/Programming-API-Docs/MiniR4_Arduino_Lib_API_Docs/
- **Example Programs:** After installing the MatrixminiR4 library, you can find example programs for the various hardware controls mentioned above in the Arduino IDE under (**File** > **Examples** > **MatrixminiR4**). These are the best starting points for learning how to use the library.
- **Official Resources:** Be sure to check the official MATRIX Robotics website and community channels regularly for the latest tutorials and library updates.

3. Programming with MATRIXblock

This chapter will guide you through how to install the MATRIXblock software, connect to the MATRIX Mini R4 controller, learn the basic programming concepts, and an important firmware update process. With MATRIXblock's graphical interface, you will be able to easily design various interesting interactive applications for the Mini R4.

3.1. Installing MATRIXblock Software

3.1.1. Obtaining the Installer

A. Visit official website: please use a web browser to open the MATRIX Robotics official website, MATRIXblock, software download page and press "Download Now":

https://www.matrixrobotics.com/matrixblock-software 。

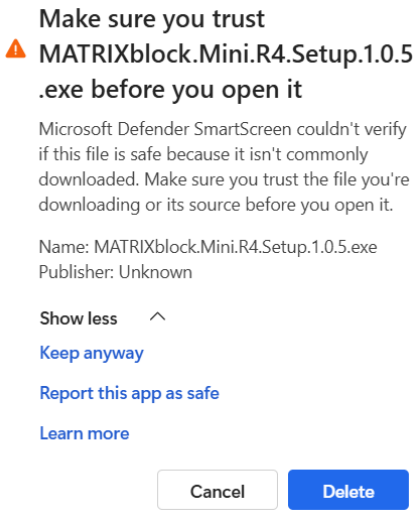
B. **Choose the correct version**：

- According to your computer operating system choose to download the Windows version or the macOS version.
- **Important Notice**： Make sure to select the software applicable for the **MATRIX Mini R4** and not the version for the older Mini (R3), to ensure complete compatibility between the software and the hardware.
- The currently supported versions of the operating system are：
  - Windows: Windows 10, 11 (x64), recommended 1803 and above.
  - macOS: Monterey or later versions support Arm® and Intel® CPU

3.1.2. Windows Installation Steps

A. **Run the installer**： After the download is complete, locate the .exe installer file (e.g., MATRIXblock_R4_Setup_vx.x.x.exe) and double click.

B. **If Microsoft blocks the download: press the three dots and keep anyway.**

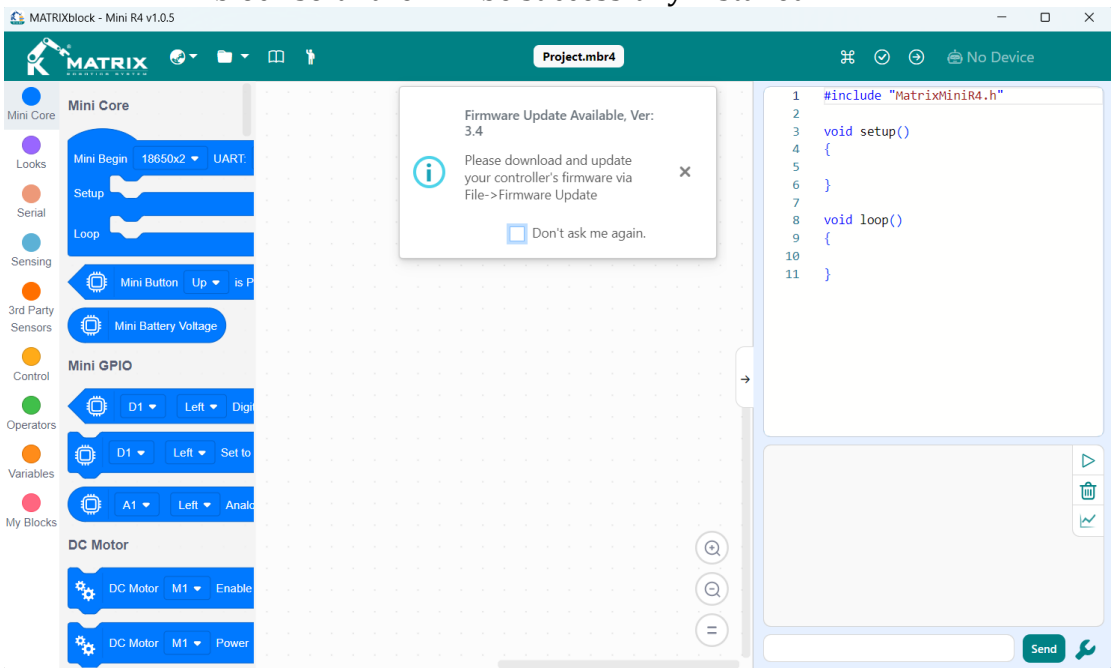


C. **User Account Control (UAC)**：If the system displays a User Account Control prompt asking, "Do you want to allow this app to make changes to your device?", click Yes.

D. **Installation Wizard Instructions**：

- Next, the installer will guide you through the installation process. This usually includes reading and agreeing to the software license agreement (if any), choosing the installation path (it's recommended to keep the default path), etc.

E. **Complete the installation**：Once all steps in the installation wizard are completed, the MATRIXblock software will be successfully installed.



3.1.3. macOS Installation Steps

A. **Open the Disk Image**：After the download is complete, locate the .dmg disk image file and double click it to open.

B. **Drag to Install**：A window will typically appear showing the MATRIXblock application icon and a shortcut to the Applications folder. **Drag** the MATRIXblock icon into the Applications folder to complete the installation.

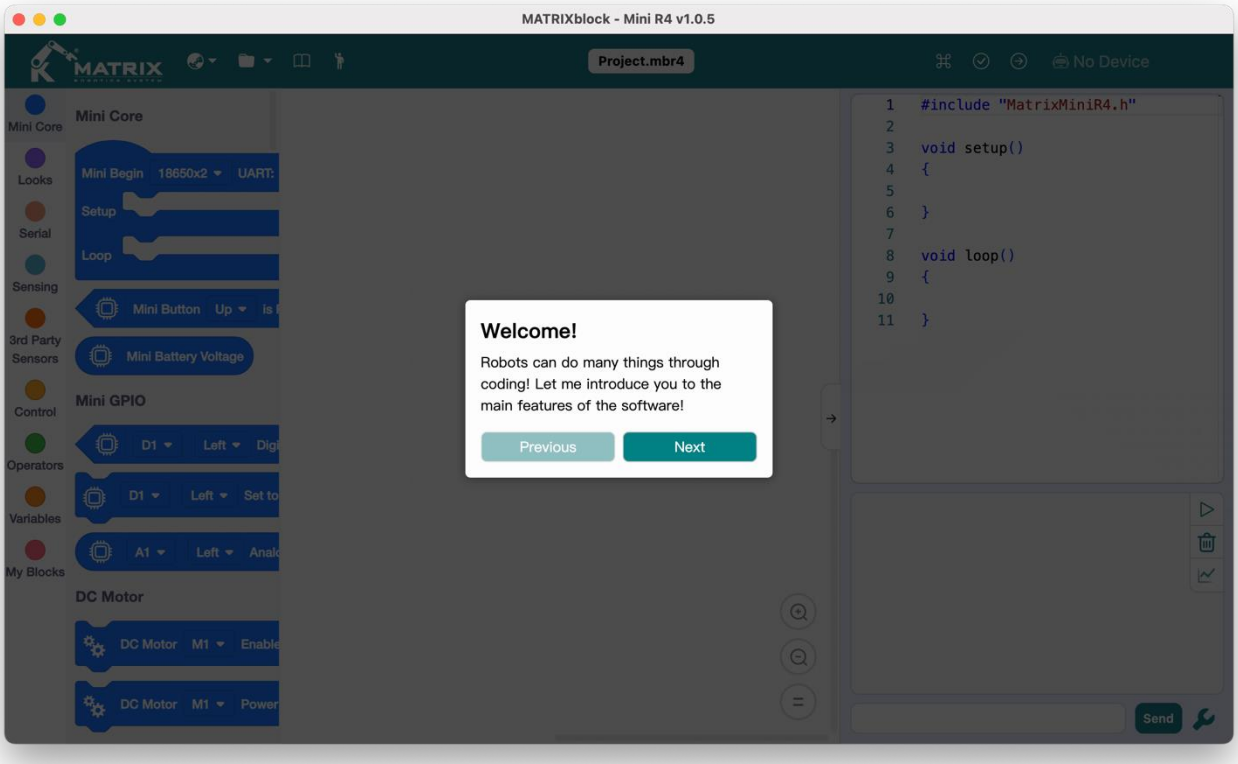
C. **Security and Privacy Settings (First Launch)**：

- When launching MATRIXblock for the first time, macOS may display a warning because the app is not from the App Store.
- You may need to allow the app to run by going to System Settings > Privacy & Security, and enabling the option to open apps downloaded from identified developers.

3.1.4. First Launch

- **Windows:** After installation is complete, you can find the MATRIXblock Mini R4 shortcut in the **Start** menu or on the desktop. Click it to launch the software.

- **macOS**：Go to the Applications folder, locate the MATRIXblock Mini R4 icon, and click it to run the application.



3.2. Connecting Mini R4 to MATRIXblock

After successfully installing the MATRIXblock software, the next step is to establish communication between your MATRIX Mini R4 controller and the software, and ensure the controller is ready to receive programs.

Key Concepts: Connection, Power On, and Power Supply

- **USB connection and Software Detection**：When you connect the Mini R4 to your computer using a USB-C cable, the MATRIXblock software may detect the USB device and display “Connected” even if the Mini R4 is not powered on. This only indicates that basic USB communication has been established between the computer and the controller.
- **Controller Power-On**：The MATRIX Mini R4 does not automatically power on when connected via USB. You need to press and hold the **Reset** button (usually also the power button) on the controller for a few seconds until you hear a beep and see the boot animation on the OLED screen. This indicates the controller has successfully powered on and is operational.
- **Requirements for Program Uploading**： The controller must be powered on to successfully receive and burn/upload a program.
- **Power Supply Options**：

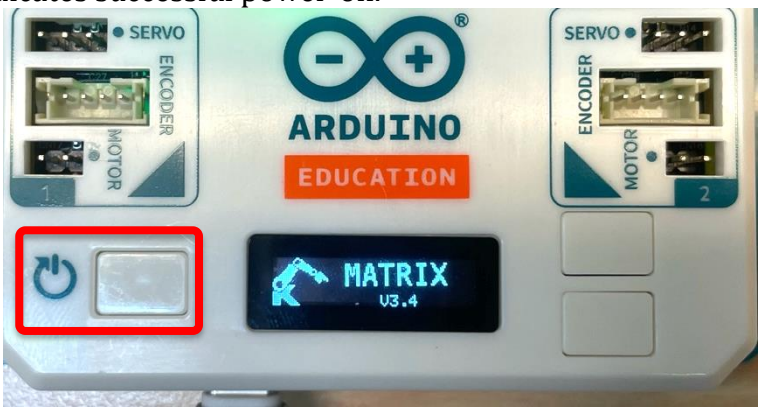
- **USB-C Only Power**： The Mini R4 can be powered solely through USB-C for basic program logic tests, onboard LED control, and reading low-power sensors.
- **External Power Supply:** If your program involves driving motors (DC motors, servo motors) or connecting high-power modules, you must connect an external power source (6V–24V DC) to the Mini R4.

3.2.1. Hardware Connection Setup

A. **Connect the Controller**： Use the USB-C cable included with the MATRIX Mini R4 (or a good-quality USB-C data cable) to connect the USB-C port on the controller to your computer's USB port.

B. **Power on the Controller**：

- **Press and Hold the Reset/Power Button**:
  After connecting the USB-C cable, press and hold the Reset/Power button on the Mini R4 until you hear a beep and see the boot animation or screen on the OLED. This indicates successful power-on.



C. **Connect External Power (safely and when needed):**

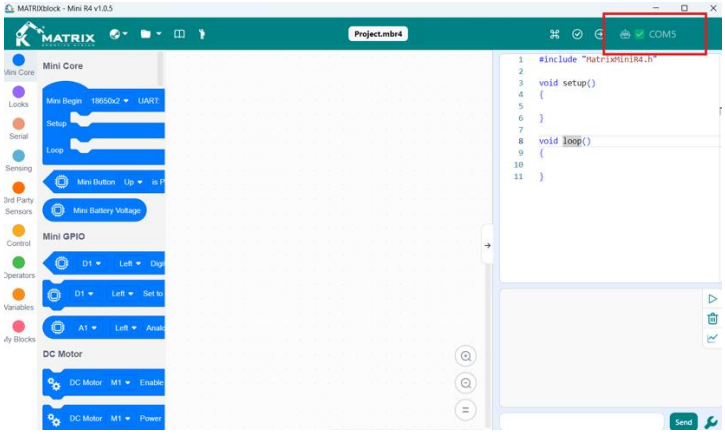
- **During Program Upload:**
  For uploading programs or performing low-power tests without motors, external power is typically not required—USB-C power is sufficient.
- **During Program Execution and Testing (Use caution):**
  - **Safety Tip:**

If your program includes motor control or movement commands, it's strongly recommended to move the Mini R4 (or the robot it's mounted on) to a safe, open, and controlled area after uploading the program but before connecting external power. Then, reconnect the external power and restart the controller to execute the program.
This prevents unexpected movement or collisions when motors activate

3.2.2. Launching Software and Connection Interface

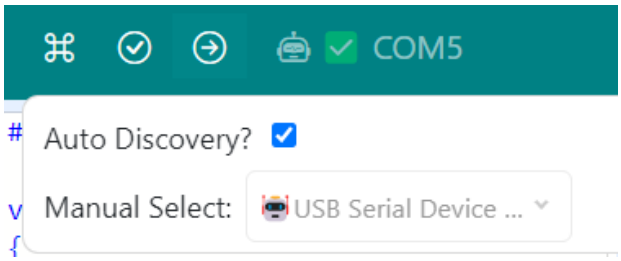
A. **Launch MATRIXblock**：If not already open, launch the installed MATRIXblock (Mini R4 version) software on your computer.

B. **Locate the Connection Feature**： In the upper-right corner of the MATRIXblock interface, you'll find the menu for the "Connect Device" or "Select Serial Port"



3.2.3. Selecting Serial Port and Establishing Connection

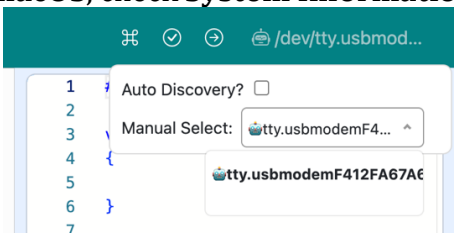
A. **Automatic Detection**： MATRIXblock will usually attempt to automatically detect and connect to the connected MATRIX Mini R4 when the software is launched or when the connect button is clicked.



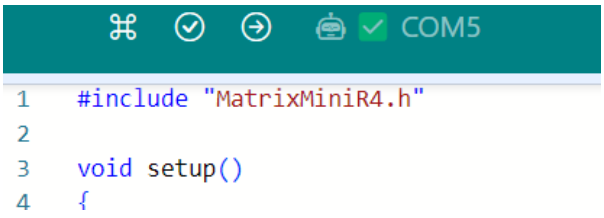
B. **Manually Selecting the Serial Port (COM Port)**：

- If automatic connection fails, you'll need to manually select the correct port from the list provided in the software.
- **Port Naming Tips**：
  - On **Windows**, ports are typically labeled as **COMx** (e.g., COM3, COM4, etc.).
  - On **macOS**, they usually appear as /dev/cu.usbmodemXXXX or /dev/tty.usbmodemXXXX.
- **How to Identify the Correct Port**：

- **Robot Icon**： When MATRIXblock successfully identifies a MATRIX Arduino device (like Mini R4), a (robot head) icon is usually displayed next to the correct port name—this helps you find the right one quickly.
- **Compare Before and After:** If you're unsure which port corresponds to your Mini R4, compare the list before and after plugging it in.
    1. On **Windows**, you can also check the **Device Manager** under "Ports (COM & LPT)."
    2. On **macOS**, check **System Information** under USB devices.



C. **Confirm the Connection Status**： Once successfully connected, MATRIXblock will display a “Connected” status, indicating that the software is ready to communicate with the controller.



3.2.4. Common Connection Issues and Tips

- **Cannot find a serial port or connection failed**：
  - Ensure the MATRIX Mini R4 is correctly powered on as described in section "3.2.1 Hardware Connection and Power-On."
  - Make sure the USB-C cable is securely connected to both the computer and the controller
  - Try using a different USB-C cable or another USB port on your computer.
  - Restart the MATRIXblock software or unplug and replug the Mini R4 controller (after reconnecting, remember to press and hold the power button again to turn it on).
  - Windows users should check the Device Manager to see if the controller is properly recognized or if there are any driver error messages. If MATRIXblock prompted you to install specific drivers during installation, make sure the installation was completed.
- **Wrong serial port selected**：If multiple serial devices are connected to your computer, choose the port marked with the (robot head) icon, or carefully verify you have selected the correct port for the Mini R4.
- **Program upload failed**：
  - First, check if the controller is properly powered on.
  - Ensure the USB connection is stable.
  - Make sure the correct board model is selected in MATRIXblock
- **Abnormal program behavior（especially when motors are involved）**：
  - Confirm that the external power supply has been properly connected and turned on. USB-C power alone cannot drive motors.
  - Always test motor-involved programs in a safe environment.

3.3. Basic Programming Concepts

Before you start programming your MATRIX Mini R4 using MATRIXblock, it's important to understand a few basic programming concepts. These will help you understand how to organize and stack blocks to make the controller perform specific tasks.

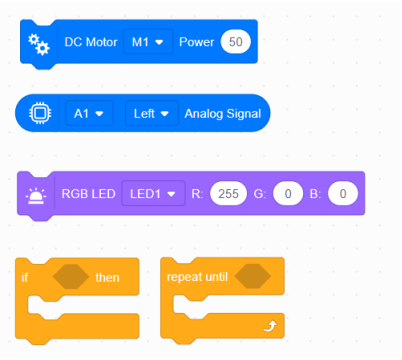
3.3.1. What is a Program?

Simply put, a program is a sequence of instructions given to a controller (in this case, the MATRIX Mini R4) telling it what to do, when to do it, and how. Logic and order of instructions are at the core of programming.

3.3.2. Understanding Blocks

In MATRIXblock, programs are composed of various "blocks" of different shapes and colors. Each block represents a specific command or function, such as：

- Rotating a motor
- Reading sensor values
- Lighting up an LED
- Checking if a condition is met
- Repeating a set of actions

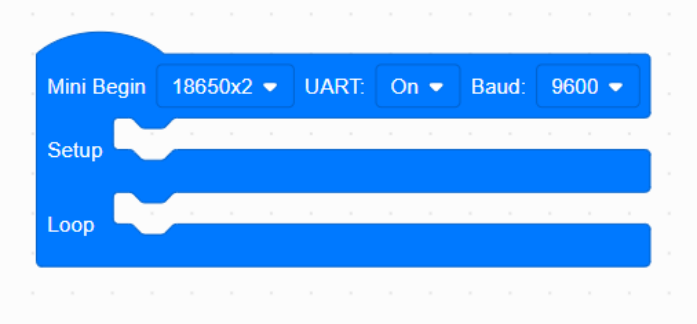


These blocks are puzzle-shaped, with interlocking edges, guiding you to stack them in a logical manner to create a complete program.

3.3.3. Program Start: The "Initialize" Block for Mini R4

Unlike many other block-based programming tools, MATRIXblock has a special **Initialize** block designed for the MATRIX Mini R4 as the starting point of any program. This block not only marks the beginning of the program but also contains essential configuration settings for the controller.

- **Appearance and Function**：You'll see a block with a rounded top (like a hat) labeled "Mini Initialize." This block must be placed at the top of all your code blocks.



- **Initialization Settings**：Within this block, you can configure basic settings such as：
  - **Battery type/voltage**：The block might show something like "18650x2," related to power management or default voltage detection.

- **Serial Port**：You can set whether the serial port is enabled (e.g., “Serial Port: On”) and its baud rate (e.g., “Baud: 9600”). The serial port is vital for communication between the controller and the computer or other devices, especially for data transfer and debugging.

- **Program Execution**：A program that starts with the “Mini Initialize” block will automatically run once it is uploaded to the Mini R4 and the controller is powered on.

  This “Initialize” block is your first step when writing a standalone program for the Mini R4 and ensures the controller is correctly configured before executing your commands.

3.3.4. Responding to External Events and Changes

Since the MATRIXblock program for Mini R4 mainly starts executing sequentially from the “Mini Initialization” block, when the controller needs to respond to external events (such as a button being pressed or a sensor detecting a specific condition), the logic for these detections and responses is usually implemented within a loop structure placed under the “Mini Initialization” block.

- You would typically use conditional blocks (such as “If… Then…”) inside looping blocks (like “Repeat Forever” or loops with specific conditions) to continuously check for changes in external states and perform corresponding actions.
- This approach allows the program to keep running and respond to various inputs in real time.

3.3.5. Sequential Execution

Under the “Mini Initialization” block, or in any part of the program, the blocks are usually executed in order from top to bottom. It's like following a recipe step by step— the controller finishes executing one block before moving on to the next.

- **Example**：If you stack the following blocks：

  1. （Turn on red LED）
  2. （Wait 1 second）
  3. （Turn off red LED）

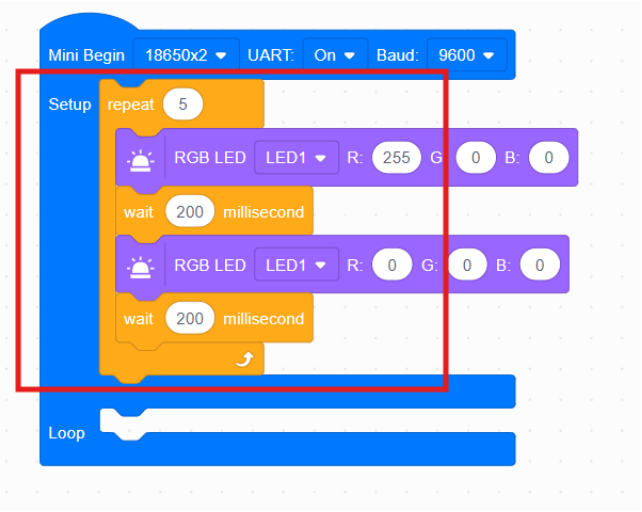
The controller will first turn on the red LED, then wait one second, and finally turn off the red LED.

Understanding the order of execution is essential for designing a program that works correctly.

3.3.6. Loops: Repeating Instructions

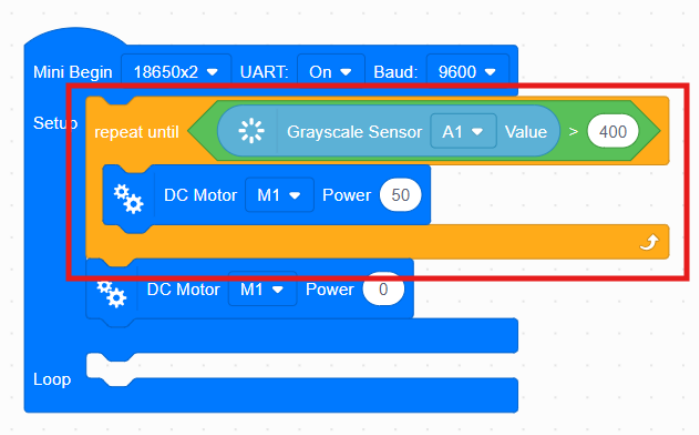
Sometimes, you need a set of instructions to run multiple times or continuously. In such cases, you use loop blocks.

- **Why use loops? ?**
  - Repeat a specific number of times: e.g., make the LED blink 5 times.
  - Continuous execution: e.g., let a robot continuously detect obstacles ahead.
  - Conditional repetition: e.g., keep a motor running until a sensor is triggered.
- **Common types of loop blocks**：
  - **"Repeat N times" block**： Executes the block inside it a specific number of times.

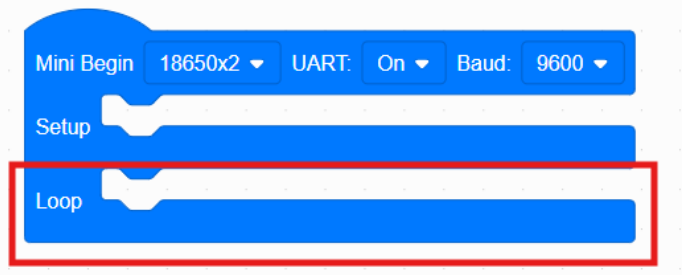


  - **"Repeat forever" block**：Continuously repeats the block inside it (until a stop command is issued or the power is turned off). This is commonly used in the main program to ensure the controller stays active and responsive.

- **“Repeat Until...” block**：Repeats the block inside until a specific condition is met.



- **Importance of loop structure**： As mentioned in section 1.3.4, a “Repeat Forever” loop is usually placed right after the “Mini Initialization” block. All continuous detection, decision-making, and actions needed after the controller starts up are placed in this main loop.

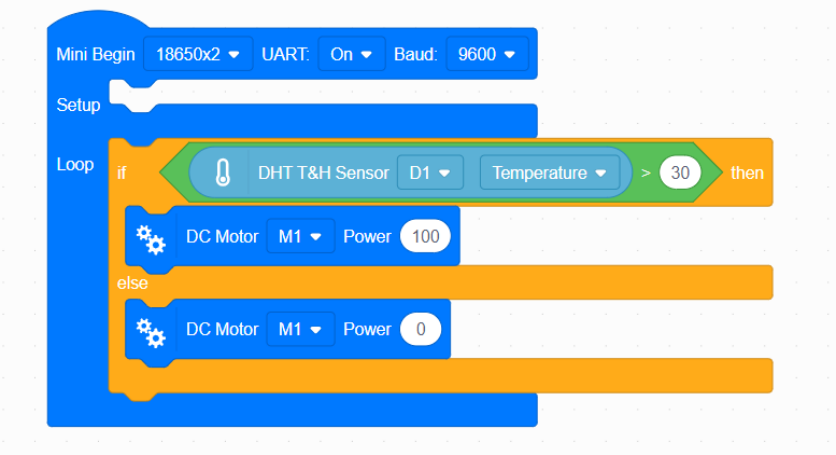


3.3.7. Conditional Logic: Making Decisions

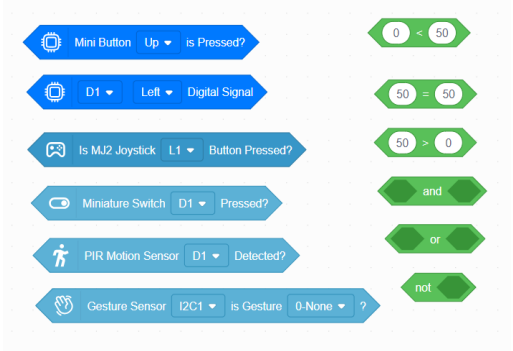
Programs don’t just run in order; they can also make decisions based on different situations using conditional logic.

- **Why use conditional logic?**
  - To make the program smarter and able to respond to different inputs or states.
  - Example: If there’s an obstacle ahead, then move backward and turn; otherwise, keep moving forward.
- **Common conditional block types**：
  - **“If...Then...” block**： If a specific condition is true, the instructions inside the “Then” section will be executed: if the condition is false, those instructions will be skipped.

- **"If...Then...Else..." block**： If the specified condition is met, the instructions in the "Then" area will be executed. If the conditions are not met, the instructions in the "Else" area will be executed.



- **Sources of Conditions**：
  - Conditions usually come from sensor readings (e.g., is the distance detected by the ultrasonic sensor less than 10 cm?), variable comparisons (e.g., is the counter variable equal to 5?), or button states (e.g., is button A pressed?).
  - In MATRIXblock, such condition blocks (usually hexagon or diamond-shaped) can be found in the "Sensing" or "Operators" categories and are designed to fit into the condition slot of the "If...Then...Else..." block.

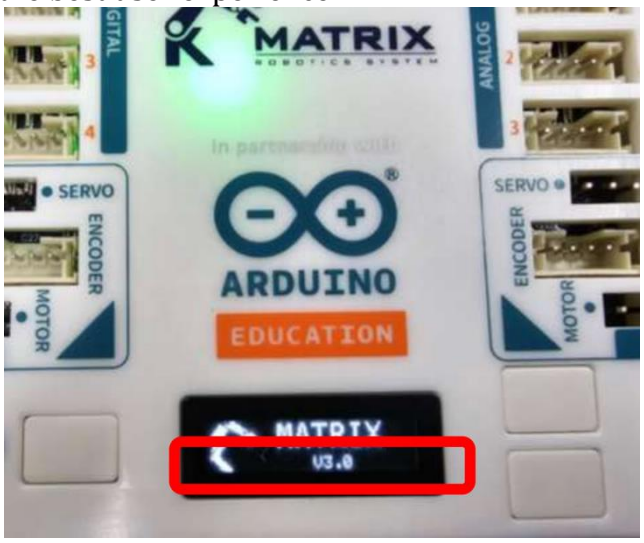


3.4. Firmware Update

Keeping the MATRIX Mini R4 controller's firmware up to date ensures optimal functionality, performance improvements, and bug fixes. This section will guide you through the firmware update steps.

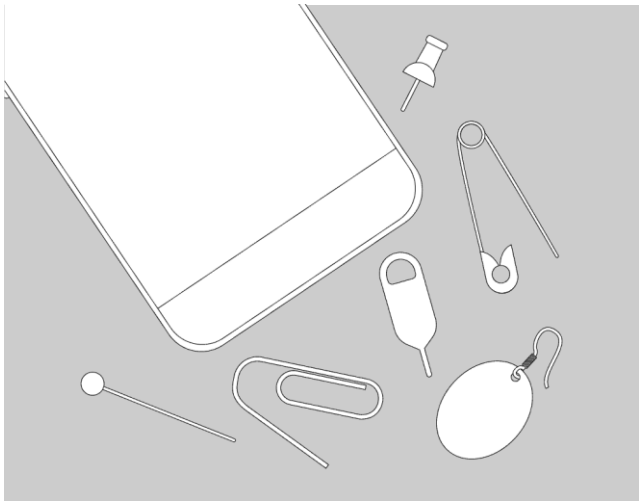
Step 0 – Firmware Version Check

- When the Mini R4 controller is powered on or reset, the OLED screen displays a startup animation, which typically includes the current firmware version.
- You can compare this version with the latest firmware released by MATRIX Robotics. If your firmware is already up to date, you may skip the following steps. If not, updating is recommended for the best user experience.



Step 1 - Prepare Tools

- You will need a paperclip, SIM card ejector tool, or a similar thin and sturdy tool to press the DFU button on the controller.



Step 2 - Disconnect All Power Sources

- Before performing the firmware update, make sure to disconnect all power sources from the MATRIX Mini R4 controller, including external batteries and the USB-C cable connected to the computer.

Step 3 - Connect USB Cable to Computer

- First, connect one end of the USB-C cable to your computer's USB port. Do **not** connect the other end to the Mini R4 controller yet.

Step 4 - Locate and Prepare to Press DFU Button

- Locate the DFU mode pinhole button on the MATRIX Mini R4 controller. This button is usually on the side of the device, near the USB-C port.
- Carefully insert the thin tool (e.g., paperclip) into the pinhole and get ready to press and hold it.

Step 5 - Entering DFU Mode

- While holding down the DFU button, connect the other end of the USB-C cable to the Mini R4 controller.

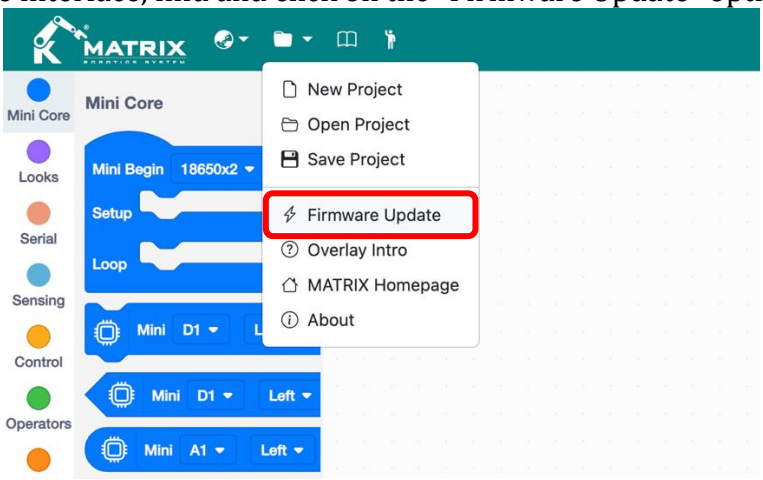
Step 6 - Confirm DFU Mode Entry

- If done correctly, the blue indicator light near the DFU button will start flashing. This indicates that the Mini R4 has successfully entered DFU mode and is ready to receive new firmware.
- You may now release the DFU button.

Step 7 - Open Firmware Update in MATRIXblock

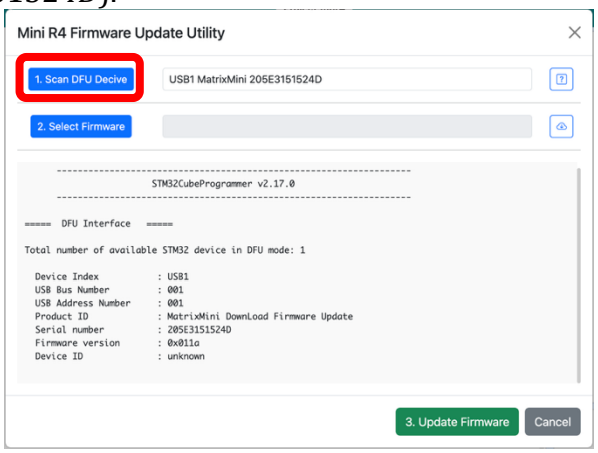
- On your computer, open the MATRIXblock (Mini R4 Version) software.

- In the software interface, find and click on the “Firmware Update” option.



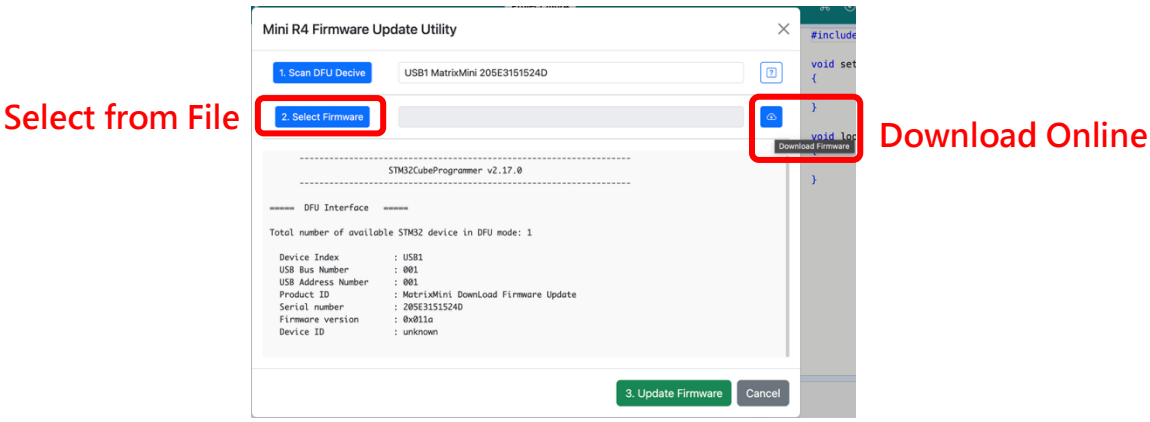
Step 8 – Scan for DFU Device

- In the "Mini R4 Firmware Update Utility" window, click on **"1. Scan DFU Device"**
- The program will scan for connected DFU devices. If the Mini R4 has successfully entered DFU mode and is recognized by the computer, you should see your device listed (e.g., USB1 MatrixMini 205E3151524D).



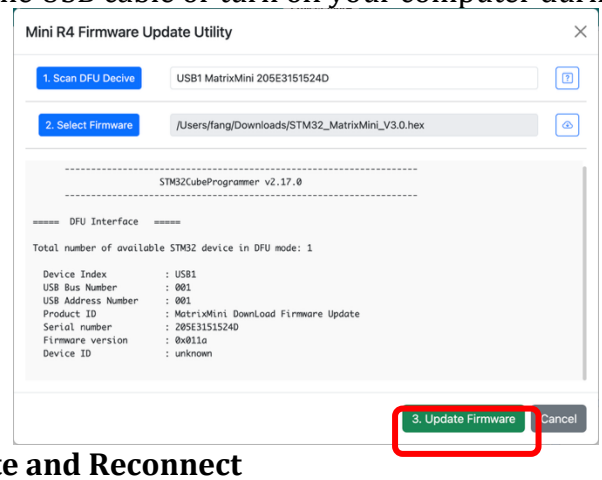
Step 9 - Select Firmware File

- Click on **"2. Select Firmware"**.
- You can choose：
  - **"Select from file":** Browse and choose a firmware file already downloaded to your computer (usually in .hex or .bin format).
  - **"Download online"**: If supported by the software, download the latest firmware version directly from the internet.



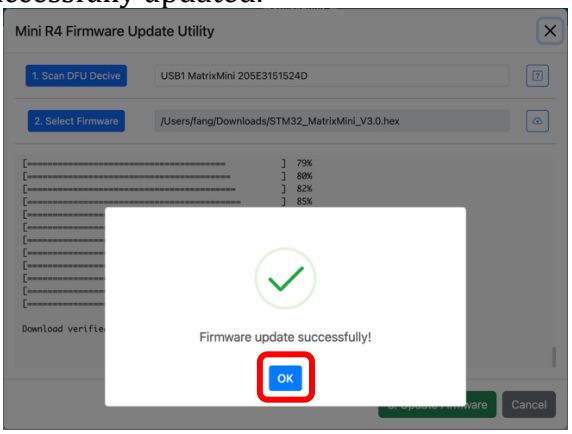
Step 10 - 開始更新韌體 (Start Firmware Update)

- After confirming the correct firmware file has been selected, click **"3. Update Firmware"** to begin the update process.
- **Do not** disconnect the USB cable or turn off your computer during the update.



Step 11 –Update Complete and Reconnect

- The update process may take some time, so please be patient. A progress bar is typically shown.
- When you see **"Firmware update successfully!"** or a similar message, it means the firmware has been successfully updated.



- Click "OK" or any confirmation button, then disconnect the USB cable from the Mini R4 and reconnect it again.

- Restart the Mini R4. You can check the firmware version again in the startup animation to confirm it has been updated to the latest version.

4. Input Modules

4.1. Overview

Input modes are the “senses” of any interactive project, allowing the controller to perceive changes in the external world or receive user commands. Through these modules, the MATRIX Mini R4 controller can obtain various types of information — such as whether a button is pressed, changes in the controller's orientation, specific physical quantities in the environment, signals from other devices, or even remote control commands from users. This input data forms the basis for the controller to process and respond accordingly.

This chapter will provide a detailed introduction to the various input modules compatible with the MATRIX Mini R4. We will start by exploring the onboard input components built into the controller, such as the user button and the IMU (Inertial Measurement Unit). Next, we will cover how to connect and use external input modules and sensors via the Mini R4’s versatile interfaces, including analog, digital, I^2C, UART, and more. We will also explain how to integrate specific input devices such as the MATRIX Joystick (MJ2).

For each major input module, we will aim to cover an overview, functional features, application examples, electrical characteristics, pin definitions (or connection methods), sample programs, and basic operating principles.

By studying this chapter, you will gain a thorough understanding of how to utilize a wide range of input modules, enabling your Mini R4 projects to have enhanced sensing capabilities and richer interactivity.

4.2. On-Board Input Modules

4.2.1. User Buttons

4.2.1.1. Overview

The MATRIX Mini R4 controller includes two onboard user-configurable buttons (User Buttons). These buttons offer a direct and convenient way to physically interact with the controller – for example, to trigger specific actions, switch modes, start or stop program segments, etc. They are among the most basic and commonly used input components on the controller, enabling users to easily add real-time human-machine interaction to their projects.

4.2.1.2. Features

- **Momentary Triggering**： The onboard user buttons are momentary buttons—they send a signal when pressed and return to their original state when released.
- **Programmable Input**： The functions of these two buttons are entirely defined by the user through programming. They can be used to trigger specific events, alter program flow, switch operating modes, or perform simple logic operations such as incrementing or decrementing values.
- **Direct Hardware Interaction**： They provide a convenient way to physically interact with the controller without needing any external modules.
- **Quantity**： The controller includes two independent user buttons.

4.2.1.3. Application Examples

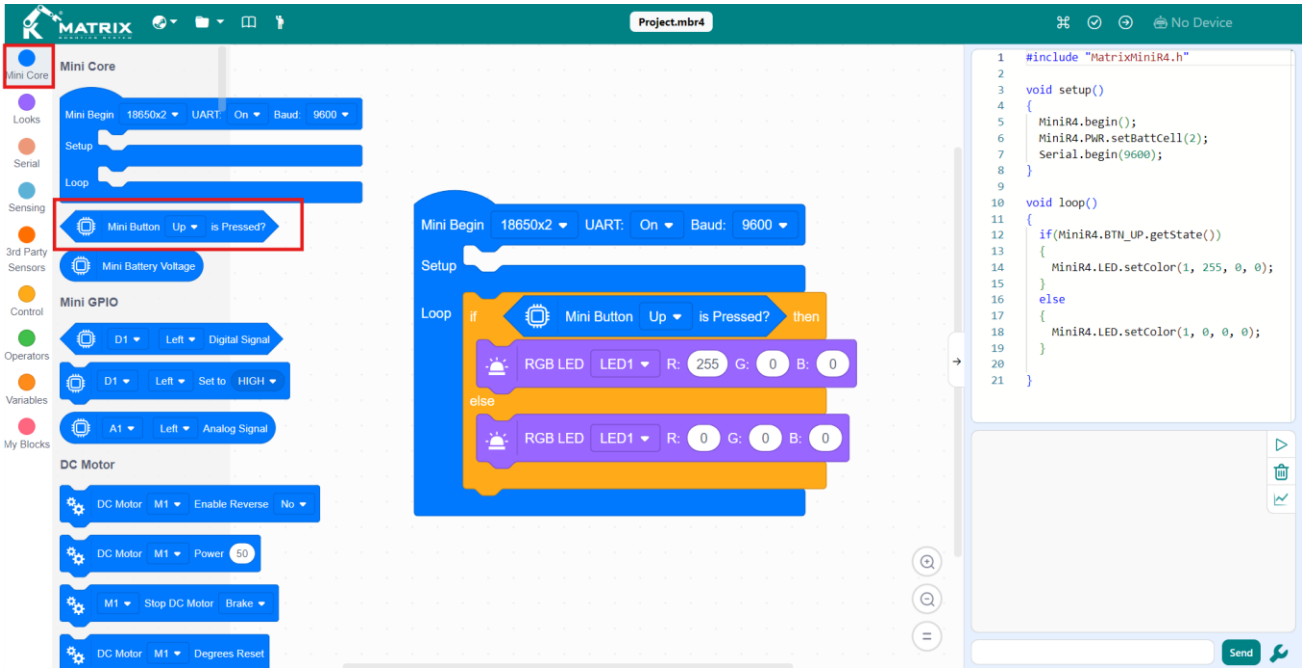
- **Mode Switching**： Used to start/stop functions or switch between different operating modes.
- **Function Triggering**： Pressing a button to execute a specific action, such as activating a robotic arm or playing a sound effect.
- **Simple Controls**： Used as input for counters, menu selection, or as a basic game controller.
- **Calibration and Settings**： Can be used to trigger calibration routines or enter parameter setting modes.
- **Teaching and Testing**： An excellent tool for verifying condition checks and output triggers during early stages of programming learning.

4.2.1.4. Electrical Characteristics

Parameter Name	Typical Value / Status	Unit	Remarks
Signal Type	Digital	-	
Pressed Logic	HIGH / 1	-	
Released Logic	LOW / 0	-	Typically designed with internal pull-down
Operating Voltage	5	V	Matches the typical I/O operating voltage of the controller
Current Consumption	Very Low	-	Functions as a switch-type input component

4.2.1.5. Sample Code & Blocks Instructions

- Sample Code：



MATRIXblock

IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
}

void loop()
{
  if(MiniR4.BTN_UP.getState())
  {
    MiniR4.LED.setColor(1, 255, 0, 0);
  }
  else
  {
    MiniR4.LED.setColor(1, 0, 0, 0);
  }

}
```

- **Explanation of Result**：

  When the user button (upper one) is pressed, the RGB LED on the controller lights up red. When released, the LED turns off.

4.2.1.6. Principle Description

The user buttons onboard the MATRIX Mini R4 are momentary switches. Their operation is based on changing the circuit state to generate a digital signal.

- **Basic Structure**： The internal structure of the button typically consists of a pair or normally open (NO) conductive contacts.
- **Circuit Design Approach**： Based on the behavior where pressing the button causes the program to read a high voltage level (logic 1), and releasing it causes the program to read a low voltage level (logic 0), the typical circuit design is as follows:
    - One side of the button is connected to the controller's power supply (e.g., 5V).
    - The other side is connected to a digital input pin of the microcontroller (in this case, the STM32F103 coprocessor).
    - This input pin is also connected to ground (GND) through a pull-down resistor.
- **How It Works**：
    - **When not pressed**： The button contacts are open. Due to the pull-down resistor, the microcontroller input is pulled to a low voltage (GND), and the program reads logic 0.
    - **When pressed**： The conductive contacts close, connecting the input pin directly to the power supply (5V). The pin voltage becomes high, and the program reads logic 1.
    - **When released**： The internal spring or similar mechanism opens the contacts again, and the input pin is pulled back to low voltage by the resistor.

The design ensures that the input pin is always in a know low state when the button is not pressed, preventing floating (undefined) input values.

4.2.2. IMU

4.2.2.1. Overview

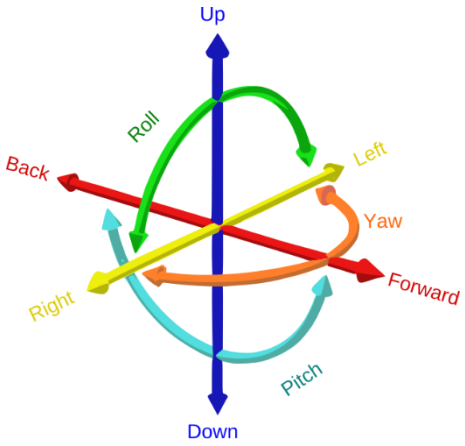
The MATRIX Mini R4 controller comes with a built-in **Inertial Measurement Unit (IMU)**, a board-mounted sensor module primarily used to detect the controller's motion and orientation. The Mini R4's IMU integrates a 3-axis accelerometer and a 3-axis gyroscope, giving it six degrees of freedom for motion sensing.

By reading IMU data, your project can detect tilting, rotation, acceleration, and other dynamic changes in the controller's state. As an onboard component, its power and communication are already handled internally by the controller, so the user can focus on obtaining rich motion data through software.

4.2.2.2. Feature

6-DoF Motion Sensing:

- **3-Axis Acceleration Sensing:**
  Detects and outputs linear acceleration along the X, Y, and Z axes. This can be used to sense gravity direction, object movement acceleration, vibrations, etc.
- **3-Axis Angular Velocity Sensing:**
  Detects and outputs angular velocity (rotation rate) around the X, Y, and Z axes. This helps detect rotational movement and orientation.



Attitude Data Acquisition:

- By fusing raw data from the accelerometer and gyroscope, the controller can calculate real-time orientation information:
  - **Pitch:** Rotation angle around the X-axis (looking up/down)
  - **Roll:** rotation angle around the Y-axis (left/right tilt)
  - **Yaw:** Rotation angle around the Z-axis (horizontal turning)

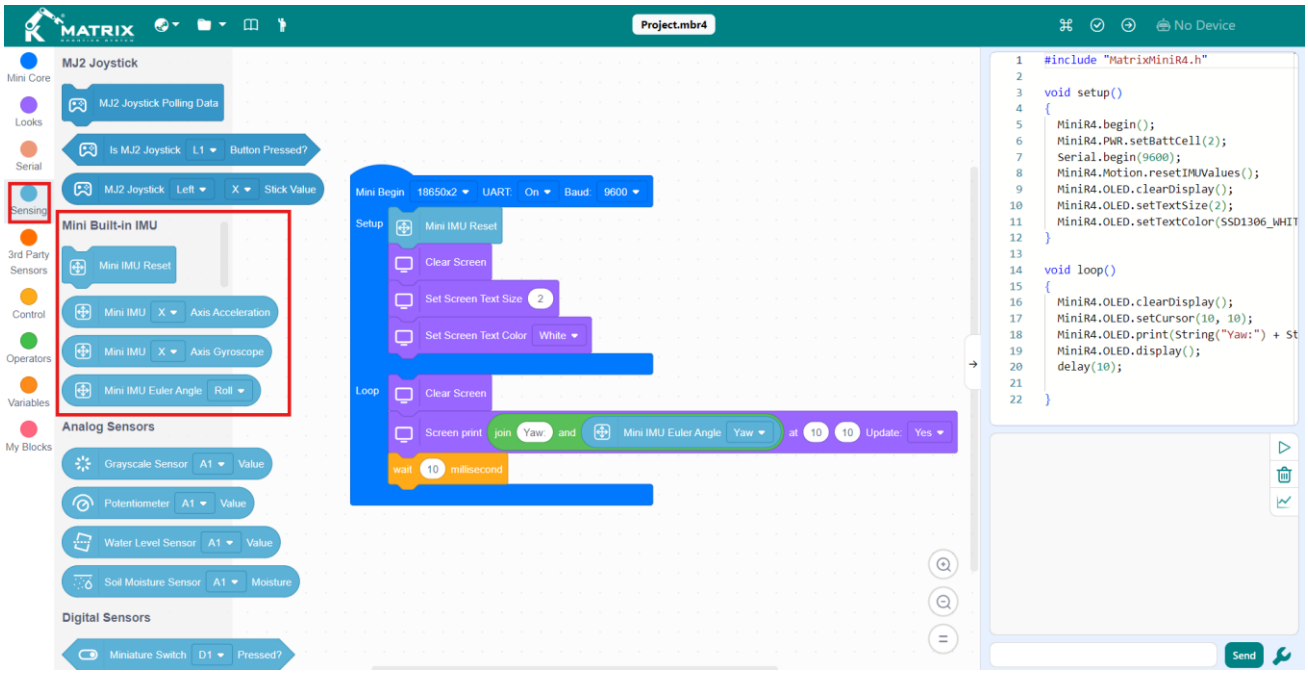
**Note：For a 6-axis IMU, long-term yaw calculations may experience cumulative error or drift due to the lack of magnetometer correction.

4.2.2.3. Application

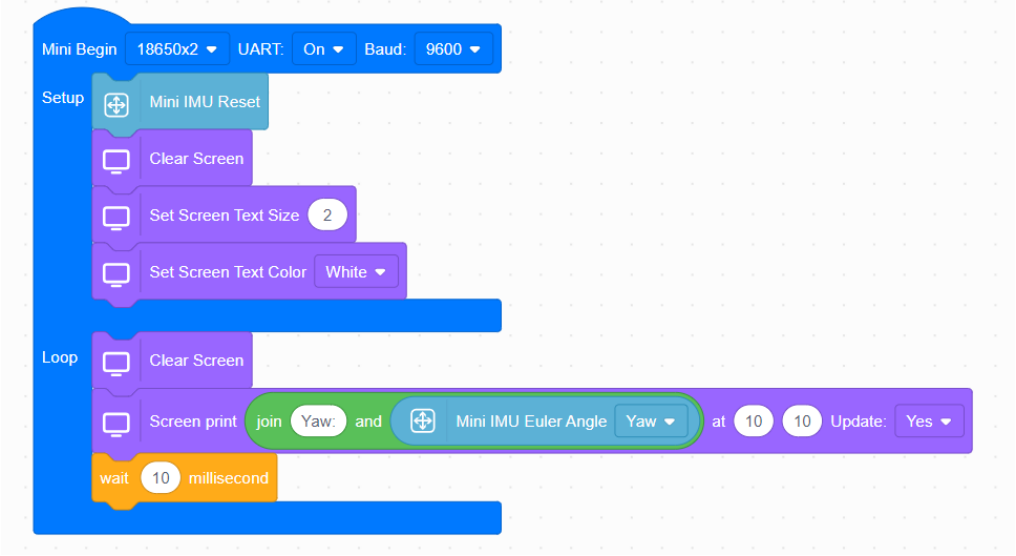
- **Posture Detection and Balance Control**： Used for detecting and correcting angles in self-balancing vehicles, or for creating a basic digital level.
- **Motion-Based Interaction**： Use the controller as a motion-sensing joystick to control games or interactive devices; trigger commands by specific shaking or flipping motions.
- **Activity Tracking**： Develop step counters or detect object motion status and falls.
- **Robot Navigation Assistance**： Assist in determining a robot's posture on slopes or tracking changes in its turning angle.

4.2.2.4. Sample Code & Blocks Instructions

- Sample Code：



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.Motion.resetIMUValues();
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(2);
  MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setCursor(10, 10);
  MiniR4.OLED.print(String("Yaw:") +
String(MiniR4.Motion.getEuler(MiniR4Motion::AxisType::Yaw)));
  MiniR4.OLED.display();
  delay(10);

}
```

- **Explanation of Result：**

  The OLED screen on the MATRIX Mini R4 continuously displays the currently measured yam angle in the format: “Yaw: [angle value]”. As you rotate the Mini R4 controller horizontally, the angle value shown on the screen will change accordingly.

