HITBOT 慧灵科技

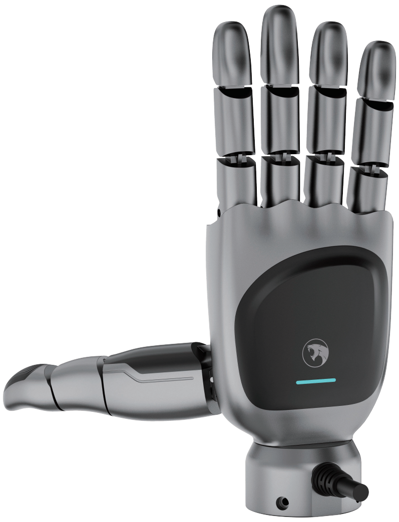
# eHand-6 Product Manual

Product：Scara Robot/6-Axis Robot/End Effector
/Intelligent Actuator/Automatic Solution
Industry：Healthcare/6C/New Retail/Education......

慧灵科技 (深圳) 有限公司
Huiling-tech Robotic Co.,Ltd.

# eHand-6 Dexterous hand

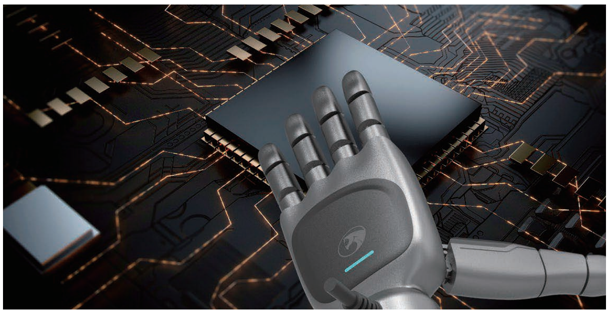
## Redefine the value of dexterous hand



### Product feature

- Degree of freedom: Four-finger bending angle 60°+110° +170°, three joints of thumb flexible rotation (32°+44°+44°) accurately replicate human grasping, pinching, gripping and pushing actions.
- Payload: The gripping force can reach up to 10N and the pulling load capacity is up to 5kg.
- Quick response: Complex gestures can be completed within 1.1s, quick and stable.
- Intelligent perception: Optional 5-channel tactile feedback sensor to build an all-round perception system and achieve 3N-10N dynamic force control.
- Development compatibility: Provides Python/ROS API interfaces and supports CAN FD protocols.

