

1.2 Introduction to Servo and Servo Controller

1. PWM Servos Introduction

The joints of the robotic arm are equipped with high-precision PWM servos independently developed by our team. Operating them is straightforward: simply send a PWM signal with a 20 ms cycle to the signal pin. The servo angle is controlled by adjusting the pulse width, which ranges from 500 to 2500 μ s, corresponding to 0 to 180°.

Each servo typically consists of a housing, reduction gear set, motor, potentiometer, and control circuitry. Put simply, PWM signals are used to control the servo's position.

When the PWM signal is sent to the motor driver circuit, the signal's duty cycle determines the motor's rotation direction and force. A higher duty cycle provides stronger torque and enables larger rotation angles, while a lower duty cycle results in smaller movements.

By dynamically adjusting the duty cycle, microcontrollers can precisely control the servo's rotation angle. The motor then drives the mechanical structure, delivering accurate position control through the servo's output shaft.

Our servos offer high control accuracy, smooth linearity, fast response, and strong torque, making them well-suited for large-angle joint designs in various bionic robot applications.

Note:

① All servos on the LeArm robotic arm are factory-calibrated to their center positions, so no additional setup is required. For further tuning or learning, refer to:

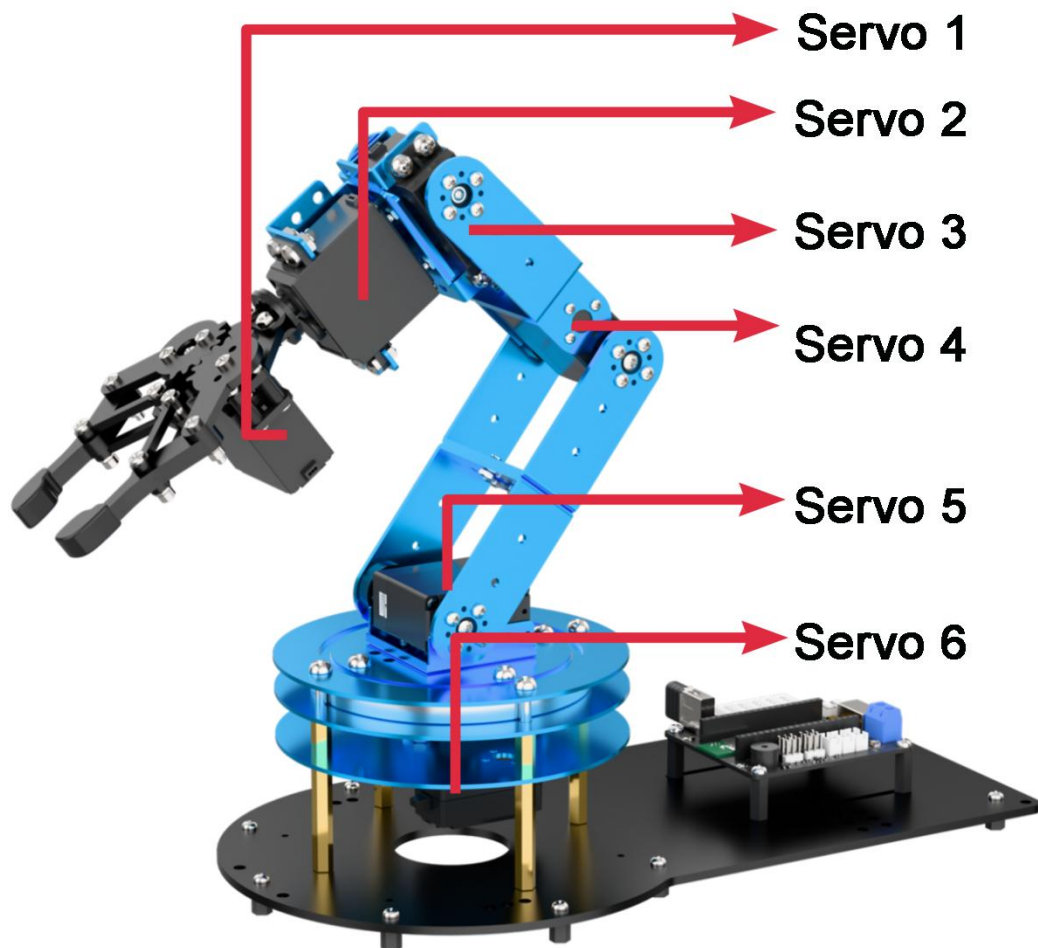
4. Extended Materials → PWM Servo Configuration

② For more on the low-level control principles, see:

"Basic Development Courses → PWM Servo Control" within the main controller programming curriculum.