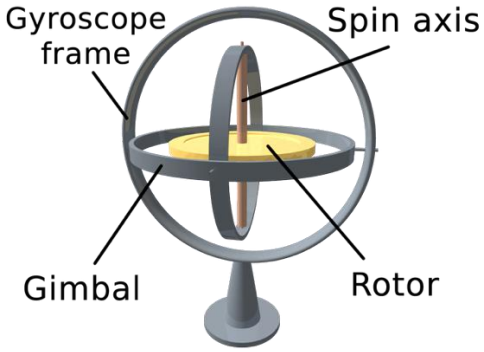
4.2.2.5. Principle Description

The MATRIX Mini R4’s onboard IMU (inertial Measurement Unit) detects the controller’s movement and orientation using its two main integrated sensing components: the accelerometer and the gyroscope.

- **How the Accelerometer Works：**
  - The accelerometer detects linear acceleration along three mutually perpendicular axes (X, Y, Z).

- When the controller is stationary, it can sense the direction of Earth's gravity, which helps determine the device's tilt (e.g., pitch and roll angles).
- When the controller moves or is subjected to external force, it measures changes in acceleration.

- **How the Gyroscope works**：



- The gyroscope detects angular velocity around the three perpendicular axes (X, Y, Z) — essentially the speed and direction of rotation.
- This allows the controller to sense its own rotational movements.

- **Using Data for Attitude Estimation**：
  - Relying only on the accelerometer to estimate orientation can result in shaky or inaccurate readings due to interference from movement-based acceleration.
  - Relying only on the gyroscope can lead to small cumulative errors over time (known as drift)
  - To obtain more accurate and stable orientation data, software algorithms (handled internally by the MatrixMiniR4 library) perform **sensor fusion**, combining data from both the accelerometer and gyroscope.

  This fusion method leverages the strengths of each sensor and compensates for their weaknesses to compute reliable orientation angles, such as:

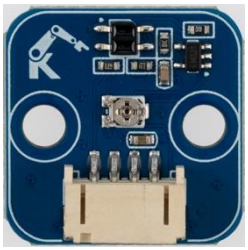
In short, an IMU senses linear and rotational motion, and through intelligent data processing, it enables the MATRIX Mini R4 to understand its dynamic state in three-dimensional space.

4.3. External Analog Inputs

4.3.1. Gray Scale Sensor (V2) MS-003 (V2)

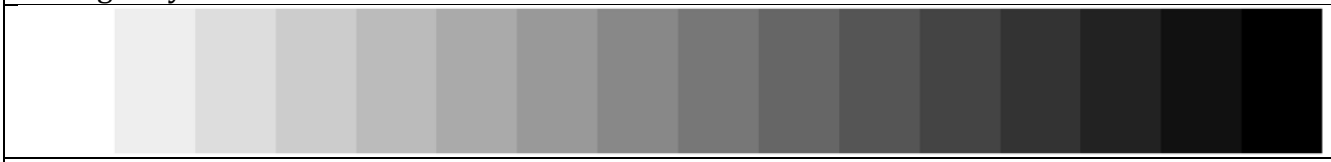
4.3.1.1. Overview

Front Back

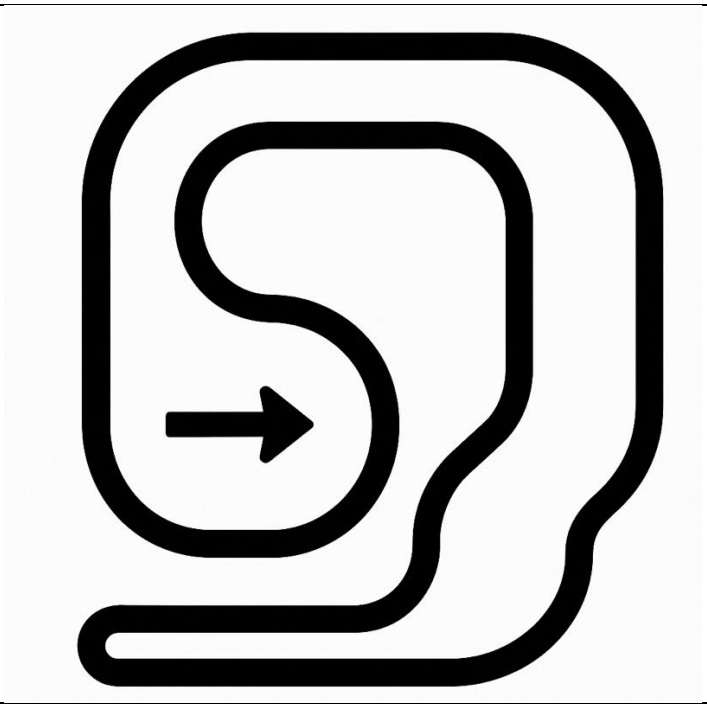


The gray scale sensor can read the brightness change of an object from white to black. It can use both digital and analog outputs. Common applications include simple grayscale detection and line following. The digital output threshold value can be set using the built-in adjustment screw, increasing flexibility of use.

Analog Grayscale



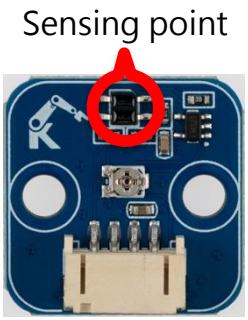
Digital Grayscale



Line tracking example: vehicle stays within black lines

4.3.1.2. Function

- Supports operating voltage 3.3V ~ 5V
- Equipped with miniature infrared sensor
- Fine-tune digital output threshold using tuning knob
- Provides both analog output (AOUT) and digital output (DOUT)
- Recommended sensing distance: 5~20mm. Adjust based on lighting conditions.



4.3.1.3. Project Application

- Line-tracking car
- Distance detection
- Grayscale sensing

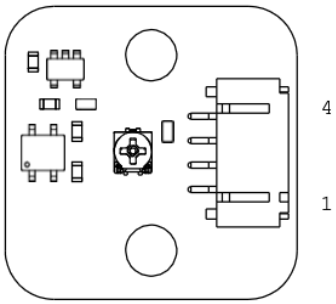
4.3.1.4. Electrical Characteristics

Parameter	Min	Typical*	Max	Units
Operating voltage (VCC)	3	3.3	5	Volts (V)
Sensing distance	-	-	50	Millimeters (mm)
IR wavelength	-	940	-	Nanometers (nm)
Analog output impedance*	-	330	-	Kiloohm (KΩ)
Digital output impedance	-	10	-	Kiloohm (KΩ)

Glossary:

*__Typical__ refers to the average/common value for each parameter. This represents the most common reference value under standard conditions. It can be used as a basis for the general performance of the component under normal working conditions.
*__Output impedance,__ also known as internal resistance or internal impedance, refers to the equivalent impedance (including static resistance and dynamic reactance) of a circuit as seen when looking into the circuit from the output port with the load connected.

4.3.1.5. Pinout

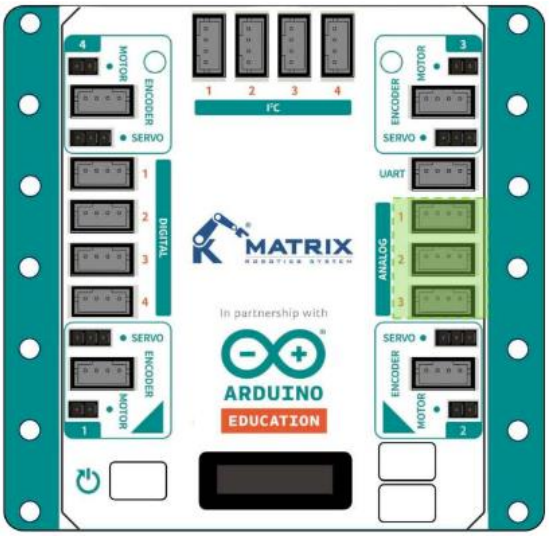


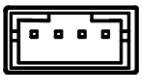
Pin Definition		
NO.	Name	Description
1	AOUT	The grayscale sensor has an analog output. When the grayscale value sensed is lower, the output voltage will be closer to VCC value.

2	DOUT	The grayscale sensor has digital output and the threshold of the sensor can be adjusted via the adjustable screw on the board.
3	VCC/5V	Voltage provided to the sensor for operation.
4	GND	The ground terminal in a circuit, used to complete the current loop, usually connected to the negative pole of the power supply.

4.3.1.6. Sample Code & Blocks Instructions

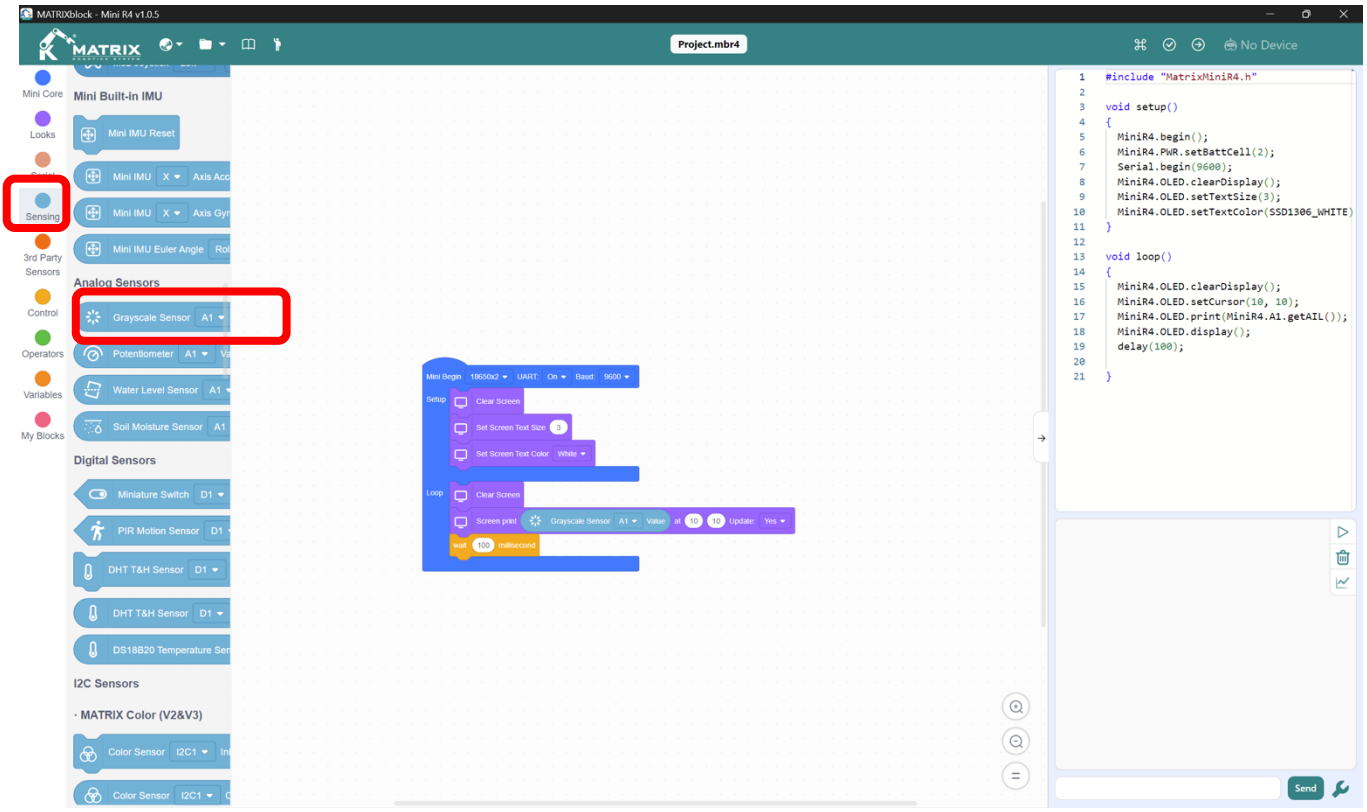
- **Hardware connection:** Please connect the grayscale sensor to the Analog 1 port of the MATRIX Mini R4 controller.



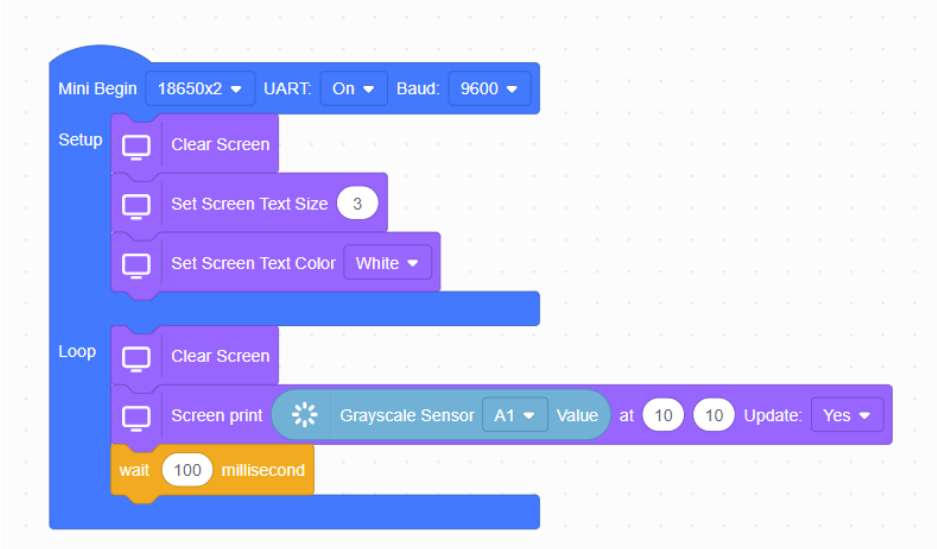


Pinout-Analog In			
No.	Name	I/O	Description
1	AIN A	Input	Analog input A
2	AIN B	Input	Analog input B
3	A5V	Output	Voltage output
4	GND	-	Grounding

- Sample Program：



MATRIXblock



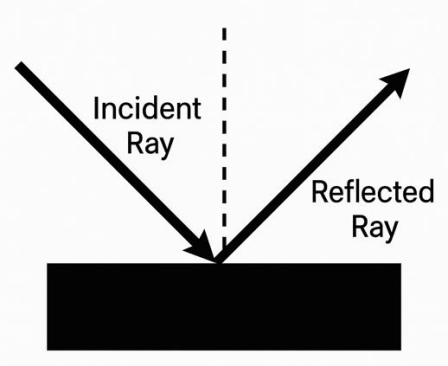
IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(3);
  MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setCursor(10, 10);
  MiniR4.OLED.print(MiniR4.A1.getAIL());
  MiniR4.OLED.display();
  delay(100);

}
```

- **Results:**
  The grayscale sensor value is displayed on the OLED, ranging from 0 to 1023. When it senses white, it returns a smaller value, and when it senses black, it returns a larger value.

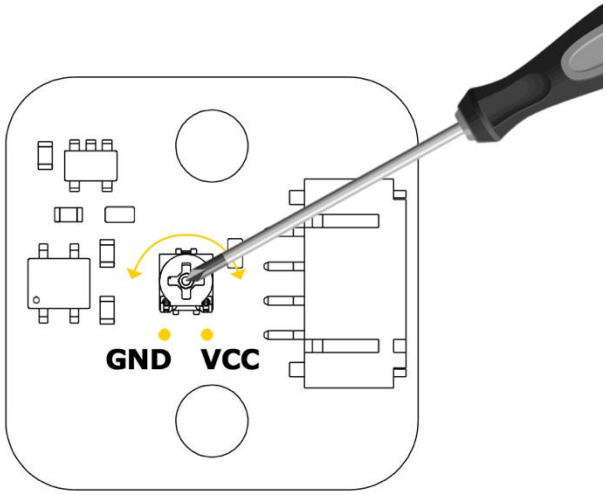
4.3.1.7. Principle Description

The photosensors inside the sensor detect the intensity of ambient or external light reflected from different surfaces. Since surfaces with different colors or materials may have different reflectivities, the sensor will output a voltage signal corresponding to the intensity of the received reflected light, reflecting changes in the brightness of the surface.

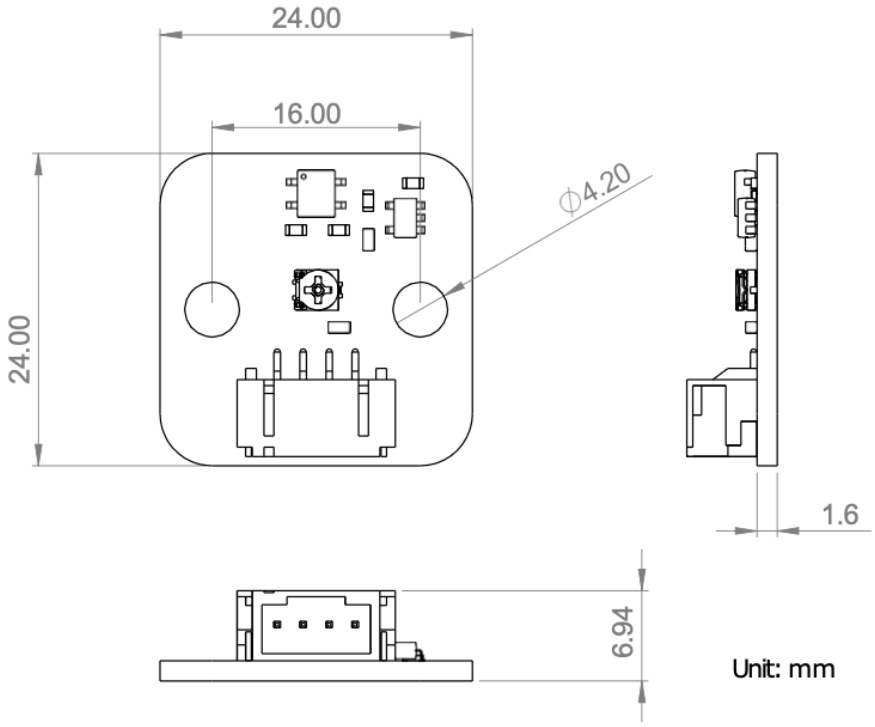


Light is reflected from object surface

- **AOUT (Analog Output)**: Output voltage changes with grayscale, white surfaces will produce lower output voltages and black surfaces will produce higher output voltages.
- **DOUT (Digital Output)**: There are only two states, high potential (1) and low potential (0). The switching timing is determined by the threshold knob set by the user. Rotating the knob clockwise brings the trigger point closer to VCC, rotating counterclockwise brings it closer to GND. For example, if the knob is turned clockwise, the digital output will switch to high (1) only when it senses a very dark surface, but if it is turned counterclockwise, the digital output will switch to high (1) when it senses a lighter colored surface. When there is high contrast between black and white areas (such as during line following), rotate clockwise to prevent the sensor treating external shadows as black areas. When there is low contrast, it is better to rotate counterclockwise to reduce misjudgment.



4.3.1.8. 尺寸 Dimensions



4.3.2. Potentiometer Sensor MS-013

4.3.2.1. Overview

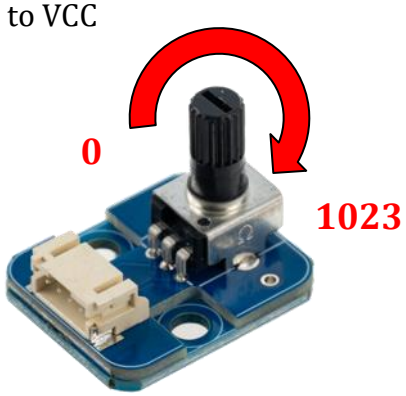
Sensor face

Back

A potentiometer sensor is a sensor that detects change in rotation angle and outputs a corresponding voltage. By adjusting the knob, users can adjust voltage to control electronic applications such as volume or brightness. The sensor outputs an analog sensor that can be easily read by the R4 controller.

4.3.2.2. Feature

- Supports 3.3V ~ 5V operating voltage
- Provides analog output (AOUT) with output voltage range of 0 to VCC
- Knob rotation angle ranges from 0° to 300°
- Output voltage changes linearly with rotation angle



4.3.2.3. Application

- Control volume or brightness
- Controller input
- Analog input devices (joysticks, knobs)
- E-learning kits for educational purposes

4.3.2.4. Electrical Characteristics

Parameter	Min	Typical*	Max	Units
Operating voltage (VCC)	3	3.3	5	Volts (V)
Output voltage	0	-	VCC	Volts (V)
Output impedance	-	10	-	Kiloohms (KΩ)
Rotation angle	0	-	300	Degrees (°)

Glossary

*__Typical__ refers to the average/common value for each parameter. This represents the most common reference value under standard conditions. It can be used as a basis for the general performance of the component under normal working conditions.

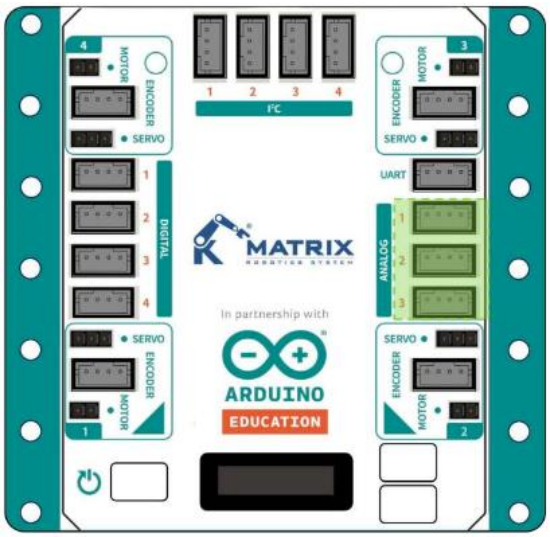
*__Output impedance,__ also known as internal resistance or internal impedance, refers to the equivalent impedance (including static resistance and dynamic reactance) of a circuit as seen when looking into the circuit from the output port with the load connected.

4.3.2.5. Pinout

Pin Definitions		
NO.	Name	Description
1	AOUT (Analog Output)	Analog output pin: voltage changes with knob angle.
2	NC (No Connection)	Not connected to circuit.
3	5V (Power Source)	Voltage provided to sensor for operation.
4	GND (Grounding)	The ground terminal in a circuit, used to complete the current loop, usually connected to the negative pole of the power supply.

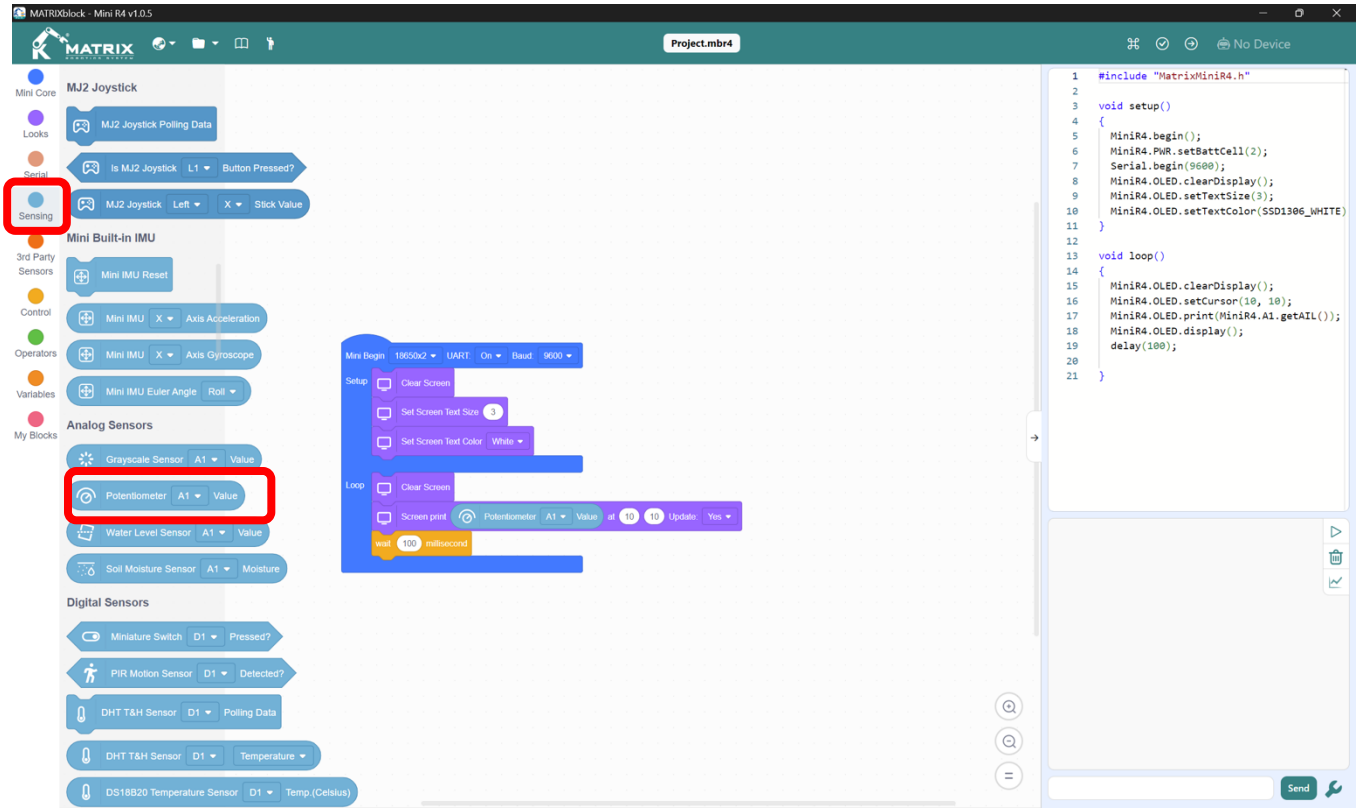
4.3.2.6. Sample Code & Blocks Instructions

- **Hardware connection:** Connect the sensor to the Analog 1 port of the Matrix Mini R4.

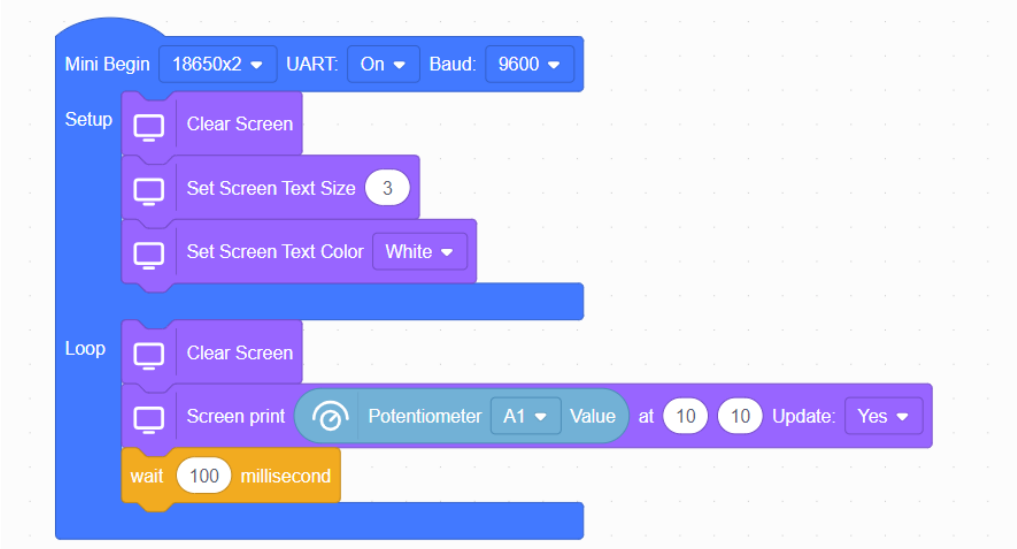


Pinout-Analog Input			
No.	Name	I/O	Description
1	AIN A	Input	Analog input A
2	AIN B	Input	Analog input B
3	A5V	Output	Power supply voltage output
4	GND	-	Grounding

- **Sample Program:**



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(3);
  MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setCursor(10, 10);
  MiniR4.OLED.print(MiniR4.A1.getAIL());
  MiniR4.OLED.display();
  delay(100);

}
```

- **Results:**
  As the sensor knob is rotated, the OLED screen will display a value between 0 and 1023. Turning it clockwise will increase the value and turning it counterclockwise will decrease the value.

4.3.2.7. Principle Description

The potentiometer sensor contains a rotating resistor element. When the user rotates the knob, the resistance value changes, causing the output voltage to change accordingly. The R4 controller reads the voltage and converts it into a digital value, allowing the sensor to be used in many applications. When the knob is turned clockwise, the value will increase, when turned counterclockwise, the value will decrease.

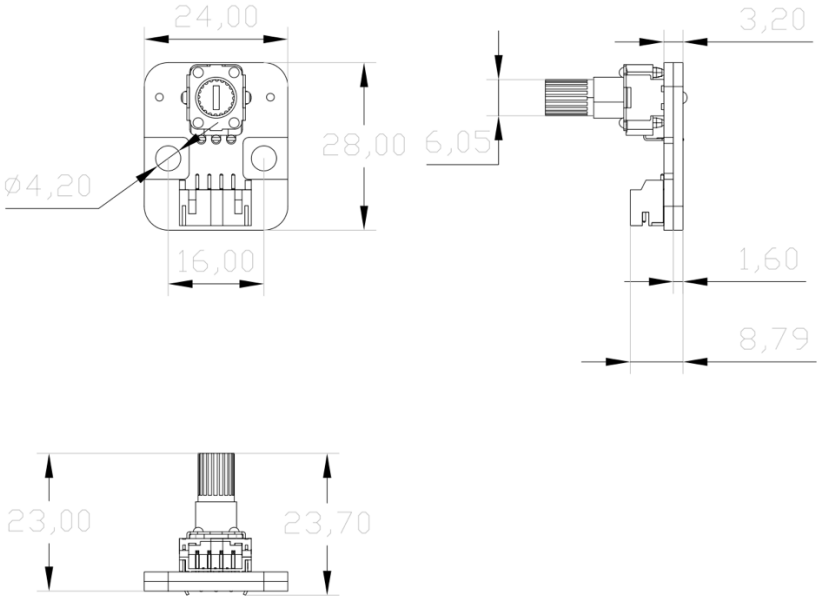
For example:

- When the knob is turned to the most counterclockwise position, the output voltage is near 0. This is used for shutting down the device or entering a minimum output state.

- When the knob is in the middle position, the voltage is around half of the maximum voltage, which can be used to set medium brightness, volume, or speed.
- When the knob is turned to the most clockwise position, the output voltage is close to VCC and can be used for maximum output requirements, such as maximum brightness, volume, or speed.

Because of these characteristics, potentiometer sensors are very useful as user input devices for manual adjustment of controls such as brightness, volume, or speed.

4.3.2.8. Dimensions



4.3.3. Water Level Sensor MS-014

4.3.3.1. Overview

The Water Level Sensor is a sensor that can detect the water level of a liquid. When a liquid contacts the sensing surface, the sensor will output an analog signal corresponding to the immersion depth. This can be used for simple water level detection and monitoring applications.

Features

- Supports 3.3V ~ 5V operating voltage
- Provides analog output (AOUT) which changes with water level
- Easy to use, simple for entry-level learning and simple water detection
- PCB anti-rust design, suitable for long-term immersion (not complete immersion)

4.3.3.2. Applications

- Water level detection and alarm system
- Automatic irrigation equipment
- Aquarium water-level management
- Rain detection
- Leak detection

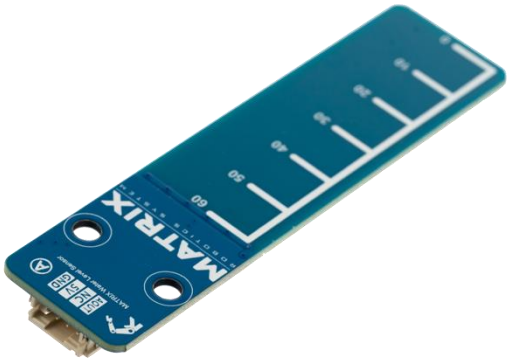
4.3.3.3. Electrical Characteristics

Parameter	Min	Typical*	Max	Units
Operating voltage (VCC)	3	3.3	5	Volts (V)
Output voltage	0	-	VCC	Volts (V)
Sensing surface length	-	-	60	Millimeters (mm)

Glossary:

*__Typical__ refers to the average/common value for each parameter. This represents the most common reference value under standard conditions. It can be used as a basis for the general performance of the component under normal working conditions.

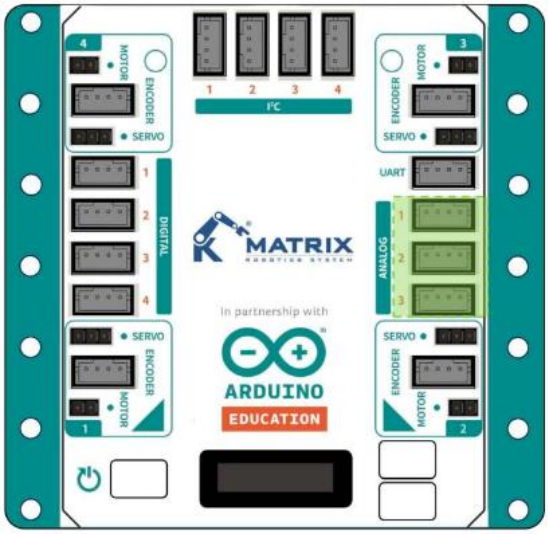
4.3.3.4. Pinout

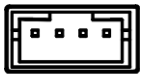


Pin Definitions		
NO.	Name	Description
1	AOUT (Analog Output)	Analog output pin, outputs corresponding voltage as water level changes.
2	NC (No Connection)	Not connected to circuit.
3	5V (Power source)	Voltage provided to sensor for operation.
4	GND (Grounding)	The ground terminal in a circuit, used to complete the current loop, usually connected to the negative pole of the power supply.

4.3.3.5. Sample Code & Blocks Instructions

- **Hardware connection:** Connect the sensor to the Analog 1 port of MATRIX Mini R4.

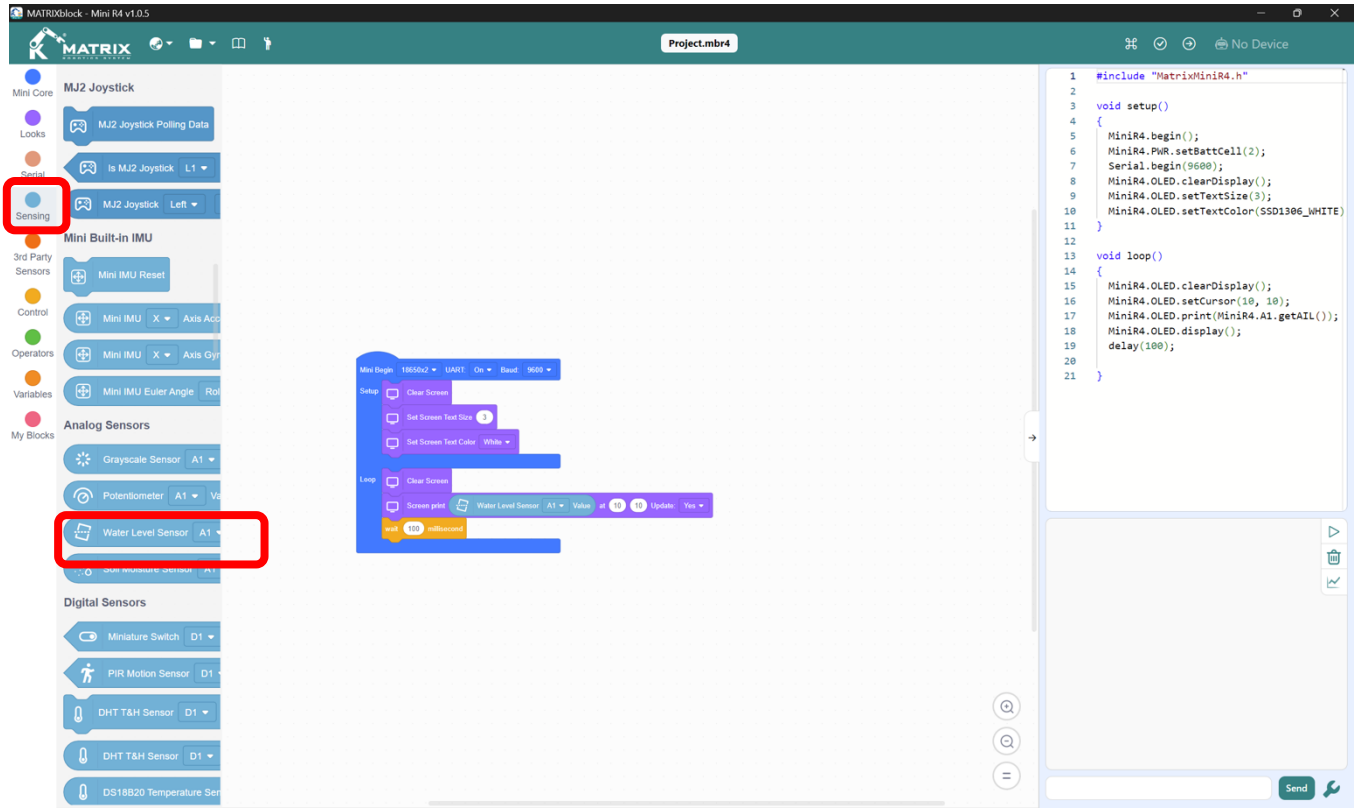




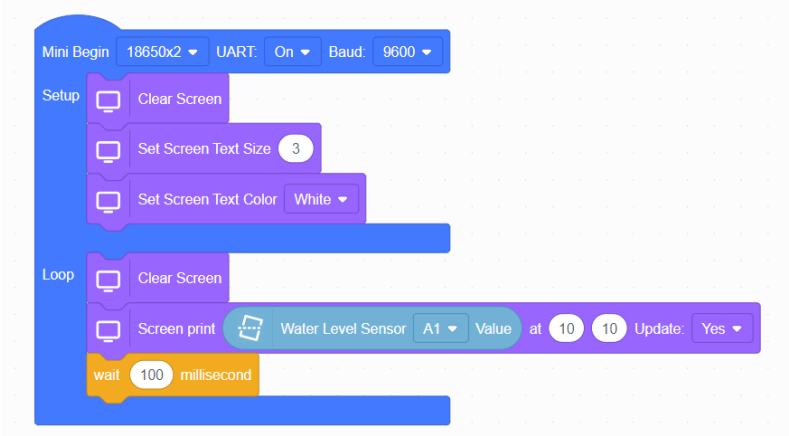
1 4

Pinout-Analog Input			
No.	Name	I/O	Description
1	AIN A	Input	Analog input A
2	AIN B	Input	Analog input B
3	A5V	Output	Power supply voltage output
4	GND	-	Grounding

- **Sample Program:**



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(3);
  MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setCursor(10, 10);
  MiniR4.OLED.print(MiniR4.A1.getAIL());
  MiniR4.OLED.display();
  delay(100);

}
```

- **Results:**
  As the water level rises, the amount of water contacting the sensing surface rises, and the value displayed by the OLED will also rise. When the water level drops, the value will drop, which is applicable to water level detection and volume change detection.

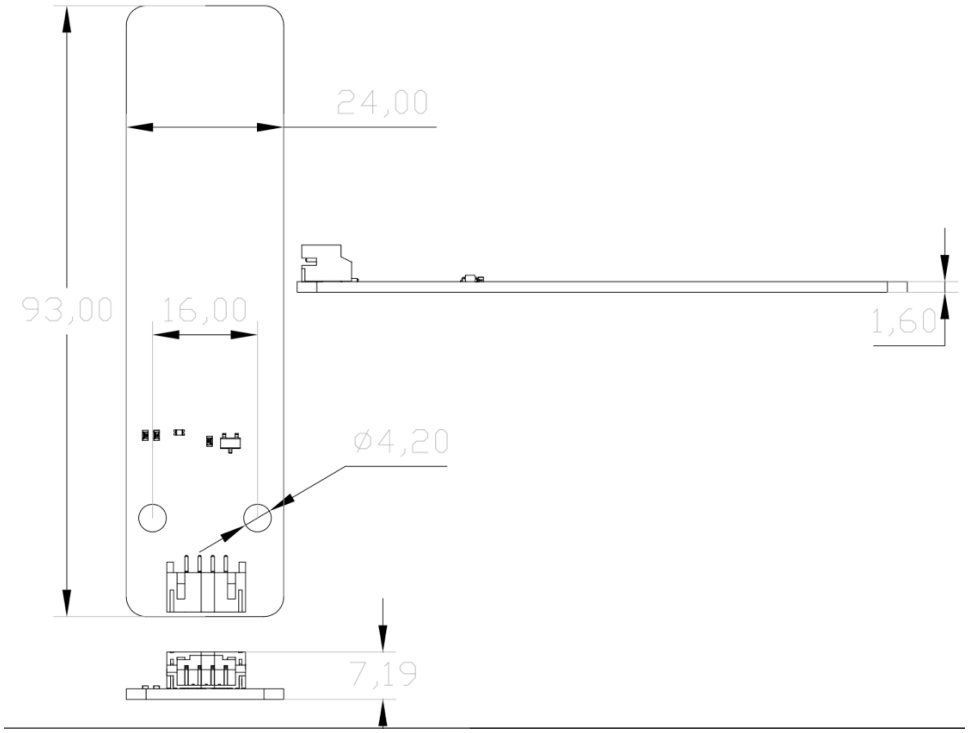
4.3.3.6. Principle Description

The water level sensor is designed to sense the conductivity of water, making it able to detect water level changes. As the water rises and contacts more of the sensing surface, the conductive area increases along with the output voltage. As the water level decreases, the output voltage will also decrease. For example, when the sensor is completely dry, there is almost no voltage output; when the water level is at a moderate level, the output voltage is approximately between 300 and

550; and when the water almost covers the entire sensing area, the output voltage exceeds 650. By using these values, the current water level status can be determined and used to trigger a corresponding control or reminder, such as an alarm or water pump.
Note: The sensor is not designed to be fully immersed in water. Please do not immerse the entire sensor for extended periods of time to avoid damage.
Output numerical reference example:

- Near 0: Dry
- 300~550：Partially submerged
- Exceeding 650：Almost completely or completely submerged

4.3.3.7. Dimensions



4.3.4. Soil Moisture Sensor MS-016

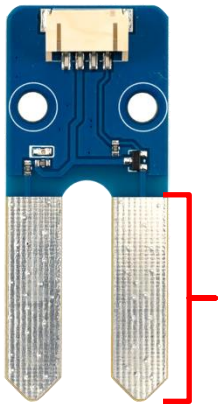
Front Back

4.3.4.1. Overview

The soil moisture sensor measures the amount of water present in soil. The probes sense the level of conductivity of the soil to determine the moisture content. It converts changes in conductivity to analog voltage signals, allowing the controller to read the values for various applications. This type of sensor is often used in smart agriculture, automatic irrigation, and environmental monitoring.

4.3.4.2. Features

- 3.3 – 5V operating voltage
- Provides analog output (AOUT) which changes with soil moisture
- Simple design, easy to install, and has various applications
- Can be used as a practical sensor in teaching or special projects



4.3.4.3. Applications

- Soil moisture monitoring in smart agriculture
- Provide start and stop conditions for automatic irrigation
- Greenhouse soil moisture management
- Soil dryness alarm and reminder

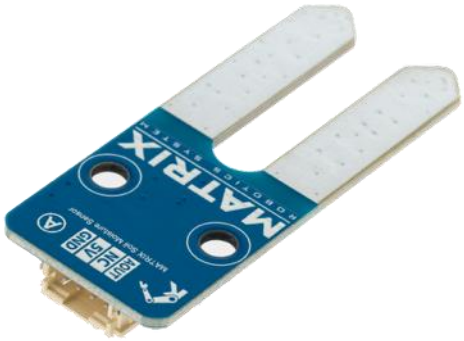
4.3.4.4. Electrical Characteristics

Parameter	Min	Typical*	Max	Units
Operating voltage (VCC)	3	3.3	5	Volts (V)
Operating current	-	35	-	Milliamperes (mA)
Output voltage	0	-	VCC	Volts (V)
Sensing area length	-	-	30	Millimeters (mm)

Glossary:

*__Typical__ refers to the average/common value for each parameter. This represents the most common reference value under standard conditions. It can be used as a basis for the general performance of the component under normal working conditions.

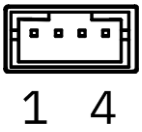
4.3.4.5. Pinout



Pin Definition		
NO.	Name	Description
1	AOUT	Analog output pin, outputs corresponding voltage as soil moisture changes.
2	NC	Not connected to circuit.
3	5V	Voltage provided to sensor for operation.
4	GND	The ground terminal in a circuit, used to complete the current loop, usually connected to the negative pole of the power supply.

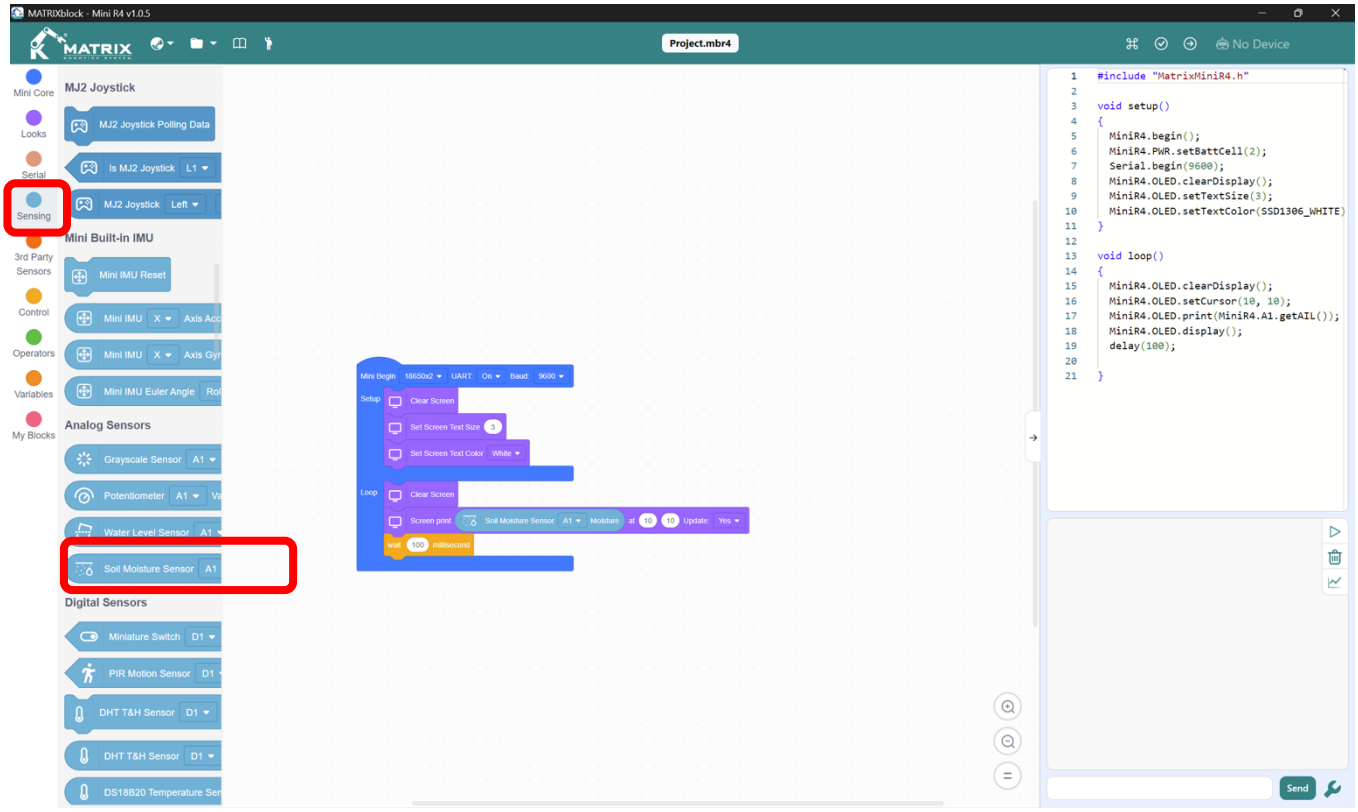
4.3.4.6. Sample Code & Blocks Instructions

- **Hardware connection:** Connect the sensor to the Analog 1 port of MATRIX Mini R4.

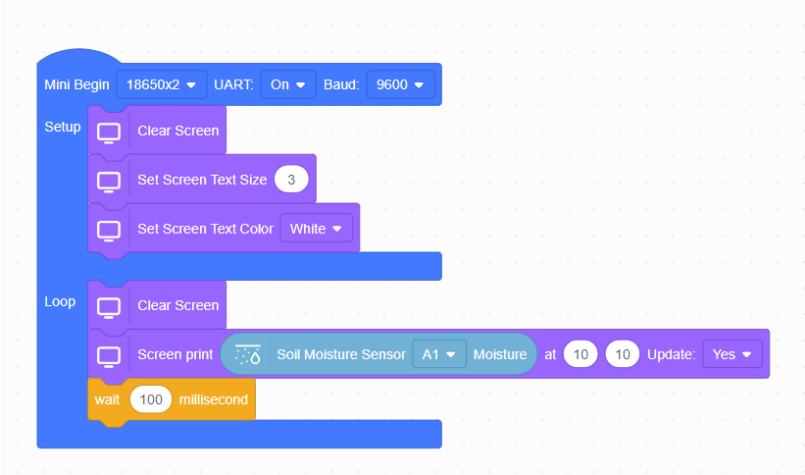


Pinout-Analog In			
No.	Name	I/O	Description
1	AIN A	Input	Analog input A
2	AIN B	Input	Analog input B
3	A5V	Output	Analog output
4	GND	-	Grounding

- **Sample Program:**



MATRIXblock

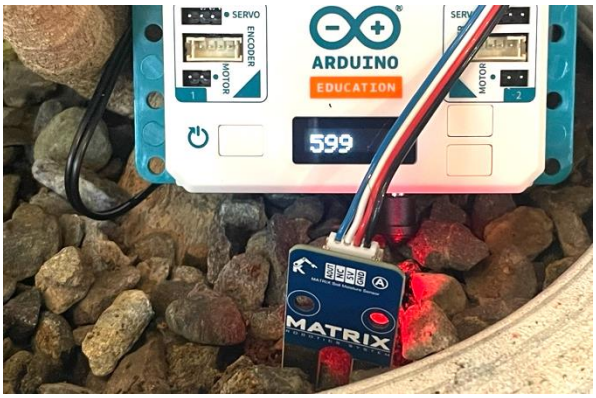


IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(3);
```

```
    MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
    MiniR4.OLED.clearDisplay();
    MiniR4.OLED.setCursor(10, 10);
    MiniR4.OLED.print(MiniR4.A1.getAIL());
    MiniR4.OLED.display();
    delay(100);

}
```

- **Results:**
  When soil moisture increases, the conductivity sensed by the probe becomes stronger and the value displayed by the OLED increases, when the soil becomes drier, the conductivity and display value decrease. These values can be used for automatic irrigation controls and watering reminders.