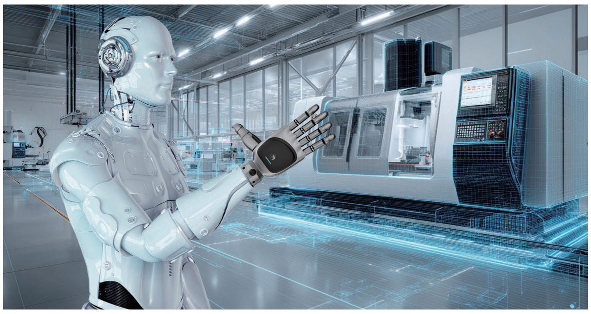
## Application Scenario



Graphic Detection

Assisted production

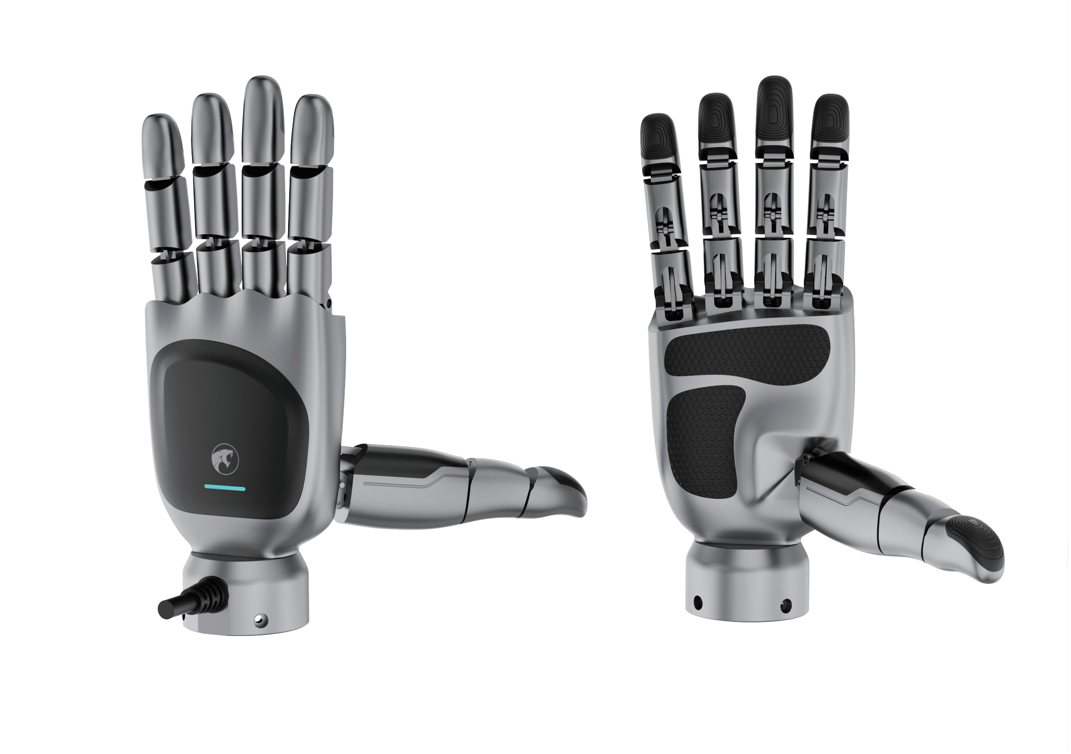
Mobile Robot



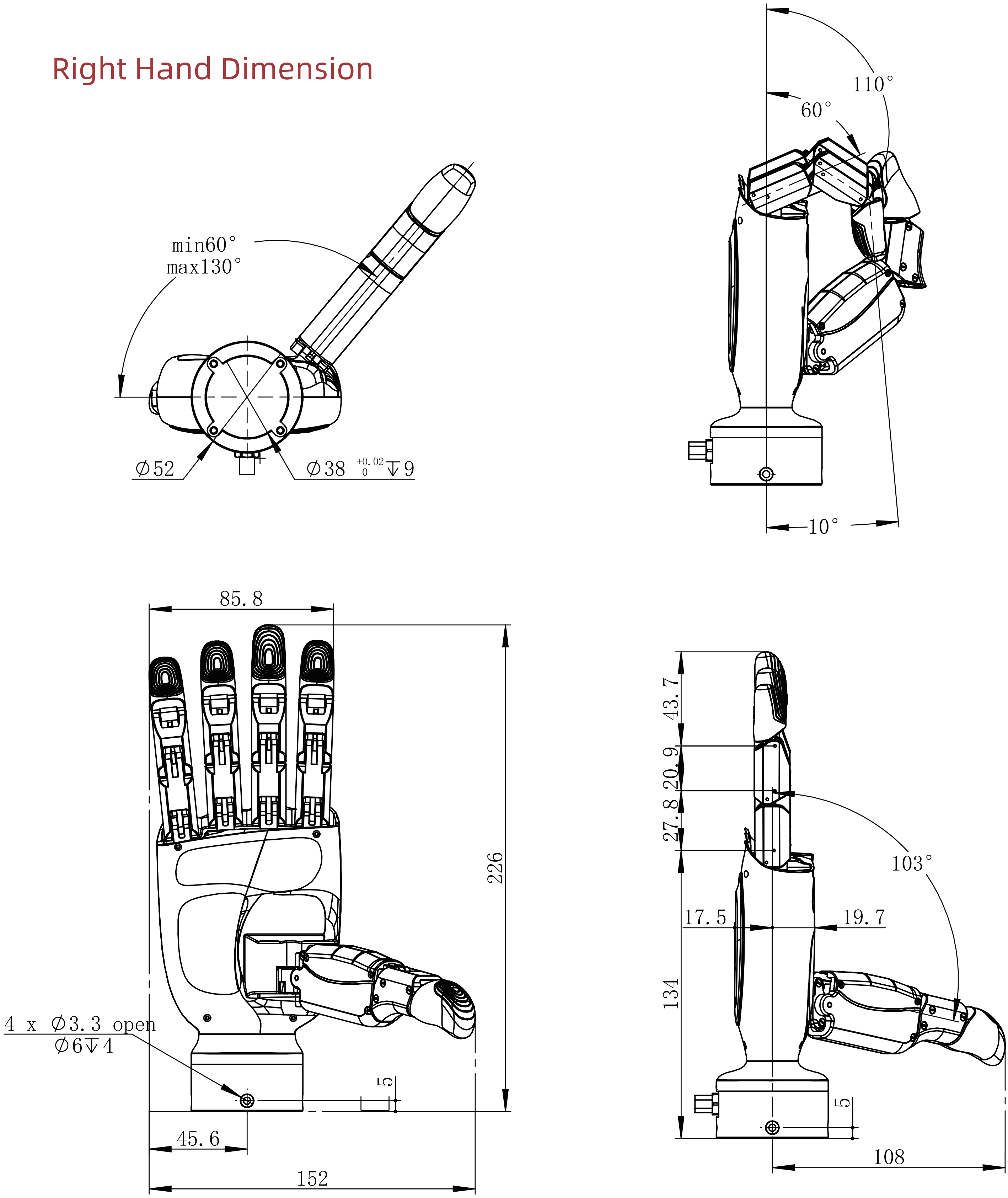
Humanoid Robot

## Specification

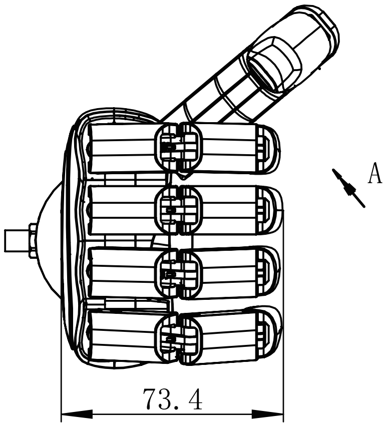
| | Description | Parameter |
|---|---|---|
| Parameter | Degree of freedom | 6 |
| | Cycle time | 1.1s |
| | Four-finger bending angle | 60°+110°+170° |