2. Servos Introduction

The servo layout of the LeArm robotic arm is illustrated below.



The table provides a brief overview of the specifications for each servo model used in the system. For detailed performance characteristics and usage instructions, please refer to Section 4: Extended Materials.

Servo	Rotation speed	Stall torque	Rotation range	Servo accuracy
LDX-218:	0.16sec/60° 7.4v:	15kg.cm/6v 17kg.cm/7.4v	0-180°	0.3°
LD-1501MG:	0.16sec/60° 7.4v:	15kg.cm 6.5V 17kg.cm/7.4v		
LDX-335MG:	0.18sec/60° 7.4v:	20kg.cm/7.4v		
LFD-06:	0.25sec/60° 7.4v:	6kg.cm 7.4V		

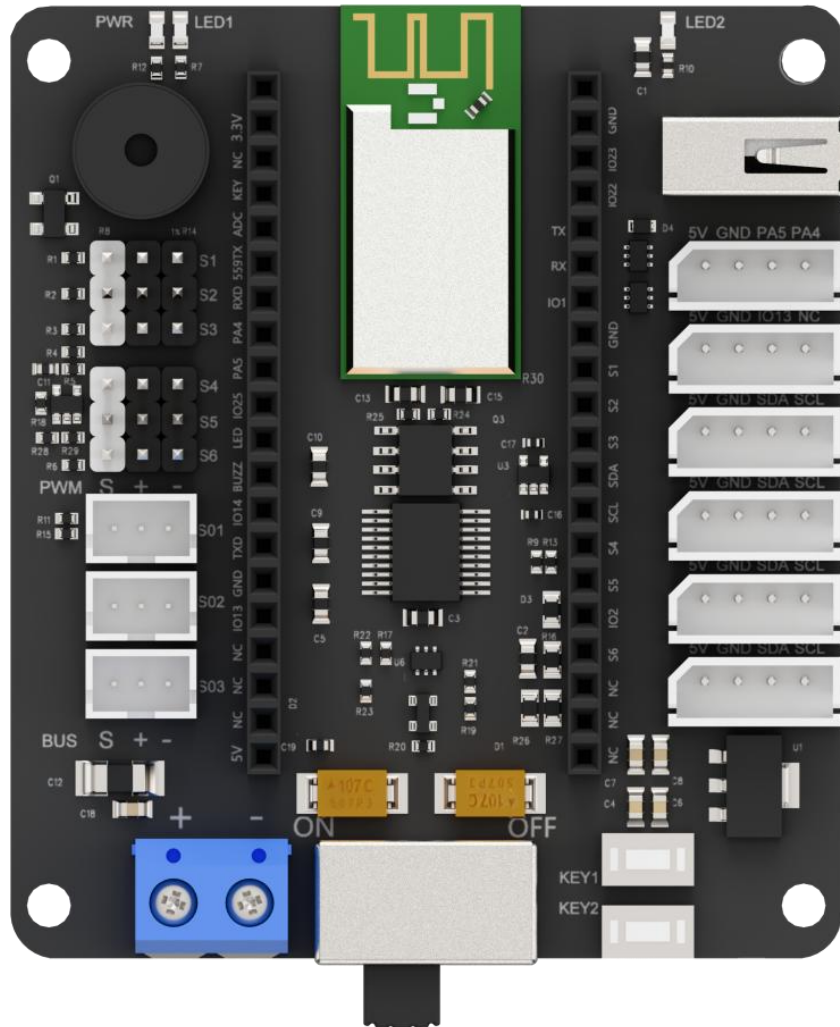
Note:

① When a PWM servo is continuously rotating, the load should not exceed its stall torque. It is recommended to keep the load between one-third and one-fifth of the stall torque.