4.3.4.7. Principle Description

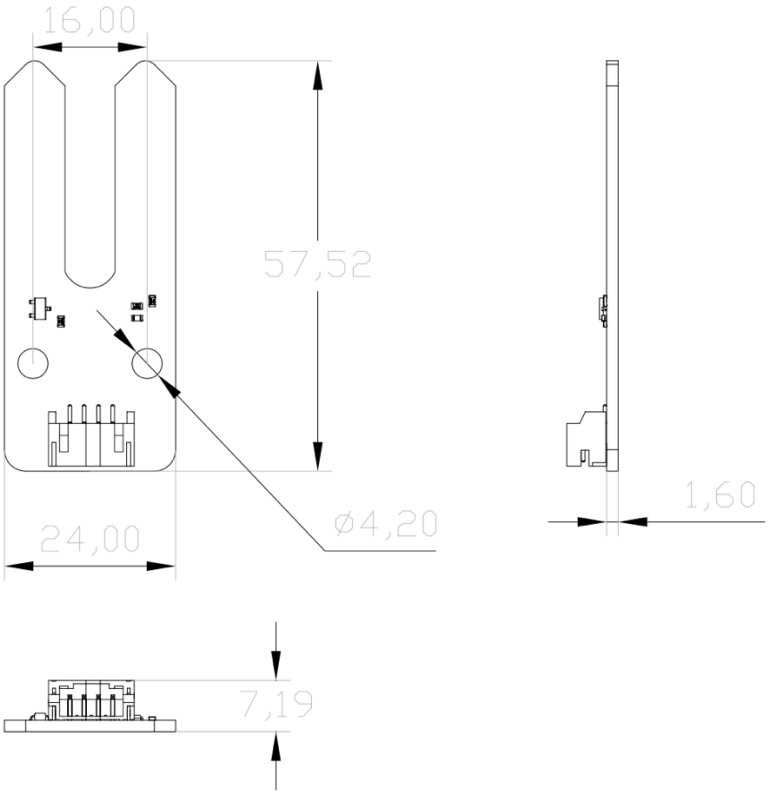
The soil moisture sensor uses the conductivity of the soil to determine water content. When the soil is moist, the soil has a stronger conductivity and outputs a higher voltage. When the soil is dry, the conductivity decreases and the output voltage is lower. The R4 controller can quickly read these values to determine soil status.

Note: This sensor is not designed to be fully immersed in water. Avoid immersing in water for extended periods of time, and clean regularly to avoid sediment buildup.

Output value reference guide

- 0 ~ 300：Dry soil
- 300 ~ 700：Moist soil
- 700 ~ 950：Placed in water

4.3.4.8. Dimensions



4.4. External Digital Inputs

4.4.1. Miniature Switch(V2/V3) MS-004(V2/V3)

4.4.1.1. Overview

The Miniature Switch is a commonly used digital input sensor with a simple and clear output behavior.

- When the switch is pressed it outputs a HIGH voltage (1).
- When the switch is released, it outputs a LOW voltage (0).
  Due to its simple structure and fast response, it is especially suitable for applications such as object collision detection, position sensing, and manual control.

4.4.1.2. Feature

- Outputs a digital signal (HIGH/LOW) with only two states: pressed and not pressed.
- Built-in spring return mechanism, automatically resets after bring pressed.
- Compact size and easy installation, suitable for space-limited applications.
- Simple and clear digital output makes control logic design easier.
  - Pressed → outputs HIGH (1)
  - Released→ outputs LOW (0)

4.4.1.3. Application

- Robot positioning detection (e.g., wall collision detection)
- Manual button control
- Travel or limit position detection in automated devices
- DIY electronics projects or interactive game devices

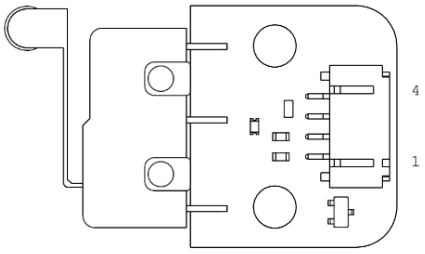
4.4.1.4. Electrical Characteristics

Parameter	Min	Typ	Max	Units
Operating Voltage (VCC)	3.3	-	5	V
Operating Force*	-	-	0.49	N
Switch Bounce Time*	-	-	5	ms

Explanation：

**Operating force* refers to the force required to press the switch, measured in Newtons (N)
**Switch bounce time* is the brief time signal instability period that occurs right after pressing or releasing the switch.

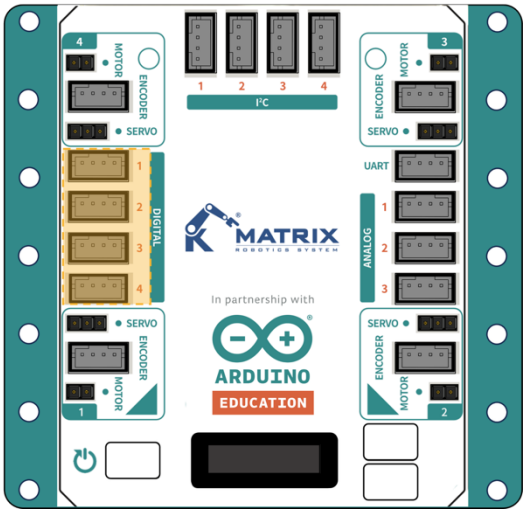
4.4.1.5. Pinout



Pinout		
NO.	Name	Description
1	OUT (Output)	Outputs HIGH (1) when pressed, LOW (0) when not pressed.
2	NOUT (Inverted Output)	Inverted output pin, outputs LOW (0) when pressed.
3	5V	Voltage supply pin for powering the sensor.
4	GND	Ground supply: This refers to the ground terminal in a circuit, used to complete the current loop. It is usually connected to the negative terminal of the power supply.

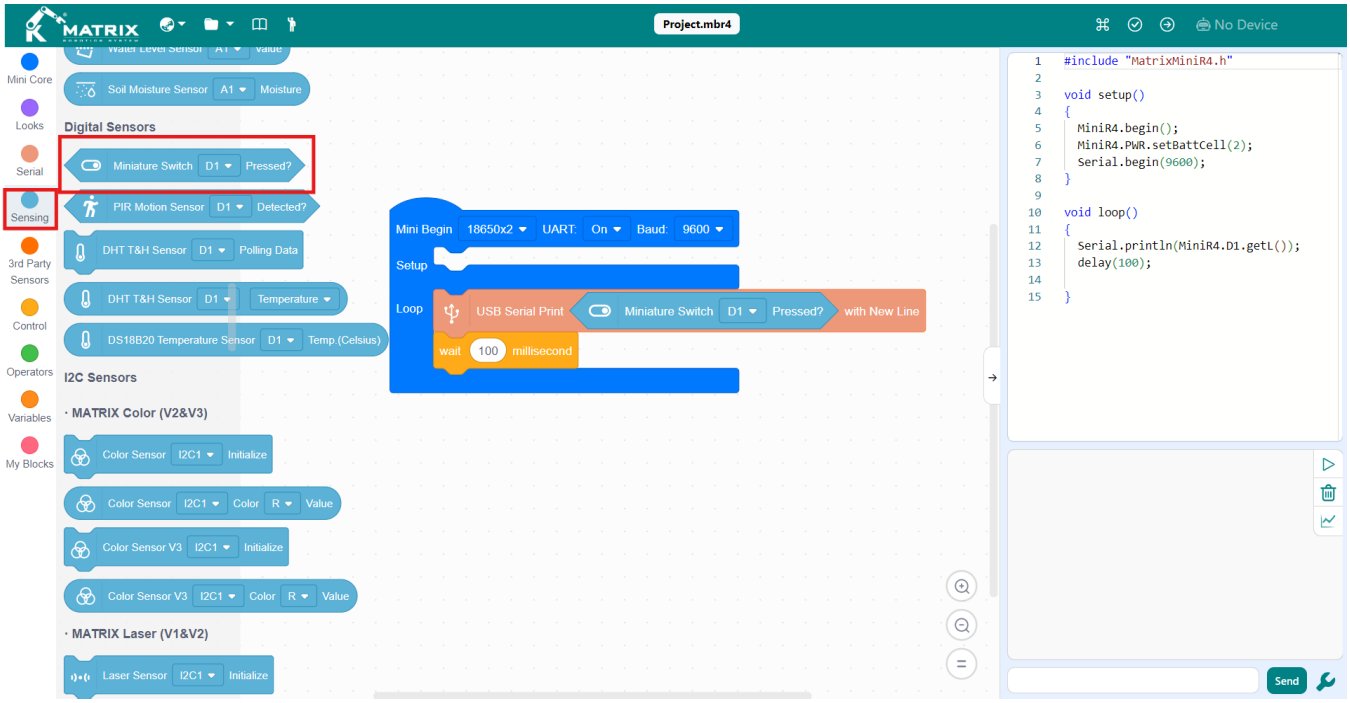
4.4.1.6. Sample Code & Blocks Instructions

- **Hardware connection**：Please connect the Miniature Switch to the **Digital 1** port of the MATRIX Mini R4 controller.

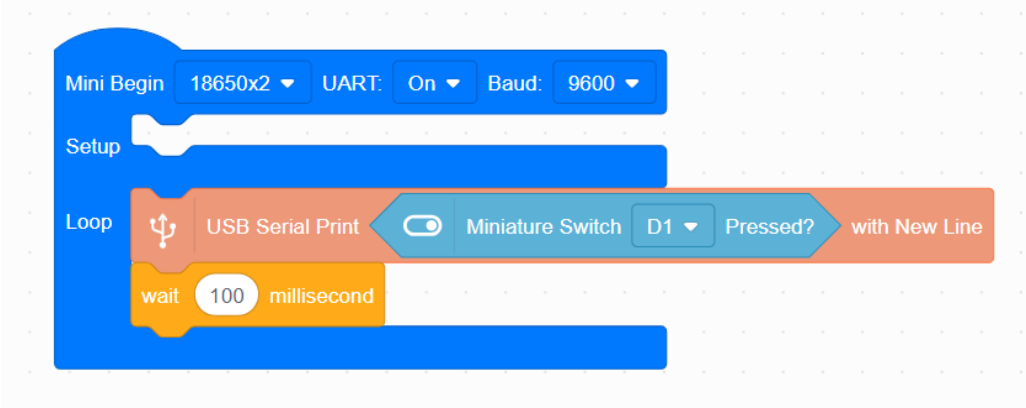


Pinout-Digital I/O			
No.	Name	I/O	Description
1	DIO A	Input/Output	Digital I/O pin A (GPIO A), for signal detection or output.
2	DIO B	Input/Output	Digital I/O pin B (GPIO B), for signal detection or output.
3	VCC	Output	Power supply pin.
4	GND	-	Ground pin

- **Sample Code**：



MATRIXblock



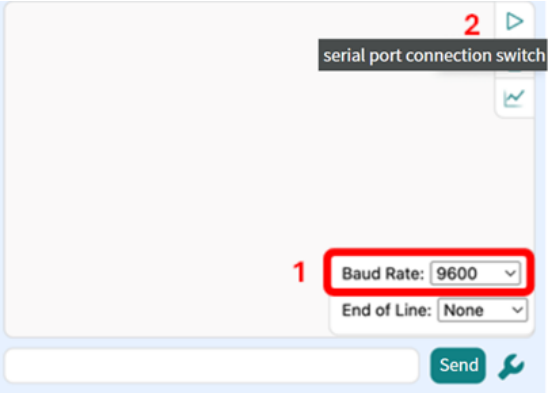
IDE C++

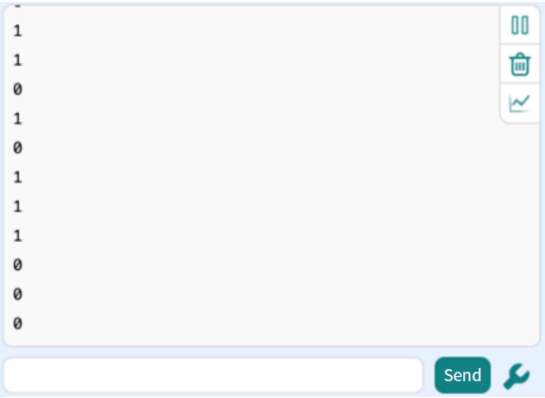
```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
}

void loop()
{
  Serial.println(MiniR4.D1.getL());
  delay(100);
```

```
14.
15. }
```

- **Results Explanation：**

  When the miniature switch is pressed, the serial monitor will display a value of 1 (HIGH). When released, the value will display 0 (LOW). The status is updated every 0.1 seconds.
  1.Confirm that the Baud Rate matches between the program and the serial monitor.
  2. Press the serial port connect button





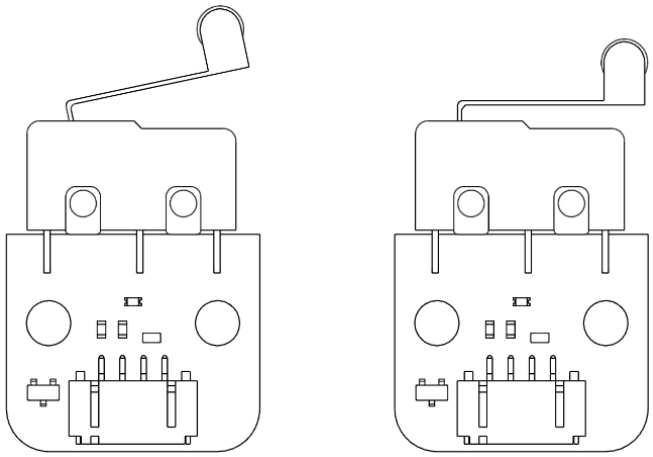
4.4.1.7. Principle Description

The miniature switch uses a mechanical internal structure. When the button is pressed, a metal contact closes the circuit, outputting a HIGH signal. When the button is released, a spring resets the contact, breaking the circuit and outputting a LOW signal. The switch has a short bounce time, making it suitable for real-time detection applications.

Notes:

- For higher stability in programs, consider implementing software debouncing to avoid false triggers.
- Avoid applying excessive force or lateral impact to maintain the component's lifespan and stability.

Output Value

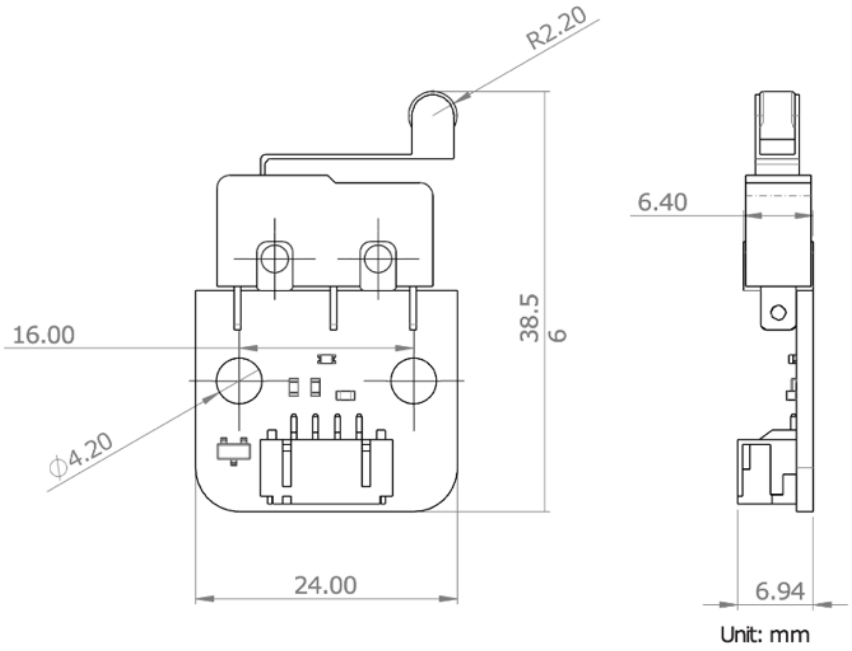


- When pressed：1（HIGH）

- When released：0（LOW）

This state can be directly used in conditional logic and program control. It is simple and intuitive.

4.4.1.8. Dimensions



4.4.2. PIR Motion Sensor MS-012

4.4.2.1. Overview

The PIR（Passive Infrared）motion sensor is a passive infrared sensing module that detects changes in infrared radiation caused by the movement of humans or animals. When motion is detected within the sensing area, the module outputs a HIGH voltage signal. Otherwise, it maintains a LOW voltage.

Sensing side

Back side

4.4.2.2. Features

- Passive infrared sensing: does not emit any signals, making it energy-efficient and stable
- Built-in digital output, easy to connect with controller
- Detection range: up to 7 meters, sensing angle: approx. 100 degrees
- Suitable for applications like automatic lighting, security systems, and smart home devices

4.4.2.3. Application

- Automatic door control systems
- Indoor automatic lighting switches
- Motion detection in smart home systems
- Intrusion detection in security systems

4.4.2.4. Electrical Characteristics

Parameter	Min	Typ	Max	Units
Operating voltage (VCC)	3.3	-	5	Volts (V)
Static Current*	-	15	-	Microamperes (μA)
Sensing Distance	-	7	-	Meters (m)
Sensing Angle	-	100	-	Degrees (°)

Definition：

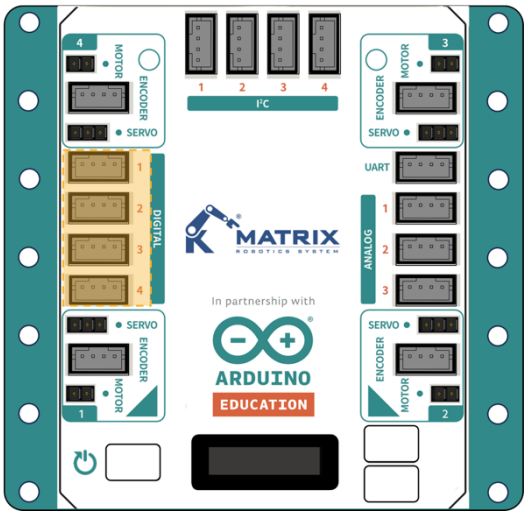
***Static current** refers to the current consumption when the sensor is in standby

4.4.2.5. Pinout

腳位定義		
NO.	Name	Description
1	DOUT (Digital Output)	Digital output pin: outputs HIGH (1) when motion is detected.
2	NC (No connection)	No pin connection.
3	5V (Power Supply)	Power supply: provides the voltage required for sensor operation.
4	GND (Ground)	Ground: completes the circuit, typically connected to power negative.

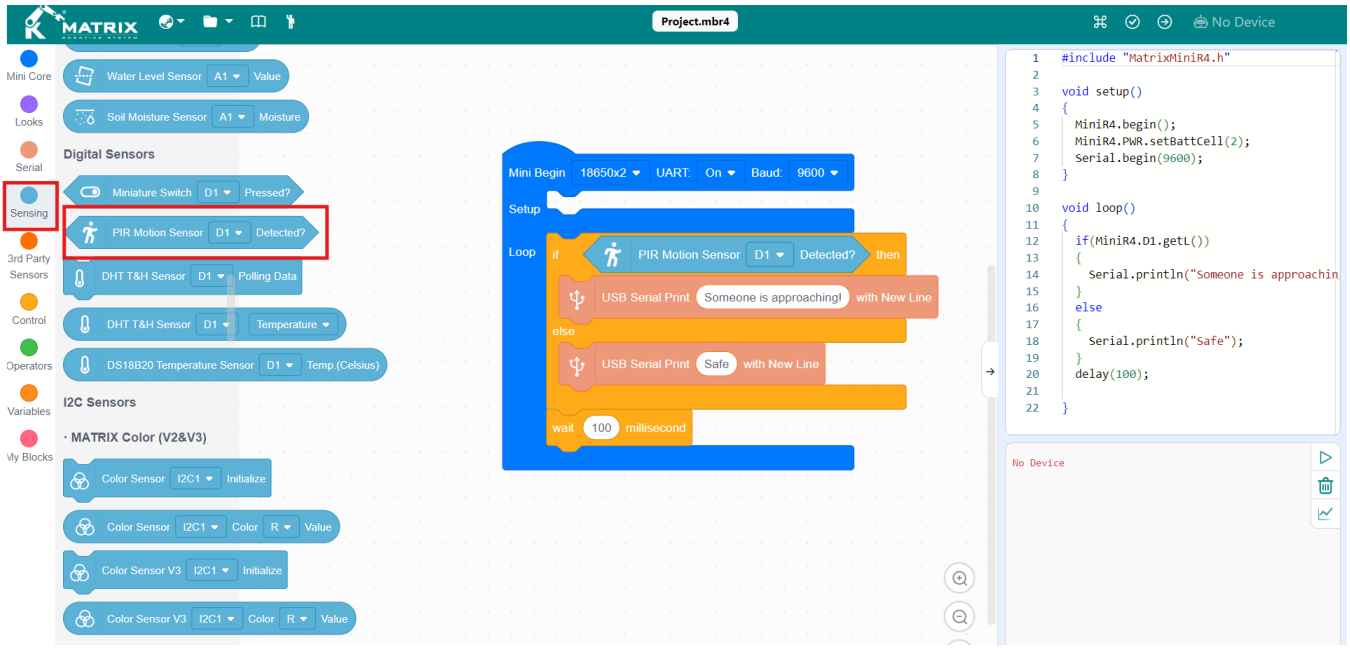
4.4.2.6. Sample Code & Blocks Instructions

- **Hardware Connection**：Please connect the micro switch to the MATRIX Mini R4 controller's Digital 1 interface.

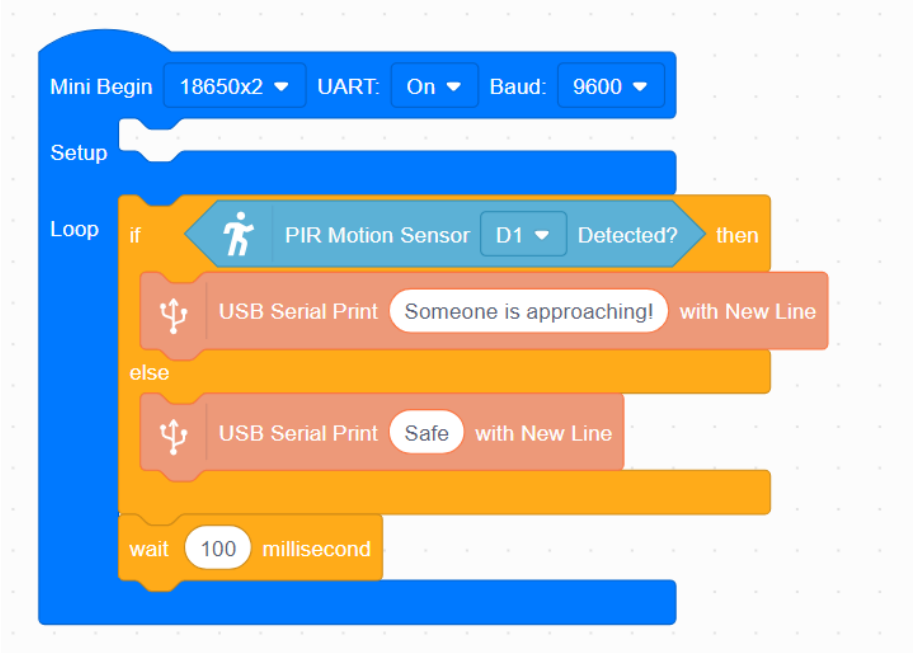


Pinout-Digital I/O			
No.	Name	I/O	Description
1	DIO A	I/O	Digital I/O pin A (GPIO A), can be used for detecting or outputting digital signals
2	DIO B	I/O	Digital I/O pin B (GPIO B), can be used for detecting or outputting digital signals
3	VCC	Output	Power supply pin
4	GND	-	Power ground

- **Sample Program：**



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
```

```
    Serial.begin(9600);
}

void loop()
{
    if(MiniR4.D1.getL())
    {
        Serial.println("Someone is approaching!");
    }
    else
    {
        Serial.println("Safe");
    }
    delay(100);

}
```

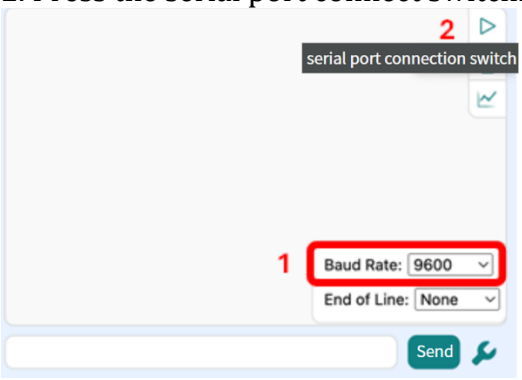
Result Description：

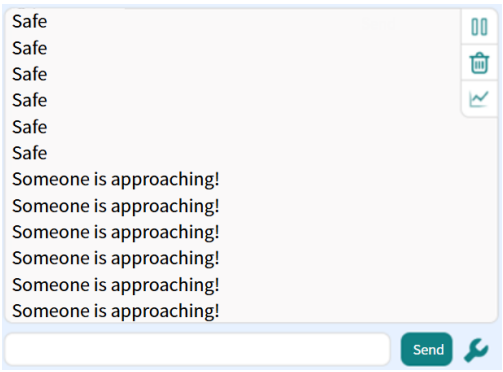
The program continuously monitors the digital output status of the PIR infrared sensor and displays the corresponding message through the serial port:

- When the sensor detects infrared changes (e.g., a human body moving within the detection range), it displays **"Someone is approaching!"**
- When there is no infrared change in the environment (i.e., no moving object), it displays **"Safe"**

The status is updated every 0.1 seconds.

1. Make sure the baud rate matches that set in the program.
2. Press the serial port connect switch.





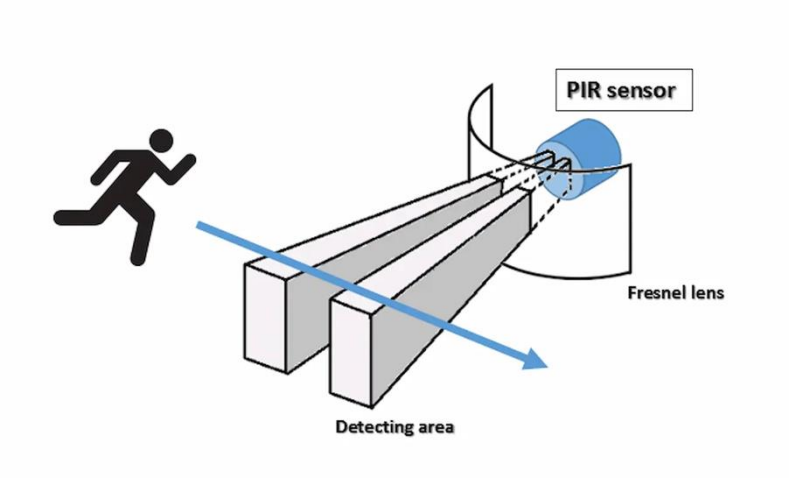
4.4.2.7. Principle Description

The core principle of the PIR (passive infrared) sensor is based on a pyroelectric sensor that detects changes in infrared radiation emitted by human bodies or animals. Human and animal body temperatures are around 36°C, continuously emitting infrared radiation.

When this infrared radiation enters the sensor's detection range and its intensity changes (e.g., when a person moves into or out of view), the sensing element generates an electrical signal, causing the digital output pin to go HIGH (1).

The Fresnel lens in front of the PIR sensor focuses infrared radiation from a wide area onto the sensing element, enabling it to cover a broader angle and distance. Once the sensor detects a change in infrared due to the movement of a heat source, it outputs a brief HIGH signal.

Note: PIR sensors do not detect stationary humans or objects, as they only respond to changes in infrared intensity. Therefore, in the absence of movement, the output remains LOW (0).



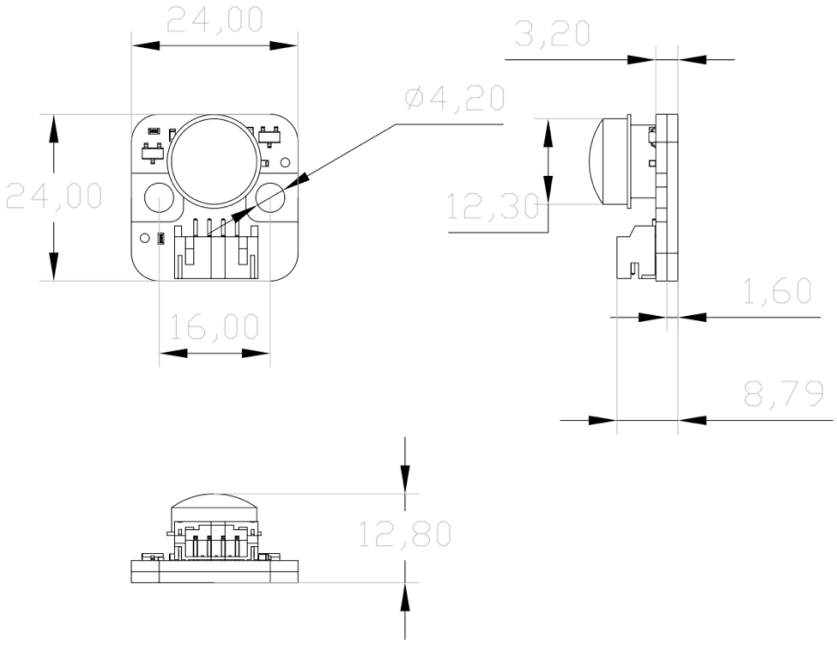
Application Supplement：

This characteristic makes PIR sensors ideal for application triggered by motion detection, such as:

- Automatic lighting systems: Lights turn on when someone passes by and turns off when no one is around.
- Smart home security systems: Alarms or recordings are triggered when someone enters the area.
- Robot interaction applications: automatically activate voice or motion modules when a person approaches.

With simple programming, users can implement conditions checks based on the HIGH/LOW output of the sensor to achieve energy-saving, interactive, and automated control.

4.4.2.8. Dimensions



4.4.3. DHT Temperature and Humidity Sensor MS-011

4.4.3.1. Overview

The DHT temperature and humidity sensor is a digital sensing module that combines both temperature and humidity detection functions. It has a built-in digital signal processing unit that can simultaneously read ambient temperature and relative humidity, transmitting the data via a single digital signal pin. It is suitable for smart home systems, environmental monitoring, educational projects, etc., and features a compact size, low power consumption, and simple operation.

Sensing side

Back side

4.4.3.2. Features

- Supports 3.3V to 5V operating voltage
- Detects both temperature and relative humidity
- Single digital pin outputs both parameters (temperature + humidity)
- Stable accuracy, suitable for general environmental monitoring
- Built-in signal filtering and temperature compensation for strong anti-interference
- Temperature measurement range: 0 ~ 50°C, accuracy ±2°C
- Humidity measurement range：20 ~ 90% RH, accuracy ±5% RH

4.4.3.3. Application

- Environmental variable monitoring
- Air quality monitoring in smart agriculture
- Indoor climate control (e.g., air conditioning or dehumidifiers)
- DIY weather stations or IoT nodes

4.4.3.4. Electrical Characteristics

Parameter	Min	Typ*	Max	Units
Operating Voltage (VCC)	3.3	-	5	V
Operating Current	-	2.5	-	mA
Temperature Range	0	-	50	°C
Temperature Accuracy*	-	±2	-	°C
Humidity Range	20	-	90	% RH
Humidity Accuracy*	-	±5	-	% RH
Data Update Cycle*	-	1	-	sec

Definitions：

Typ* refers to the typical value under standard conditions; it is not the minimum or the maximum limit.

Temperature and Humidity Accuracy* indicate the maximum deviation after internal calibration.

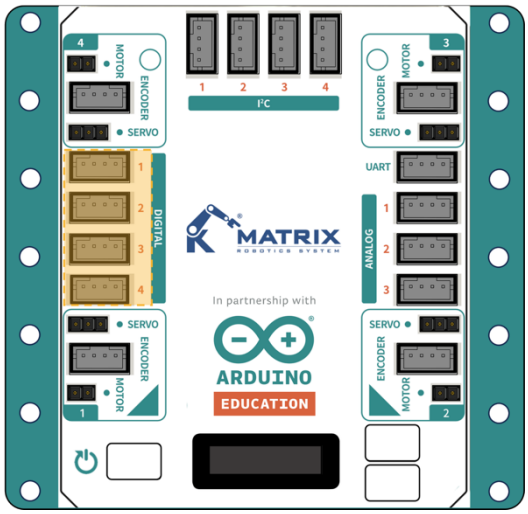
Data Update Cycle*: This module uses the DHT11 chip, which updates data once per second. It is recommended that the reading interval is not less than 1 second.

4.4.3.5. Pinout

Pinout Definition		
NO.	Name	Description
1	DOUT (Digital Output)	Digital output pin used to transmit encoded temperature and humidity data.
2	NC (No connection)	No connection to this pin.
3	5V (Power Supply)	Provides the operating voltage required by the sensor.
4	GND (Ground)	Power ground, typically connected to the power source's negative terminal.

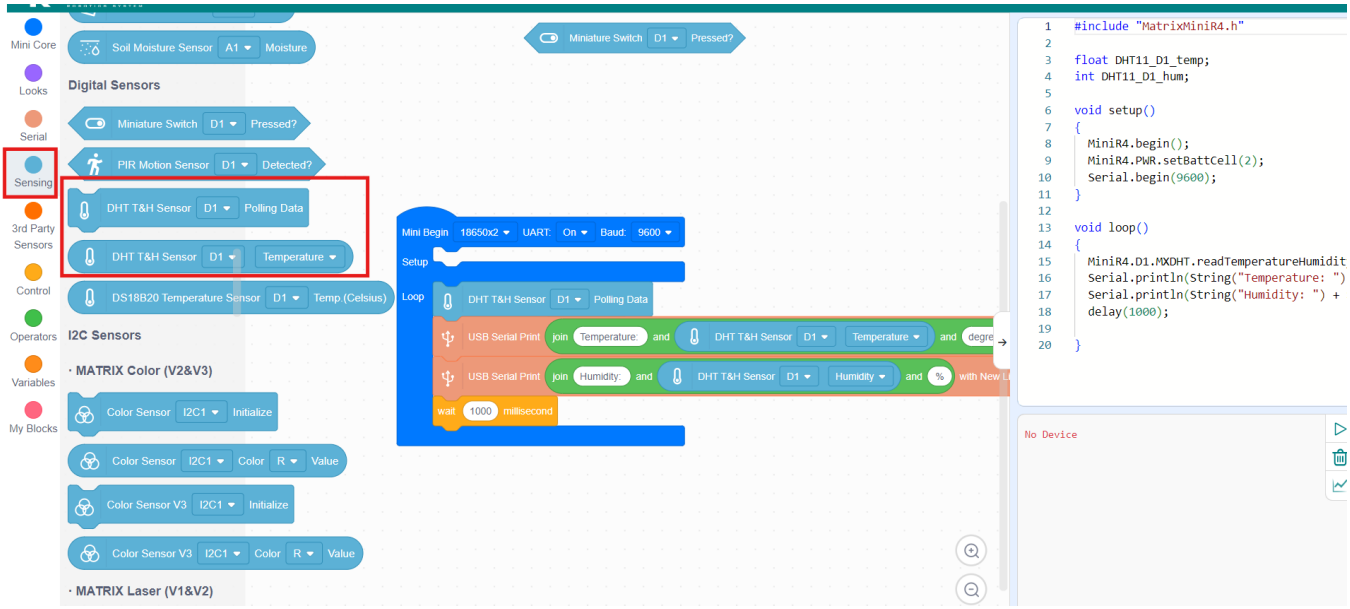
4.4.3.6. Sample Code & Blocks Instructions

- **Hardware connection:** Please connect the temperature and humidity sensor to the Digital 1 port of the MATRIX Mini R4 controller.

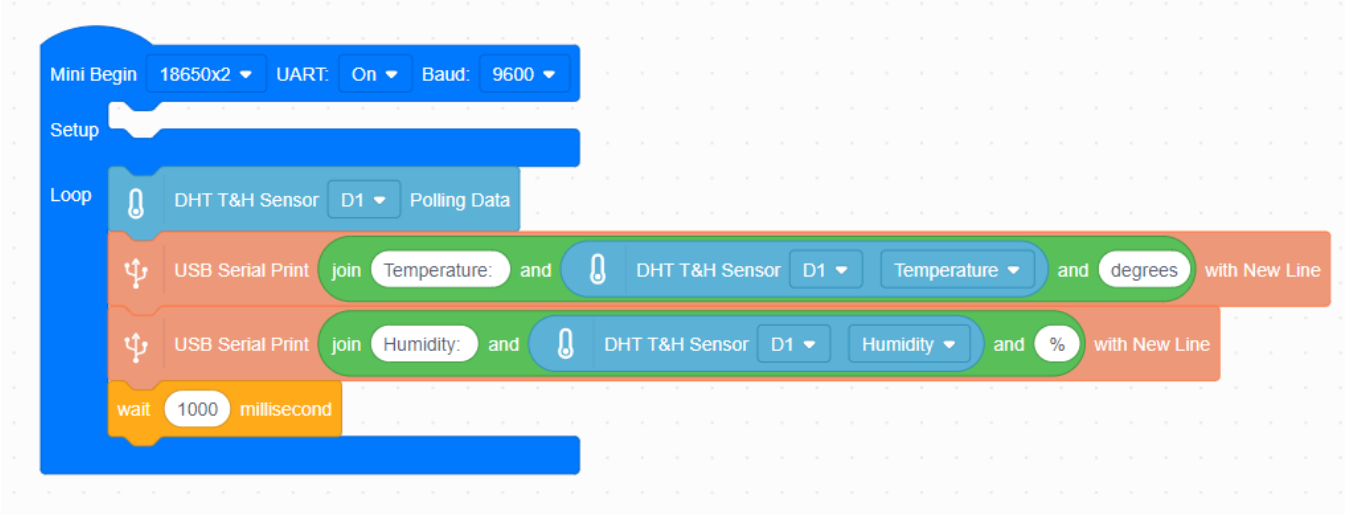


Pinout-Digital I/O			
No.	Name	I/O	Description
1	DIO A	I/O	Digital I/O pin A (GPIO A), used for detecting or outputting digital signals
2	DIO B	I/O	Digital I/O pin B (GPIO B), used for detecting or outputting digital signals
3	VCC	O	Power supply pin
4	GND	-	Power ground

- **Sample code：**



MATRIXblock



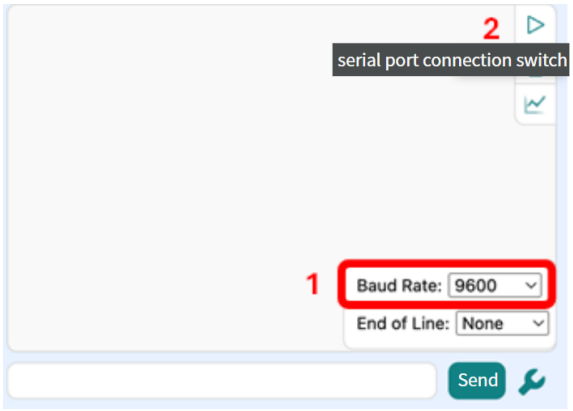
IDE C++

```
#include "MatrixMiniR4.h"

float DHT11_D1_temp;
int DHT11_D1_hum;

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
}

void loop()
{
  MiniR4.D1.MXDHT.readTemperatureHumidity(DHT11_D1_temp, DHT11_D1_hum);
  Serial.println(String("Temperature :") + String(DHT11_D1_temp) + String("degrees"));
  Serial.println(String("Humidity :") + String(DHT11_D1_hum) + String("%"));
  delay(1000);

}
```

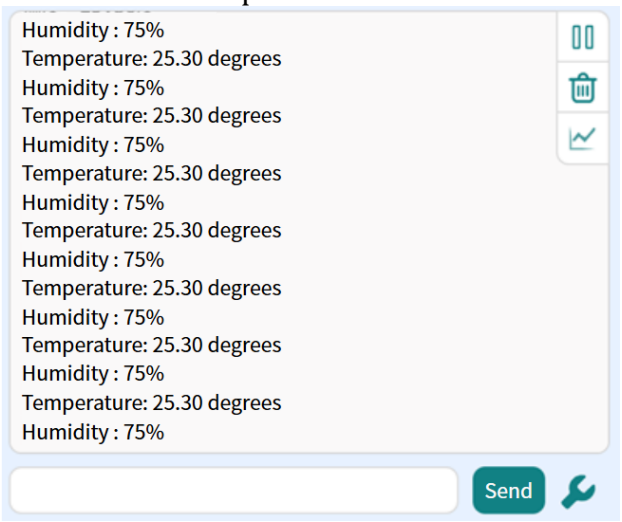
Result description：

The program will continuously display the current ambient temperature and humidity in the serial monitor window, with data updated every 1 second.

1. Make sure the baud rate matches the one set in the program.



2.Press the serial port connection switch.



4.4.3.7. Principle Description

The DHT temperature and humidity sensor integrates two components internally: a capacitive humidity sensor and an NTC thermistor, which are used to detect relative humidity and temperature in the air, respectively. When the ambient humidity changes, the capacitance of the humidity sensor changes accordingly; when the temperature changes, the resistance of the thermistor changes.

These changes are digitally converted and calibrated by the microcontroller inside the module, and the data is packaged into a formatted signal and transmitted through a single digital pin to an external controller (such as the MATRIX Mini R4). The data includes temperature and humidity values and is updated once per second.

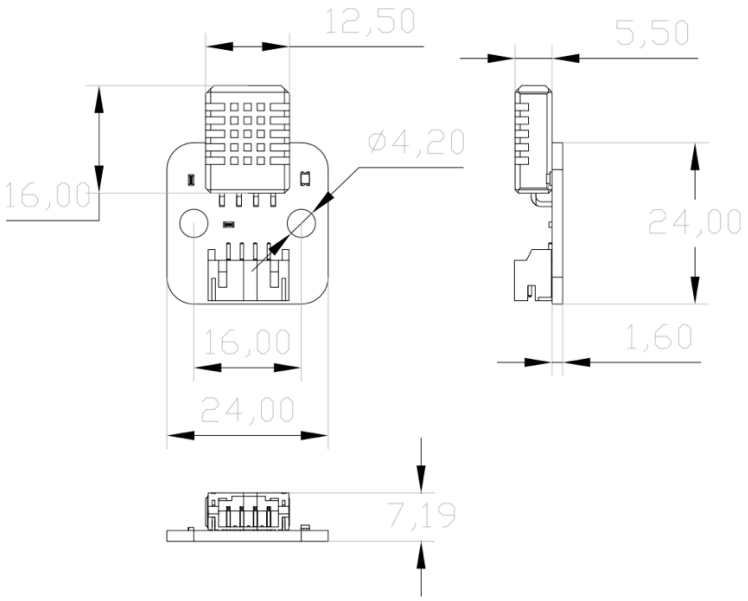
The DHT11 is an entry-level sensor. Although its accuracy is not as high as that of industrial-grade modules, it has a simple structure, high stability, and is easy to operate, making it very suitable for educational purposes, learning, and general environmental monitoring.

Usage notes：

- The sensor updates data once every second. If the program attempts to read data too frequently, it may receive duplicate or incorrect data.

- It is recommended to wait at least 1 second between each reading.
- To maintain stability and accuracy, avoid placing the sensor in high-temperature, high-humidity, or condensing environments for long periods.
- For long-term monitoring, consider using data filtering logic or an anomaly exclusion mechanism.

4.4.3.8. Dimensions

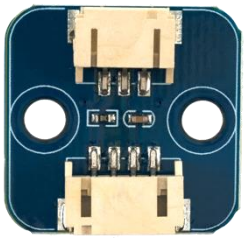


4.4.4. One- Wire Temperature Sensor MS-015

4.4.4.1. Overview

The One-Wire Temperature Sensor is a sensing module that uses the 1-Wire digital communication protocol for data transmission. It integrates a DS18B20 temperature chip and features a waterproof casing and high temperature resistance. The sensor offers temperature readings with 9- to 12-bit resolution, making it suitable for applications such as multi-point temperature monitoring and automated environmental control.

Connection side



Back side

4.4.4.2. Features

- Supports 3.3V to 5V operative voltage
- High-precision temperature measurement (resolution configurable from 9 to 12 bits)
- Low power consumption, ideal for long-term monitoring
- Waterproof probe design, suitable for use in humid or enclosed environments
- Communication via 1-wire interface – simple wiring (just one digital line + GND)

Sensing Area

Chip Pin Connection

4.4.4.3. Application

- Home automation systems (heating and cooling control)
- Temperature monitoring for liquids or pipelines
- Temperature sensing in smart agriculture
- Environmental monitoring in refrigerators, greenhouses, or aquariums

4.4.4.4. Electrical Characteristics

Parameter	Min	Typ*	Max	Units
Operating Voltage (VCC)	3.3	-	5	Volts
Temperature Range	-55	-	125	°C
Temperature Accuracy	-	±0.5	-	°C
Data Output Resolution *	9	-	12	Bits
Operating Current	-	1.5	3.0	mA

Term explanation：

Typ* refers to “Typical,” indicating commonly observed performance under standard conditions (not minimum or maximum limits).

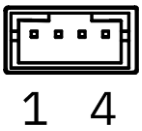
Data Output Resolution* is configurable via command from 9 to 12 bits. Higher resolution results in longer response time (up to approx. 750 ms at 12-bit).

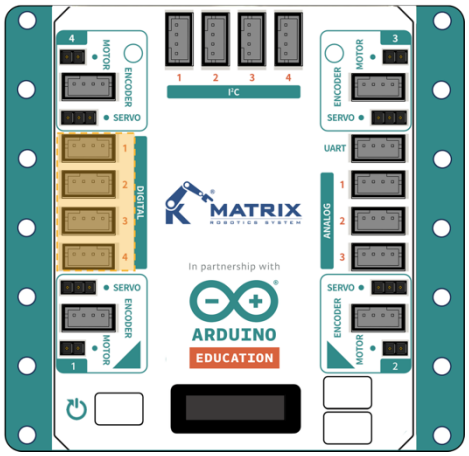
4.4.4.5. Pinout

Pinout Definition		
NO.	NO.	NO.
1	DOUT (Digital Output)	Digital output pin for temperature data, transmitted using the 1-Wire protocol.
2	NC (No connection)	No function; not connected to internal circuitry.
3	5V (Power)	Supply voltage pin that provides operating power to the sensor.
4	GND (Ground)	Power ground pin, used to complete the current loop; usually connected to the negative terminal of the power source.

4.4.4.6. Sample Code & Blocks Instructions

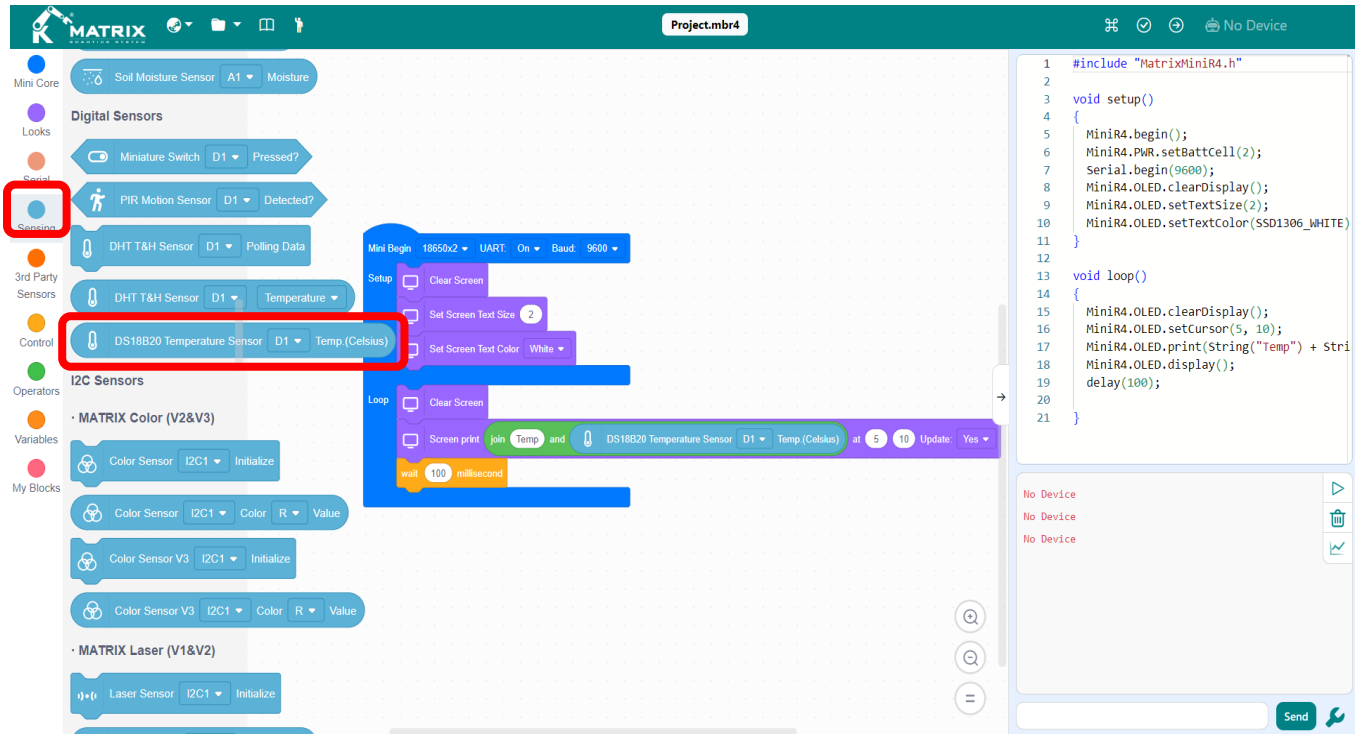
- **Hardware Connection**：Please connect the One-Wire Temperature Sensor to the **Digital 1** port of the MATRIX Mini R4 controller.



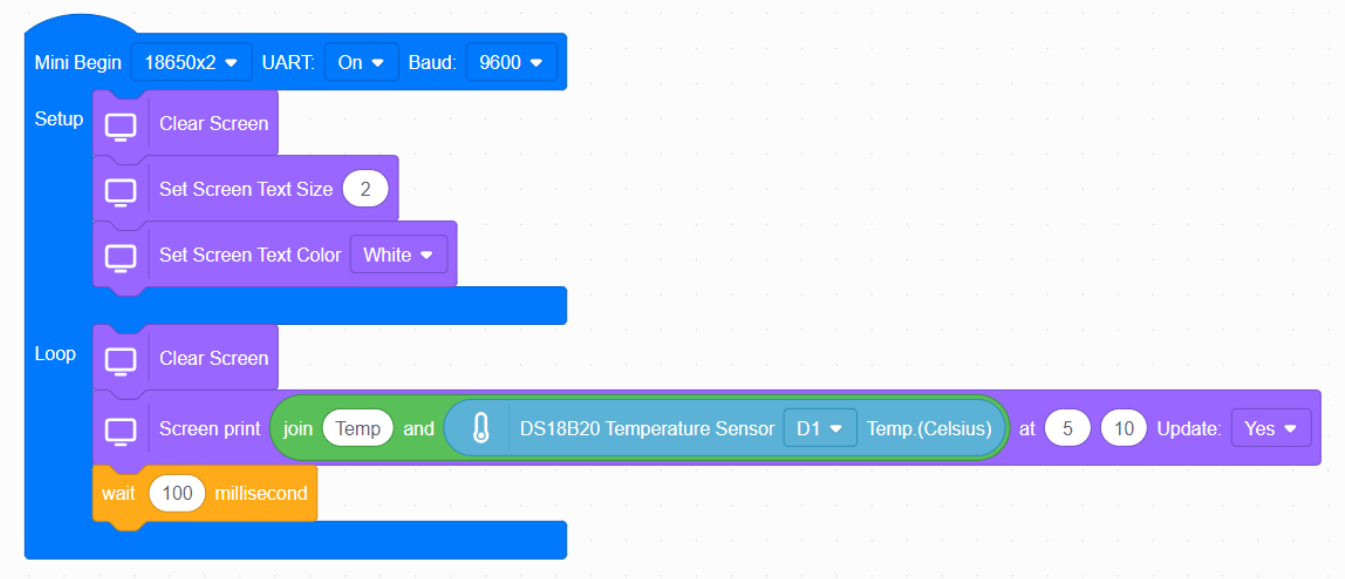


Pinout-Digital I/O			
No.	Name	I/O	Description
1	DIO A	Input/Output	Digital I/O pin A (GPIO A), can be used for detecting or outputting signals
2	DIO B	Input/Output	Digital I/O pin B (GPIO B), can be used for detecting or outputting signals
3	VCC	Output	Power supply pin
4	GND	-	Power ground

- **Sample Code**：



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
```

```
    MiniR4.PWR.setBattCell(2);
    Serial.begin(9600);
    MiniR4.OLED.clearDisplay();
    MiniR4.OLED.setTextSize(2);
    MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
    MiniR4.OLED.clearDisplay();
    MiniR4.OLED.setCursor(5, 10);
    MiniR4.OLED.print(String("Temp") +
String(MiniR4.D1.MXOnewireDT.requestAndGetTemp()));
    MiniR4.OLED.display();
    delay(100);

}
```

Result Description

The temperature data currently detected by the temperature sensor is displayed on the OLED screen and updates every 0.1 seconds.

4.4.4.7. Principle Description

The sensor uses the DS18B20 digital temperature chip, which measures thermal changes through a resistive sensing element and converts the results into digital signals. Data is transmitted via a **1-Wire protocol** using a single digital pin. The sensor features high precision, low power consumption, and good stability.

It supports adjustable resolution from 9 to 12 bits, with a default of **12 bits** (accuracy of approximately **±0.5°C**). Advanced users can modify the resolution using 1-Wire commands (writing to the Scratchpad configuration register) to adjust measurement speed and accuracy:

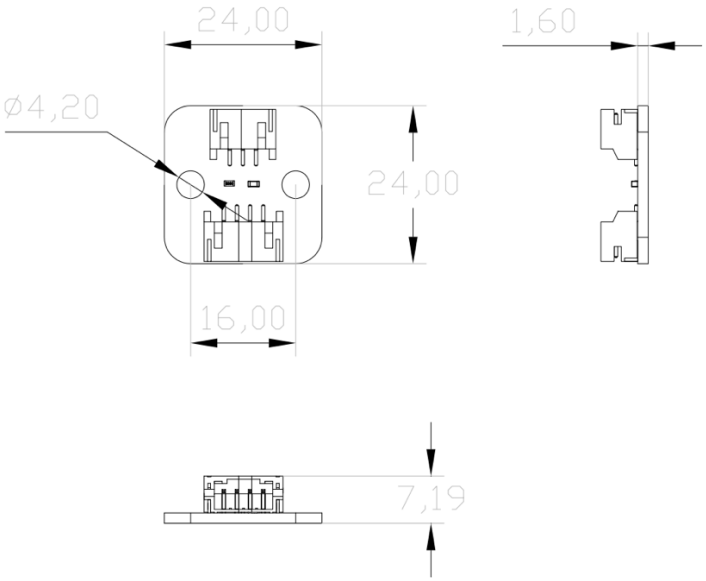
Resolution	Conversion time	Config Byte

9 bit	~94 ms	0x1F
10 bit	~188 ms	0x3F
11 bit	~375 ms	0x5F
12 bit	~750 ms	0x7F（default）

⚠ **Note：**
In the MATRIXblock and MiniR4 libraries (e.g., MXOnewireDT.requestAndGetTemp()), the module resolution is fixed at 12 bits, so users can directly obtain stable temperature data without manual configuration.

Due to its waterproof metal probe casing and simple communication wiring, the sensor is especially suitable for humid, hot, or hard-to-reach environments.

4.4.4.8. Dimensions



4.5. External I²C Inputs

4.5.1. Color Sensor V2 MS-002 V2

4.5.1.1. Overview

The MATRIX Color Sensor (Color Sensor V2) is an RGB sensing module that supports I²C communication. It features a built-in fill light, color classification functionality, and multi-format output capabilities. It can return sensing data in RGB, CMYK, grayscale formats, and includes automatic light source adjustments and gamma correction. It can reliably detect and classify a variety of common colors, making it suitable for applications such a symbol recognition, object classification, and automation control.

Sensing side

Back side

4.5.1.2. Feature

- Supports 3.3V to 5V operating voltage
- Supports RGB, CMYK, and Grayscale output formats
- Built-in 14-color auto classification (Color Number) function
- Supports gamma correction, closely mimicking human vision
- Automatic or PWM dimmable fill light to adapt to different lighting conditions
- Uses I²C communication interface, supports 100~400kHz transmission rate
- RGB value range is 0-255, representing the intensity of each channel
- Recommended sensing distance is approximately 5-10 mm; sensing may become unstable or affected by ambient light if too far

Sensing Point

4.5.1.3. Application

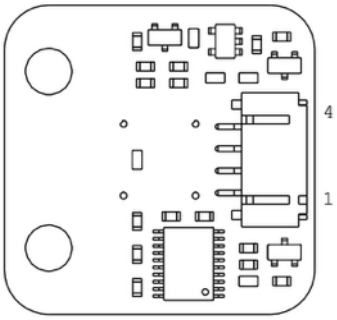
- Identification of signs or route colors
- Industrial or robotic automatic classification systems
- Grayscale monitoring and light intensity evaluation
- Color sensing and logic judgement training in educational experiments

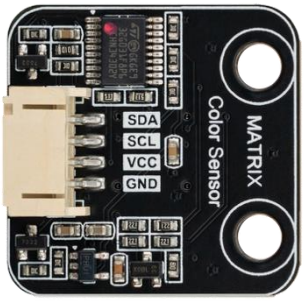
4.5.1.4. Electrical Characteristics (Module's Electronic Component Characteristics)

Parameter	Min Value	Typ Value	Max Value	Units
Operating Voltage (VCC)	3.0	3.3	5.0	Volts (V)
Red Channel Wavelength*	-	465	-	Nanometers (nm)
Green Channel Wavelength*	-	525	-	Nanometers (nm)
Blue Channel Wavelength*	-	615	-	Nanometers (nm)
I²C Transfer Speed*	100	-	400	kHz
I²C Input Low Voltage (VIL)*	-0.5	-	0.33 × VCC	Volts (V)
I²C Input High Voltage (VIH)*	0.7 × VCC	-	VCC	Volts (V)

Glossary of Terms:
Red / Green / Blue Channel Wavelength* Indicates the main wavelength positions of red, green, and blue light received by the sensor, measured in nanometers (nm).
I²C Transmission Speed* Indicates the data transfer rate, measured in kilohertz (kHz). A higher value means faster communication.
I²C Input Low Voltage (VIL)* If the voltage sent by the controller is below this value, the sensor will recognize it as 0 (LOW).
I²C Input High Voltage (VIH)* If the voltage sent by the controller is above this value, the sensor will recognize it as 1 (HIGH).

4.5.1.5. Pinout

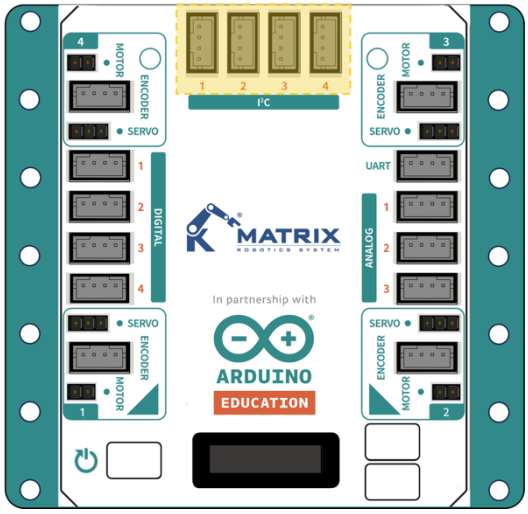




Pin Definitions		
NO.	Name	Description
1	SDA (Data Line)	I²C data line (Serial Data), used for data read/write
2	SCL (Clock Line)	I²C clock line (Serial Clock), controls data transmission timing
3	VCC (Power Supply)	Power input, supports 3.3V ~ 5V
4	GND (Ground)	Power ground, connected to controller ground

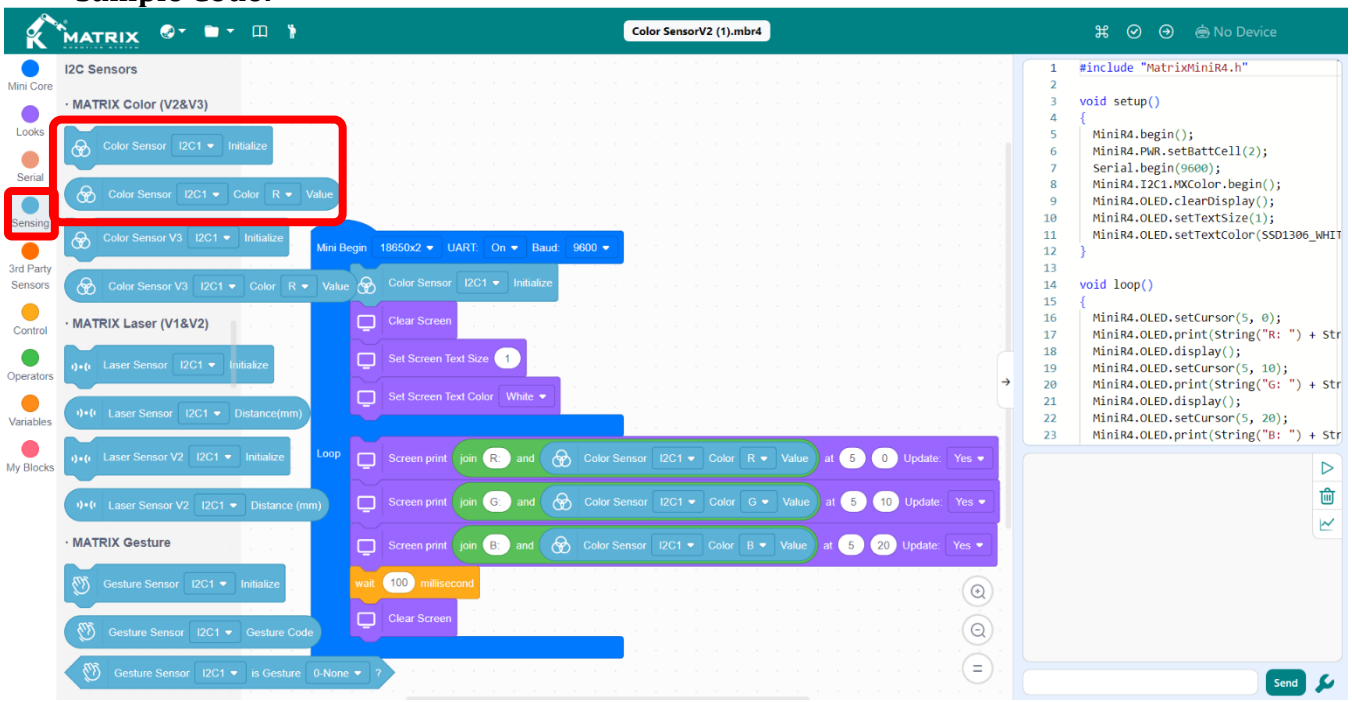
4.5.1.6. Sample Code & Blocks Instructions

- **Hardware Connection:** Please connect the Color Sensor V2 to the I2C 1 port of the MATRIX Mini R4 controller.

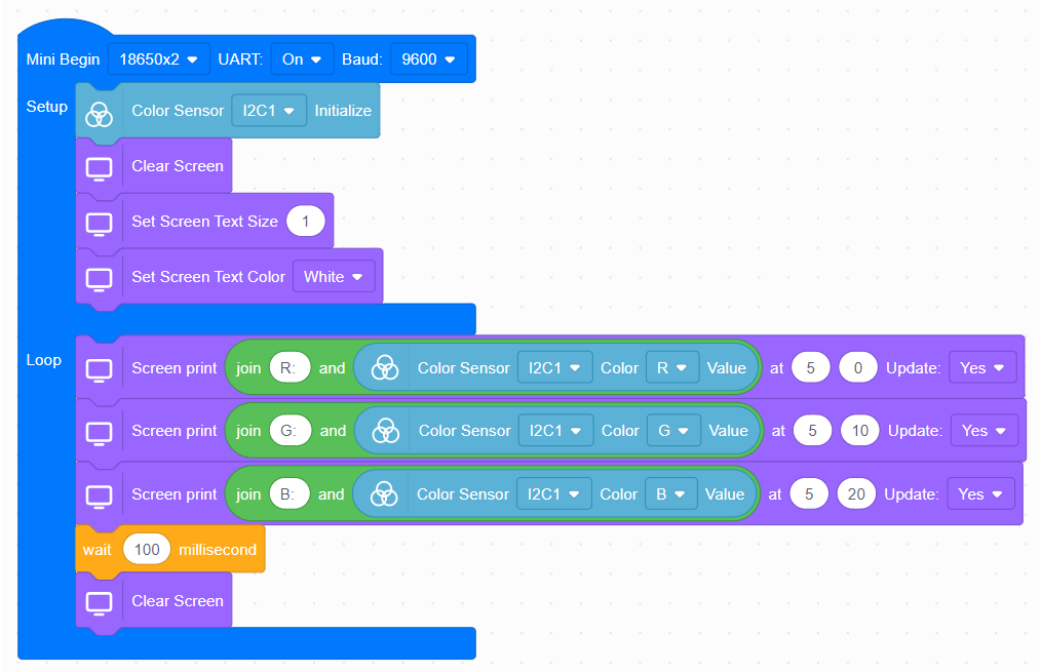


I²C Pinout - I²C			
No.	Name	I/O	Description
1	SDA	Input/Output	Data line (Serial Data Line)
2	SCL	Input	Clock line (Serial Clock Line)
3	VCC	Output	Power supply pin
4	GND	-	Power Ground

- **Sample Code:**



MATRIXblock



IDE C++

```cpp
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.I2C1.MXColor.begin();
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(1);
  MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
  MiniR4.OLED.setCursor(5, 0);
  MiniR4.OLED.print(String("R: ") + String(MiniR4.I2C1.MXColor.getColor(R)));
  MiniR4.OLED.display();
  MiniR4.OLED.setCursor(5, 10);
  MiniR4.OLED.print(String("G: ") + String(MiniR4.I2C1.MXColor.getColor(G)));
  MiniR4.OLED.display();
  MiniR4.OLED.setCursor(5, 20);
  MiniR4.OLED.print(String("B: ") + String(MiniR4.I2C1.MXColor.getColor(B)));
  MiniR4.OLED.display();
  delay(100);
  MiniR4.OLED.clearDisplay();
```

```
27.
28. }
```

Result Explanation:

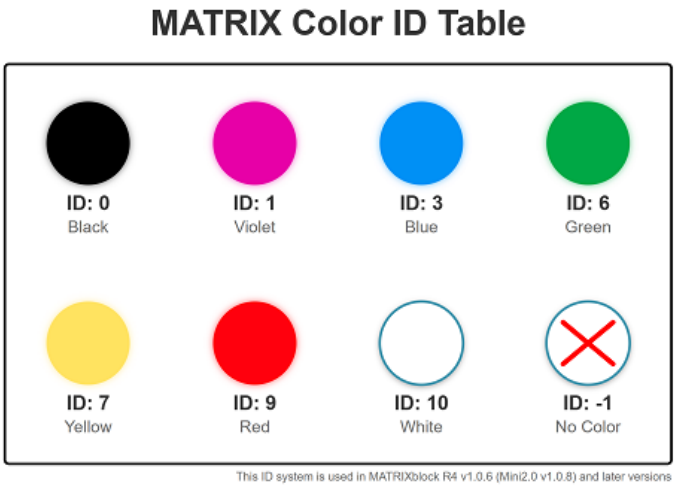
The RGB values from the color sensor are automatically read every 0.1 seconds and displayed in real time on the OLED screen. Each screen update shows the values of the Red (R), Green (G), and Blue (B) channels simultaneously. Each color channel's value ranges from 0 to 255, and the values continuously change according to the color of the object in front of the sensor.

4.5.1.7. Principle Description

The Color Sensor V2 utilizes internal light-sensing elements to measure the intensity of Red, Green, and Blue light, transmitting the RGB values via I^2C communication. The module's built-in algorithm can convert RGB data into CMYK color values and grayscale levels, and it supports Gamma correction to ensure the output colors closely align with actual human visual perception.

The sensor also features integrated LED fill lights, which can be set to "Auto-dimming mode" or manually adjusted via PWM. This ensures stable color data reading even in low-light environments.

Furthermore, the module supports an internal color classification mechanism (Color ID). This feature automatically categorizes colors into 8 preset colors (such as Red, Yellow, Green, Blue, etc.), allowing beginners to skip complex RGB calculations and quickly deploy the sensor in their projects.



*The sensor follows the standard I^2C communication protocol, where the register address to be read is first written, followed by sending a read command to retrieve data. When using the MATRIX library and blocks, these processes are handled automatically, requiring no manual setup.

Advanced Usage

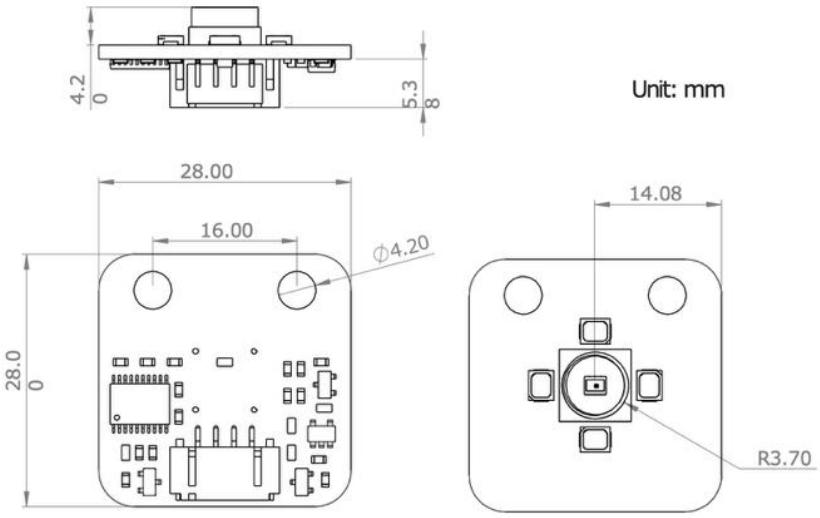
For users who need further control over registers, adjustment of fill light modes, or understanding of data format definitions, please refer to the official technical manual below:

MS-002 V2 Datasheet（PDF）

- Contains complete I^2C register map, Fill Light PWM settings, Gamma control and color classification logic
- Suitable as a reference for developing custom features or debugging communication

※ Beginner users do not need to configure these parameters to start using this sensor.

4.5.1.8. Dimensions



4.5.2. Color Sensor V3 MS-002 V3

4.5.2.1. Overview

The Color Sensor V3 uses the TCS34725 chip, which can simultaneously detect red, green, blue light, and ambient light (Clear light). It is equipped with an infared blocking filter to improve color detection accuracy. The built-in fill light supports low-light detection and features programmable analog gain and integration time, allowing users to adjust sensing behavior accodording to application needs. It is suitable for advanced color detection and color temperature analysis applications.

Sensing side

Back side

4.5.2.2. Feature

- Four-channel sensing (RGB + Clear) with an integrated infrared (IR) blocking filter.
- Built-in automatic fill light with adjustable Analog Gain and Integration Time.
- Uses I^2C communication interface, supports 3.3V ~ 5V voltage systems
- RGB value range is 0-255, representing the intensity of each channel
- Recommended sensing distance is approximately 5-40 mm; sensing may become unstable or affected by ambient light if too far

4.5.2.3. Application

- Identification of sign or object colors
- Industrial or automated color recognition processes
- Light environment or color temperature variation sensing (suitable for smart lighting control)

4.5.2.4. Electrical Characteristics

Parameter	Min	Typ	Max	Units
Operating Voltage (VCC)	3.3	-	5.0	Volts (V)
Red Channel Wavelength*	-	465	-	Nanometers (nm)
Green Channel Wavelength*	-	525	-	Nanometers (nm)
Blue Channel Wavelength*	-	615	-	Nanometers (nm)
Clear Channel Wavelength*	0	0 ~ 2048 (MATRIXblock)	65535	Unitless
I^2C Transmission Speed**	-	-	400	Kilohertz (kHz)
I^2C Input Low Voltage VIL**	-	-	0.4 × VCC	Volts (V)
I^2C Input High Voltage VIH**	0.7 × VCC	-	-	Volts (V)

Glossary of Terms

Red / Green / Blue Channel Wavelength* Represents the primary wavelength positions of red, green, and blue light received by the sensor, measured in nanometers (nm).

Clear Channel* Represents the total intensity of unfiltered ambient light. The value range depends on sampling settings, with a maximum of 65,535. Under default settings in MATRIXblock, the typical range is approximately 0 to 2048. To adjust sampling time and gain, advanced settings via IDE programming are required.

I^2C Transmission Speed* Represents the data transfer rate, measured in kilohertz (kHz). A high

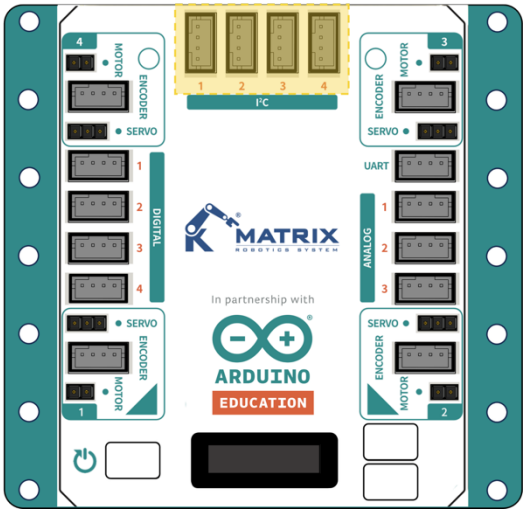
I^2C Specification Note** The specifications listed here are based on the TCS34725 chip. Actual implementation may vary slightly depending on module design.

4.5.2.5. Pinout

Pinout Definition		
NO.	Name	Description
1	SDA (Data)	I^2C Serial Data Line, used for sensor data transmission
2	SCL (Clock)	I^2C Serial Clock Line, controls the timing of data transmission
3	VCC (Power)	Power input, supports 3.3V ~ 5V
4	GND (Ground)	Power ground, connected to the controller's ground

4.5.2.6. Sample Code & Blocks Instructions

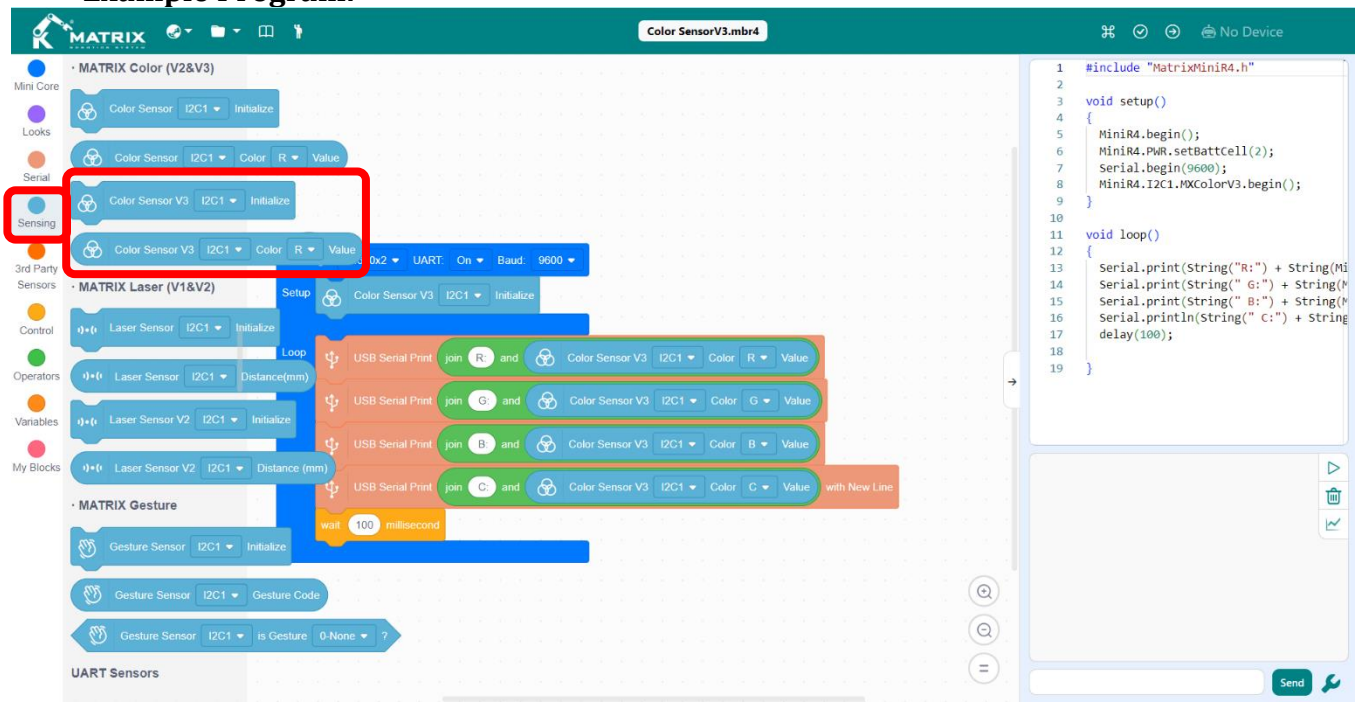
- **Hardware Connection:** Please connect the **Color Sensor V3** to the **I^2C 1 Port** on the **MATRIX Mini R4** controller.



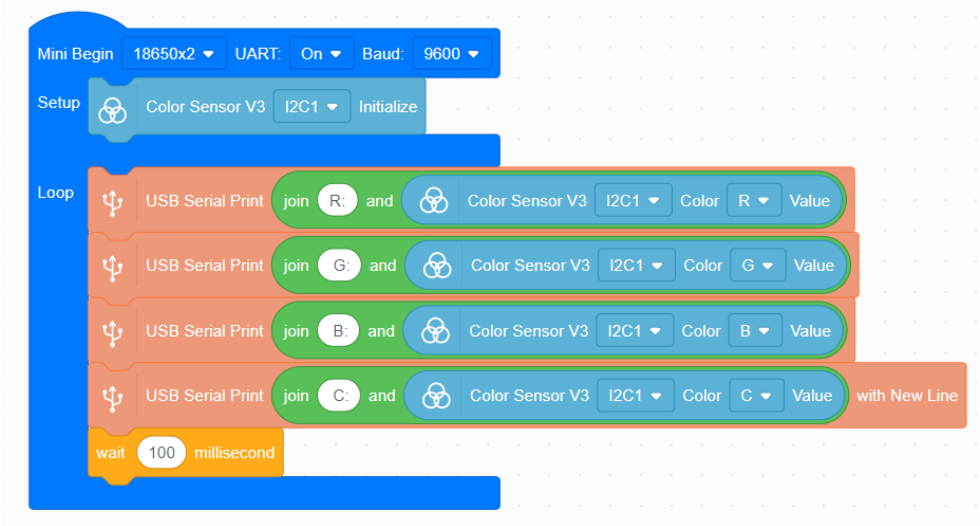
1 4

Pinout - I²C			
No.	Name	I/O	Description
1	SDA	Input/Output	Serial Data Line
2	SCL	Input	Serial Clock Line
3	VCC	Output	Power Supply Pin
4	GND	-	Power Ground

- **Example Program:**



MATRIXblock



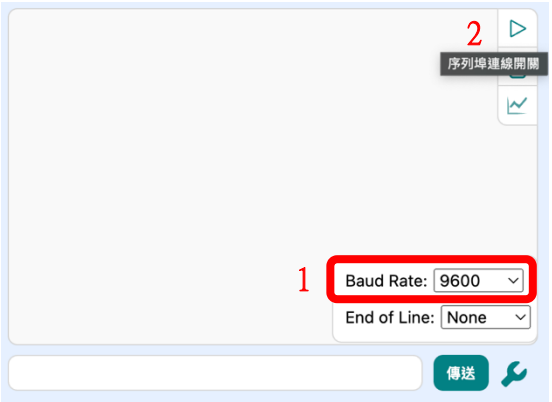
IDE C++

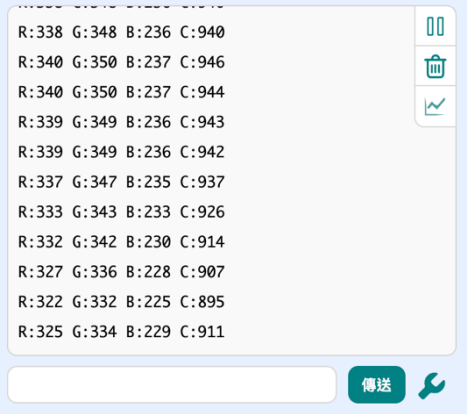
```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.I2C1.MXColorV3.begin();
}

void loop()
{
  Serial.print(String("R:") + String(MiniR4.I2C1.MXColorV3.getR()));
  Serial.print(String(" G:") + String(MiniR4.I2C1.MXColorV3.getG()));
  Serial.print(String(" B:") + String(MiniR4.I2C1.MXColorV3.getB()));
  Serial.println(String(" C:") + String(MiniR4.I2C1.MXColorV3.getC()));
  delay(100);

}
```

Result Description:

The program reads values from the color sensor every **0.1 seconds** for the four channels: **Red (R), Green (G), Blue (B), and Clear (brightness)**, and outputs them through the serial port. For example: `R:128 G:112 B:240 C:1820`. **RGB values** range from **0 to 255**, representing the light intensity of each channel. The **Clear value** represents overall ambient brightness. The range varies depending on the sampling settings, and under the default MATRIXblock configuration, it is approximately **0 to 2048**.

1. Make sure the **baud rate** matches the one set in your program.
2. Press the **serial port connect button**.



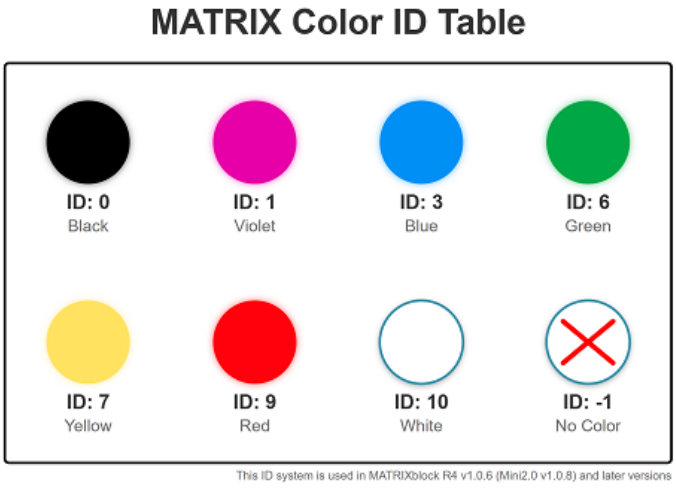


4.5.2.7. Principle Description

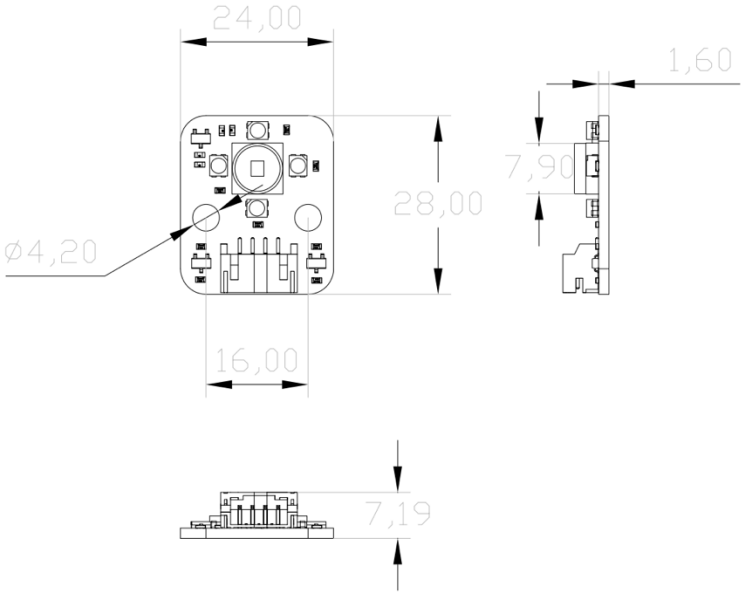
The Color Sensor V3 is equipped with a TCS34725 sensing chip, capable of measuring red, green, and blue visible light channels, as well as ambient brightness (Clear Light). It also includes a built-in infrared blocking filter to enhance color detection accuracy. Through its internal ADC and programmable settings, users can adjust analog gain and integration time to meet application needs under different lighting conditions.

The module includes a built-in fill light that supports automatic or manual control, allowing it to stably detect color changes under low light sources. In addition, the Clear channel captures the total amount of ambient brightness without filtering and is commonly used to compensate for varying light intensities, estimate color temperature, or serve as a reference for auto exposure and color correction.

Furthermore, the module supports an internal color classification mechanism (Color ID), which automatically categorizes detected colors into 8 preset colors (such as Red, Yellow, Green, Blue, etc.). This allows beginners to bypass complex RGB calculations and quickly deploy the sensor in their projects.



4.5.2.8. Dimensions



4.5.3. Laser Sensor MS-009

4.5.3.1. Overview

Sensing Side

Back Side

The Laser Sensor is an I²C sensing module with absolute distance measurement capability. It uses laser ranging principles to measure distances between 21 mm and 1999 mm with 1 mm accuracy and returns data at a fixed frequency of 50Hz. The module features a timeout detection mechanism that actively issues alerts when data responses are absent, making it suitable for positioning applications requiring high stability and real-time performance.

4.5.3.2. Feature

Sensing Point

- Supports absolute distance measurement, range 21-1999 mm, accuracy up to 1 mm.
- Fixed sampling frequency of 50Hz (returns data up to 50 times per second)
- Distance data returned via 16-bit values (high and low bytes)
- Equipped with timeout detection function to identify detection failures or abnormal conditions
- Uses I²C communication interface, supporting 3.3V to 5V voltage systems

4.5.3.3. Application

- Robot positioning and obstacle avoidance judgment
- Camera focusing or depth of field applications
- Level measurement and posture stabilization devices
- High-precision distance sensing needs (e.g., autonomous navigation or automatic doors)

4.5.3.4. Electrical Characteristics

Parameter	Min	Typ	Max	Units
Operating Voltage (VCC)	3.0	3.3	5.0	Volt (V)
Detection Distance Range	21	-	1999	Millimeter (mm)
Sampling Frequency*	-	50	-	Hertz (Hz)
I²C Transmission Speed*	100	-	400	Kilohertz (kHz)
I²C Input Low Voltage (VIL)*	-0.5	-	0.33 × VCC	Volt (V)
I²C Input High Voltage (VIH)*	0.7 × VCC	-	VCC	Volt (V)

Glossary:

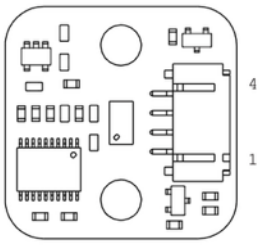
Sampling Frequency* The number of times data is returned per second. Fixed at 50Hz and cannot be changed.

I²C Transmission Speed* Indicates the data transmission rate, measured in kilohertz (kHz). A higher value means faster communication.

I²C Input Low Voltage (VIL)* If the controller sends a voltage lower than this value, the sensor reads it as 0 (LOW).

I²C Input High Voltage (VIH) * If the controller sends a voltage higher than this value, the sensor reads it as 1 (HIGH).

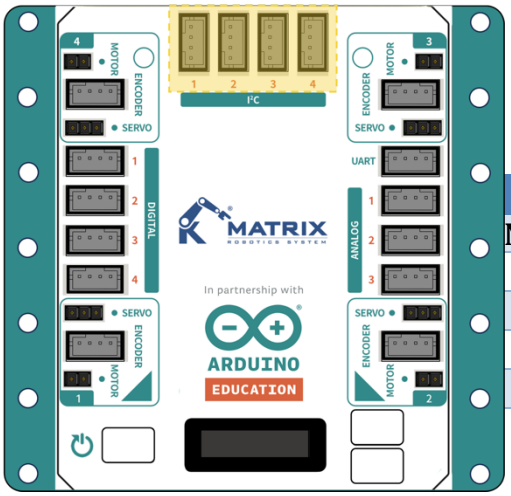
4.5.3.5. Pinout

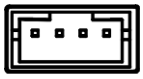


Pinout		
NO.	Name	Description
1	SDA (Data Line)	I²C data line (Serial Data), used for transmitting sensor data
2	SCL (Clock Line)	I²C clock line (Serial Clock), controls the timing of data transmission
3	VCC (Power Supply)	Power input, supports 3.3V ~ 5V
4	GND (Ground)	Power ground, connected to the controller's ground line

4.5.3.6. Sample Code & Blocks Instructions

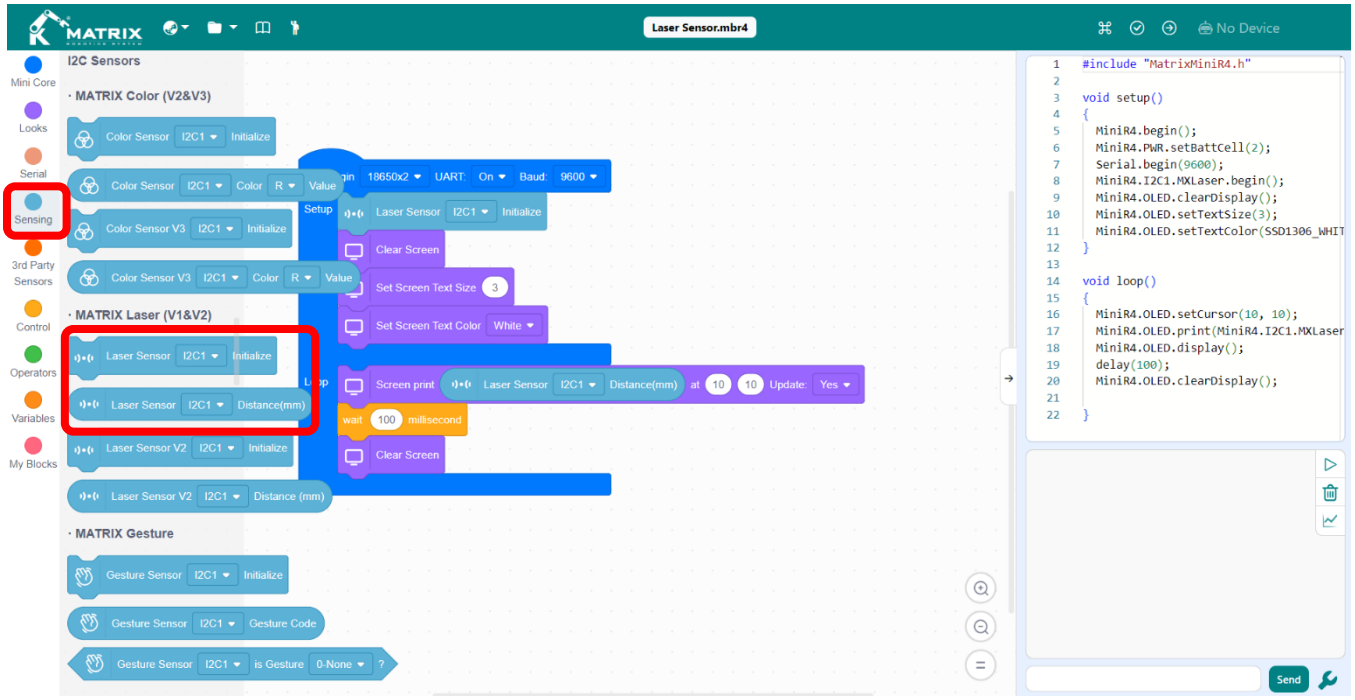
- **Hardware Connection:** Please connect the laser sensor to the **I²C 1** port of the MATRIX Mini R4 controller.