② Stall torque refers to the maximum torque the servo can provide when it is unable to rotate further (i.e., stalled or blocked). Due to energy losses during rotation, the load should be properly controlled to avoid exceeding the servo's rated capacity. Excessive load causes the servo to work harder, increasing energy consumption and reducing efficiency.

③ When changing the position of a PWM servo, the pulse width should be adjusted gradually—either increasing or decreasing step-by-step—rather than abruptly, to ensure smooth movement.

3. Servo Controller Introduction



The LeArm servo controller is an open-source, six-channel controller designed with a modular main control architecture. Its onboard microcontroller base supports seamless switching between ESP32, STM32, and C51 microcontrollers, providing flexibility for various development needs.

3.1 Interfaces and Buttons

Interface/Button	Purpose
------------------	---------

PWM Servo Port	Connects PWM servos
Serial bus servo port	Connects bus servos
Power switch	Turns the controller power on or off
Custom Button	Allows personalized configuration and control
GPIO Ports	For attaching external expansion modules
Controller Receiver Port	Connects the receiver module for remote communication
Microcontroller Socket	For installing the main microcontroller module

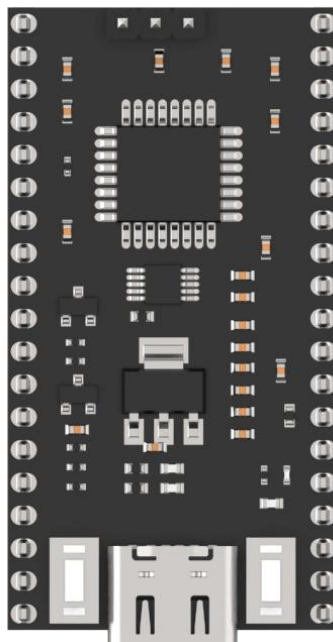
3.2 Specification Parameter

Programming Software	Official Arduino IDE, Keil
Inputs	Controller, function buttons
Outputs	Buzzer, PWM servo output, bus servo output, indicator lights
Microprocessor	Compatible with ESP32 mainboard, STM32 mainboard, C51 mainboard
Communication Methods	Controller communication, Bluetooth, USB, serial communication

Control Method	PC host software, mobile app, controller, motion-sensing glove
Power Supply	6.4 - 8.4V

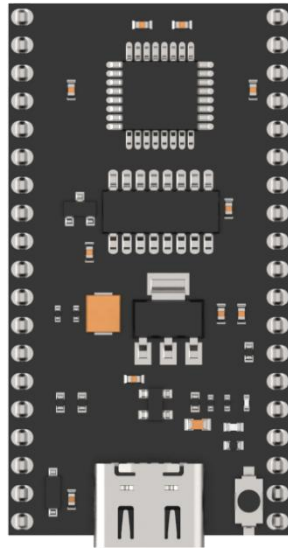
4. Core Board Introduction

4.1 STM32 Core Board



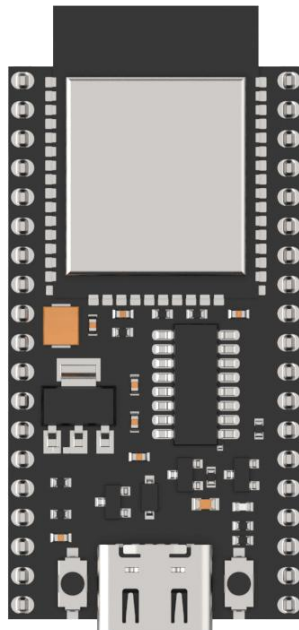
This development board is powered by the STM32F103RBT6 microcontroller. It features an onboard Type-C port for programming and serial communication. On the left side, a BOOT button is connected to the chip's Boot0 pin, allowing users to select the download mode during programming by pressing this button. The RST button on the right performs a hardware reset when pressed.

4.2 C51 Core Board



This development board uses the Ai8051U-34K64 microcontroller. It also includes a Type-C port for programming and serial communication. The K1 button on the right side triggers a hardware reset when pressed.

4.3 ESP32 Core Board



This development board is equipped with the ESP32-WROOM module. It has a Type-C port for programming and serial communication. The button on the left is the IO0 button, which can toggle the corresponding pin's voltage level and can be programmed by the user to trigger custom events. The button on the right is the EN reset button, which performs a hardware reset when pressed.