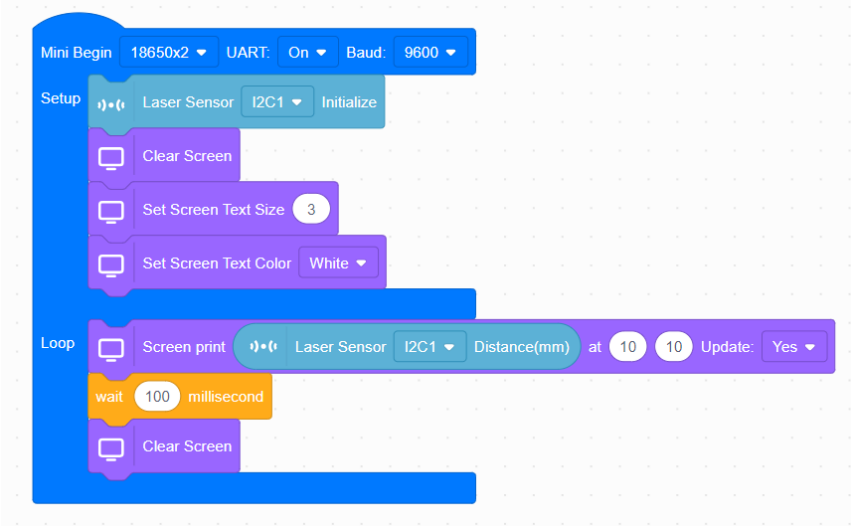


I²C Pinout - I²C			
No.	Name	I/O	Description
1	SDA	Input/Output	Serial Data Line
2	SCL	Input	Serial Clock Line
3	VCC	Output	Power supply pin
4	GND	-	Power ground

- **Sample Code:**



MATRIXblock



4.5.3.7.
IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
```

```
    MiniR4.I2C1.MXLaser.begin();
    MiniR4.OLED.clearDisplay();
    MiniR4.OLED.setTextSize(3);
    MiniR4.OLED.setTextColor(SSD1306_WHITE);
}

void loop()
{
    MiniR4.OLED.setCursor(10, 10);
    MiniR4.OLED.print(MiniR4.I2C1.MXLaser.getDistance());
    MiniR4.OLED.display();
    delay(100);
    MiniR4.OLED.clearDisplay();

}
```

Result Description:

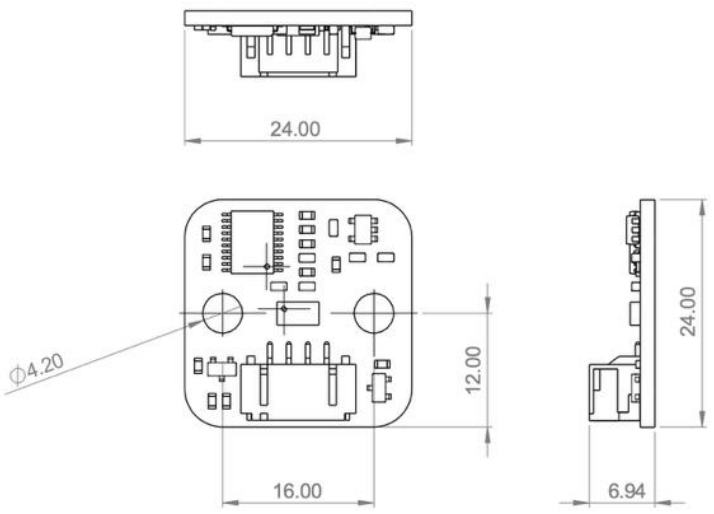
The program reads the distance value of the laser every 0.1 seconds. It displays the distance value on the OLED screen in millimeters (mm). The screen will show a number, such as 856, which means the object is 856 mm away. If no object is detected, the sensor returns 8191. This value means the sensor is in a timeout state. Before each update, the screen is cleared to display only the latest reading for easy real-time monitoring.

4.5.3.8. Principle Description

This module uses laser ranging principles. It calculates distance by measuring the time difference between emitting and receiving reflected laser light and returns the result as a 16-bit value. The sensor can stably measure distances in the range of 21-1999 mm, with a resolution of 1 mm and a fixed update frequency of 50 Hz.

This module also includes a timeout mechanism. If no valid measurement data is obtained within a specific time, it returns an error code (8191), allowing users to identify sensing anomalies. Communication with the controller is handled through I^2C. The module is fully integrated into the MATRIXblock and its associated library, allowing users to directly retrieve distance information through function calls.

4.5.3.9. 尺寸 Dimensions



Unit: mm

4.5.4. Laser Sensor V2 MS-009 V2

4.5.4.1. Overview

The Laser Sensor V2 adopts Time-of-Flight (ToF) ranging technology, featuring high precision, compact size, and excellent stability. It can detect distances from 50mm to1200mm with a resolution of 1mm. The module transmits data via the I^2C interface, making it suitable for real-time distance detection and interactive control applications.

Sensing Side

Back Side

4.5.4.2. Feature

- Using VL53L0X ToF sensor chip, with a ranging range of 50-1200mm
- Supports 1mm resolution with high accuracy and fast response
- Built-in 940nm infrared laser emitter, certified as a Class 1 laser device for safety
- Compact module design, ideal for embedded or space-constrained scenarios
- I^2C communication interface, supports system voltages from 3.3V to 5V.

Sensing Point

4.5.4.3. Application

- Simple distance measuring tools
- Robot positioning and obstacle avoidance
- Proximity-activated interactive devices (e.g., automatic doors, gesture controls)
- Autonomous navigation systems requiring high precision at short distances
- Applications triggered by distance changes (e.g., motion detection, automatic switches)

4.5.4.4. Electrical Characteristics

Parameters	Min	Typ	Max	Units
Operating Voltage (VCC)	3.3	-	5.0	Volts (V)
Detection Range	50	-	1200	Millimeters (mm)
Laser Wavelength	-	940	-	Nanometers (nm)
Resolution*	-	1	-	Millimeters (mm)
Sampling Frequency*	-	30	-	Hertz (Hz)

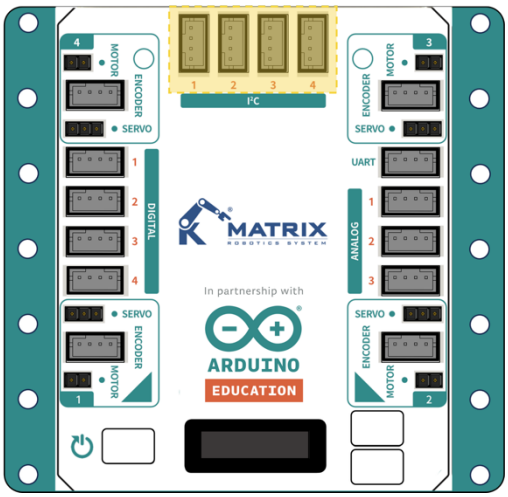
Glossary:
Resolution* The smallest change in distance that the sensor can detect.
Sampling frequency* The number of times data is returned per second. Fixed at 30Hz and cannot be changed.

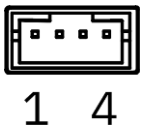
4.5.4.5. Pinout

Pinout		
NO.	Name	Description
1	SDA (Data Line)	I²C data line (Serial Data), used for transmitting sensor data.
2	SCL (Clock Line)	I²C clock line *Serial Clock), controls the timing of the data transmission
3	VCC	Power input, supports 3.3V ~ 5V
4	GND (Ground)	Power ground, connected to the controller's ground line

4.5.4.6. Sample Code & Blocks Instructions

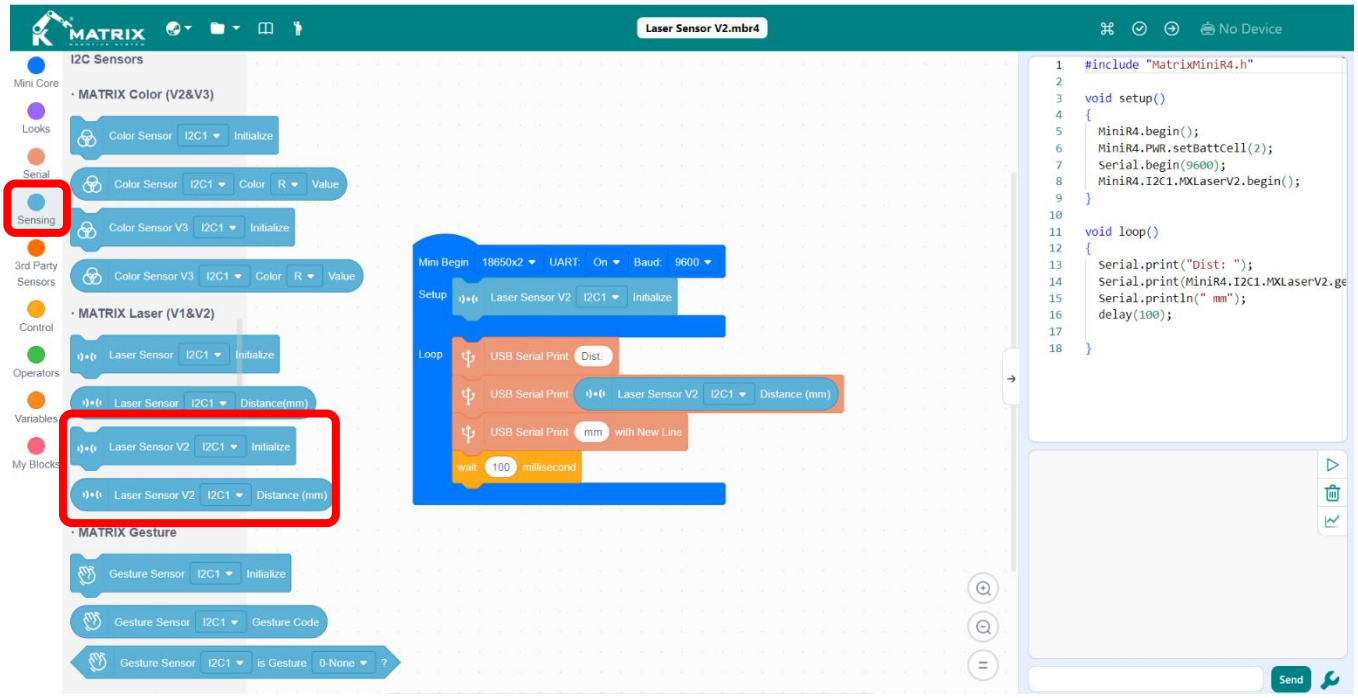
Hardware Connection: Connect the **Laser Sensor V2** to the **I²C 1** port of the MATRIX Mini R4 controller.



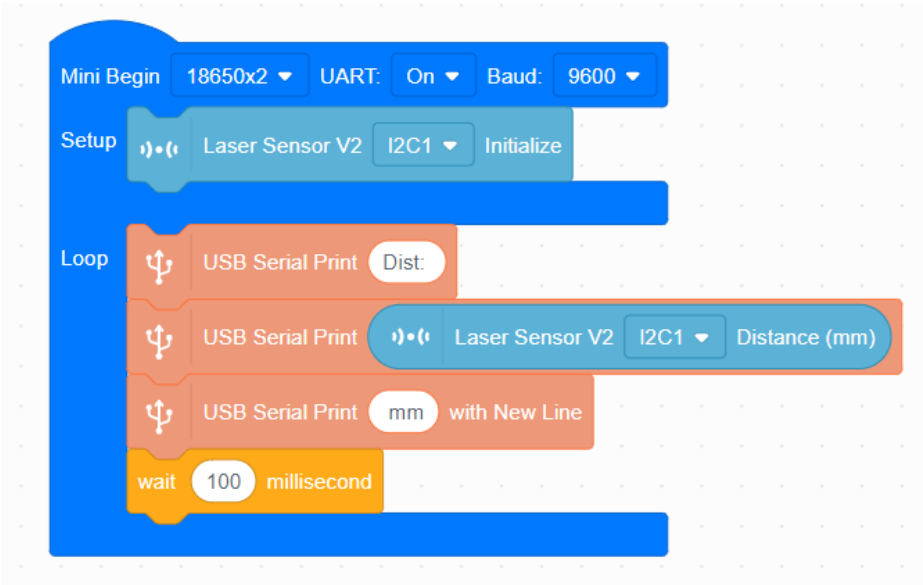


I²C Pinout - I²C			
No.	Name	I/O	Description
1	SDA	Input/Output	Serial Data Line
2	SCL	Input	Serial Clock Line
3	VCC	Output	Power Supply Pin
4	GND	-	Ground

- **Sample Code:**



MATRIXblock



IDE C++

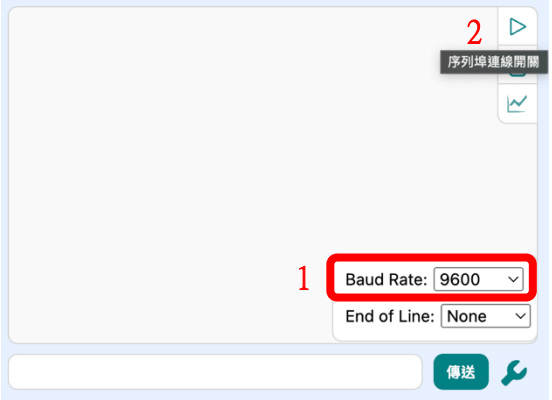
```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
```

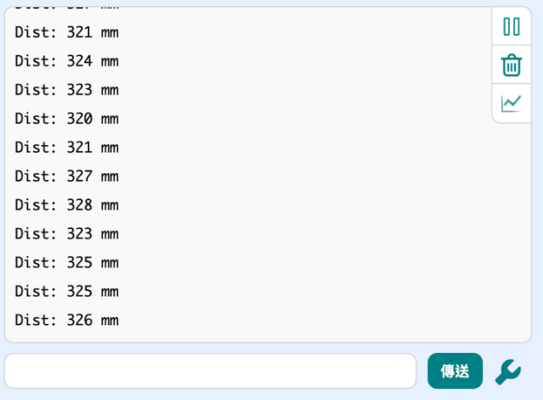
```
8.      MiniR4.I2C1.MXLaserV2.begin();
9.  }
10.
11. void loop()
12. {
13.     Serial.print("Dist: ");
14.     Serial.print(MiniR4.I2C1.MXLaserV2.getDistance());
15.     Serial.println(" mm");
16.     delay(100);
17.
18. }
```

Result Description:

The program reads distance data every 0.1 seconds and outputs it to the serial port in the format "Dist: xxx mm". The displayed value is in millimeters (mm), ranging from 50 to 1200 mm, representing the distance between the sensor and the target object. If the target is too close or not properly detected, it may return 8190 or other invalid values.

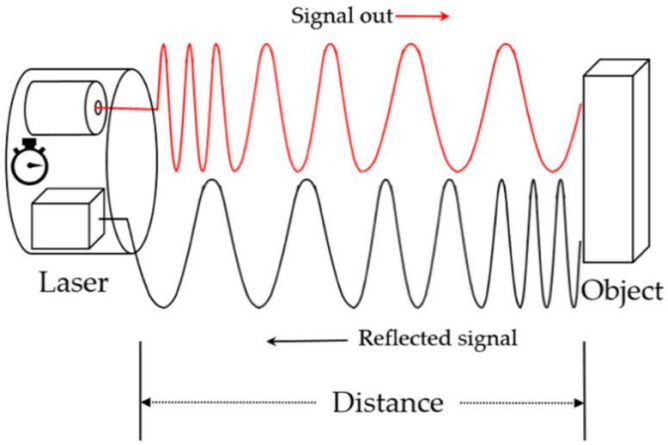
1. Confirm that the Baud Rate matches the one in the program.
2. Press the serial port connection switch





4.5.4.7. Principle Description

The Laser Sensor V2 uses the VL53L0X distance sensing chip, which is based on ToF (Time-of-Flight) technology. The sensor emits an infrared laser beam with a wavelength of 940nm. When the beam hits an object and reflects back, the module calculates the time it takes for the light to travel to and from the object and converts it into distance. This method is different from the traditional "reflective intensity" infrared sensors, and is less affected by the color or reflectivity of the object, providing more stable and reliable distance measurements.

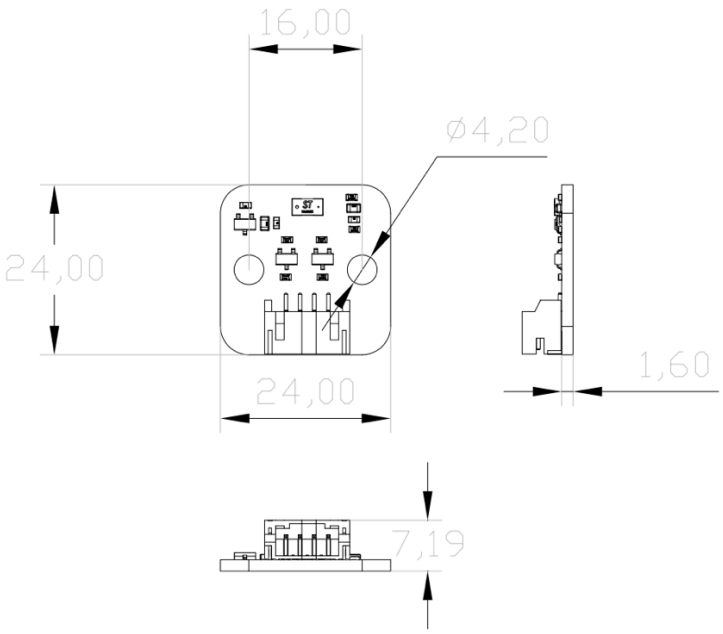


The module can measure distances ranging approximately from 50mm to 1200mm, with a resolution of 1mm, allowing it to detect even very small changes in distance. The default measurement update rate is 30Hz (up to 30 data readings per second), making it suitable for most interactive control or short-range distance measurement applications.

Data is transmitted to the controller (such as the MATRIX Mini R4) through the I^2C communication interface. Users can directly retrieve distance values through commands (unit: millimeters, mm). If the module is unable to detect a target at any given moment-such as when the object is too close or the reflected light is too weak, the system will return a specific "invalid value" of 8190, indicating that no valid measurement is currently available.

Additionally, the infrared laser used in this module falls under Class 1 laser devices in international safety standards. This means the laser intensity is extremely low and is safe for human eyes under normal usage conditions, requiring no additional protective equipment. It is suitable for educational, development, and everyday applications.

4.5.4.8. 尺寸 Dimensions


16,00
ø4,20
24,00
24,00
1,60
7,19

4.5.5. Gesture Sensor MS-017

4.5.5.1. Overview

Sensing Side

Back Side

The Gesture Sensor features 3D spatial gesture recognition capabilities and can return multiple preset gesture through I²C communication for contactless interaction. It has a sensing range of approximately 5-15 cm, making it suitable for gesture control and interactive operations in low-light environments. It has fast response and low power consumption, making it ideal for use in educational robots or smart devices.

4.5.5.2. Feature

- Supports nine basic gesture recognitions (for example waving up/down/left/right, clockwise/counterclockwise rotation, ect.)
- Contactless detection with a sensing range of approximately 5-15 cm
- Stable operation even in low-light environments
- Fast response speed for real-time gesture recognition
- Uses I²C communication interface, compatible with MATRIX controllers
- Supports 3.3V ~ 5V operating voltage, simple wiring, and easy integration

Sensing Point

4.5.5.3. Application

- Gesture-controlled interactive devices (for example page turning, volume control, menu navigation)
- Contactless interaction applications between educational robots and users
- Smart lighting or appliance switch control
- Contactless interfaces in public spaces

4.5.5.4. Electrical Characteristics

Parameter	Min	Typ	Max	Units
Operating Voltage (VCC)	3.3	-	5.0	Volts (V)
Operating Current	4.6	-	9.2	mA
Sensing Distance Range	5	-	15	Centimeters (cm)
I²C Transmission Speed**	-	-	400	kHz
I²C Input Low Voltage VIL**	-	-	0.4 × VCC	Volts (V)
I²C Input High Voltage VIH**	0.7 × VCC	-	-	Volts (V)

Glossary

I²C Transmission Speed* Refers to the data transmission rate in kilohertz (kHz). A higher value corresponds to faster communication speed.

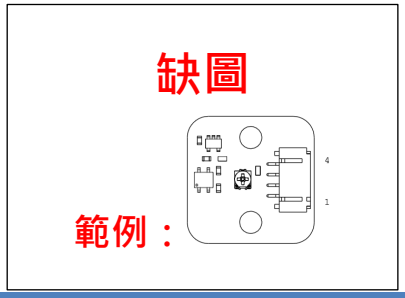
I²C Input Low Voltage VIL* If the controller sends a voltage lower than this value, the sensor will interpret it as 0 (LOW).

I²C Input High Voltage VIH* If the controller sends a voltage higher than this value, the sensor

will interpret it as 1 (HIGH).

I²C Specification Note** The above I²C electrical characteristics are estimated based on the standard design of the PAJ7620U2 chip.

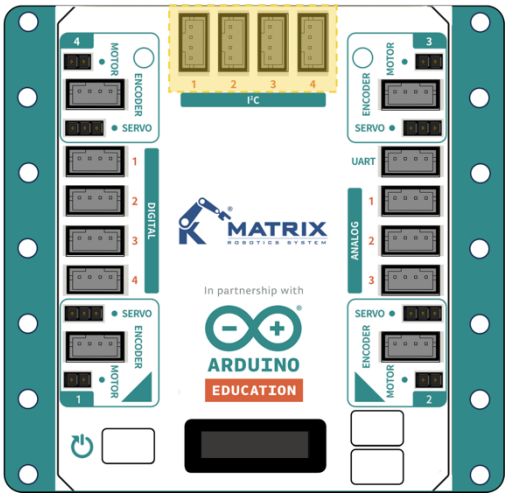
4.5.5.5. Pinout



Pin Definition		
NO.	Name	Description
1	SDA (Data Line)	I²C Data Line (Serial Data), used for transmitting sensing data
2	SCL (Clock Line)	I²C Clock Line (Serial Clock), controls the timing of data transmission
3	VCC (Power)	Power input, supports 3.3V ~ 5V
4	GND (Ground)	Power ground, connected to the controller's ground line

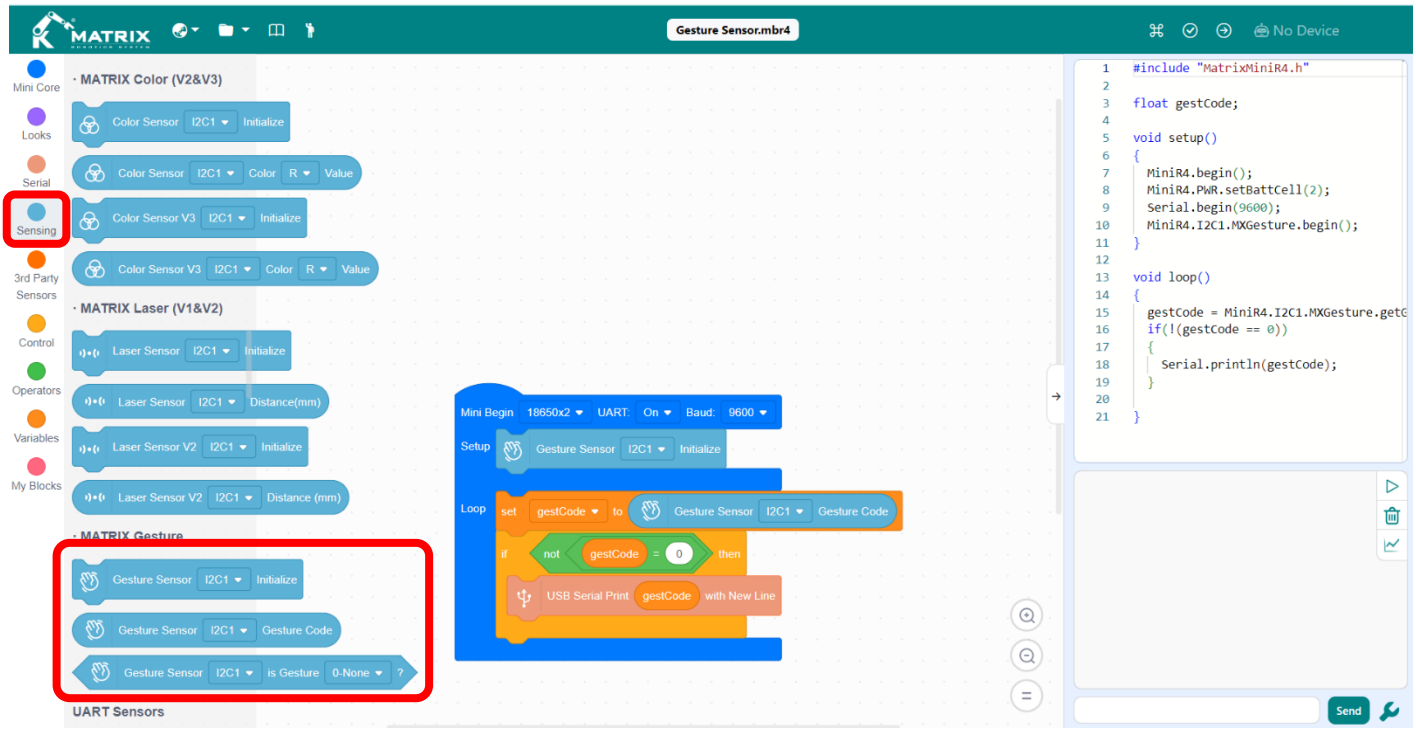
4.5.5.6. Sample Code & Blocks Instructions

- **Hardware Connection:** Please connect the Gesture Sensor to the **I²C 1** port of the MATRIX Mini R4 controller.

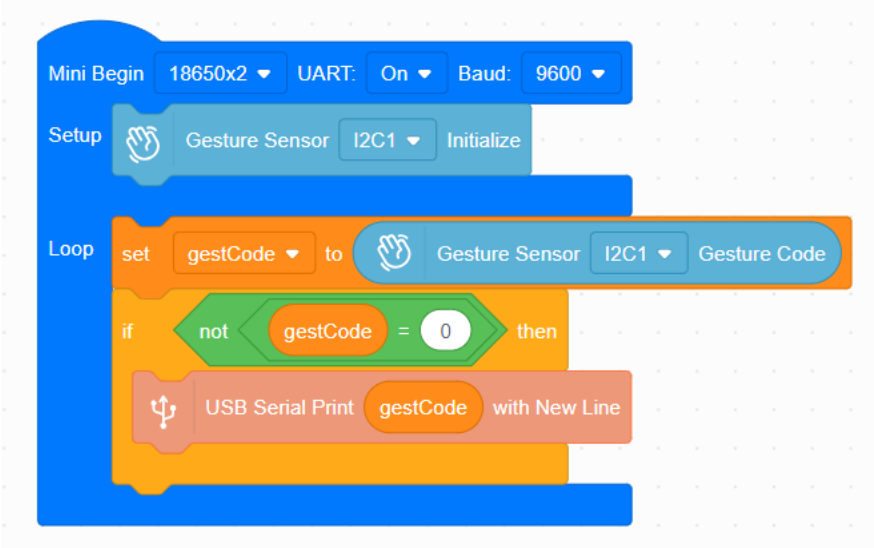


I²C Pinout - I²C			
No.	Name	I/O	I²C
1	SDA	Input/Output	Serial Data Line
2	SCL	Input	Serial Clock Line
3	VCC	Output	Power Supply Pin
4	GND	-	Ground

- **Sample Code:**



MATRIXblock



IDE C++

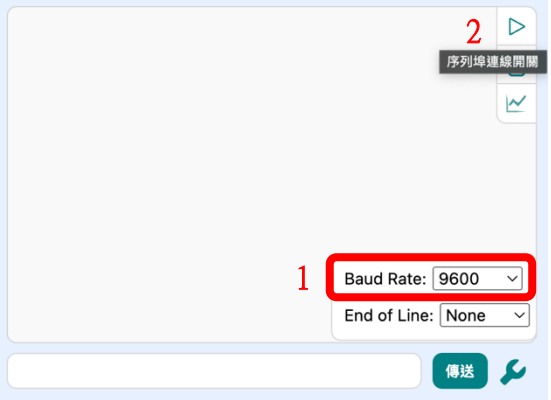
```
#include "MatrixMiniR4.h"

float gestCode;

void setup()
{
   MiniR4.begin();
```

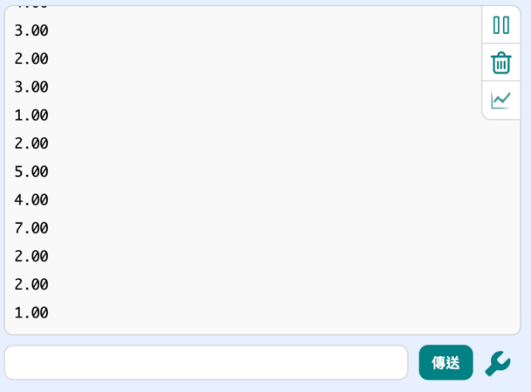
```
8.    MiniR4.PWR.setBattCell(2);
9.    Serial.begin(9600);
10.   MiniR4.I2C1.MXGesture.begin();
11. }
12.
13. void loop()
14. {
15.   gestCode = MiniR4.I2C1.MXGesture.getGesture();
16.   if(!(gestCode == 0))
17.   {
18.     Serial.println(gestCode);
19.   }
20.
21. }
```

Result Description:

The program continuously reads the gesture codes recognized by the sensor and only outputs result to the serial port when a value other than “0” (i.e, a valid gesture) is detected. For example, when a left-hand wave is detected, it will display 2; when a general wave is detected, it will display 9. For the gesture codes and their corresponding values, please refer to the Principal Description section.

1. Confirm that the Baud Rate matches the one set in the program.
2. Press the serial port connection switch.





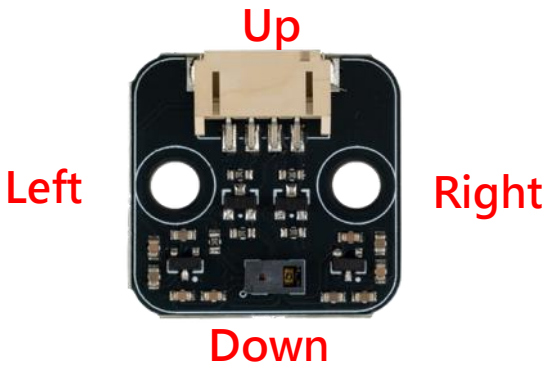
4.5.5.7. Principle Description

The Gesture Sensor is equipped with the PAJ7620U2 gesture recognition chip, which continuously detects hand movement trajectories in space through built-in infrared emitter and receiver components.

The module communicates with the main controller through the I^2C interface. When a gesture is detected, it returns a corresponding numeric code. These gestures include swiping up, down, left, and right, pushing forward, waving, or rotating clockwise/counterclockwise.

Code	Gesture
0	None
1	Right
2	Left
3	Up
4	Down
5	Forward
6	Backward
7	Clockwise
8	Counterclockwise
9	Wave

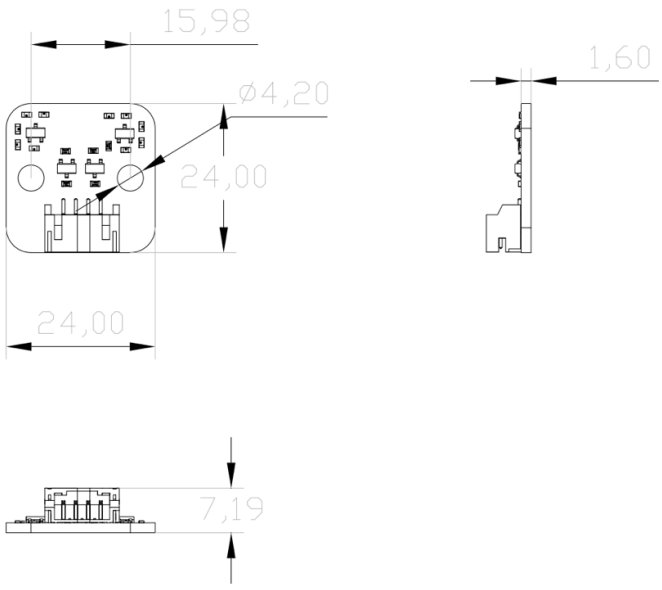
Relative Direction of the Sensor



⚠ **Notes: Regarding the timing and conditions of "gesture completion"**

- The sensor continuously tracks hand movements and uses a time window of approximately **0.4 - 0.6 seconds** to determine whether a valid gesture had occurred
- A valid gesture is recognized only if the user performs a continuous and directionally stable movement within this time window.
- If the movement is too short, too slow, paused for too long, or moves outside the sensing area, the sensor will return 0 (No gesture)
- If you need to perform the same gesture repeatedly, it's recommended to pause for **at least 300 milliseconds** between gestures to avoid them being merged into a single gesture
- For best sensing performance, keep your hand at a distance of **5 - 15 cm**, from the module during operation

4.5.5.8. Dimensions

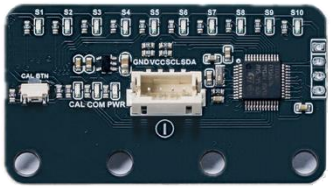

15,98
ø4,20
24,00
24,00
1,60
7,19

4.5.6. Line Tracer 10 CH MS-018

4.5.6.1. Overview

Sensing Side

Back Side



The Line Tracer 10 CH is a high-precision line-tracking sensor module equipped with 10 sets of sensing channels. It transmits ground detection data to the controller via I^2C communication. It can accurately determine the line's Error (offset from the center) and Line Width, and features junction recognition to automatically identify complex terrains such as Left turns, Right turns, T-junctions, or Cross-junctions. This module features low power consumption and operates stably even in low-light environments.

4.5.6.2. Feature

Sensing Area

- 10-Channel Sensor Array: Provides more detailed line-tracking capabilities compared to traditional sensors.
- Automatic Junction Identification: Built-in algorithms automatically determine Left, Right, T, or Cross-junction types.
- Line Width Calculation: Measures line width and records the last sensor position before a line was lost.
- Low Power & Low Light Optimization: Designed for energy efficiency and reliable detection in weak lighting.
- Intuitive Calibration: Features a built-in CAL button and blue indicator light for easy on-site calibration without complex programming.
- Wide Voltage Support: Supports 3.3V ~ 5V operating voltage for easy integration.

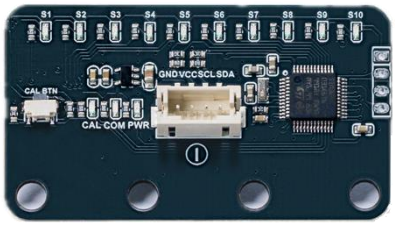
4.5.6.3. Application

- Precision Line-Following Robots: Utilizing high-resolution 10-channel offset data for smooth PID steering control.
- Smart Intersection Navigation: Identifying junction types via Junction Type data for precise turning in mazes.
- Competition Logistics Robots: Using the Last Sensor record to quickly recover paths if a line is momentarily lost.
- Ground Feature Analysis: Real-time monitoring of line width changes and ground optical reflection for data analysis.

4.5.6.4. Electrical Characteristics

Parameter	Min	Typ	Max	Unit
Supply Voltage (VCC)	3	3.3	5	V
Working Current	25	35	50	mA
Detection Range	4	-	12	mm
I2C operating speed	100	-	400	KHz

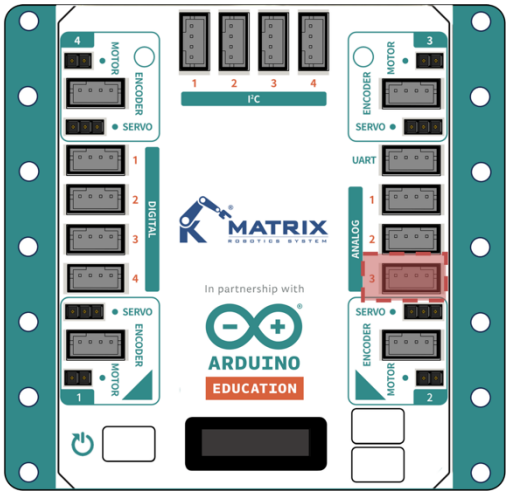
4.5.6.5. Pinout

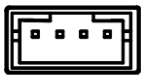


Pin Definition		
NO.	Name	Description
1	SDA (Data Line)	I²C Data Line (Serial Data), used for transmitting sensing data
2	SCL (Clock Line)	I²C Clock Line (Serial Clock), controls the timing of data transmission
3	VCC (Power)	Power input, supports 3.3V ~ 5V
4	GND (Ground)	Power ground, connected to the controller's ground line

4.5.6.6. Sample Code & Blocks Instructions

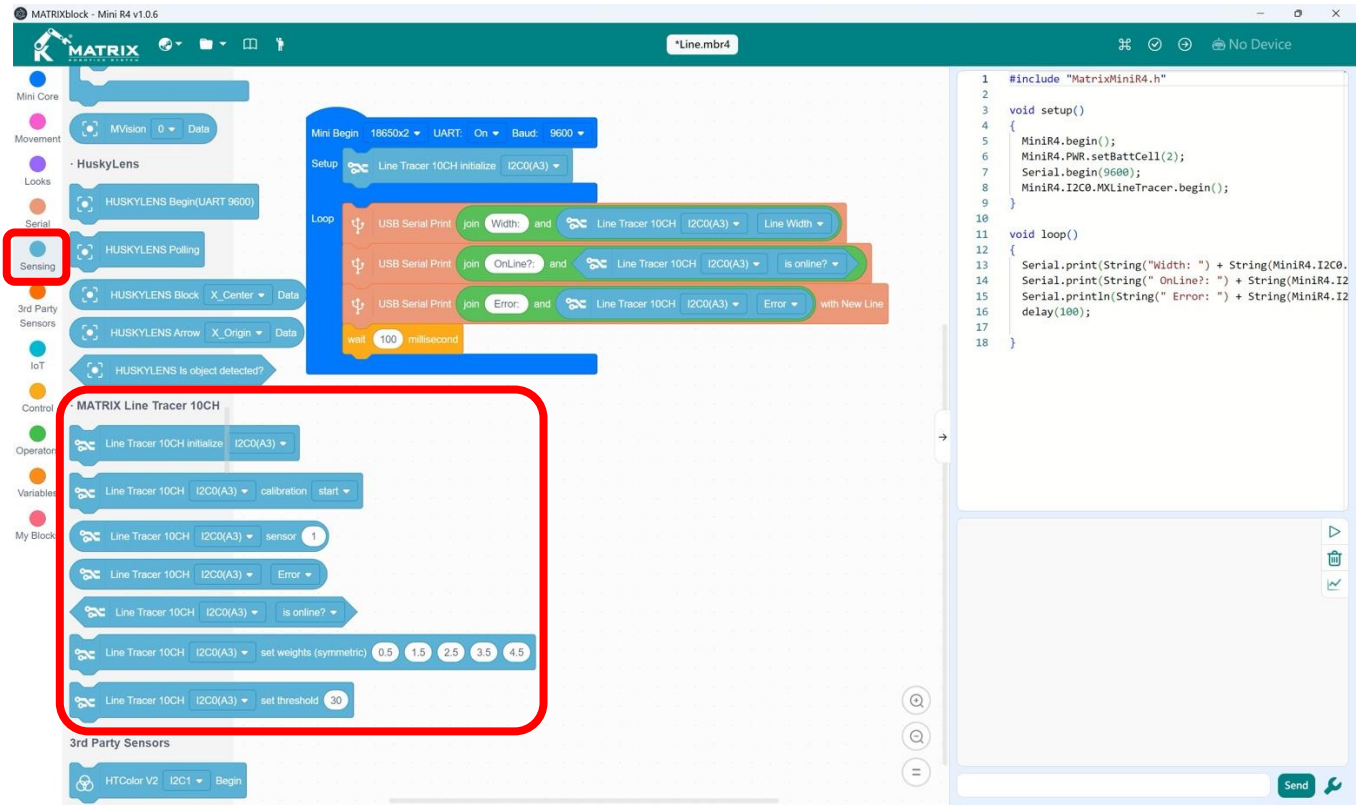
- **Hardware Connection:** Please connect the Line Tracer 10CH to the **A3(I²C 0)** port of the MATRIX Mini R4 controller.



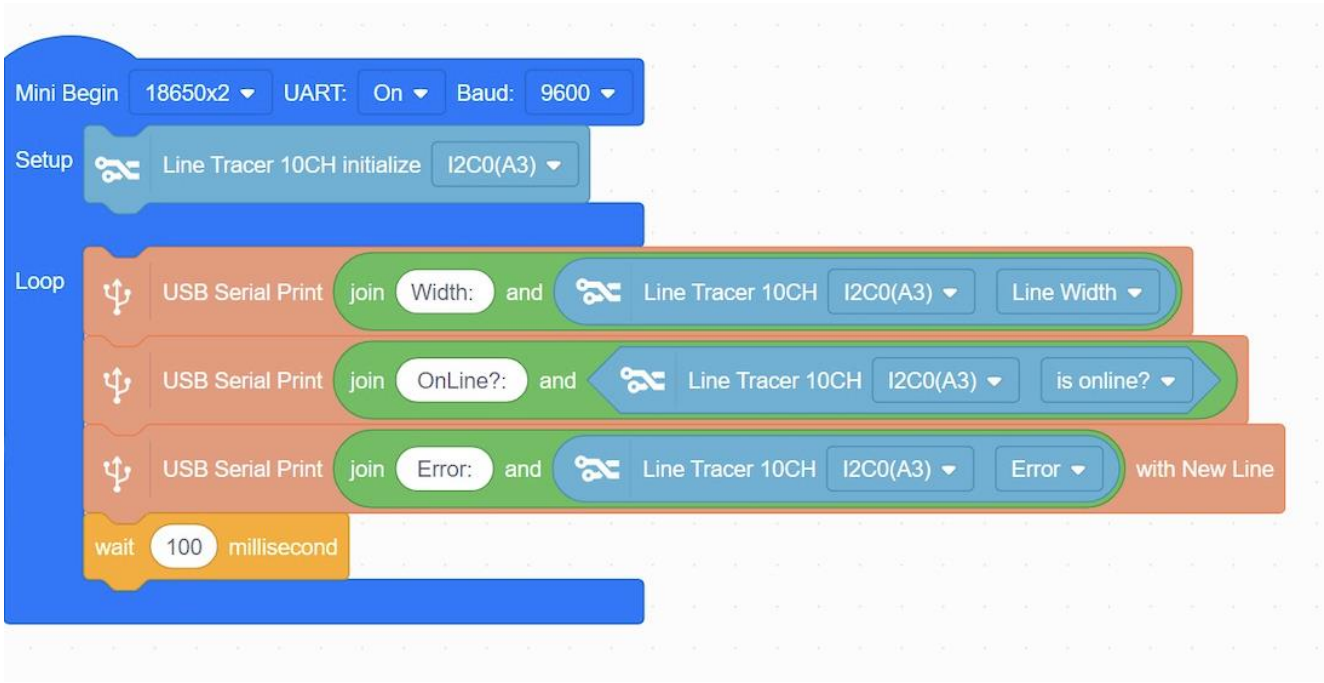


A3 (I²C 0) Pinout - I²C 0			
No.	Name	I/O	I²C
1	SDA	Input/Output	Serial Data Line
2	SCL	Input	Serial Clock Line
3	VCC	Output	Power Supply Pin
4	GND	-	Ground

- **Sample Code:**



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
```

```
5.  MiniR4.begin();
6.  MiniR4.PWR.setBattCell(2);
7.  Serial.begin(9600);
8.  MiniR4.I2C0.MXLineTracer.begin();
9.  }
10.
11. void loop()
12. {
13. Serial.print(String("Width: ") + String(MiniR4.I2C0.MXLineTracer.getLineWidth()));
14. Serial.print(String(" OnLine?: ") + String(MiniR4.I2C0.MXLineTracer.isOnline()));
15. Serial.println(String(" Error: ") + String(MiniR4.I2C0.MXLineTracer.getError()));
16. delay(100);
17.
18. }
```

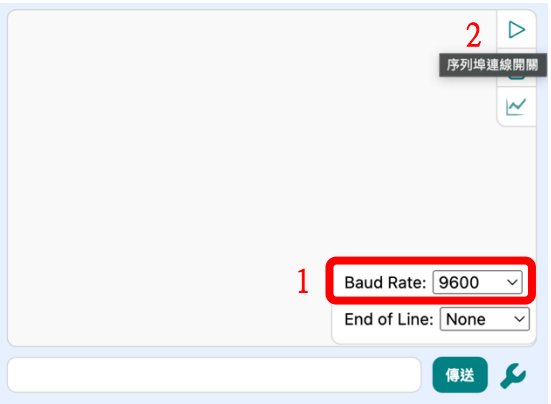
Result Description:

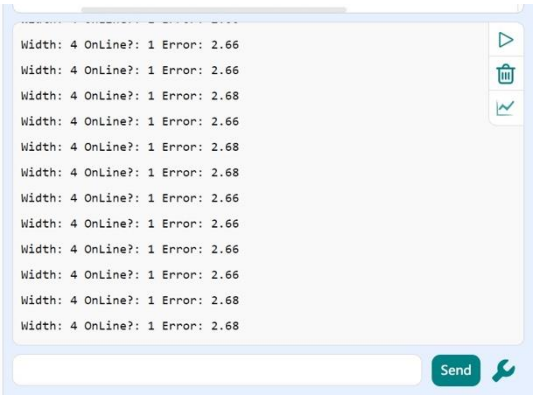
The system updates detection data every 0.1 seconds (100 milliseconds). The following real-time information will be displayed on the screen:

- **Width**: Displays the number of sensors currently covered by the line. A larger value indicates a wider detected ground line.
- **OnLine?**: Indicates whether the sensor array is positioned over a line. It displays **1 (True)** if a black line is detected, and **0 (False)** if it has completely deviated from the path.
- **Error**: Represents the degree to which the current line center deviates from the sensor array center. **The default output range for Error is between -4.5 and 4.5.**

You can utilize these values to program the robot's steering logic. For example, as the **Error** value increases, you can increase the motor differential speed to guide the robot back onto the correct path.

1. Confirm that the Baud Rate matches the one set in the program.
2. Press the serial port connection switch.





4.5.6.7. Principle Description

The MATRIX Line Tracer 10 CH module operates based on visible light reflection technology. Internal transmitters emit light toward the ground, while high-sensitivity receivers detect the intensity of the reflected light. Because dark colors (such as black lines) and light colors (such as white backgrounds) differ significantly in light absorption and reflectivity, the internal MCU (Microprocessor) collects and analyzes data from all 10 sensing channels simultaneously.

The module automatically normalizes the detected raw signals to calculate the current Line Width and the Error (the offset of the line center relative to the sensor array's center point). When the sensor array passes over an intersection, the built-in logic identifies the Junction Type based on the specific combination of triggered sensors. All processed results are transmitted via the I^2C bus, allowing the main controller (Mini R4) to access high-level information directly, greatly reducing the complexity of program development.

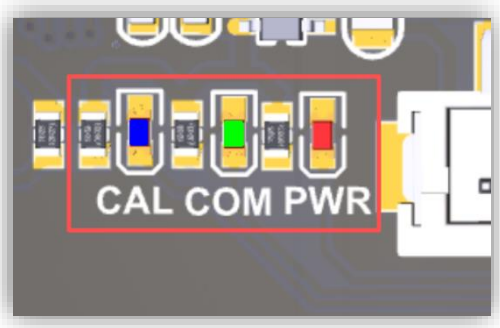
4.5.6.8. Indicator Lights & Calibration

The module includes tri-color indicator lights and a convenient hardware calibration button for real-time status monitoring and optimal performance.

A. Indicator Lights

By observing the onboard LED indicators, you can quickly determine the sensor's current operating status:

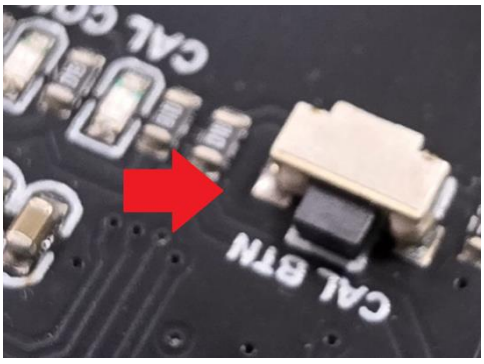
Status LED	Status	Meaning
PWR (Red)	Blinking	Working properly.
	Solid	Malfunction, power cycle device.
COM (Green)	Blinking	I2C Communication properly.
	Solid	I2C hang up, power cycle device.
CAL (BLUE)	Off	Calibration Mode End.
	Solid	Calibration Mode Start.



B. Calibration Mode

Because floor colors and line reflectivity vary between venues, calibrating before use ensures accuracy. Calibration data is stored persistently in the sensor and is not lost when power is removed.

- To Start/End Calibration Mode:



1. Hardware: Long press the "CAL BTN" button on the sensor.
2. Software: Use program commands (Registers 0x20 and 0x21).

- How to Calibrate:

1. Start Calibration mode (Blue CAL LED will turn on).
2. Smoothly move the sensor between the black line and white background several times so it learns the reflection values of both.
3. Stop Calibration mode (Blue CAL LED turns off); the sensor will automatically remember the values and optimize detection.

4.5.6.9. Register Summary (Advanced Reference)

Advanced users can directly access internal registers to obtain more metadata.

Default I2C Address: 0x30

Register Tabel (Summary) Default Device Address: 0x30			
Register	**Name**	**Data Type**	**Description**
0x00-0x09	Norm Value	uint8	Normalized light value of S1-S10
0x10	Line Width	uint8	Number of sensors on the line
0x13	Last Sensor	uint8	The last sensor to detect the line before it went missing.
0x15	Junction Type	uint8	0 = None, 1 = Left, 2 = Right, 3 = T or Cross, 4 = Unknown/None
0x20	Start Calib	-	Start Calibration Mode
0x21	End Calib	-	End Calibration Mode
0x30-0x39	Raw Value	uint8	Raw value of S1-S10
0xF0	FW Version	String	Sensor Firmware Version
0xF1	I2C Address Change	uint8	Change Device Address (0x30 ~ 0x37) *Recommended for professional users only

4.5.6.10. Dimensions

4.6. External UART Inputs

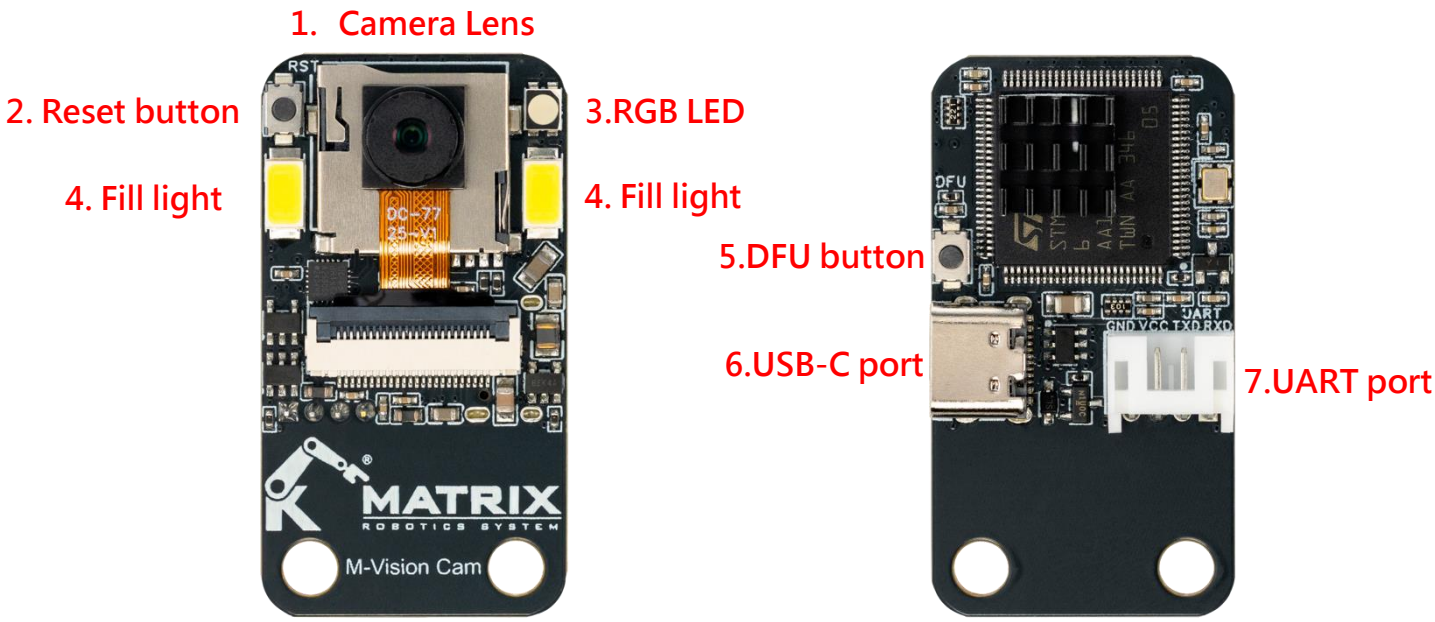
4.6.1. M-Vision Cam MS-010

4.6.1.1. Overview

The M-Vision Cam is a programmable smart camera that integrates an image recognition sensor with a microcontroller. It is capable of independently performing basic edge vision processing tasks with low power consumption. Use **OpenMV IDE** and **Micro Python** to program the M-Vision Cam for functions such as color tracking, object detection, April tag recognition, and simple image classification.

The M-Vision Cam supports UART communication, allowing it to easily exchange data with host controllers such as the MATRIX Mini R4 and send image recognition results in real time. This enables the Mini R4 to execute control instructions corresponding to image data, such as driving to track targets or automated decision-making. Due to the M-Vision Cam's built-in computing power, it can be used an entry-level implementation of AI edge computing and machine vision applications, and is widely used in education, maker projects, and robotics competitions.

4.6.1.2. Physical Layout and Key Components



Main Components:

1. **Camera Lens:** Located on the front of the module, captures images
2. **Reset Button:** Press to restart the Micro Python program running on the camera.
3. **RGB LED:** Programmable multi-color LED to display operating status or other user-defined functionality.
4. **Fill Light LEDs:** LEDs on either side of the lens that provide light when ambient lighting is low.
5. **DFU (Device Firmware Upgrade Button):** Use with the USB-C port to put the module in firmware update mode.

6. **USB-C Port:** Use to connect to a computer for programming, image transmission, firmware updates, or to power the module.
7. **UART Port:** Provides serial communication capability for data exchange with the MATRIX Mini R4 or other microcontrollers. For detailed pin definition, please refer to 4.6.1.5. Electrical Characteristics & Pinout.

4.6.1.3. Features

The M-Vision Cam is a smart lens module with built-in image processing capabilities. It can complete image analysis and send results without an external computer. Here are its main features:

- **Image processing:**
  M-Vision can perform a variety of image recognition tasks, such as:
  - **Color block detection and tracking**: Find and track specific colors
  - **Line detection:** Identify and track lines or edges on the ground
  - **Shape recognition:** Detect common shapes like circles and rectangles
  - **April Tag recognition**: Accurately read visual tag ID
  - **Simple face detection**
  - **QR Code scanning**

- **Programming development environment:**
  - **Micro Python: Use concise and easy-to-learn Python syntax to program the camera**
  - **OpenMV IDE**: Write programs, preview camera images, and adjust parameters in real time on a computer
  - **Main program execution**: Name the main program main.py and load it onto the camera. It will automatically execute after booting.

- **Data transmission and communication:**
  - **UART serial communication**: Transmits identification results (such as coordinates or tag ID) to the main control board
  - **USB Virtual Serial Port (VCP)**: When connected to a computer, data can be transmitted or debugged via USB

- **Adjustable image settings:**
  Provides flexible image capture settings to meet the needs of various applications and scenarios.
  - **Resolution switching**: Supports 320x240 (QVGA) and 160x120 (QQVGA)
  - **Pixel format switching**: Can select color (RGB565) or grayscale image
  - **Screen flip**: Supports horizonal and vertical mirroring (use set_vflip() and set_hmirror())
  - **Manual parameter settings**: Exposure, gain, white balance, etc. (depending on sensor and firmware support)

- **Onboard auxiliary components:**
  - **RGB status light**: Customizable color and flashing, displays program status/debugging

- **LED fill light**: Provides lighting in dark settings, adjust brightness automatically with fill_light.py
- **Reset button**: Restarts the main program
- **DFU button**: Enter firmware update mode

4.6.1.4. Application Scenarios

- Basic applications: Automatic tracking and tracing
  - Color tracking car: Detects position of user-designated color blocks and automatically moves the car toward the block.
  - Line-following robot: Identify black lines and drive car along them.
  - Obstacle avoidance: Use color block or edge detection to determine locations of obstacles.
- Advanced applications: identification, interaction, and classification
  - Automatic color sorting: Identify objects of different colors and push them to different areas.
  - April Tag precise positioning: Use visual tags to assist the robot in navigating, entering a station, or completing a task.
  - QR Code command start: Read the QR code to perform a specific task, like playing a sound or moving a mechanism.
  - Face detection interaction: Activate motor, voice, or screen prompts after sensing a face.
- Special application: Interactive system and AI edge computing
  - Automatic track and shoot: After the camera detects the target, it controls the servo mechanism to track and shoot the target.
  - Interactive pet device: Tracks a designated object (such as a toy ball) and automatically activates a feeder or makes a sound.
  - AI model interface (advanced): M-Vision Cam supports TensorFlow Lite for Microcontrollers. Experienced users can deploy self-trained models for image classification or recognition tasks.

4.6.1.5. Electrical Characteristics & Pinout

- **M-Vision Cam Hardware S pecifications:**

Item	Specifications/Typical values	Units	Notes
Operating voltage	5	V DC	Powered via USB-C or UART VCC pin
Core processor	STM32H7	-	Speed 480 MHz
Flash memory	2	MB	100 KB available memory
SRAM	1	MB	-
Scalability	Supports MicroSD card up to 32 GB	GB	-
Max resolution	640×480 (and below)	pixels	-
Operating current	~200 (Without fill light) ~350 (With fill light)	mA	Varies depending on application and computing load
Communication interface	USB Type-C, UART (PH2.0-4P)	-	UART is the main mode of communication with the main control board
UART communication speed	9600 ~ 115200	bps	Mini R4 default is 115200

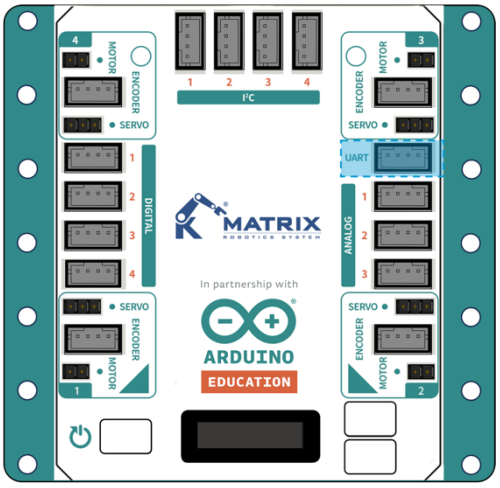
Glossary:

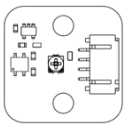
1MByte = 8Mbit ◦

Image sensor* Different sensor models may vary in low-light performance, dynamic range, shutter type (global/rolling), etc.

- **Pin and wiring description**

NO.	M-Vision Cam Pinout	Description	Connection to Mini R4
1	RXD	Information recieving	TX
2	TXD	Information sending	RX
3	VCC	Power input (5V)	VOUT
4	GND	Grounding	GND





UART Pinout			
No.	Name	I/O	Description
1	TX	Output	Serial transmission, send data to other devices
2	RX	Input	Serial receiving line, receive data from other devices
3	VCC	Output	Power supply pin
4	GND	-	Grounding

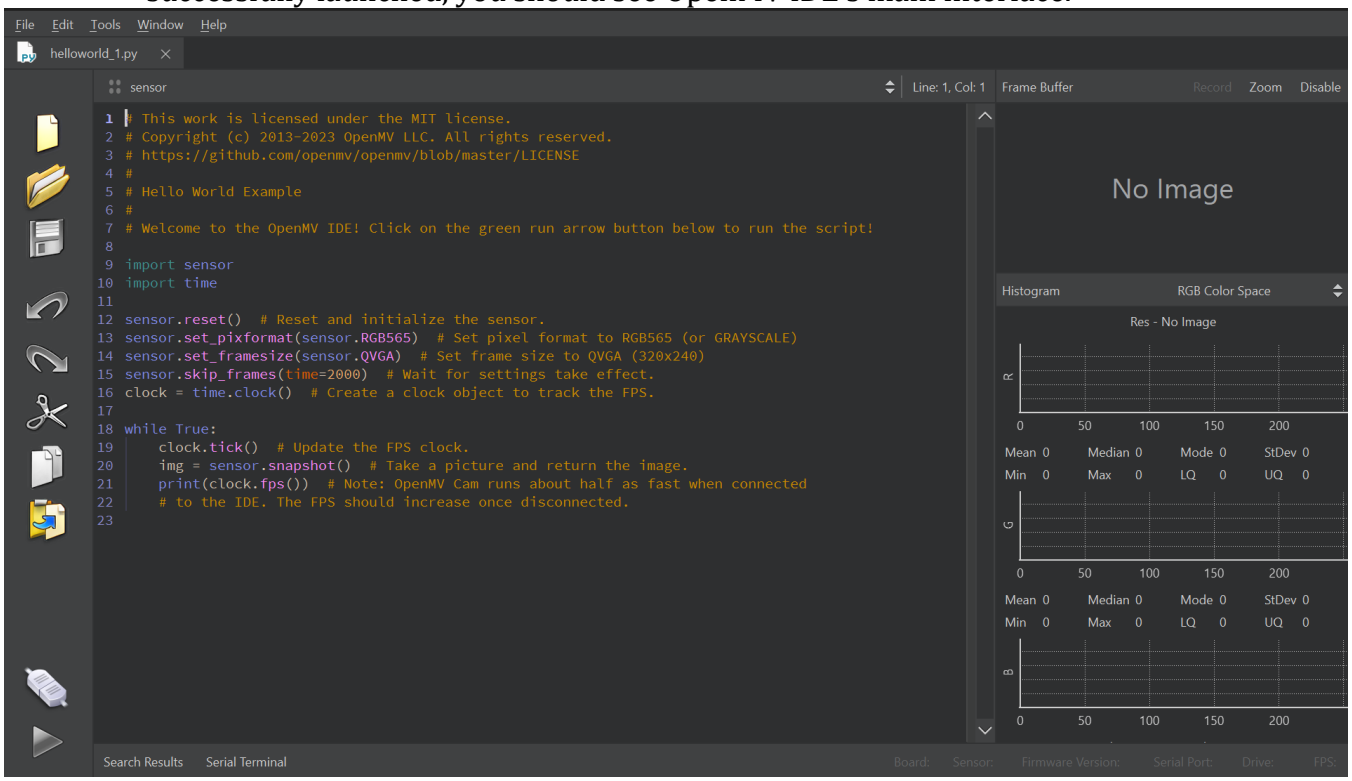
4.6.1.6. Setup & Usage

This section will guide you through the initial setup of the M-Vision smart lens, loading Micro Python programs, and connecting to and basic programming with the MATRIX Mini R4 controller.

A. M-Vision initial setup (using OpenMV IDE)

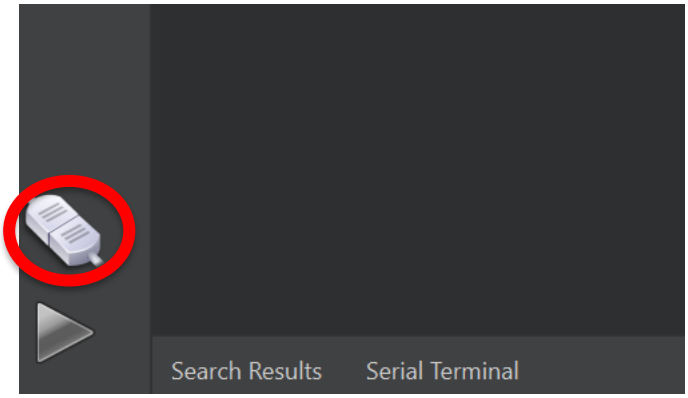
1. **Install OpenMV IDE software:**
   - Go to the OpenMV official website (https://openmv.io/pages/download) to download the latest version of OpenMV IDE for your operating system.

   - Follow the installation program's instructions to complete the installation. Once successfully launched, you should see OpenMV IDE's main interface.



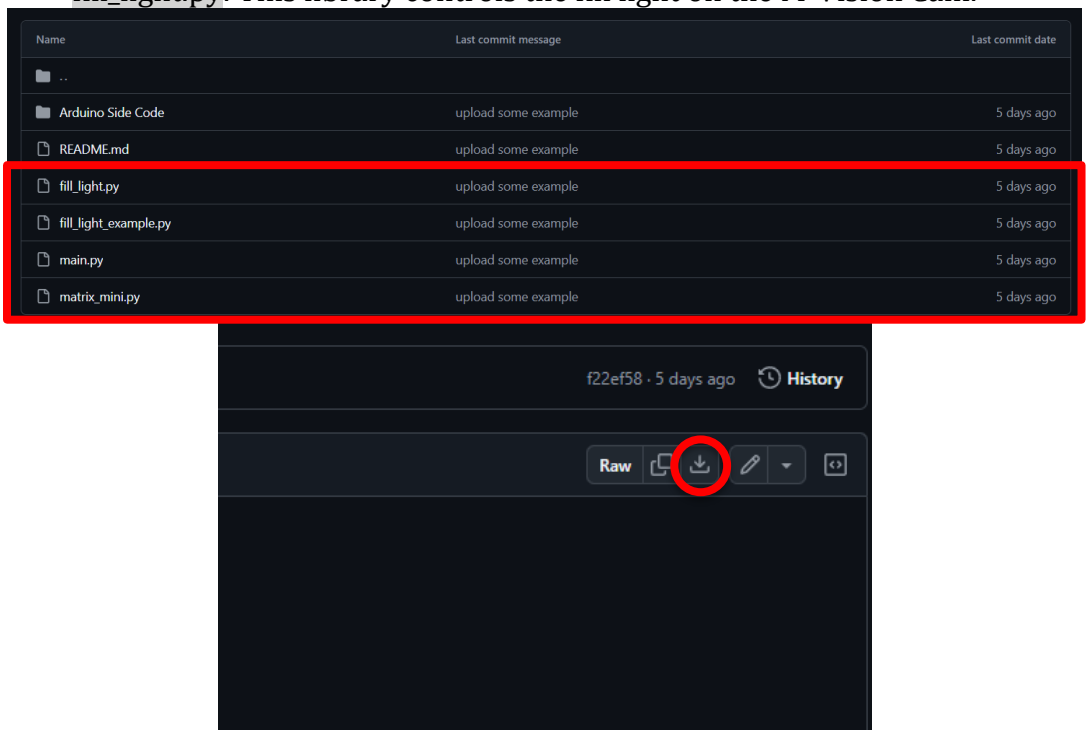
2. **Connect M-Vision Cam to your computer:**
   - Use the included USB-C data transfer cable (not a charging cable) to connect the M-Vision Cam to your computer's USB port.

- Open OpenMV IDE and click the “connect” icon in the lower left corner.



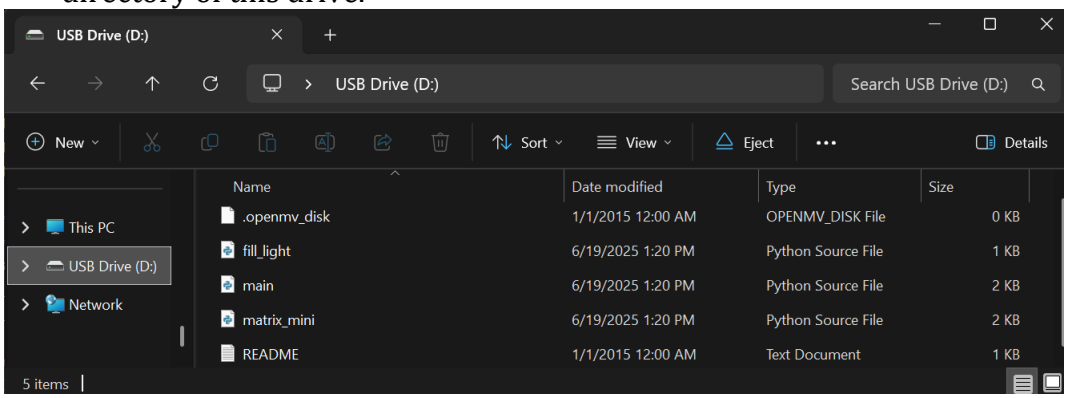
- When connecting for the first time, OpenMV IDE may pop up several prompts to register the OpenMV Cam. Please select “No” or “Cancel” to respond.

3. **Prepare and load the Micro Python core script:**
   - The M-Vision Cam requires specific Micro Python scripts (.py files) to properly function and communicate with the MATRIX Mini R4.
   - Download link: GitHub 程式庫 download the following core scripts:
     - main.py: The main program that is automatically executed after the M-Vision Cam is turned on, containing the core image processing logic.
     - matrix_mini.py: This library is responsible for UART communication protocol processing between the M-Vision Cam and the MATRIX controller.
     - fill_light.py: This library controls the fill light on the M-Vision Cam.

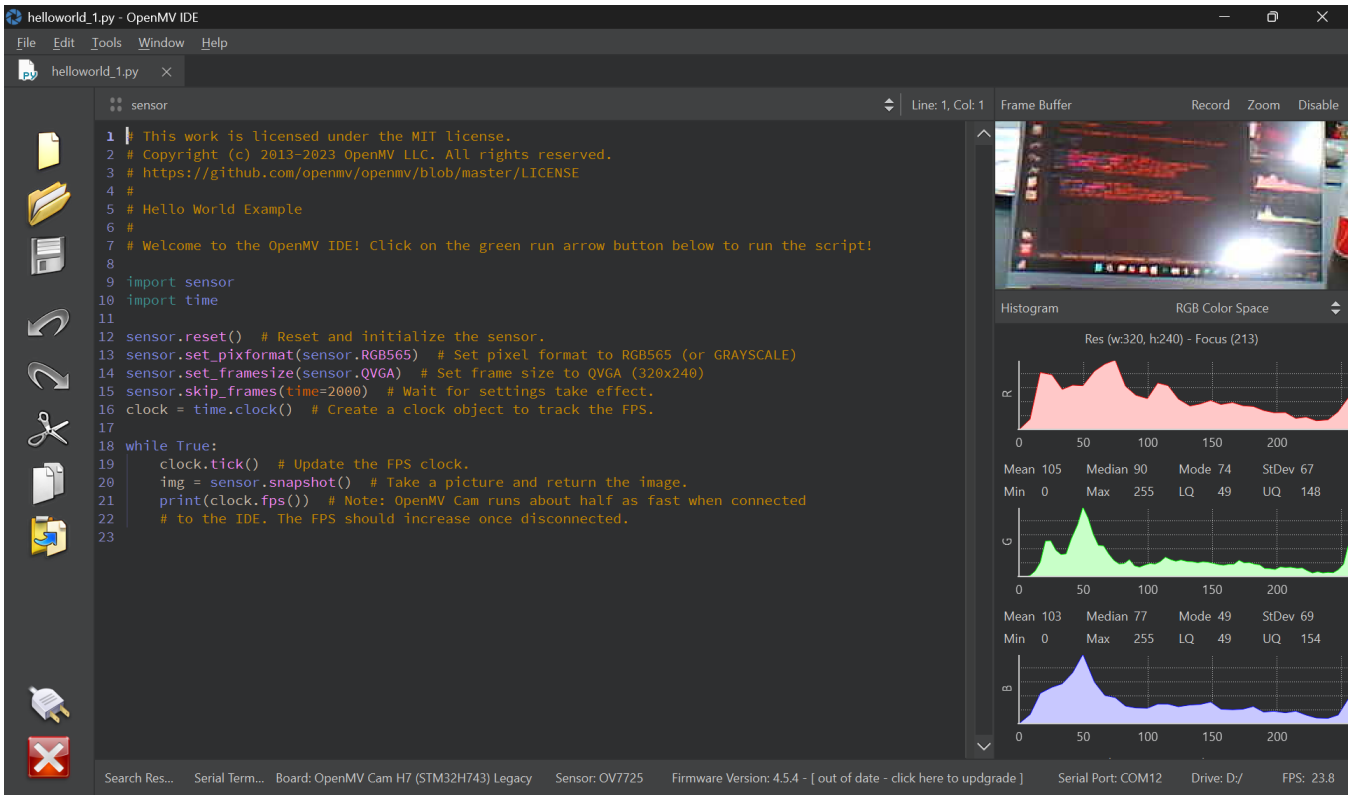


   - **Load the script onto M-Vision:**
     - Once the M-Vision Cam is successfully connected to a computer and recognized by OpenMV IDE, it will usually emulate a USB flash drive.

- Open file explorer (Windows) or Finder (macOS) to locate the drive corresponding to the M-Vision Cam.
- Copy the three downloaded files (main.py, matrix_mini.py, fill_light.py) to the root directory of this drive.



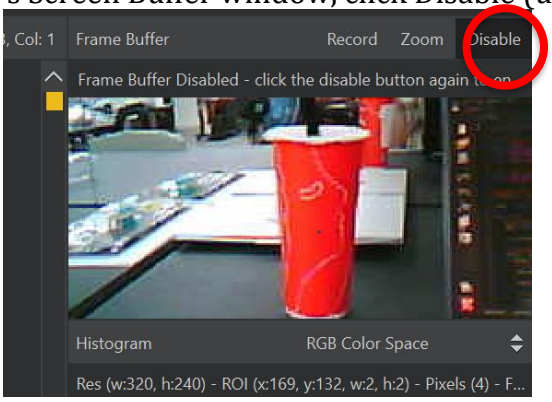
4. **Basic video capture and testing (within OpenMV IDE):**
   - In OpenMV IDE, you can open main.py (or other scripts) in the M-Vision drive via “File” -> “Open file”, or directly click on the file browser on the left.
   - **Executing the script:** Click “Execute script” (green play button) on the lower left corner of OpenMV IDE.
   - **View video:** If everything is working correctly, you should be able to see the video captured by the M-Vision camera in the upper right corner of OpenMV IDE.



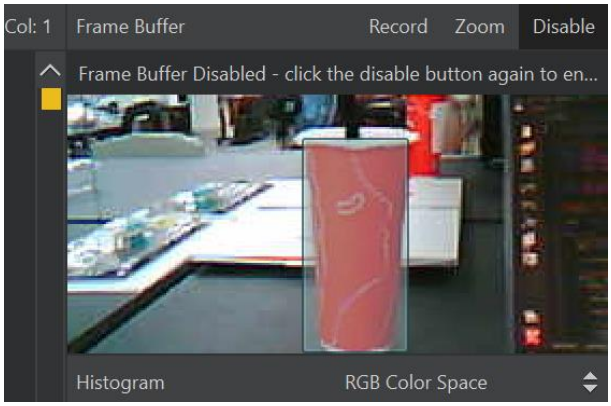
- **Basic test code example (you can create a new file or modify main.py)**

```
import sensor
import time

sensor.reset() # Reset and initialize the image sensor
sensor.set_pixformat(sensor.RGB565) # Set pixel format to RGB565 (color)
sensor.set_framesize(sensor.QVGA) # Set screen size to QVGA (320x240)
sensor.set_vflip(True) # Flip the image vertically (adjust according to installation position)
sensor.set_hmirror(True) # Flip the image horizontally (adjust according to installation)
sensor.skip_frames(time = 2000) # Wait for sensor to stabilize, skip first few frames

clock = time.clock() # Create a clock object to track frames per second

while(True):
clock.tick() # Update FPS timer
img = sensor.snapshot() # Capture an image
print(clock.fps()) # Output the current FPS in the serial terminal
# Here, the img object is the captured image, which can be processed later
# Ex: img.draw_string(10, 10, "Hello World!", scale=2)
```

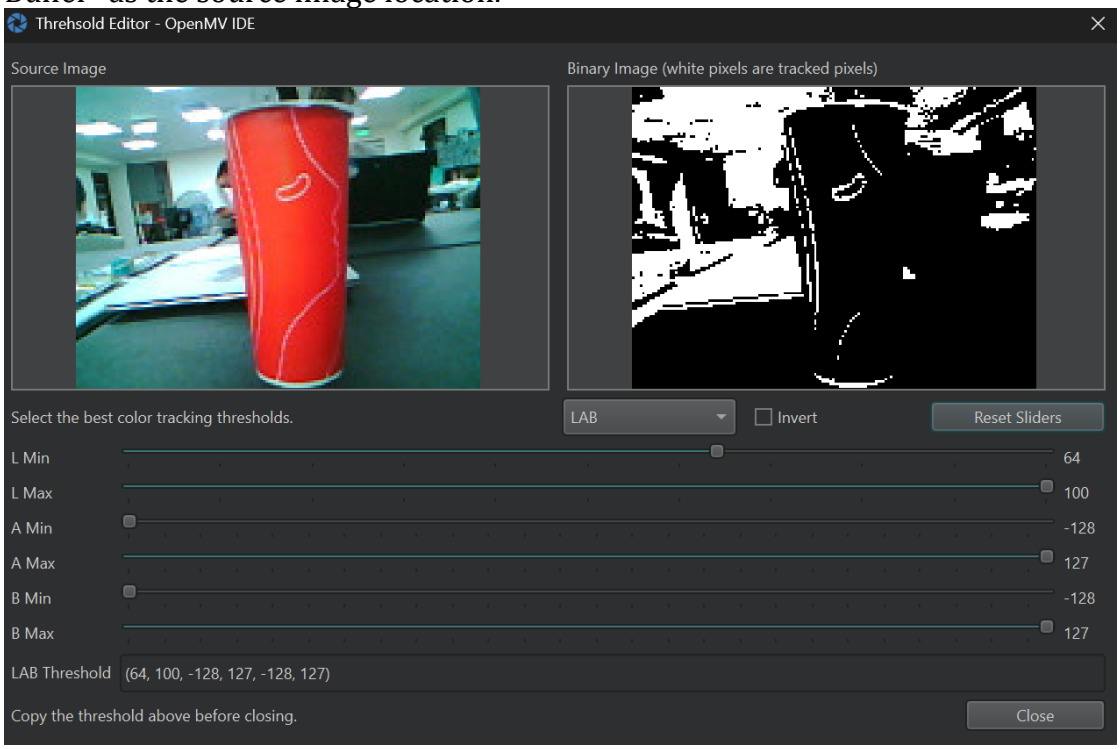
5. **Set the target color threshold (Ex: finding color blocks):**
   - In OpenMV IDE, execute main.py, which already contains the img.find_blobs() function.
   - Aim the M-Vision lens at an object of the color you wish to detect.
   - In OpenMV IDE's Screen Buffer window, click Disable (upper right corner) to freeze the current screen.

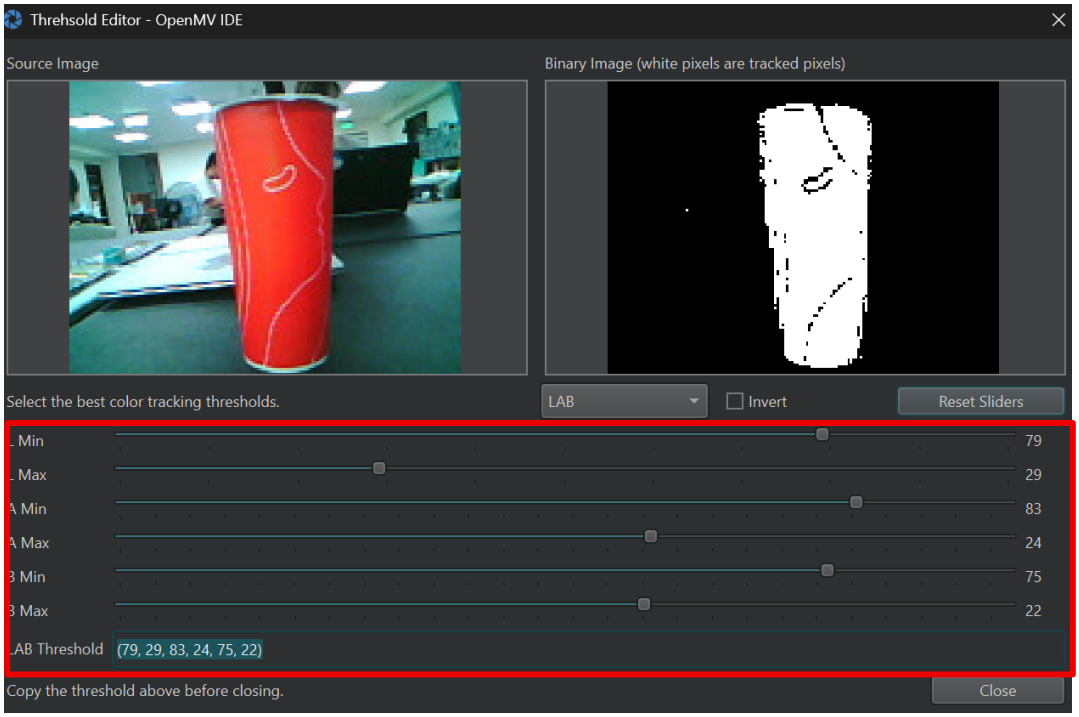


   - Use your mouse to select the target color area on the frozen image.



- Open the “Tools” menu -> “Machine Vision” -> “Threshold Editor” and select “Screen Buffer” as the source image location.

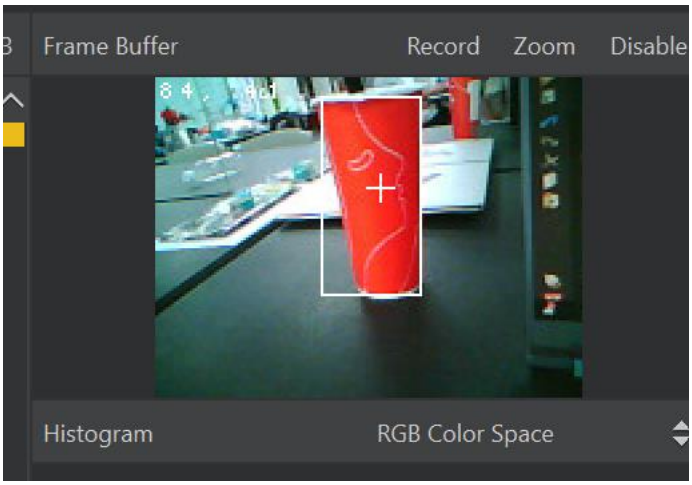




- In the threshold editor, fine-tune the LAB color sliders (min and max values for L, A, and B) until the target color is white and the rest of the colors are black in the viewing screen.
- Copy the adjusted thresholds (Usually a string of 6 numbers representing LMin, LMax, AMin, AMax, BMin, BMax).
- Paste this threshold value into your main.py script where the threshold variable is defined. For example: threshold = (LMin, LMax, AMin, AMax, BMin, BMax)

```
14 # Define color threshold range (HSV), suitable for detecting the target color
15 threshold = (45, 0, 12, 117, 90, -101)  # Hue, Saturation, and Value range
16
```

- Save the main.py file by clicking “File” -> “Save Opened script to MV cam” or ctrl+S.
- Re-execute the script, and M-Vision will start detecting and marking the target color.



B. MATRIX Mini R4 settings and programming

1. **Hardware connection:**
   - According to the “Hardware Connection Example” in part B of section 4.6.1.5 Electrical Characteristics and Pinout, use the dedicated cable to correctly connect the UART port of the M-Vision to the UART port of the MATRIX mini R4 controller.
2. **MATRIX Mini R4 programming (using Arduino IDE):**
   - **Install the library:** Make sure to install the MatrixMiniR4 library using the library manager in Arduino IDE.
   - **Programming example:**
   - **Compile and upload:** Select the correct board ("Arduino UNO R4 WiFi") and serial port in Arduino IDE, then compile and upload the program to MATRIX Mini R4.
3. **MATRIX Mini R4 Programming (Using MATRIXblock):**
   - **(Please refer to《M-Vision Tutorial - zh.docx》Chapter 3-3 MATRIXblock: A screenshot of the program and a brief description of its logic are shown below:)**
   - **Setup:** Under the MiniR4 “Setup” block, usually no other blocks are needed to connect to the M-vision cam. The MatrixMiniR4 library will handle it in MiniR4.begin().
   - **Reading data:**
     - Use the “Read data from Vision Module into List [list_name]” block if available, otherwise use the equivalent UART Read block.
     - Or use a block like “Vision module read to 0^{th} data” to get data at a specific index.
   - **Processing and display:**
     - Store the read data (ex. The X coordinate of a color block) in a variable.
     - Use the OLED screen to display blocks and variable values.
     - Insert sample block code with matrix block reading and displaying data.

C. System operation: After completing the programming and settings of the M-Vision and MATRIX Mini R4, power them both on. M-Vision will execute the image processing logic within main.py and send the results to the Mini R4 via UART. The Mini R4 will receive this data and

perform specified operations according to your program logic (such as operating motors, displaying information, etc.)

4.6.1.7. Application with MATRIXblock / Arduino

Programming example: Red and blue object detection and tracking: The M-Vision cam will detect red and blue objects. The MATRIX mini R4 will show a red light if a red object is detected and a blue light if a blue object is detected.

OpenMV IDE main.py:

```
import sensor, image, time, struct
from pyb import UART

# Initialize camera
sensor.reset() #Reset sensor
sensor.set_pixformat(sensor.RGB565) #Set pixel format to RGB565 (color)
sensor.set_framesize(sensor.QQVGA) #Set screen size to QQVGA (160x120)
sensor.set_vflip(False) #Adjust image based on installation position
sensor.set_hmirror(True)
sensor.skip_frames(time=2000) #Wait for sensor to stabilize
print("Camera Ready")

# UART initialization
uart = UART(3, 115200) #Set UART communication speed to 115200 bps at port 3

clock = time.clock() #Create clock object to track frames per second

# LAB color threshold
red_threshold = (30, 100, 15, 127, 15, 127)       # red threshold
blue_threshold = (20, 70, -10, 10, -80, -15)      # blue threshold

while True:
    clock.tick() #update FPS timer
    img = sensor.snapshot() #take a snapshot

    # Detect red and blue color blocks
    #Creates a list of red and a list of blue objects above an area of 100 pixels
    red_blobs = img.find_blobs([red_threshold], pixels_threshold=100,
area_threshold=100)
    blue_blobs = img.find_blobs([blue_threshold], pixels_threshold=100,
area_threshold=100)

    #Finds maximum of each list of color blobs
    red_blob = max(red_blobs, key=lambda b: b.pixels()) if red_blobs else None
```

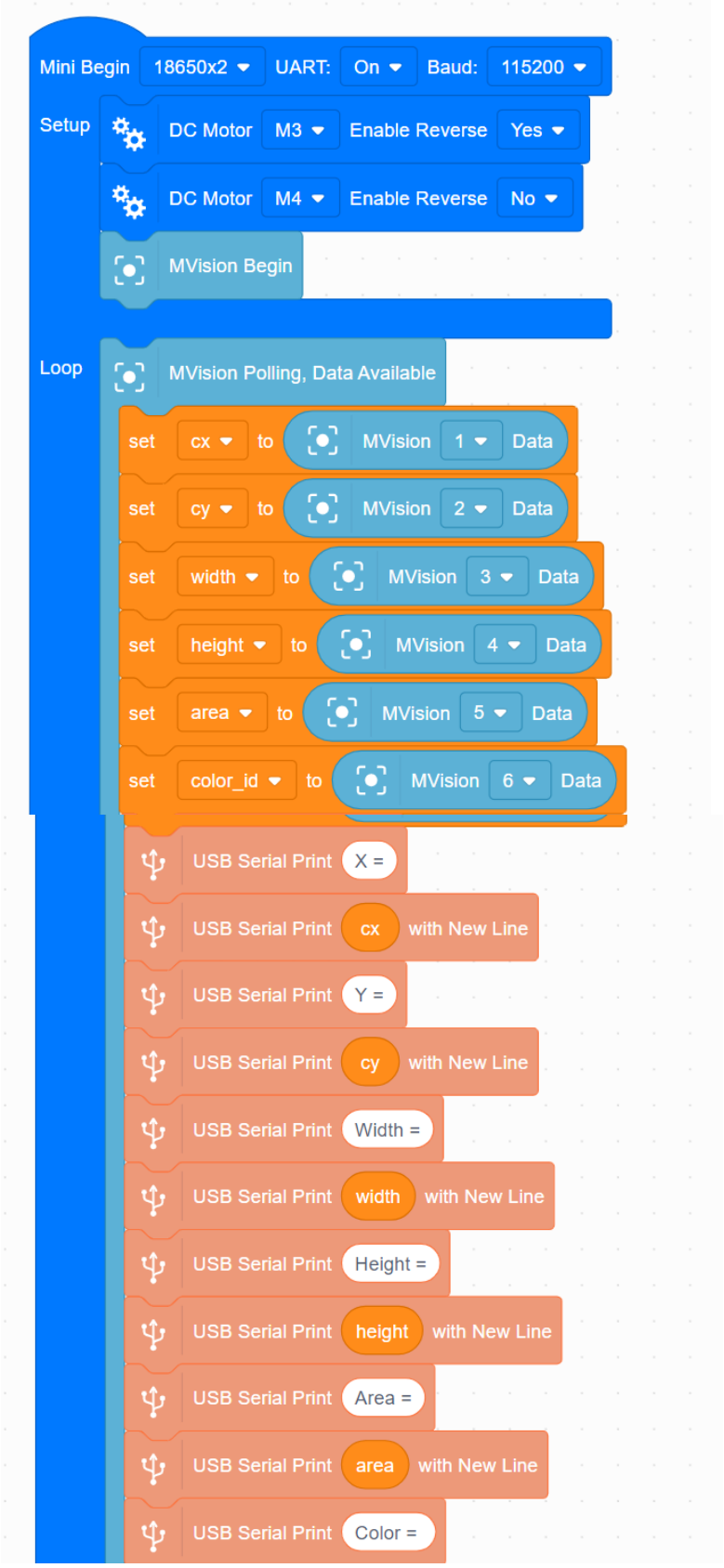
```
    blue_blob = max(blue_blobs, key=lambda b: b.pixels()) if blue_blobs else None
    #Initialize target_blob and color_id variables
    target_blob = None
    color_id = 0

    if red_blob and blue_blob: #If there are both red and blue objects
        if red_blob.pixels() > blue_blob.pixels(): #If red object's area is larger
            target_blob = red_blob #Set target blob to red
            color_id = 1  # Red
        else: #If blue object's area is larger
            target_blob = blue_blob #Set target blob to blue
            color_id = 2  # Blue
    elif red_blob: #If there is only a red object
        target_blob = red_blob #Set target blob to red
        color_id = 1
    elif blue_blob: #If there is only a blue object
        target_blob = blue_blob #Set target blob to blue
        color_id = 2

    if target_blob: #If a red or blue object was detected
        cx = target_blob.cx() #X coordinate of blob's center
        cy = target_blob.cy() #Y coordinate of blob's center
        w = target_blob.w() #Blob width
        h = target_blob.h() #Blob height
        area = w * h #Blob area

        #Blob framing
        #Draw a red rectangle around the object
        img.draw_rectangle(target_blob.rect(), color=(255, 0, 0))
        #Draw a red cross in the center of the object
        img.draw_cross(cx, cy, color=(255, 0, 0))

        #Send 6 uint16:  [cx, cy, w, h, area, color_id]  (total 12 bytes)
        uart.write(struct.pack("<6H", cx, cy, w, h, area, color_id))
        print("Largest color block: (color_id={}):".format(color_id), cx, cy, w, h,
area)
    else:
        #No object detected, send all 0's
        uart.write(struct.pack("<6H", 0, 0, 0, 0, 0, 0))
        print("No red or blue objects detected.")

    print("FPS:", clock.fps())
```

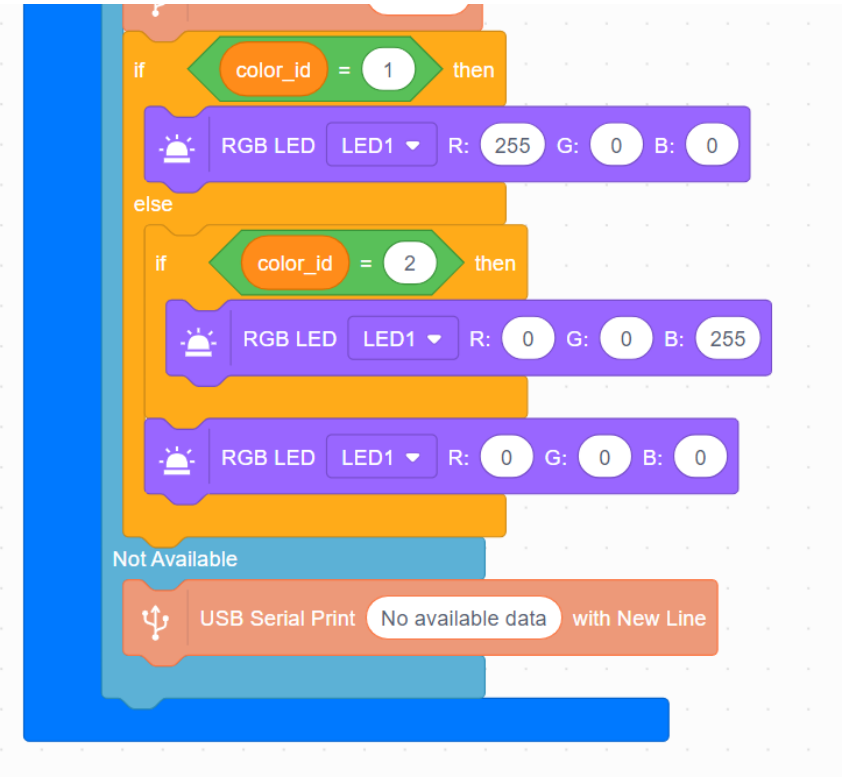
Arduino Script: (Choose to use either Arduino or MATRIXblock to program Mini R4)

```
#include <MatrixMiniR4.h>
```

```
float direction;
void setup() {
  MiniR4.begin(); // Initialize MiniR4
  Serial.begin(115200); // Set communication speed for Serial
  Serial1.begin(115200); //Set communication speed for Serial 1
}

void loop() {
  if (Serial1.available() >= 12) { // Checks if data is available to read
    uint8_t buffer[12]; // Creates an array to store 12 bytes of data, each byte is an
integer
    Serial1.readBytes(buffer, 12); // Reads 12 bytes from Serial 1 and stores in array

    uint16_t cx = buffer[0] | (buffer[1] << 8); // X coordinate of object center
    uint16_t cy = buffer[2] | (buffer[3] << 8); // Y coordinate of object center
    uint16_t w  = buffer[4] | (buffer[5] << 8); // Object width
    uint16_t h  = buffer[6] | (buffer[7] << 8); // Object height
    uint16_t area = buffer[8] | (buffer[9] << 8); // Object area
    uint16_t color_id = buffer[10] | (buffer[11] << 8); // Object color

    // Print information recieved from MVision to Serial terminal
    Serial.print("Image Center(");
    Serial.print(cx); Serial.print(", ");
    Serial.print(cy); Serial.print("), Width=");
    Serial.print(w); Serial.print(" Height=");
    Serial.print(h); Serial.print(" Area=");
    Serial.print(area);
    Serial.print(" Color=");

    if (color_id == 1) MiniR4.LED.setColor(1, 255, 0, 0); // Show a red light if red
object detected
    else if (color_id == 2) MiniR4.LED.setColor(1, 0, 0, 255); // Show a blue light if
blue object detected
    else Serial.println("Not detected"); // Print message if no object is detected
    MiniR4.LED.setColor(1, 0, 0, 0); // Turn off LED if no colors are detected
  }
}
```

Using MATRIXblock: (Choose to use either Arduino or MATRIXblock to program Mini R4)


Mini Begin
18650x2
UART:
On
Baud:
115200
Setup
DC Motor
M3
Enable Reverse
Yes
DC Motor
M4
Enable Reverse
No
MVision Begin
Loop
MVision Polling, Data Available
set
cx
to
MVision
1
Data
set
cy
to
MVision
2
Data
set
width
to
MVision
3
Data
set
height
to
MVision
4
Data
set
area
to
MVision
5
Data
set
color_id
to
MVision
6
Data
USB Serial Print
X =
USB Serial Print
cx
with New Line
USB Serial Print
Y =
USB Serial Print
cy
with New Line
USB Serial Print
Width =
USB Serial Print
width
with New Line
USB Serial Print
Height =
USB Serial Print
height
with New Line
USB Serial Print
Area =
USB Serial Print
area
with New Line
USB Serial Print
Color =



4.6.1.8. Principle & Data Protocol

The M-Vision Smart Camera integrates a high-performance image processing chip. When the lens captures an image, internal algorithms perform real-time analysis to recognize pre-configured targets (such as specific colors, shapes, or faces). Once recognition is successful, the camera encodes these high-level results (e.g., the center coordinates, width, and height of a color block, or the angle of a line) into a specific data format and transmits them to the main controller via the UART serial port. The dedicated function library for the MATRIX Mini R4 is responsible for receiving and decoding this data, allowing users to easily access these visual recognition results directly in their programs without needing to handle complex image processing algorithms.

4.7. Composite Inputs

4.7.1. MATRIX Joystick 2

4.7.1.1. Overview

The MATRIX Joystick 2 (MJ2) is a 2.4GHz wireless remote controller designed specifically for the MATRIX series of robots, including the Mini R4. It offers a variety of control elements, such as dual-axis joysticks and multiple buttons, enabling intuitive and flexible remote control of a robot's movements and functions. MJ2 connects to the MATRIX Mini R4 controller via a matching wireless receiver adapter board, using the SPI communication protocol. This controller is an ideal choice for projects involving complex robot remote control and interactive game control.

4.7.1.2. Feature

Wireless Technology: 2.4GHz wireless communication for stable remote control.
Main Controls:

- **Dual Analog Joysticks ("mushroom heads"):** One on each side, each providing X and Y axis analog values (0-255) for precise direction and speed control.
- **D-pad:** Digital directional input for up, down, left, and right.
- **Four Primary Action Buttons:** △ (Triangle), ○ (Circle), X (Cross), □ (Square)
- **Shoulder Buttons:** L1, L2, R1, R2
- **Other Buttons:** SELECT, START, MODE, and clickable L3 (left stick) and R3 (right stick)

Power: Powered by two AAA batteries.
Pairing: Can pair with its dedicated receiver; usually pre-paired at the factory.
Operation Modes: Supports Analog (default) and Digital modes

- **Important Note:** To ensure correct joystick signal readings, we recommend remaining in the default Analog mode. Digital mode may cause signal transmission issues is accidentally switched.

4.7.1.3. LED Status Indication

The MJ2 and its wireless receiver are equipped with LED indicators to help users quickly determine their current working status.

Controller LED Status:

- **Solid light:** Controller is receiving and in analog mode (default).
- **Flashes every 0.5 secs:** Controller is receiving, but in digital mode.
- **Flashes every 0.25 secs:** Low battery warning.
- **Flashes every 0.1 secs**: Pairing mode. Returns to default after 3 seconds.
- **Flashes every 1 sec:** Standby mode after 15 minutes of inactivity.

Receiver LED Status:

- **Off:** No signal or incorrect ID.

- **Solid Light:** Receiving signal, joystick in analog mode.
- **Flashes every 0.5 secs:** Receiving signal, joystick in digital mode. (Default)
- **Flashes every 0.125:** Pairing mode. Restores to default after 10 seconds.

4.7.1.4. Application

- **Remote Control:** Real-time control of SumoBot or task-based robots in competitions like WRO, MARC, or for custom vehicle projects.
- **Robotic Arms:** Precise remote control of multi-axis robotic arms (e.g., gripping, moving).
- **Interactive Game Controller:** Combine Mini R4 with MH2 as input handles for custom games.
- **Remote Equipment Control:** Wireless operation of display props, small automation devices, etc.
- **Education and Training:**
    - Practice precise motor speed and direction control.
    - Design button combinations to trigger different robot behaviors and train logical thinking.

4.7.1.5. Electrical Characteristics

- MJ2 Controller:

Parameters	Specification	Unit
Wireless Technology	2.4G	-
Power Supply	AAA batteries x 2	-
Operating Voltage	2.4 - 3	Volts

- Receiver:

Parameters	Specification	Unit	Notes
Operating Voltage	5	Volts	Supplied via MATRIX Mini R4
Communication	SPI	-	Interface with the controller

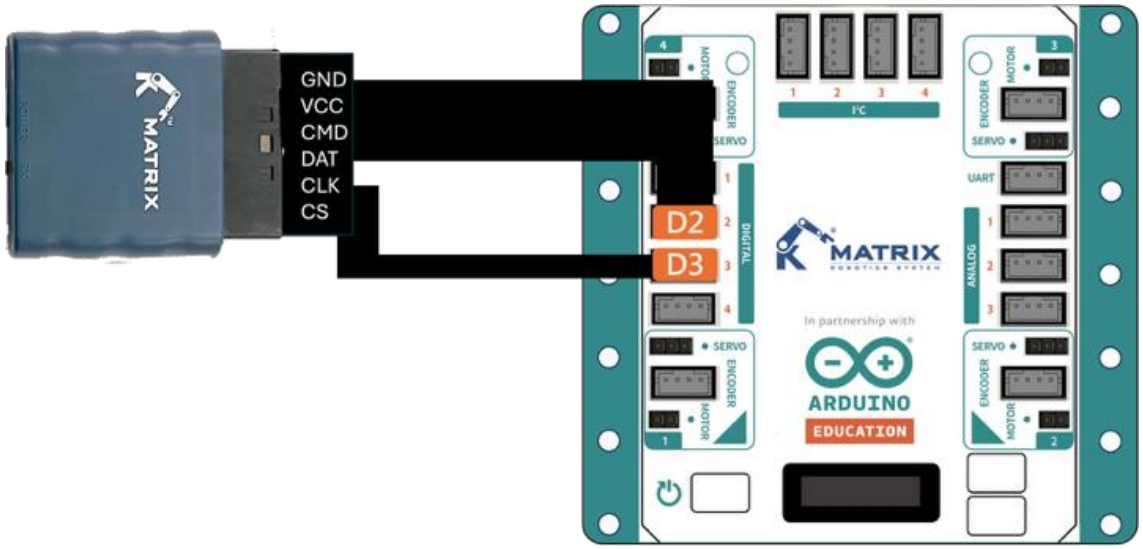
4.7.1.6. Pinout

MJ2 connects to the MATRIX Mini R4 controller through its adapter board. The adapter translates control signals into SPI protocol data readable by the Mini R4.

Receiver adapter board pin description and Mini R4 connection:

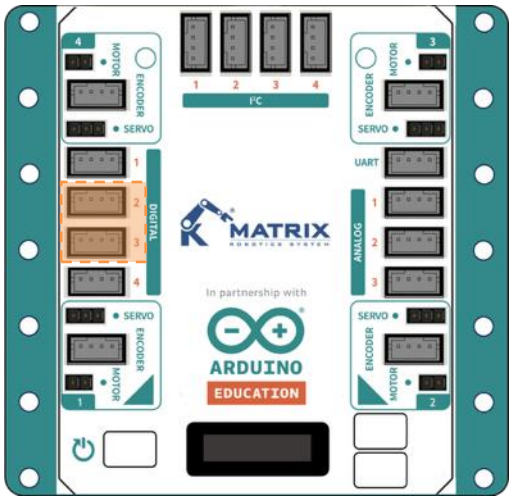
The following table lists the main signal pins of the MJ2 wireless receiver adapter board, their corresponding SPI functions, and the recommended digital interface for connection to the MATRIX Mini R4.

Signal on Adapter	Mini R4 Port	SPI Function	Main Description
VCC	D2	Power	5V from Mini R4
GND		Ground	Ground
CMD (Command)		MOSI	Master Out Slave In: Mini R4 sends commands to the receiver
DAT (Data)		MISO	Master In Slave Out: Mini R4 reads data from the receiver
CLK (Clock)	D3	SCK (Serial Clock)	The clock signal line for SPI communication is generated by Mini R4
CS (Chip Select)		CS/SS (Chip/Slave Select)	The chip select line for SPI by Mini R4, used to select MJ2 receiver.

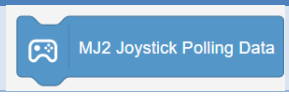
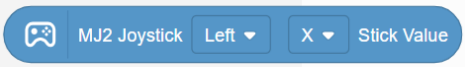


4.7.1.7. Sample Code & Blocks Instructions

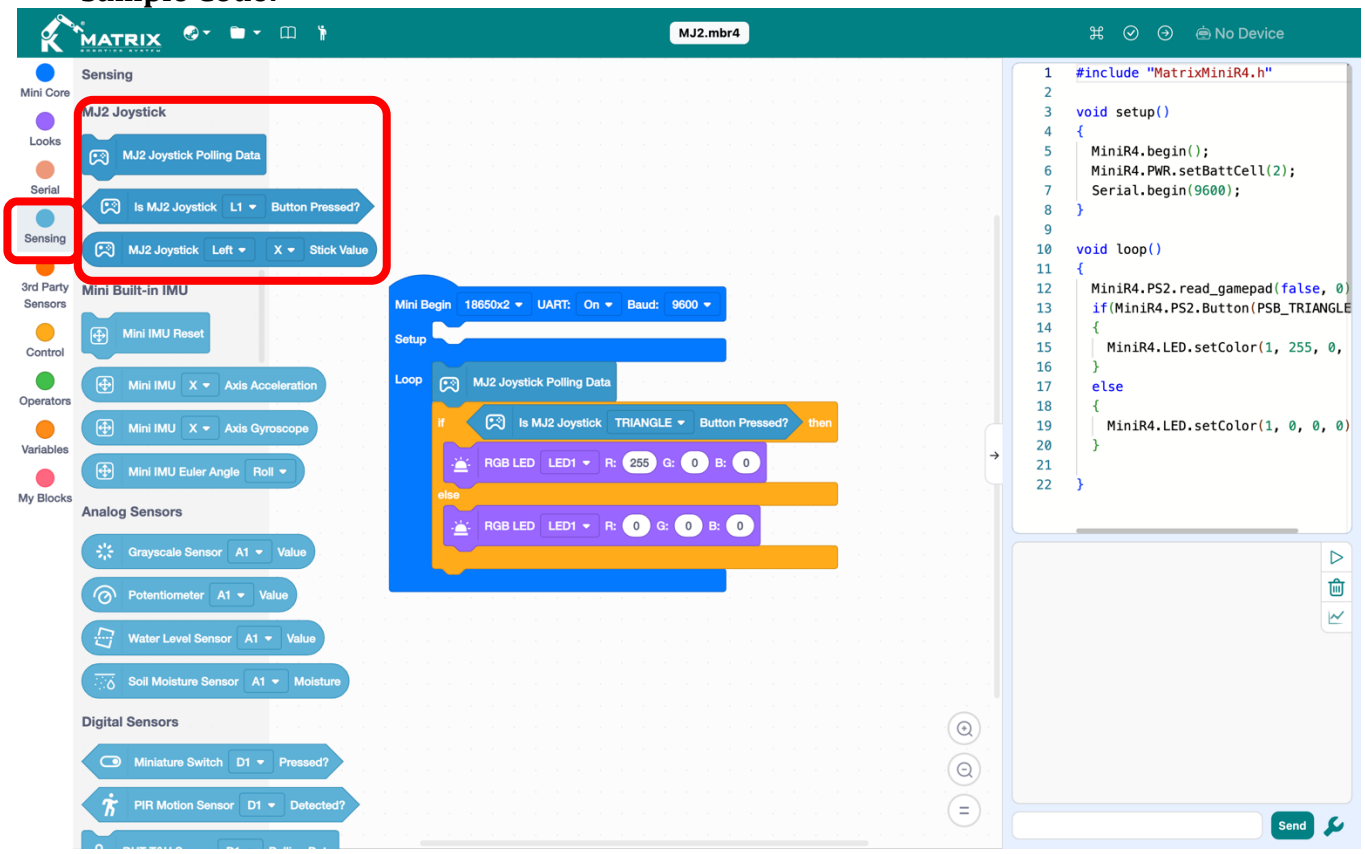
- **Hardware Connection:** Please connect the receiver to the **D2 & D3** ports of the MATRIX Mini R4 controller.



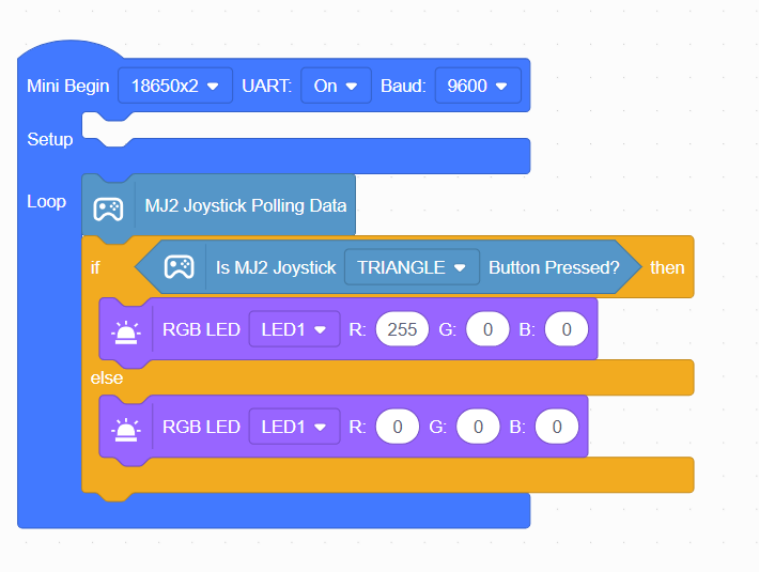
- **Command Descriptions**

Block	Description
	Place at the beginning of the loop.
	Returns true if the selected button is pressed.
	Returns the current position value (X, Y).

- **Sample Code:**



MATRIXblock



IDE C++

```
1. #include "MatrixMiniR4.h"
2. void setup()
```

```
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
}

void loop()
  {
    MiniR4.PS2.read_gamepad(false, 0);
    if(MiniR4.PS2.Button(PSB_TRIANGLE))
  {
  MiniR4.LED.setColor(1, 255, 0, 0);
  }
  else
  {
    MiniR4.LED.setColor(1, 0, 0, 0);
  }

}
```

Explanation:
Pressing the △ (Triangle) button on the Mini R4 lights up the onboard RGB LED (left side) red. Releasing the button turns this light off.

4.7.1.8. Principle Description

MATRIX Joystick 2 (MJ2) wireless remote controller works in conjunction with the MATRIX Mini R4 controller, which mainly involves the following principles:

- **Wireless Communication and Pairing:**
  - The MJ2 handle uses 2.4GHz wireless communication technology to transmit the user's operating instructions to its dedicated wireless receiver. This technology usually has better signal stability and anti-interference ability.
  - o In order to ensure a unique communication link between the controller and the receiver, they need to be "paired". The controller is usually paired with its matching receiver before leaving the factory. However, when necessary (for example, when replacing the receiver or controller), a manual re-pairing process is also supported. The controller can enter pairing mode by pressing and holding the L3 button for more than 2 seconds, and the receiver can enter pairing mode by pressing and holding the Bind button for more than 2 seconds after powering on.
- **Joystick Data Generation and Wireless Transmission:**
  - Internal microprocessor detects joystick and button activity.
  - This status information is encoded into a specific digital signal packet.
  - The controller then continuously sends these data packets through its 2.4GHz wireless module.
- **Receiver and SPI Transmission:**
  - The MAJ2 Wireless receiver adapter board continuously monitors the wireless signal from the paired controller.

    - After receiving the data packet, the receiver decodes and restores it to the original data.
    - Then, this data is sent from the receiver to the Mini R4 via the SPI (Serial Peripheral Interface) communication protocol.
- **Mini R4 Data Reading, Parsing, and Application:**
    - Code reads data with polling commands (e.g., MiniR4.PS2.read_gamepad(false, 0); in Arduino).
    - Libraries handle SPI details and expose readable functions like:
        - Button(PSB_TRIANGLE) – Check if a button is pressed.
        - Analog value blocks/functions – Read joystick position
    - Joystick outputs (0-255) are often mapped to other ranges (e.g., motor speed -100 to 100) for more intuitive control

Through the steps above, the user's operations on the MJ2 can be instantly converted into instructions that the MATRIX Mini R4 controller can understand and execute, thereby realizing wireless remote control of robots or other projects.

5. Output Modules

5.1. Overview

If the input module is the “senses” of the controller, the output module is the “expression” and “voice” that allows it to communicate with the outside world. The output module is responsible for receiving instructions from the controller and converting them to physical responses such as light, sound, or text that can be perceived by the user. Through these modules, your project can truly “express itself” and convey useful information or status to the user.

This chapter will introduce the main **onboard output modules** on the MATRIX Mini R4 in detail. We will explore the three key output components built into the controller one by one: the **programmable RGB LED** (for colorful light indications), the **buzzer** (for generating sound prompts) and the **OLED display** (for displaying text and graphical information.)

By studying this chapter, you will be able to make full use of these onboard output modules, allowing your Mini R4 project to not only think, but also express itself visually and auditorily.

5.2. RGBLEDs

5.2.1. Overview

The MATRIX Mini R4 has 2 built-in **programmable full-color RGB LEDs**, numbered 1 on the left and numbered 2 on the right. These LEDs can be programmed to display tens of thousands of different colors. They do not require any external wiring and are suitable for use as indicator lights for system operating status, adding exciting lighting effects to projects, or as visual debugging tools during program development.

5.2.2. Features

The programmable RGB LED onboard the MATRIX Mini R4 has the following main features:

- **Full-Color Control:** Each LED is composed of three basic color light units: red, green, and blue. By controlling the brightness of these three colors from 0 to 255, thousands of colors can be mixed.
- **Independent Addressing and Control:** The two RGB LEDs are individually numbered (e.g. LED 1, LED 2) and can be programmed to have different colors and brightnesses.
- **Brightness Adjustment:** In addition to color, you can also control the brightness of the LED by adjusting the values of R, G, and B. For example, (255, 0, 0) is the brightest red, while (128, 0, 0) is a darker red.

5.2.3. Applications

The onboard RGB LEDs can be flexibly applied to various scenarios, providing visual feedback.

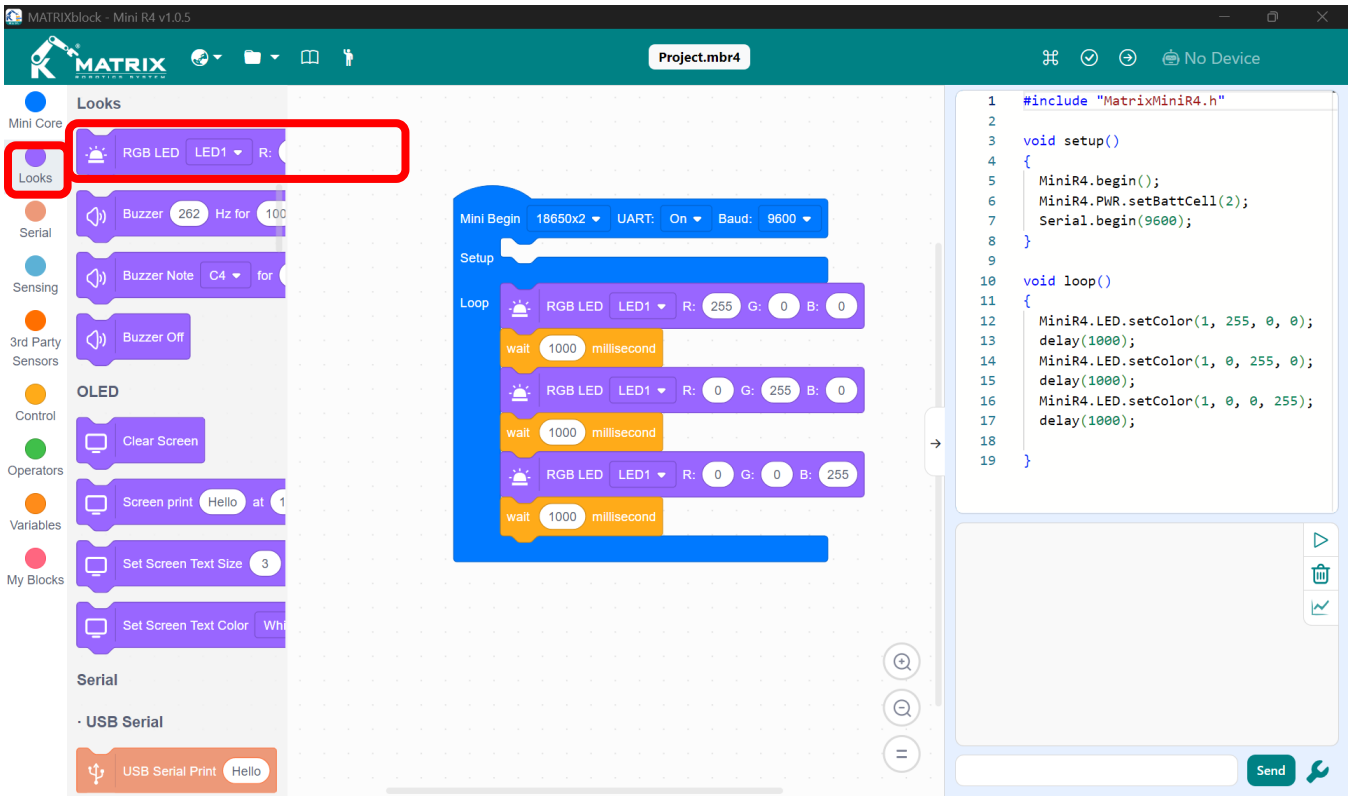
- **System status indicator:** Use different colors or flashing patterns to indicate wi-fi connection, battery level, or the robots operating mode (such as standby, running, etc.)
- **Data visualization:** Changes in sensor values (such as distance or temperature) can be presented using changes in light color or brightness.
- **Add visual lighting effects:** Create breathing lights, rainbow lights, warning flashing lights, or simulate vehicle turn and braking lights.
- **Assist program debugging:** During development, change the LED color at specific points in the program to confirm execution progress.

5.2.4. Electrical Characteristics

Parameter	Min	Typical	Max	Units
Operating voltage	-	5	-	V
Control pin	-	D7	-	-
LED red light wavelength	620	-	625	nm
LED green light wavelength	522	-	525	nm
LED blue light wavelength	465	-	467	nm

5.2.5. Sample Code & Blocks Instructions

- Sample program



MATRIXblock

IDE C++

```
1. #include "MatrixMiniR4.h"
2.
```

```
3.  void setup()
4.  {
5.    MiniR4.begin();
6.    MiniR4.PWR.setBattCell(2);
7.    Serial.begin(9600);
8.  }
9.
10. void loop()
11. {
12.   MiniR4.LED.setColor(1, 255, 0, 0);
13.   delay(1000);
14.   MiniR4.LED.setColor(1, 0, 255, 0);
15.   delay(1000);
16.   MiniR4.LED.setColor(1, 0, 0, 255);
17.   delay(1000);
18.
19. }
```

Results:

RGB LED 1 switches between red, green, and blue in sequence, with a color change interval of 1 second.



5.2.6. Principle Description

The reason why the programmable RGB LED on the MATRIX Mini R4 can display a variety of colors is because its core principle is based on the **additive color mixing of light**, which is often called **"The three primary colors of light."**

- **RGB Composition:**
  - RGB is the abbreviation for Red, Green, and Blue.
  - Each RGB LED incorporates three light emitting chips in a tiny package: one that emits red light, one that emits blue light, and one that emits green light.
  - Red, green, and blue are called the three primary colors of light because most colors in nature can be created by mixing these three colors of light in different proportions.
- **Color generation- Additive mixing of light:**
  - You can imagine three spotlights of different colors (red, green, and blue) shining simultaneously on a white wall.

- When light beams of different colors overlap, their brightness “adds” to produce different colors. This is the principle of additive color mixing.
  - **Red + Green = Yellow**
  - **Red + Blue = Magenta**
  - **Green + Blue = Cyan**
  - **Red + Green + Blue = White**
  - If none of the lights are on, it is black (no light).

- **Brightness control and color change:**
  - The controller can not only control the on/off state of the red, green, and blue light emitting chips, but also control their respective **brightness.**
  - In programs, we usually use the value of **0 (completely off)** to **255 (brightest)** to represent the brightness of each color channel.
  - By setting different brightness combinations for R, G, and B, you can mix many unique colors. For example:
    - (255, 0, 0) -> Pure red
    - (0, 255, 0) -> Pure green
    - (255, 255, 0) -> Yellow
    - (255, 128, 0) -> Orange
    - (128, 128, 128) -> Gray (medium-bright white)
- **Pulse width modulation:**
  - At the hardware level, precise brightness control is achieved via a technology called **Pulse Width Modulation (PWM).** It changes the average brightness perceived by our eyes by switching the LED power on and off very quickly and adjusting the proportion of the “on” time.
  - However, users do not need to worry about these operations. The MatrixMiniR4 library has already taken care of these low-level functions. The user only needs to provide R, G, and B values from 0-255 to easily set the desired color.

5.3. Buzzer

5.3.1. Overview

The MATRIX Mini R4 has a built-in **buzzer**. A buzzer is a component that can convert electronic signals into sound. It is the most commonly used output device to provide **auditory feedback** in projects. It does not require any external wiring and can be easily programmed to produce sounds of various tones. It is often used to make sounds that serve as alarms or reminders, prompts, or even simple melodies to add dimension to your project.

5.3.2. Features

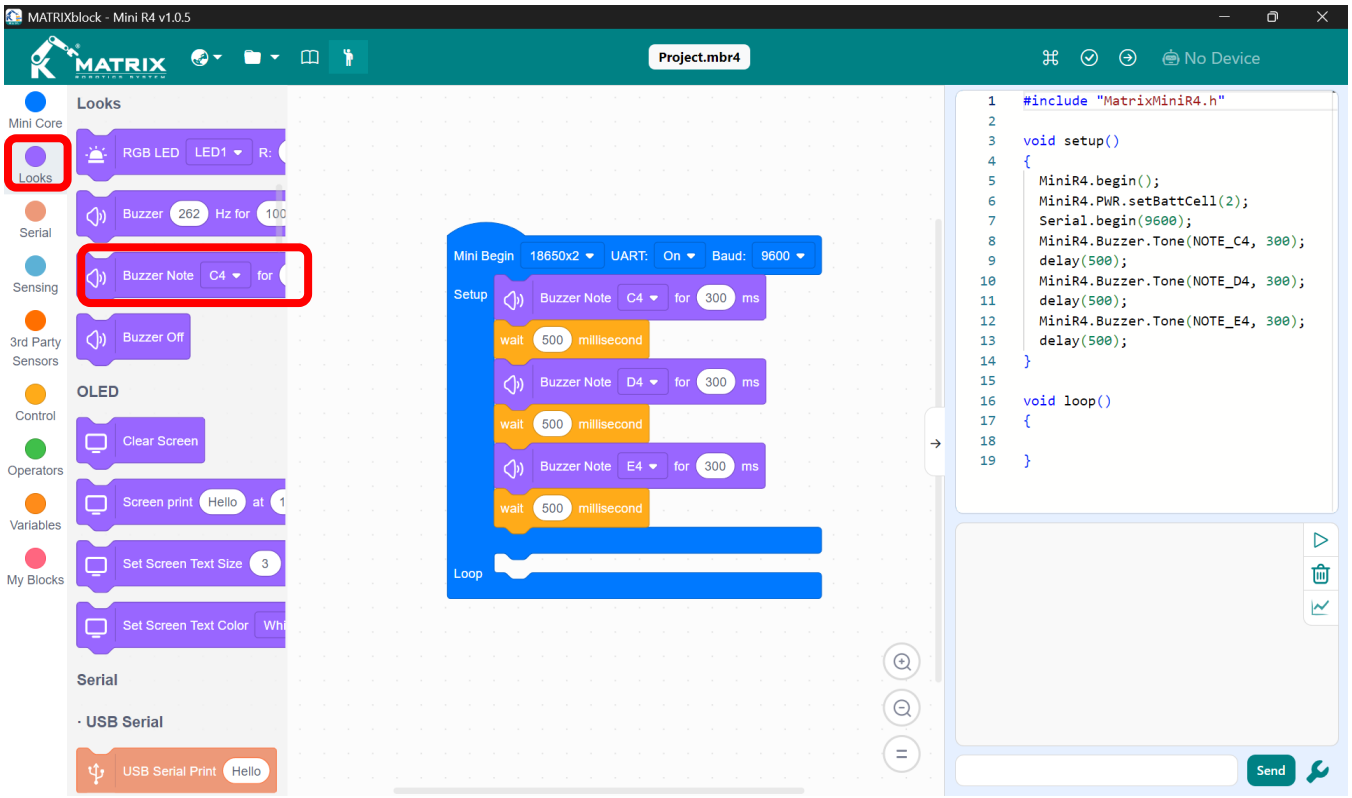
- **Programmable sound effects:** The **frequency** (pitch) and **duration** of the sound can be programmed to produce a variety of tones, alarms, or simple melodies.
- **Onboard integration:** Built into the controller, no external wiring is required for use.
- **Simple software control:** Can be easily programmed using MATRIXblock's block coding software or the MatrixMiniR4 Arduino library.

5.3.3. Applications

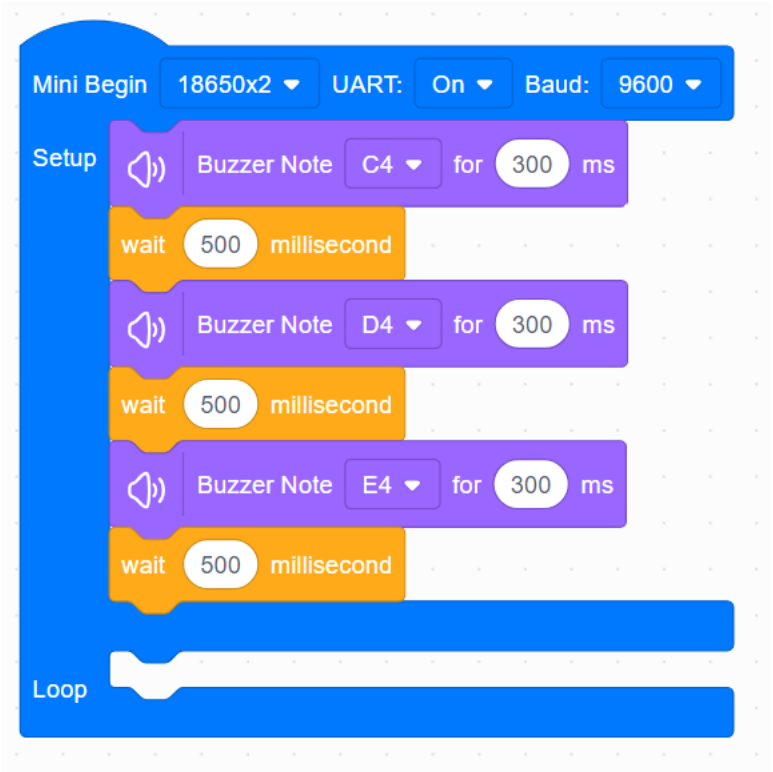
- **Operation and status prompt sounds:** For example: prompt sound for power-on completion, confirmation sound for button presses, or a reminder for task completion.
- **Alarms and warning sounds: For example: obstacle avoidance robot that sounds an alarm when it detects an obstacle, or a low-battery warning.**
- **Simple music and sound effects:** Play simple melodies or create sound effects for specific actions in your project (such as success, failure, movement.)
- **Assist program debugging: During development, different tones can be played at different points in the program, serving as an auditory tool to confirm execution progress.**

5.3.4. Sample Code & Blocks Instructions

- Example program:



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{

  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.Buzzer.Tone(NOTE_C4, 300);
  delay(500);
  MiniR4.Buzzer.Tone(NOTE_D4, 300);
  delay(500);
  MiniR4.Buzzer.Tone(NOTE_E4, 300);
  delay(500);
}

void loop()
{

}
```

Results:

The buzzer will play the notes Do, Re, and Mi in a sequence, then stop.

5.3.5. Principle Description

The reason why the MATRIX Mini R4 onboard buzzer can emit different tones through program control is because it operates as a **Passive Buzzer.**

- **Active vs. Passive buzzers:**
  - The **Active buzzer** has its own oscillation circuit inside. As long as it is powered on, it will emit a fixed “beep” sound that cannot be changed.
  - A **Passive buzzer** (such as the one on the Mini R4) has no internal oscillation circuit and acts as a mini speaker. You need to provide an electronic signal of a specific frequency to drive it, and it will produce a sound of the corresponding pitch. Because of this, it can be programmed to play different notes.
- **Piezoelectric Effect:**
  - The core of the buzzer is a **Piezoelectric Ceramic Disc**. This special material has a “piezoelectric effect”, which means that when voltage is applied to it, it will produce a tiny physical deformation (for example, bend slightly). When the voltage is removed, it will regain its original shape.
- **Sound generation (via PWM):**
  - When a the controller rapidly applies, removes, applies voltage to the piezoelectric ceramic, the ceramic vibrates back and forth at the same frequency.
  - The vibration pushes the surrounding air, creating audible sound waves.
  - The frequency set in the program is actually the switching frequency of the voltage applied by the controller to the piezoelectric piece. For example, if the frequency is set to 440Hz, the controller will switch the voltage 440 times per second, driving the

ceramic piece to vibrate 440 times per second, producing the sound of the standard pitch A4 (La).

- The controller generates this precise frequency wave signal on its corresponding pin (D6), using **PWM** to drive the buzzer.
- The tone function in the MatrixMiniR4 library has encapsulated the process of generating a PWM signal at a specific frequency, allowing the user to simply specify the desired note or frequency.

In short, the buzzer of the Mini R4 generates an electronic signal of a program-specified frequency, which causes the internal piezoelectric piece to vibrate, allowing it to emit sounds of various pitches.

5.4. OLED Screen

5.4.1. Overview

The MATRIX Mini R4 controller has a built-in **0.91 inch, 128x32 pixel OLED display.** OLED is a display technology that offers advantages such as high contrast, wide viewing angles, and fast response times. The onboard screen provides a powerful visual output interface for any project. It can display text, numbers, symbols, and even graphics in real time without any external display module. This allows the controller to intuitively display information to users.

5.4.2. Features

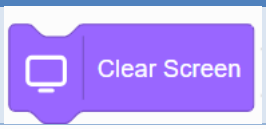
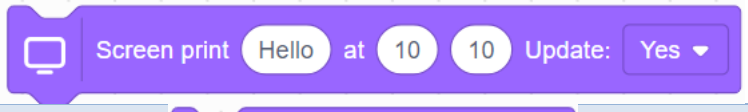
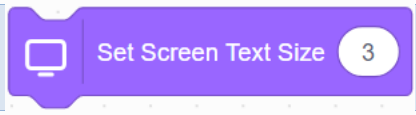
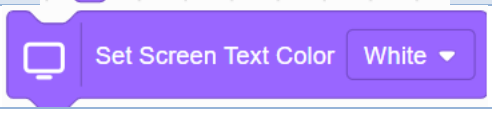
- **Display Specifications:**
  - **Size:** 0.91 inches
  - **Resolution:** 128x32 pixels
  - **Display color:** Monochrome (white light)
- **Text Display and Formatting:**
  - **Display content:** Able to display English letters, numbers, common symbols, and text strings.
  - **Text size adjustment:** Can change text size within the program.
  - **Text color setting:** Supports changing text color (black or white).
  - **Precise coordinate positioning:** Can adjust the starting pixel at which text is displayed.
- **Screen Control:**
  - **Clear screen:** Command to clear all content on the screen.
  - **Buffer and update mechanism:** First, the content to be printed is prepared in internal memory (buffer), then is displayed on the screen all at once (update). This method prevents image flickering.
- **On-Board Integration:** The OLED screen is directly integrated onto the Mini R4 control board and does not require any external wiring.

5.4.3. Applications

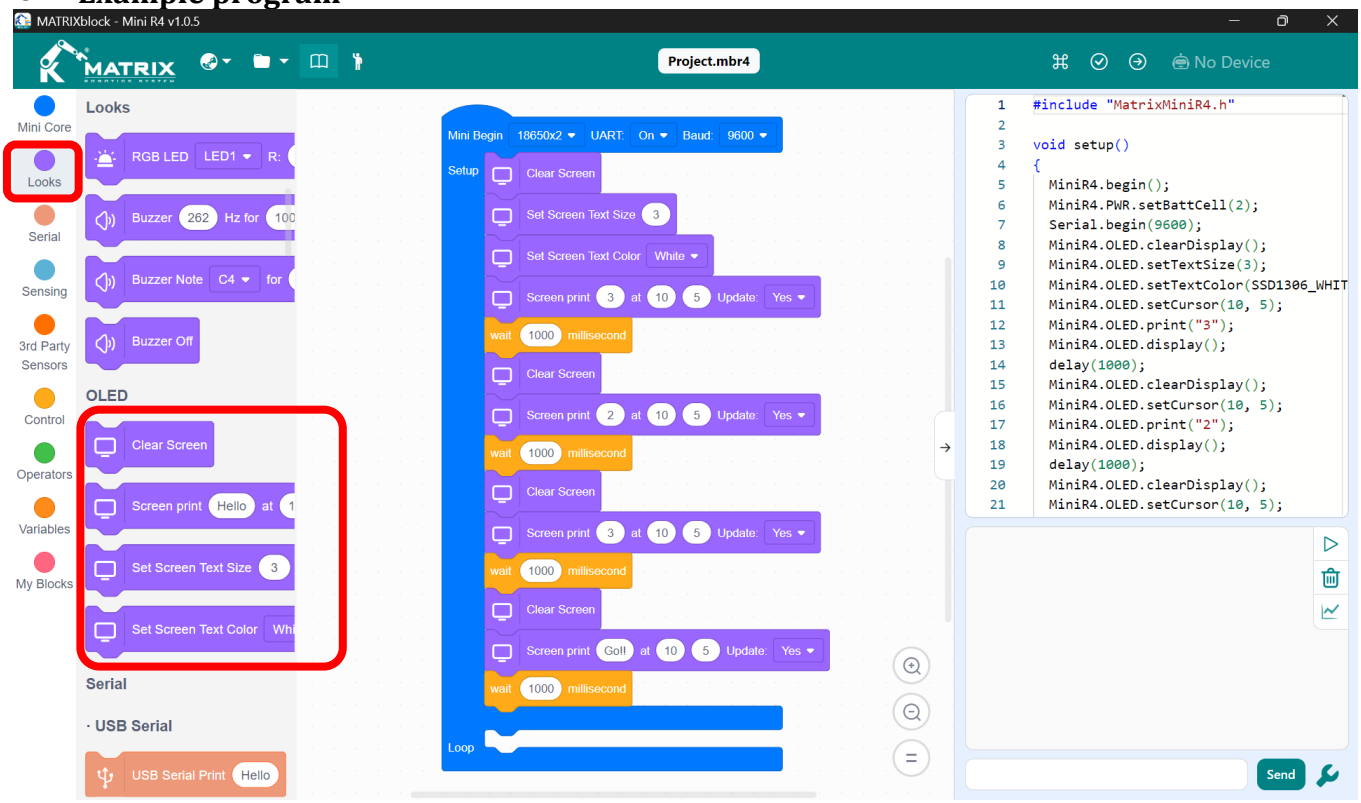
- **Display sensor values:** The values of sensors such as grayscale, soil moisture, potentiometer, or water level can be displayed in real time for convenient observation and debugging.
- **Create interactive menus:** Use onboard buttons to create simple user interfaces for selecting options or adjusting parameters.
- **Display messages:** Display text such as welcome messages, operation prompts, task completion, or error messages.

5.4.4. Sample Code & Blocks Instructions

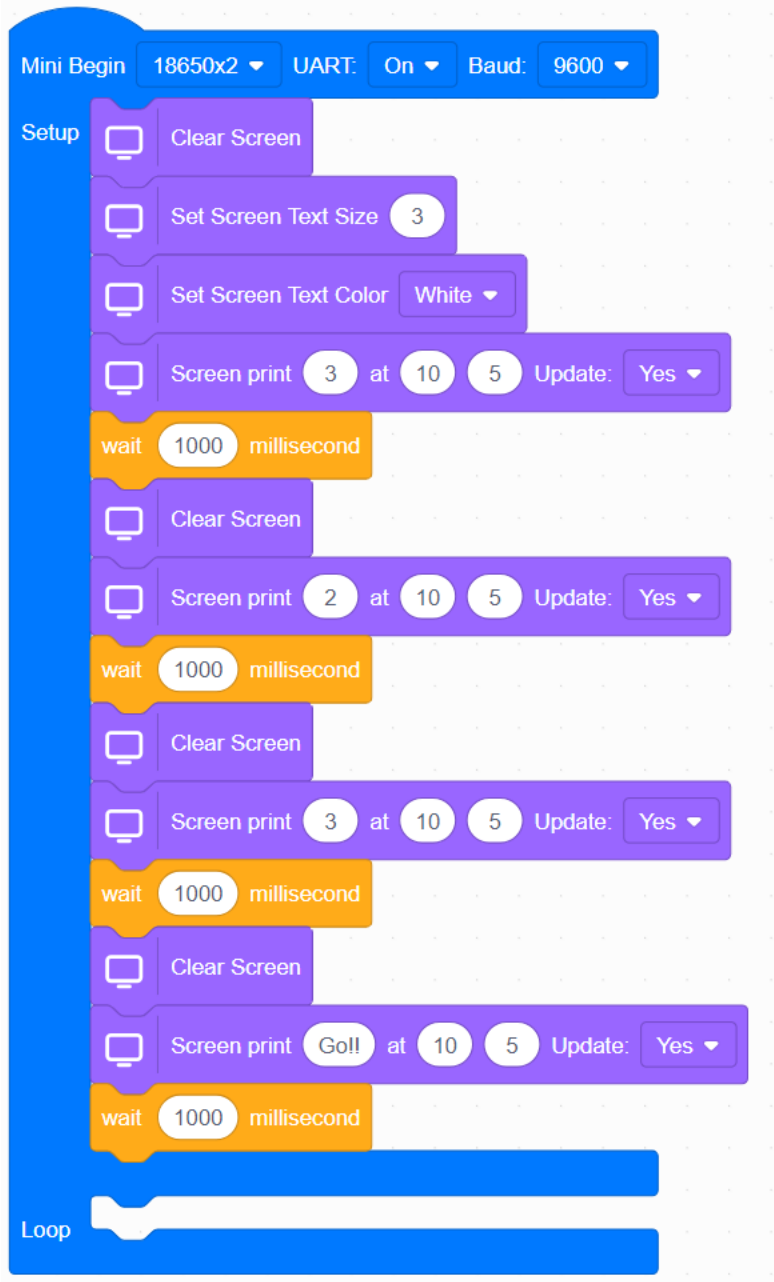
- **Block descriptions**

Block	Description
	Clear the last displayed content from the screen.
	Display a string on the OLED screen, set the text position, and mark whether or not to refresh the screen.
	Set the font size of text to be displayed.
	Set the text color.

- **Example program**



MATRIXblock



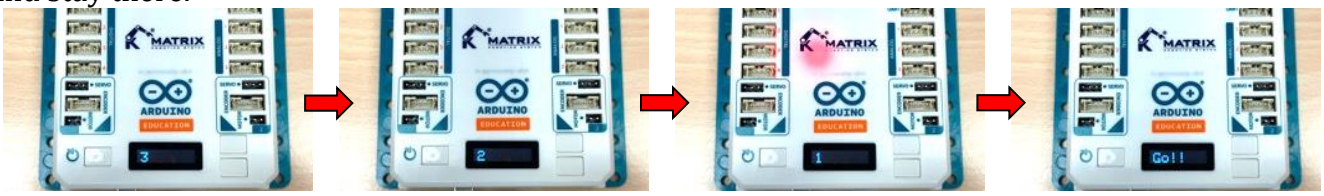
IDE C++

```cpp
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.OLED.clearDisplay();
  MiniR4.OLED.setTextSize(3);
  MiniR4.OLED.setTextColor(SSD1306_WHITE);
```

```
11.     MiniR4.OLED.setCursor(10, 5);
12.     MiniR4.OLED.print("3");
13.     MiniR4.OLED.display();
14.     delay(1000);
15.     MiniR4.OLED.clearDisplay();
16.     MiniR4.OLED.setCursor(10, 5);
17.     MiniR4.OLED.print("2");
18.     MiniR4.OLED.display();
19.     delay(1000);
20.     MiniR4.OLED.clearDisplay();
21.     MiniR4.OLED.setCursor(10, 5);
22.     MiniR4.OLED.print("1");
23.     MiniR4.OLED.display();
24.     delay(1000);
25.     MiniR4.OLED.clearDisplay();
26.     MiniR4.OLED.setCursor(10, 5);
27.     MiniR4.OLED.print("Go!!");
28.     MiniR4.OLED.display();
29.     delay(1000);
30. }
31.
32. void loop()
33. {
34.
35. }
```

Results:

The OLED screen will display “3”, “2”, “1”, waiting one second between each, then display “Go!!” and stay there.



5.4.5. Principle Description

The reason why the OLED screen onboard the MATRIX Mini R4 can display clear text and images is because of its unique self-luminating technology and pixel control principle.

- **OLED and its self-luminescence:**
  - OLED’s full name is **Organic Light Emitting Diode**.
  - Its core principle is “self-luminescence”. This means that each pixel that makes up the screen is itself a tiny organic-material bulb that can emit light independently, unlike traditional LCD screens that require an additional backlight module to illuminate.
  - **Advantages:** Since there is no need for a backlight layer, OLED screens can be made thinner and more power efficient. Because non-luminous pixels can reach pure

black, they can achieve extremely high contrast, making the picture look clearer and sharper.

- **Pixel Matrix:**
  - The OLED screen on the Mini R4 has a resolution of **128x32 pixels**, which means that the entire screen is made up of a matrix of tiny pixels that is **128 columns (horizontally)** and **32 rows (vertically).**
  - You can think of it as a light panel made up of 4096 (128x32) tiny lightbulbs packed together. By precisely controlling the on/off states of each "lightbulb", various words, numbers, and patterns can be created.
- **Buffer and Display Mechanism:**
  - In the program, we don't directly turn individual pixels on and off. Instead, the MatrixMiniR4 library will create a **"Screen buffer"** in the memory of the controller (Arduino UNO R4 WiFi). This buffer is like a virtual canvas whose size directly corresponds to the size of the 128x32 screen.
  - When you use commands such as print(), setTextSize(), setCursor(), etc., the program will first "draw" the corresponding pixel information in the memory buffer. This process is invisible to the user.
  - Finally, when you call the **display()** command, the controller will send the entire buffer data to the OLED screen driver chip at once through the internal communication interface (usually **I^2C**). The driver chip will then light up or turn off the corresponding pixels based on this data to fully display the image you prepared. This mechanism of first drawing behind the scenes, then presenting, will effectively prevent screen flickering.

Through these principles, the OLED screen provides an efficient and high-quality visual output method for the MATRIX Mini R4.

6. Motion Modules

6.1. Overview

If the input modules are the robot's "senses", then the motion modules are its "muscles and joints", enabling physical movement. They convert electronic commands from the Matrix Mini R4 controller into actions like wheel rotations, arm swings, or claw gripping. This chapter introduces several core motion modules used with the MATRIX Mini R4: the TT Motor, Encoder TT Motor, and the RC Servo.

Each module will be discussed in terms of: overview, features, applications, electrical characteristics, connection pinouts, sample code, and operational principles.

6.2. TT Motor

6.2.1. Overview

The TT Motor is a low-cost DC motor with a reduction gearbox. It is one of the most common power components in education, maker, and introductory robotics projects because of its affordable price, moderate torque, and easy installation. It is usually used to power the wheels of wheeled robots, allowing them to move forward and backward, and turn. Through the dedicated motor interface on the MATRIX Mini R4 controller, the speed and direction of the TT motor can easily be controlled.

6.2.2. Feature

- Gearbox reduces speed to around 130 rpm at 4.5V and increases torque.
- Dual-shaft design allows flexible mechanical configurations.
- Standard dimensions fit many wheel types.
- Controlled via PWM through MATRIX Mini R4 motor ports.

6.2.3. Application

- Two-wheeled robots
- Sumo robots
- Tracked vehicles
- Conveyor belts or robotic arms
- Teaching motor control principles

6.2.4. Electrical Characteristics

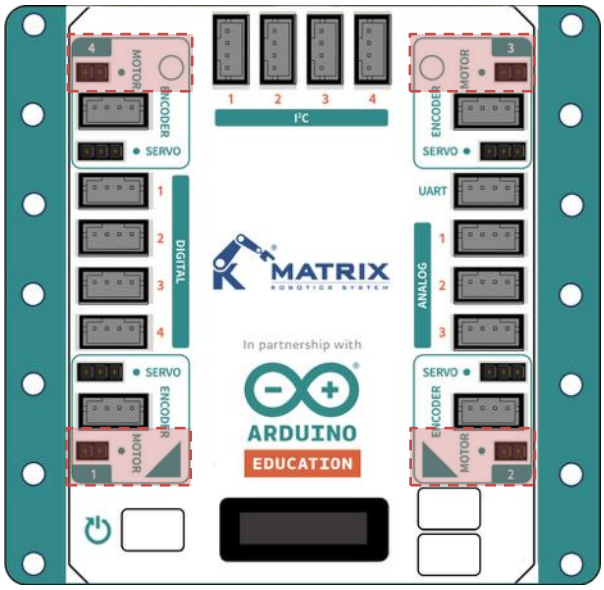
Parameter	Min	Typ	Max	Unit
Voltage	3	4.5	6	VDC
Current	-	0.3	-	A

6.2.5. Pinout

Line Color	Interface pins on Mini R4	Description
Black line	Motor Negative (M-)	Negative power input terminal of the motor
White line	Motor Positive (M+)	Positive power input terminal of the motor

6.2.6. Sample Code & Blocks Instructions

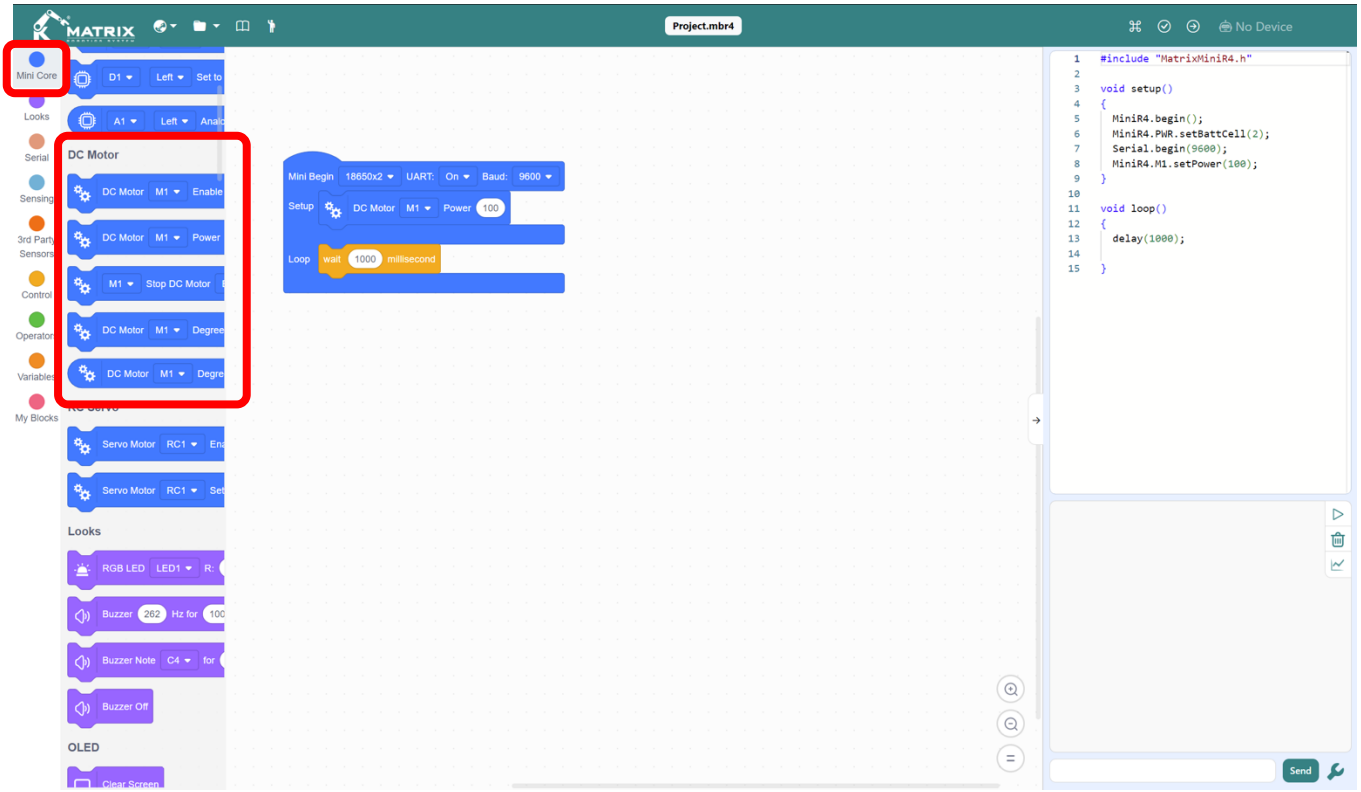
- **Hardware connection:** Please connect the TT motor to the Motor 1 port on the lower left of the MATRIX Mini R4 controller, with the black line close to the dot.



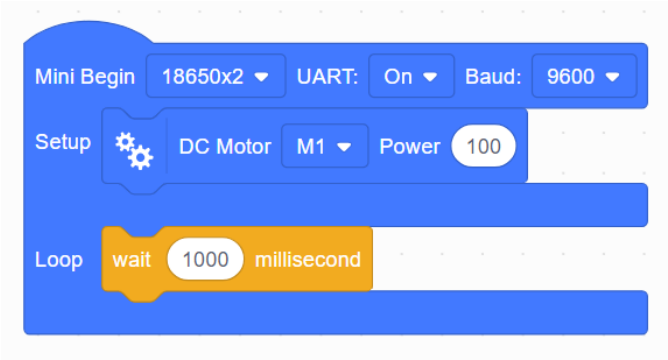
MOTOR 1&4 MOTOR 2&3

Pinout-Motor Out			
No.	Name	I/O	Description
1	M-	Output	H-bridge Output M- Terminal
2	M+	Output	H-bridge Output M+ Terminal

- **Sample Code:**



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
  MiniR4.M1.setPower(100);
```

```
    delay(1000);
    MiniR4.M1.setBrake(true);
}

void loop()
{

}
```

Result: Motor runs forward for 1 second, then brakes.

6.2.7. Principle Description

The operating principle of the MATRIX TT motor is mainly based on the internal DC motor and reduction gearbox, and the MATRIX Mini R4 controller provides precise drive signals.

- **Motor Rotation Principle:** The core of the motor is “electromagnetism”. When current passes through the coil inside the motor, an electromagnetic field is generated. This electromagnetic field interacts with the fixed permanent magnet to generate thrust, driving the motor to rotate continuously.
- **The Reduction Gearbox:** The motor itself has a very high speed but insufficient power (torque). The gearbox converts the high speed into a lower output speed through the transmission of multiple sets of gears, and at the same time amplifies the torque to make it sufficient to drive the robot’s wheels. The 1:55 reduction ratio of this TT motor means that the motor rotates 55 times internally and the output shaft rotates 1 time.
- **Mini R4 control method:**
    - **Direction Control:** The controller changes the direction of the current flowing to the motor M+ and M- pins through the built-in H-bridge drive circuit, thereby controlling the forward and reverse rotation of the motor.
    - **Speed Control:** The controller uses Pulse Width modulation (PWM) technology to control the average power of the motor by quickly switching the power supplied to the motor and adjusting the proportion of time the power is “on” (duty cycle), thereby changing the speed.

6.3. Encoder TT Motor

6.3.1. Overview

The Encoder TT Motor builds on the standard TT motor by integrating a rotary encoder, allowing it to not only provide power but also send feedback to the controller about its actual movement. This feedback lets the controller monitor how fast and how far the motor turns, and in which direction.

Unlike standard TT motors that rely on open-loop control (sending commands without knowing the result), the encoder enables closed-loop control – adjusting performance in real time based on feedback. This results in more accurate and reliable motion, making the encoder motor ideal for projects that require precise distance control, constant speed, or multi-motor synchronization.

6.3.2. Feature

- Metal gears and dual shafts
- Integrated encoder sends A/B phase pulse signals
- Measures rotation speed, angle, and direction
- Enables precise distance control, speed regulation, and synchronization
- Uses 6-pin (PH2.0-6P) connector for power and signals

6.3.3. Application

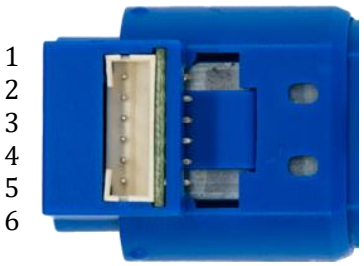
Due to its precise and controllable characteristics, encoder TT motors are particularly suitable for the following projects:

- Precise-motion robots (drawing bots, maze solvers)
- Multi-motor synchronized projects
- Speed-regulated robots
- Movement recording and playback

6.3.4. Electrical Characteristics

Parameter	Min	Typ	Max	Unit
Motor Voltage	1.8	6	-	VDC
Encoder Voltage	3.3	-	5	VDC
No Load Current	-	0.47 @ 6V	-	A
Rated Load Current	-	1.31 @ 6V	-	A
Stall Current	-	-	3.4 @ 6V	A
Rated Speed	-	163 rpm @6V	-	rpm
Rated Torque	-	1.5	-	kg*cm
Stall Torque	-	4.55	-	kg*cm

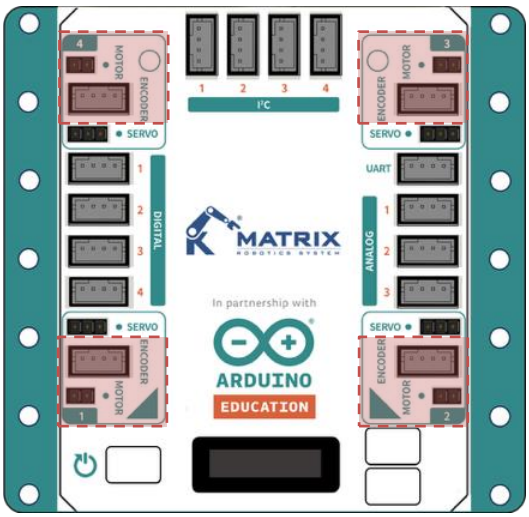
6.3.5. Pinout



Pin	Pin Name	Description
1	M-	Motor negative power input
2	M+	Motor positive power input
3	HB	Encoder B-phase signal output
4	HA	Encoder A-phase signal output
5	VCC	Encoder positive power input
6	GND	Common Ground

6.3.6. Sample Code & Blocks Instructions

- **Hardware connection:** Please connect the encoder TT motor to the Motor 1 and Encoder ports on the lower left of the MATRIX Mini R4 controller, with the black line close to the dot.



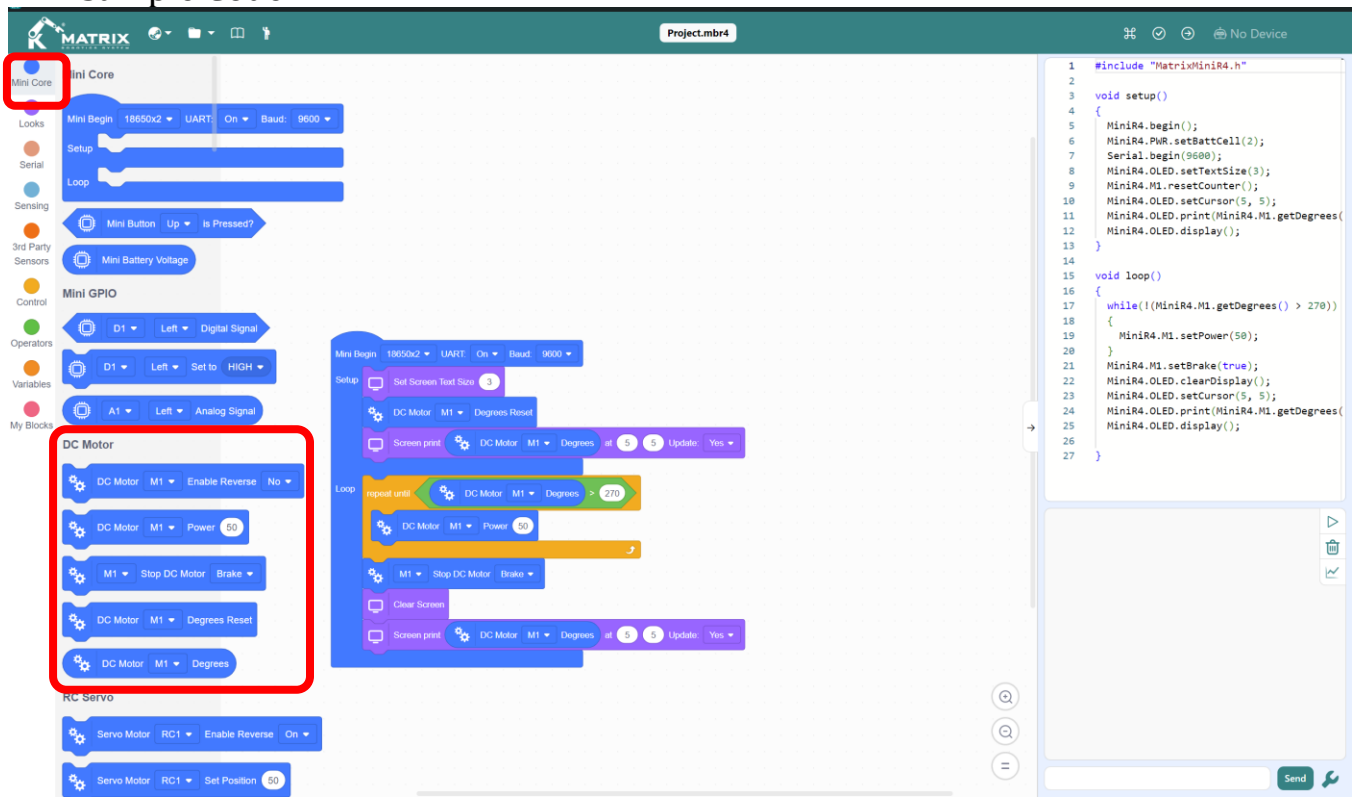
MOTOR 1&4

MOTOR 2&3

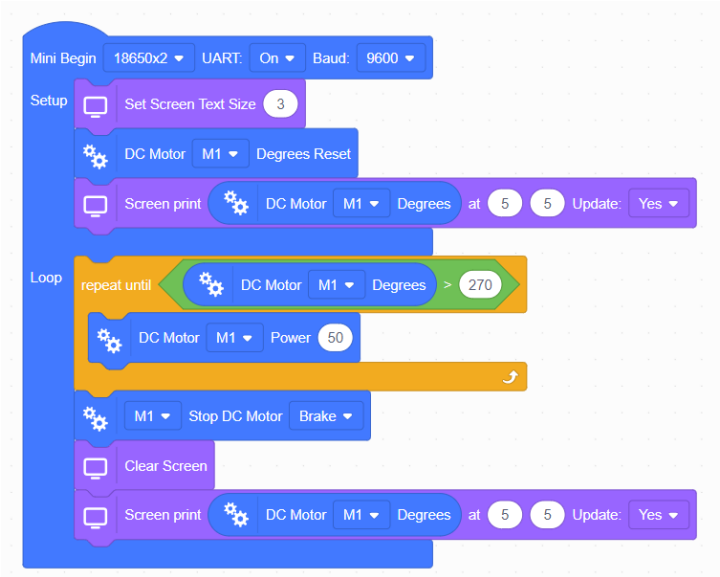
Pinout-Motor Out			
No.	Name	I/O	Description
1	M-	Output	H-bridge Output M- Terminal
2	M+	Output	H-bridge Output M+ Terminal

Pinout-Encoder			
No.	Name	I/O	Description
1	CH A	Input	Encoder A Phase Input
2	CH B	Input	Encoder B Phase Input
3	M5V	Output	Power Supply V output
4	GND	-	Power Supply Ground

- Sample Code:



MATRIXblock



IDE C++

- #include "MatrixMiniR4.h"

```
1.
2.  void setup()
3.  {
4.    MiniR4.begin();
5.    MiniR4.PWR.setBattCell(2);
```

```
    Serial.begin(9600);
    MiniR4.OLED.setTextSize(3);
    MiniR4.M1.resetCounter();
    MiniR4.OLED.setCursor(5, 5);
    MiniR4.OLED.print(MiniR4.M1.getDegrees());
    MiniR4.OLED.display();
    while(!(MiniR4.M1.getDegrees() > 270))
    {
       MiniR4.M1.setPower(50);
    }
    MiniR4.M1.setBrake(true);
    MiniR4.OLED.clearDisplay();
    MiniR4.OLED.setCursor(5, 5);
    MiniR4.OLED.print(MiniR4.M1.getDegrees());
    MiniR4.OLED.display();
}

void loop()
{

}
```

Result: Rotates until 270 degrees, then stops and displays the angle.

6.3.7. Principle Description

The Encoder TT Motor operates by combining the driving principles of a standard DC motor with the feedback capabilities of a rotary encoder to achieve closed-loop control.

- **Power Generation (DC Motor Component):** Like a regular TT motor, it works on the principle of electromagnetism—when an electric current flows through the internal coil, it generates a magnetic field that interacts with permanent magnets, producing a force that drives the motor to rotate. The MATRIX Mini R4 controller manages the rotation direction using an H-bridge circuit and adjusts the speed using PWM (Pulse Width Modulation) signals.
- **Motion Feedback (Rotary Encoder Component):** As the motor spins, it also turns a code disc attached to the back of the motor shaft. The encoder detects changes in the markings on this disc and generates two pulse signals (A and B channels) with a phase difference.
    - The controller (specifically, the STM32 co-processor in the Mini R4) reads the pulse count, frequency, and phase relationship of these signals to accurately calculate the motor's speed, rotation angle, and direction of movement.
- **Closed-loop Control Implementation:** With this feedback, the controller can compare the target movement (e.g., rotate 270 degrees) to the actual movement reported by the encoder (e.g., 250 degrees). Based on this comparison, it dynamically adjusts the power signal sent to the motor—such as continuing to drive it—until the actual position matches the target precisely.

- This forms the essence of closed-loop control, which ensures the motor does not operate blindly but instead executes instructions with precision and oversight, adapting in real time to changes like friction or load.

6.4. RC Servo

6.4.1. Overview

RC (Radio Control Servo Motor) is a motor for precise angle control, not continuous rotation. In the Mini R4, this is used in the arms, joints, steering, camera mounts, etc.

6.4.2. Features

- Controlled via PWM to hold specific angles (e.g., 0-180 degrees)
- Internal closed-loop system (with motor, gears, position sensor, controller)
- Can resist external forces to hold position

6.4.3. Applications

- Robotic arms
- Wheel steering
- Grippers
- Pan-tilt mechanisms
- Biomechanical robots (e.g., insect legs)
- Rudders on model planes/boats

6.4.4. Electrical Characteristics

Parameters	Min	Typ	Max	Unit
Operating Voltage	4.8	-	6	VDC
Rotation Angle	-	180 (±10)	-	°
Operating Speed @ 6.0V	-	-	0.09	sec/60°
Stall Torque @ 6.0V	-	-	2	kg·cm
Operating Current @ 6.0V	-	550 (±50)	-	mA
Stall Current @ 6.0V	-	750 (±75)	-	mA
Standby Current	5	-	6	mA

Note: Specs above are based on 6V operation. At 5V (Mini R4 output), values will be slightly lower.

Important notes:

- **Voltage compatibility:** This servo motor operates within a voltage range of 4.8V to 6.0V, making it fully compatible with the 5V power supply provided by the MATRIX Mini R4 servo motor interface.
- **Current considerations:** The stall current of this servo motor is approximately 0.9A (900mA). Each servo motor port on the Mini R4 can supply up to 1A, which is sufficient for a single motor. However, when driving multiple servos under heavy load simultaneously, it is important to ensure that the total output capacity of the external power supply is adequate.
- **Torque and speed:** The torque and speed values in the table above are measured at 6.0V. When powered by the Mini R4's 5V output, the actual torque and speed will be slightly lower, which is expected and normal.

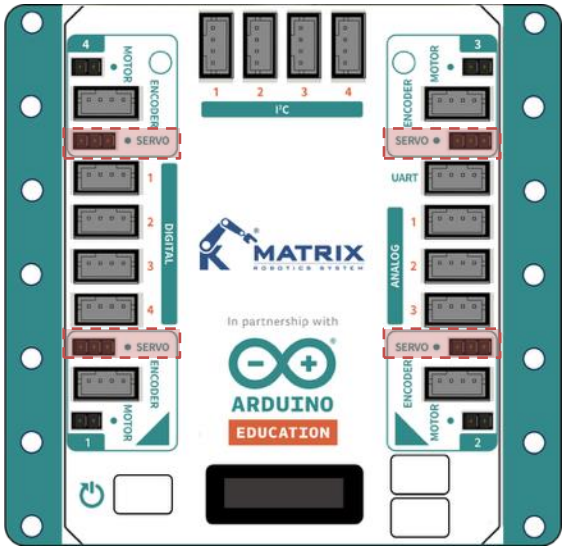
6.4.5. Pinout Definition

RC servo motors usually use a standard 3-pin connector to connect the controller, which includes power, ground, and a control signal.

Common Wire Color	Name	Description
Black or Brown	GND (Ground)	Power ground connection.
Red	VCC (Power)	Positive power input for the servo motor, supplied by the Mini R4 (5V).
White, Yellow, or Orange	Signal	Receives the PWM control signal from the controller to set the angle.

6.4.6. Sample Code & Blocks Instructions

- **Hardware Connection:** Connect the servo motor to the Servo 1 port on the bottom-left corner of the MATRIX Mini R4 controller. Ensure the ground wire (brown) is placed closest to the small dot on the connector.

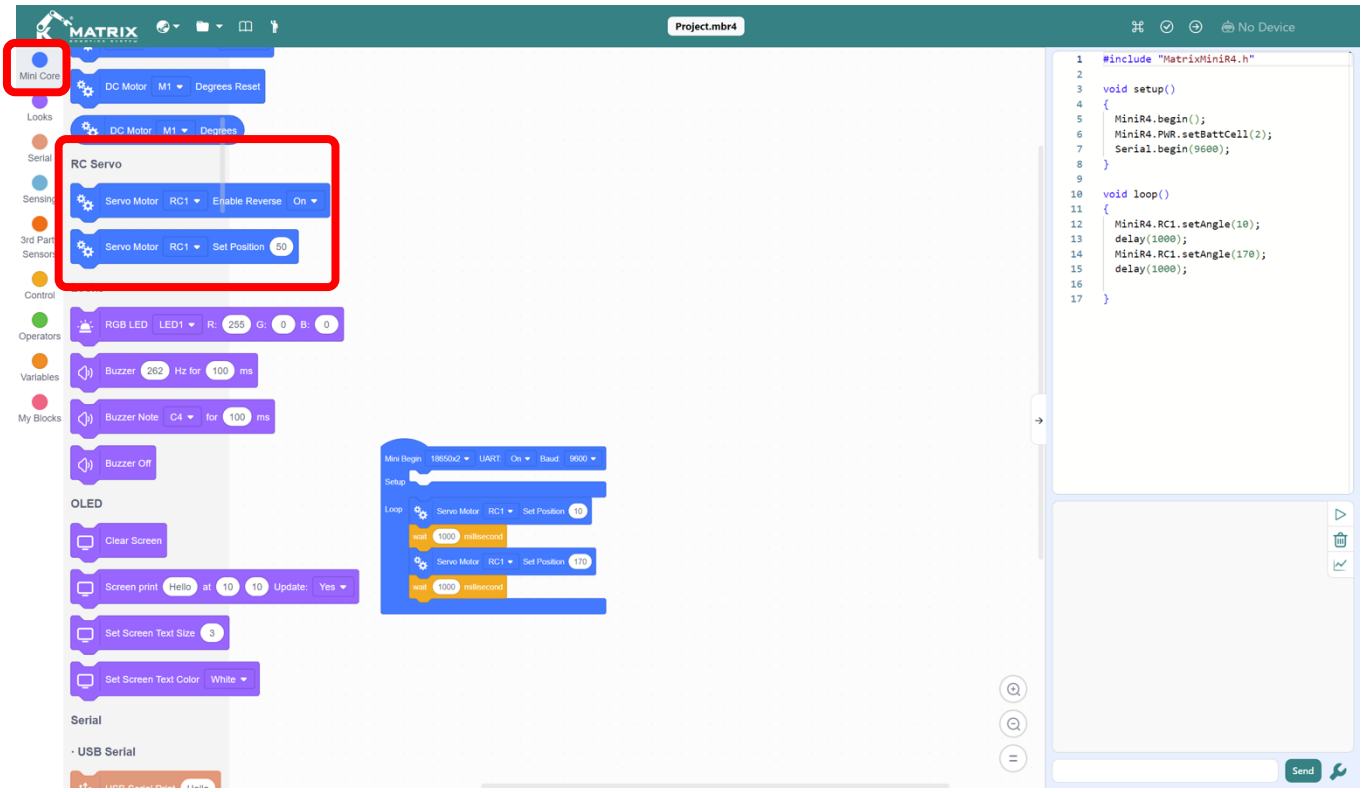


SERVO 1&4

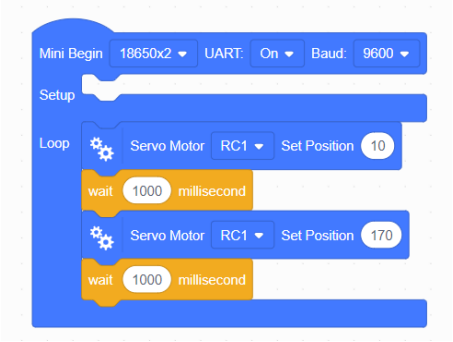
SERVO 2&3

Pinout-Servo Out			
No.	Name	I/O	Description
1	GND	-	Ground
2	5V	Output	Power supply output (5V)
3	PWM	Output	PWM signal output for RC servo

- **Sample Program:**



MATRIXblock



IDE C++

```
#include "MatrixMiniR4.h"

void setup()
{
  MiniR4.begin();
  MiniR4.PWR.setBattCell(2);
  Serial.begin(9600);
}

void loop()
{
  MiniR4.RC1.setAngle(10);
  delay(1000);
  MiniR4.RC1.setAngle(170);
  delay(1000);

}
```

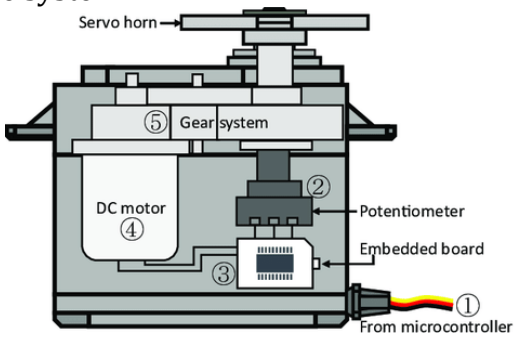
Result:

The servo motor connected to the RC1 port will continuously alternate between 10 degrees and 170 degrees, switching once per second.

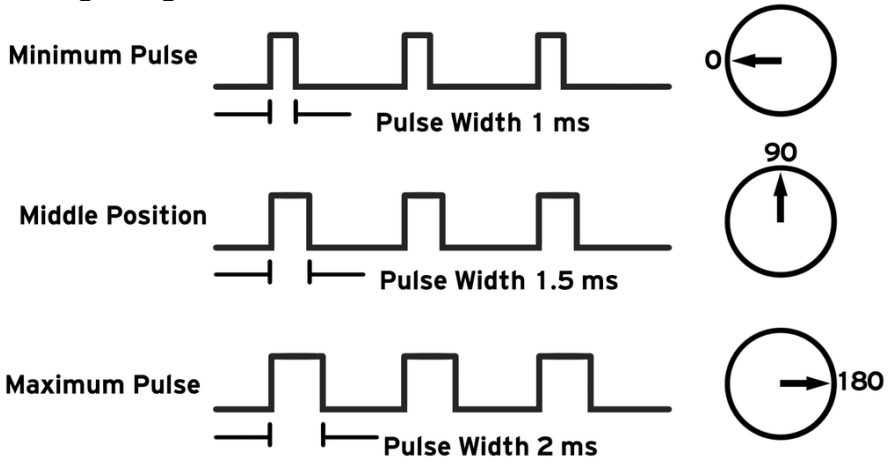
6.4.7. Principle Description

The reason an RC servo motor can accurately control and maintain a specific angle is that it contains a fully integrated closed-loop control system.

- **Internal structure**: Inside the servo motor is a combination of a DC motor, a gear reduction gearbox, a position sensor (usually a variable resistor), and a control circuit that acts as the "brain" of the system.



- **Control Principle**：
  1. **Receiving Commands:** The controller (such as the Mini R4) sends a PWM (Pulse Width Modulation) signal to the servo motor. The width of the pulse corresponds to the target angle.



  2. **Comparison and Correction:** The servo motor's internal control circuit continuously compares the target angle with the current angle reported by the position sensor.
  3. **Driving and Holding:** If the two angles are not the same, the control circuit drives the internal motor to rotate until the current angle matches the target angle. It then continues to apply force to resist external forces and hold the motor shaft in place.

In summary: The servo motor achieves precise angle control and stable positioning by constantly comparing the desired angle with the current angle and making corrections as needed.

6.5. DriveDC Chassis Control

6.5.1. Overview

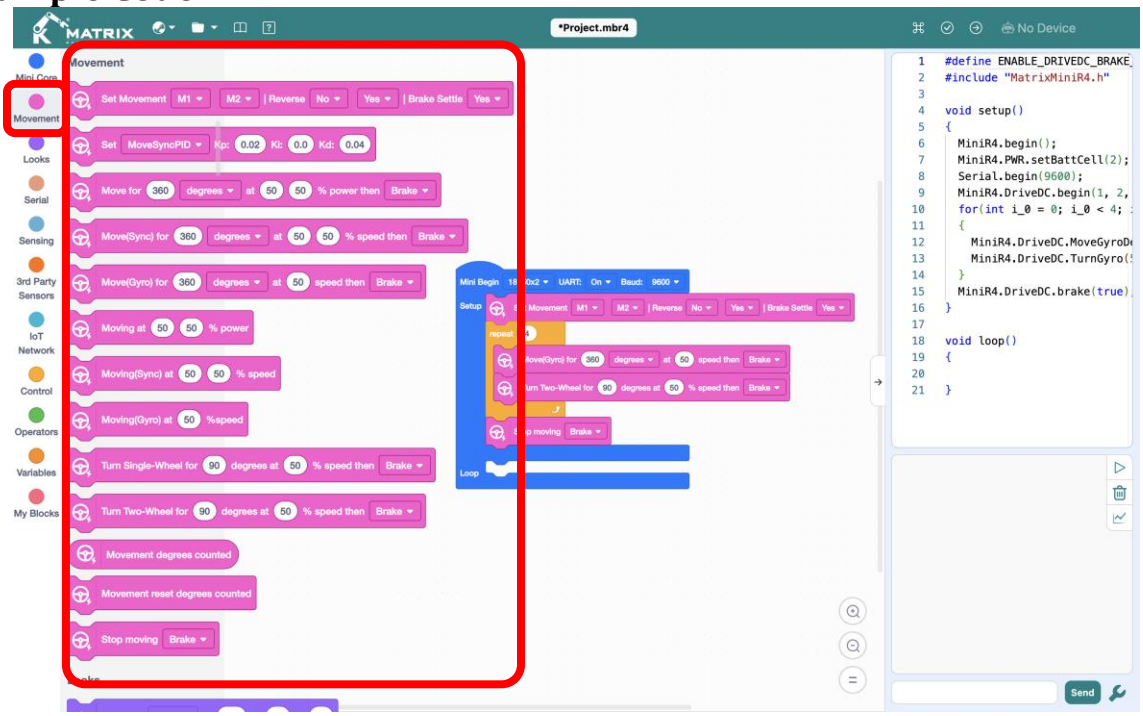
In robotics competitions, precise chassis control is the key to achieving outstanding results. Whether it is the stability of straight-line driving, the accuracy of turns, or the execution efficiency of complex routes, these factors directly impact competition performance. The **DriveDC** class of the Matrix Mini R4 is specifically designed to address these challenges. It encapsulates complex technologies such as motor control, IMU (Inertial Measurement Unit) attitude correction, and PID regulation into simple and easy-to-use functions. This allows participants to focus on strategy development rather than low-level hardware control.

6.5.2. Features

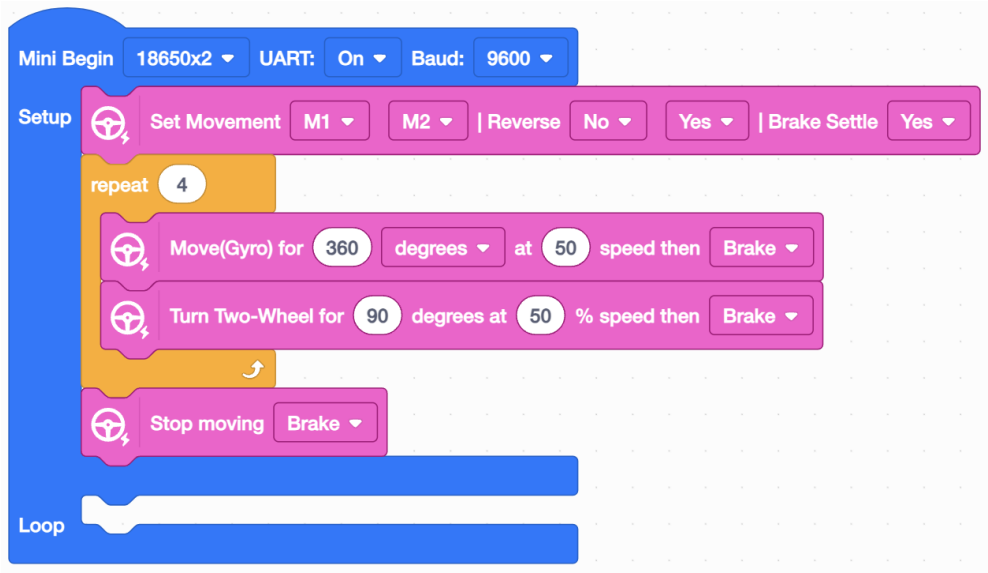
- **Rapid Chassis and Motor Setup:** Provides a simple API where chassis initialization is completed by merely setting the left and right motor assignments and their rotation directions. If necessary, parameters such as motor RPM, PPR (Pulses Per Revolution), and movement PID can also be individually configured.
- **Multiple Algorithm Assistance:** This module offers various algorithms, including basic chassis movement (no algorithm), motor synchronization auxiliary control, IMU gyroscope-assisted straight driving, and IMU gyroscope-assisted turning attitude control.
- **Preset Execution Conditions:** To simplify programming complexity, common movement conditions—including duration (seconds), movement distance (degrees/rotations), and continuous movement—are encapsulated and ready to be called directly.
- **High Response Speed:** Compared to manual PID encapsulation or individual motor command calls, this module benefits from low-level calculations performed by the STM32 co-processor, ensuring superior control responsiveness.

6.5.3. Sample Code & Blocks Instructions

- **Sample Code:**



MATRIXblock



IDE C++

```
#define ENABLE_DRIVEDC_BRAKE_DELAY
#include "MatrixMiniR4.h"

void setup()
{
MiniR4.begin();
MiniR4.PWR.setBattCell(2);
Serial.begin(9600);
MiniR4.DriveDC.begin(1, 2, false, true);
for(int i_0 = 0; i_0 < 4; i_0++)
{
MiniR4.DriveDC.MoveGyroDegs(50, 0, 360, true);
MiniR4.DriveDC.TurnGyro(50, 90, 1, true);
}
MiniR4.DriveDC.brake(true);
}

void loop()
{

}
```

7. IoT Communication

7.1. Overview

The **Internet of Things (IoT)** refers to a technical architecture that connects various physical devices through a network, enabling them to communicate with each other, share data, and perform intelligent control tasks. The **MATRIX Mini R4** controller features powerful built-in wireless communication capabilities, including **Wi-Fi** and **Bluetooth Low Energy (BLE)**. These features allow the controller to easily connect with cloud platforms, mobile devices, or other smart equipment, bringing IoT capabilities to your robotics projects.

This chapter will introduce how to use the Mini R4 for **Wi-Fi connectivity**, **MQTT protocol communication**, and **BLE communication**. Through these technologies, you can implement various applications such as remote monitoring, data uploading, and wireless remote control, seamlessly integrating your robotics projects into the modern IoT ecosystem.

7.2. Wi-Fi and MQTT Communication

7.2.1. Overview

Wi-Fi is a wireless communication technology that allows devices to connect to a network via radio waves without the need for physical cables, enabling devices to move freely while sending and receiving data. The core of the MATRIX R4, the **Arduino UNO R4 WiFi**, includes an integrated wireless communication module that natively supports both Wi-Fi and Bluetooth. This makes communication simple and accessible, providing a significant advantage for developing IoT-themed projects.

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol specifically designed for IoT applications. It utilizes a **"Publish/Subscribe"** model, allowing devices to exchange messages efficiently over a network. Once the MATRIX Mini R4 is connected to the network via its built-in Wi-Fi, it can use the MQTT protocol to communicate with cloud platforms or other devices.

Using MQTT, your robot can publish sensor data to the cloud, subscribe to control commands from other devices, or interact in real-time with mobile applications, achieving true IoT functionality.

7.2.2. Features

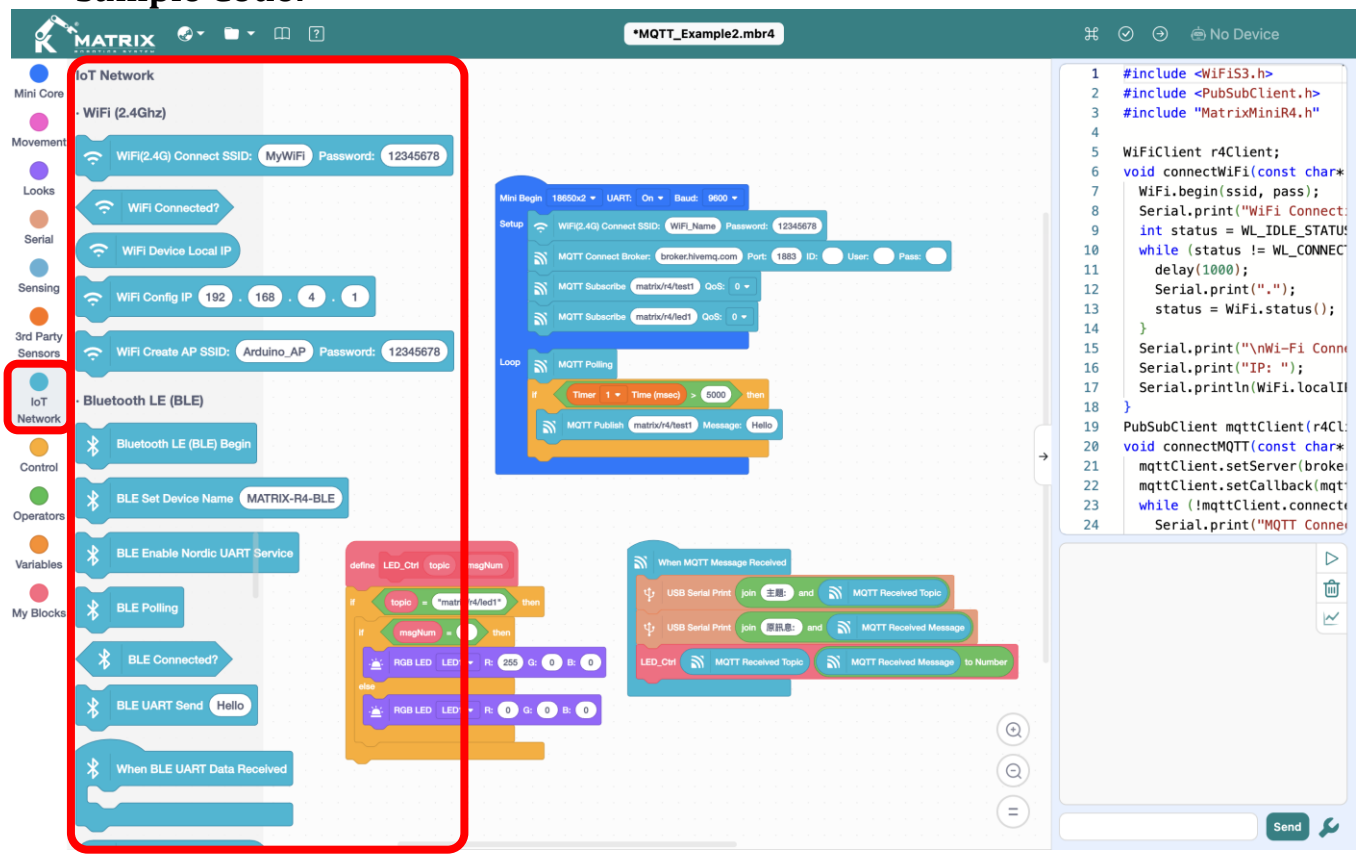
- **Wi-Fi Connectivity:** Supports 2.4GHz Wi-Fi networks, allowing connection to home or school wireless environments.
- **MQTT Protocol Support:** Implements a lightweight publish/subscribe communication model suitable for low-bandwidth or high-latency network environments.
- **Topic Subscription and Publication:** Capable of subscribing to multiple topics to receive messages and publishing messages to specific topics.
- **Non-blocking Design:** Utilizes an event-driven approach to process messages without blocking the execution of the main program.

7.2.3. Application

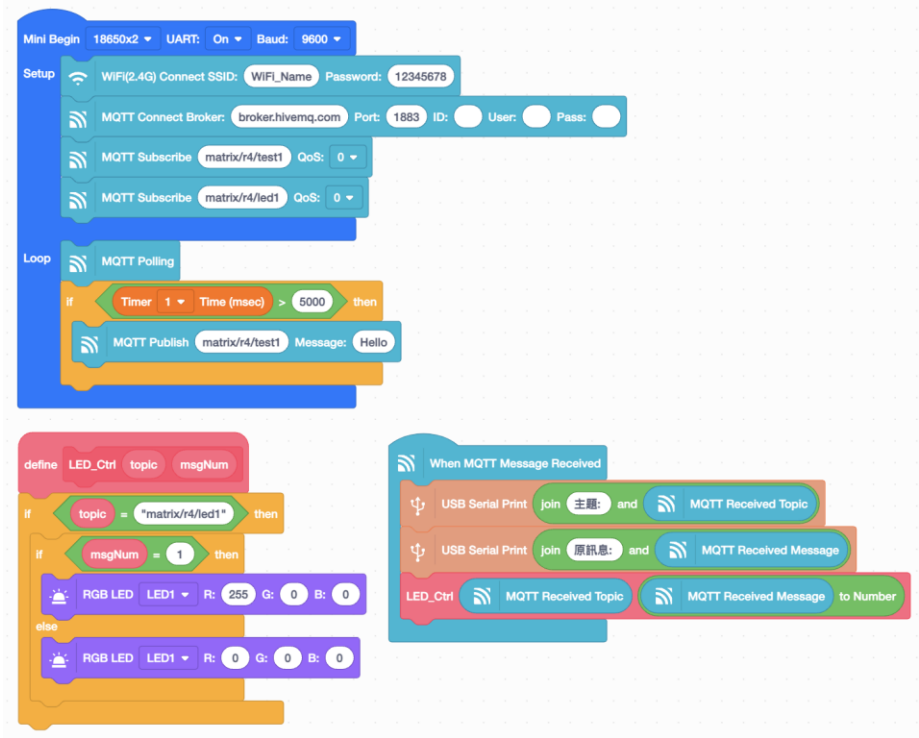
- **Remote Monitoring:** Monitor the robot's status remotely by uploading sensor data to cloud dashboards.
- **Remote Control:** Control the robot's movements remotely via mobile applications or web interfaces.
- **Multi-Robot Coordination:** Enable collaborative communication and task allocation between multiple robots.

7.2.4. Sample Code & Blocks

- **Sample Code:**



MATRIXblock



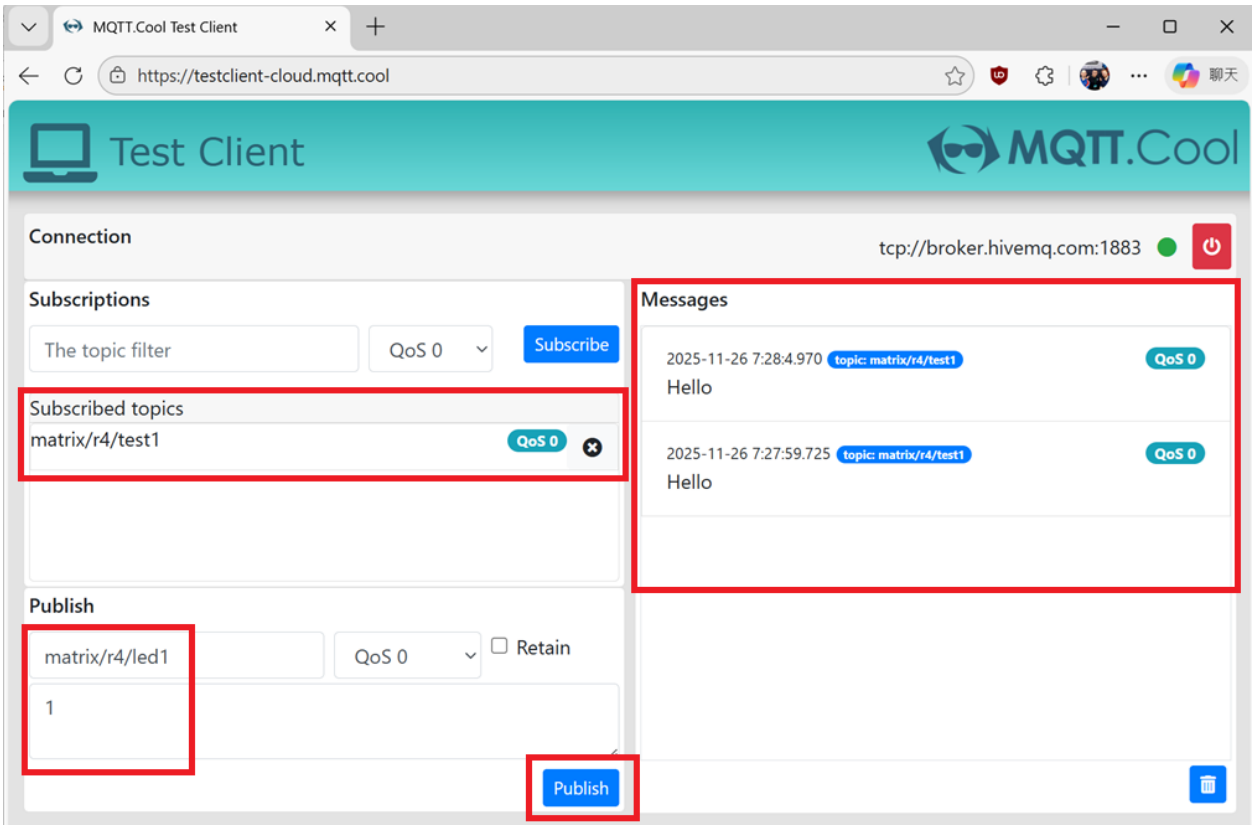
IDE C++

```
#include <WiFiS3.h>
#include <PubSubClient.h>
#include "MatrixMiniR4.h"

WiFiClient r4Client;
void connectWiFi(const char* ssid, const char* pass) {
WiFi.begin(ssid, pass);
Serial.print("WiFi Connection..");
int status = WL_IDLE_STATUS;
while (status != WL_CONNECTED) {
delay(1000);
Serial.print(".");
status = WiFi.status();
}
Serial.print("\nWi-Fi Connected! ");
Serial.print("IP: ");
Serial.println(WiFi.localIP());
}
PubSubClient mqttClient(r4Client);
void connectMQTT(const char* broker, int port, const char* id, const char* user, const
char* pass) {
mqttClient.setServer(broker, port);
mqttClient.setCallback(mqttCallback);
while (!mqttClient.connected()) {
Serial.print("MQTT Connection..");
if (!mqttClient.connect(id, user, pass)) {
delay(1000);
Serial.print(".");
} else {
Serial.println("\nMQTT Connected! ");
}
}
}
unsigned long timer_1 = 0;
void LED_Ctrl_s_n(String topic, float msgNum) {
if(topic == "matrix/r4/led1")
{
if(msgNum == 1)
{
MiniR4.LED.setColor(1, 255, 0, 0);
}
else
{
MiniR4.LED.setColor(1, 0, 0, 0);
```

```
}
}
}

String mqtt_rcvTopic = "";
String mqtt_rcvMessage = "";
void mqttCallback(char* topic, byte* payload, unsigned int length) {
mqtt_rcvTopic = String(topic);
mqtt_rcvMessage = "";
for (unsigned int i = 0; i < length; i++) {
mqtt_rcvMessage += (char)payload[i];
}
mqtt_rcvMessage.trim();

Serial.print(String("Theme: ") + String(mqtt_rcvTopic));
Serial.print(String("Original Message: ") + String(mqtt_rcvMessage));
LED_Ctrl_s_n(mqtt_rcvTopic, mqtt_rcvMessage.toFloat());
}

void setup()
{
MiniR4.begin();
MiniR4.PWR.setBattCell(2);
Serial.begin(9600);
connectWiFi("WiFi_Name", "12345678");
connectMQTT("broker.hivemq.com", 1883, "", "", "");
mqttClient.subscribe("matrix/r4/test1", 0);
mqttClient.subscribe("matrix/r4/led1", 0);
}

void loop()
{
mqttClient.loop();
if((millis() - timer_1) > 5000)
{
mqttClient.publish("matrix/r4/test1", String("Hello").c_str());
}

}
```

Result:

After the program executes, the controller will connect to the specified Wi-Fi network and the MQTT server (broker.hivemq.com), subscribing to the topics **"matrix/r4/test1"** and **"matrix/r4/led1"**. Every 5 seconds, the controller will publish a "Hello" message to the "matrix/r4/test1" topic. When a message is received on the "matrix/r4/led1" topic, the controller will control RGB LED 1 based on the content (lighting up red if the message is "1", and turning off if the message is "0").



You can use an MQTT web testing tool, such as [mqtt.cool](), to verify the program is working correctly. The tool allows you to observe messages sent from the R4 controller every 5 seconds and publish control messages (0 or 1) to the topics the R4 is monitoring.

7.2.5. Principle Description

The operation of MQTT is based on the **"Publish/Subscribe"** model, which differs from traditional point-to-point communication:

- **MQTT Broker:** Acts as a central message exchange station. All devices connect to the Broker, which is responsible for receiving, filtering, and distributing messages.
- **Publisher:** Sends messages to a specific **Topic**. Publishers do not need to know which devices will receive the messages.
- **Subscriber:** Subscribes to topics of interest. When a message is published to that topic, the Broker automatically pushes it to all subscribers.
- **Topic:** A classification label for messages using a hierarchical structure, such as "matrix/r4/led1". Devices can subscribe to specific topics or use wildcards to monitor multiple topics.

On the **MATRIX Mini R4**, the Wi-Fi module establishes the network connection, while the **PubSubClient** library handles the encapsulation of the MQTT protocol. The controller continuously checks for new messages via the mqttClient.loop() method and processes received data through the mqttCallback() function, achieving non-blocking, event-driven communication. This design allows multiple devices to communicate flexibly without knowing each other's network addresses, making it ideal for device coordination and remote control in IoT applications.

7.3. Bluetooth Low Energy (BLE)

7.3.1. Overview

Bluetooth Low Energy (BLE) is a technology specifically designed for low-power, short-range wireless communication, widely used in wearable devices, smart homes, and IoT applications. The **MATRIX Mini R4** features built-in BLE functionality , enabling wireless connection and data exchange with smartphones, tablets, or other BLE-enabled devices.

Through BLE, you can remotely control robots via mobile apps, read sensor data, or implement wireless remote control without requiring a network environment. This makes it highly suitable for robot debugging during competitions or for outdoor applications.

7.3.2. Features

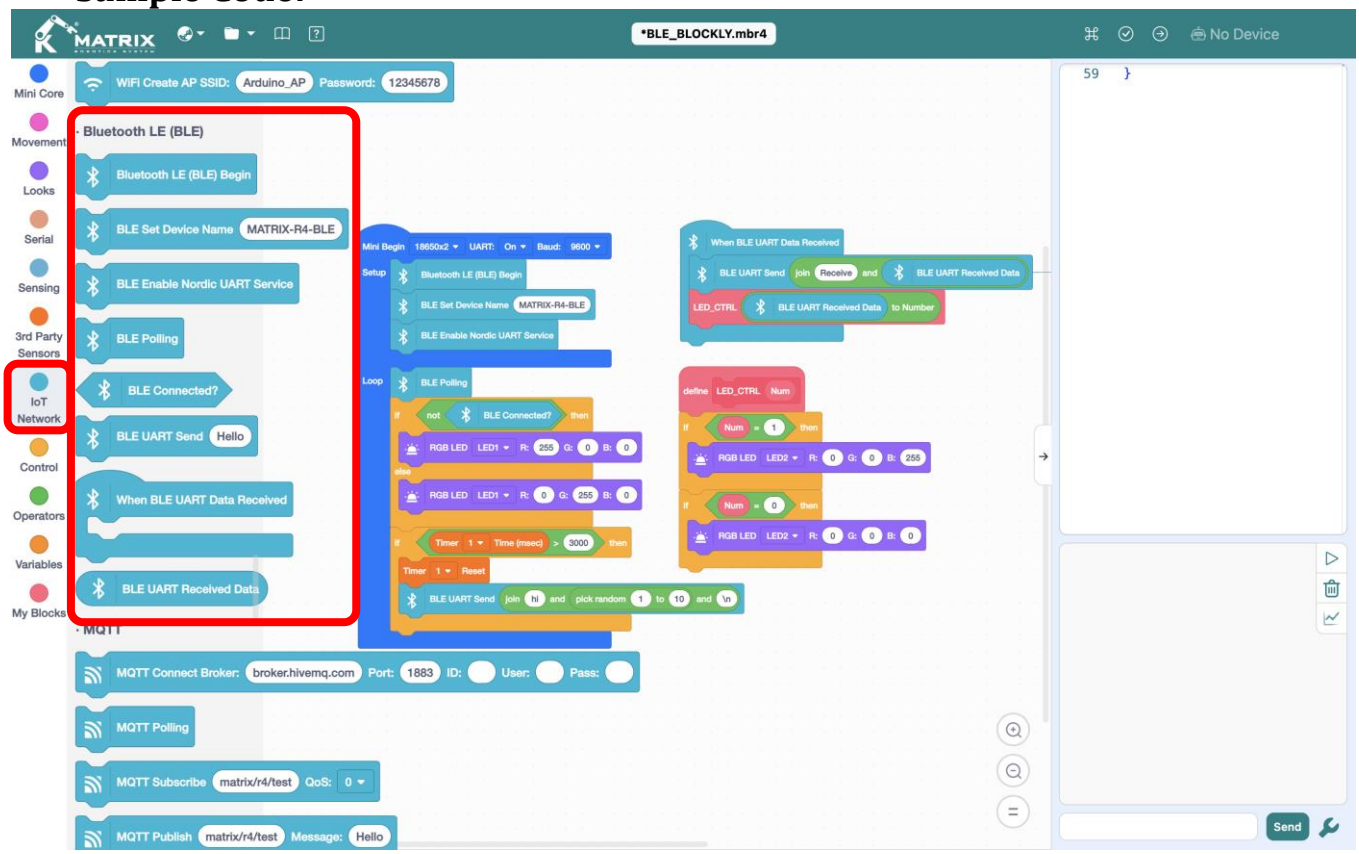
- **Low Power Design:** Power consumption is significantly lower compared to traditional Bluetooth , making it ideal for battery-powered devices.
- **Two-Way Communication:** Supports sending commands from a phone to the controller (RX) and sending data from the controller to a phone (TX).
- Connection Status Detection: Real-time detection of the BLE connection state allows the program logic to adjust based on connectivity.
- **Event-Driven Design:** Uses non-blocking processing to ensure it does not interfere with the execution of the main program logic.
- **Modern Device Support:** As a newer protocol, BLE offers superior compatibility with modern smartphones and tablets.
- **Built-in Nordic UART Service (NUS):** To ease the transition from traditional Bluetooth SPP (like HC-05), the R4 controller natively supports the standard Nordic UART service mode. This is the default mode in MATRIXblock, making it very easy to get started.

7.3.3. Applications

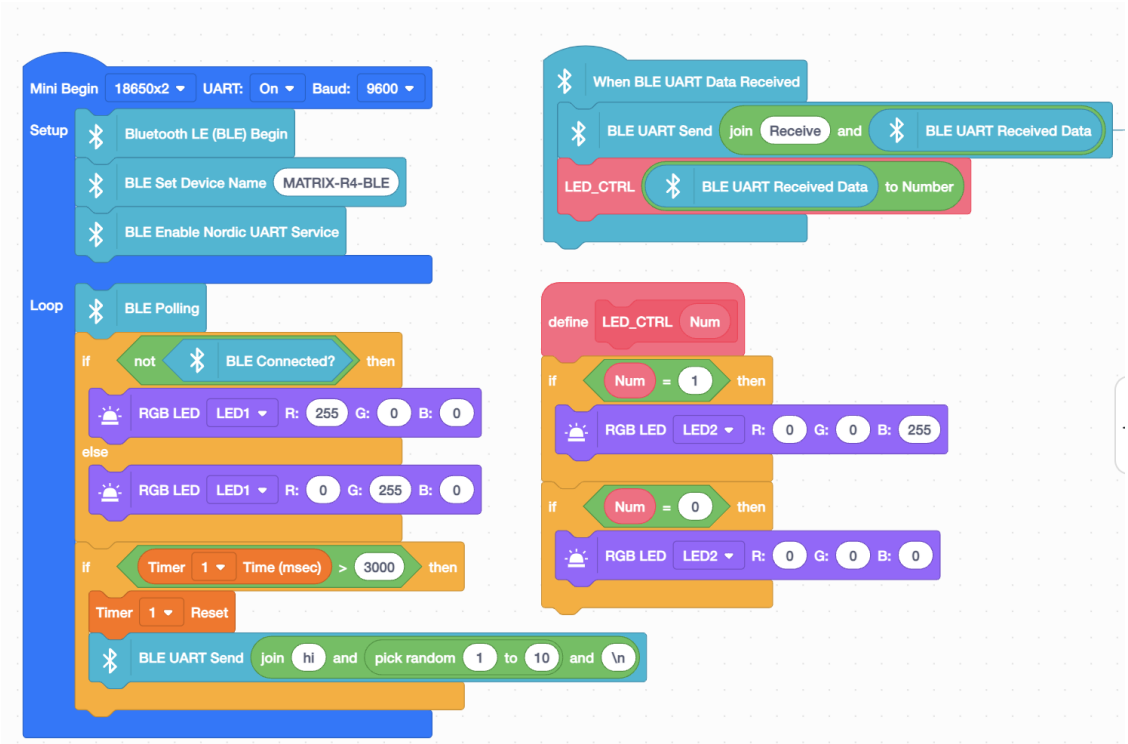
- **Wireless Remote Control:** Move robots using a mobile app interface.
- **Real-Time Data Transmission:** Send sensor data to a phone for immediate display.
- **Wireless Tuning and Monitoring:** Monitor states and adjust parameters wirelessly during competitions.
- **Multi-User Collaborative Control:** Use different phones to control various functions of the same robot simultaneously.

7.3.4. Sample Code & Blocks Instructions

- **Sample Code:**



MATRIXblock



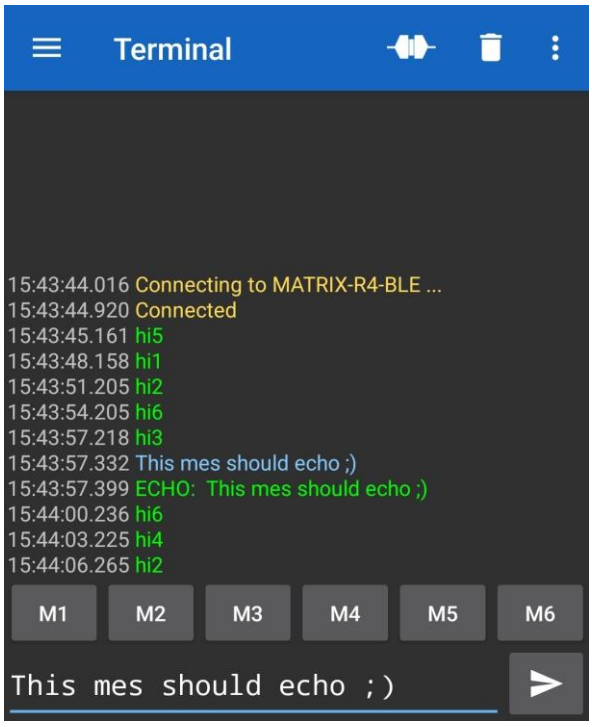
IDE C++

```
#include <ArduinoBLE.h>
#include "MatrixMiniR4.h"

BLEService uartService("6E400001-B5A3-F393-E0A9-E50E24DCCA9E");
BLEStringCharacteristic rxChar("6E400002-B5A3-F393-E0A9-E50E24DCCA9E", BLEWrite, 128);
BLEStringCharacteristic txChar("6E400003-B5A3-F393-E0A9-E50E24DCCA9E", BLERead | BLENotify, 128);
unsigned long timer_1 = 0;
String ble_rcvData = "";
void onBLEWritten(BLEDevice central, BLECharacteristic characteristic) {
ble_rcvData = rxChar.value();

txChar.writeValue(String(String("Receive") + String(ble_rcvData)));
LED_CTRL_n(ble_rcvData.toFloat());
}
void LED_CTRL_n(float Num) {
if(Num == 1)
{
MiniR4.LED.setColor(2, 0, 0, 255);
}
if(Num == 0)
{
MiniR4.LED.setColor(2, 0, 0, 0);
}
}

void setup()
{
MiniR4.begin();
MiniR4.PWR.setBattCell(2);
Serial.begin(9600);
BLE.begin();
BLE.setLocalName("MATRIX-R4-BLE");
BLE.setAdvertisedService(uartService);
uartService.addCharacteristic(rxChar);
uartService.addCharacteristic(txChar);
BLE.addService(uartService);
rxChar.setEventHandler(BLEWritten, onBLEWritten);
BLE.advertise();
}

void loop()
{
```

```
43. BLE.poll();
44. if(!BLE.connected())
45. {
46. MiniR4.LED.setColor(1, 255, 0, 0);
47. }
48. else
49. {
50. MiniR4.LED.setColor(1, 0, 255, 0);
51. }
52. if((millis() - timer_1) > 3000)
53. {
54. timer_1 = millis();
55. txChar.writeValue(String(String("hi") + String((random(1, 10))) + String("\n")));
56. }
57.
58. }
```

Result:

After the program executes, the controller will begin BLE broadcasting with the device name **"MATRIX-R4-BLE."** When a smartphone connects, RGB LED 1 will display green; when disconnected, it will display red. Every 3 seconds, the controller automatically sends a message containing a random number. When the phone sends **"1"**, LED 2 will light up blue; when it sends **"0"**, LED 2 will turn off. Additionally, all messages received from the phone will be sent back (Echoed) to the device.



On the mobile side, apps such as **"Serial Bluetooth Terminal"** or **"BLESerial nRF"** can be used to communicate with the R4 controller.

7.3.5. Principle Description

The operation of BLE is based on the **GATT (Generic Attribute Profile)** architecture:

- **Service**: A collection of functions, each identified by a unique UUID. For example, the Nordic UART Service (NUS) provides serial communication capabilities.
- **Characteristic**: Data containers within a service that can be used for reading, writing, or receiving notifications. The UART service specifically includes **RX** (Receive) and **TX** (Transmit) characteristics.
- **Broadcasting and Connection**: Devices announce their presence through advertising packets, allowing smartphones to scan and initiate a connection.

On the **MATRIX Mini R4**, the Arduino BLE library implements the BLE protocol stack. The program continuously handles BLE events, including connection management and data reception, via the BLE.poll() method. When a smartphone writes data via the **RX characteristic**, the onBLEWritten() callback function is triggered, allowing the program to process the received commands. Conversely, the controller can proactively send data notifications to the phone via the **TX characteristic**.

Because this example utilizes the **Nordic UART Service**, it is significantly easier to get started compared to manually building a raw GATT structure. However, the Arduino BLE library also supports native GATT operations, including both central and peripheral roles, allowing advanced users to perform further development.

Compared to Wi-Fi and MQTT, BLE does not require network infrastructure, making it ideal for point-to-point, short-range communication. This is particularly useful in competition venues or outdoor environments where Wi-Fi is unavailable. Its low-power characteristics also allow battery-powered robots to maintain longer operational times.