| | Thumb bending angle | 32°+44°+44° |
| | Thumb side swing angle | 70° |
| | Maximum thumb grip strength | 10N |
| | 4-finger max grip strength | 10N |
| | Pulling load | 5 kg |
| | Weight | 715g |
| | Transmission mode | Coreless Motor+Planetary reducer +Screw + Connecting rod |
| | Dimension | 226*88*52mm |
| Working environment | Communication protocol | CAN FD |
| | Working voltage | 24VDC±10% |

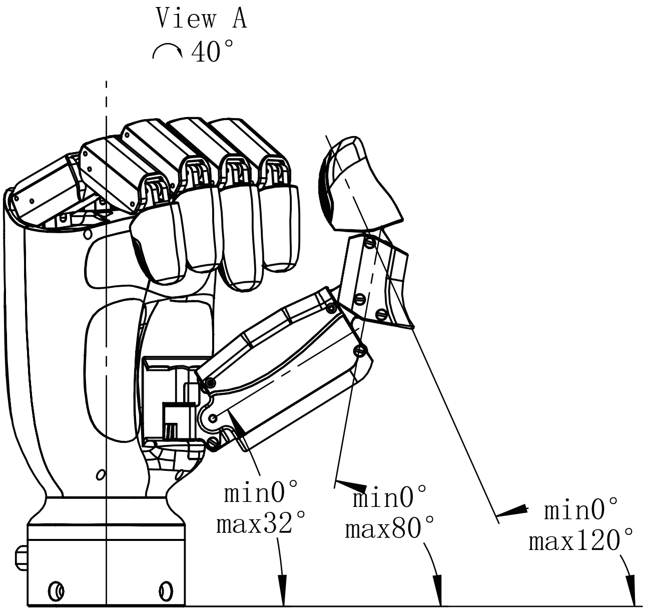


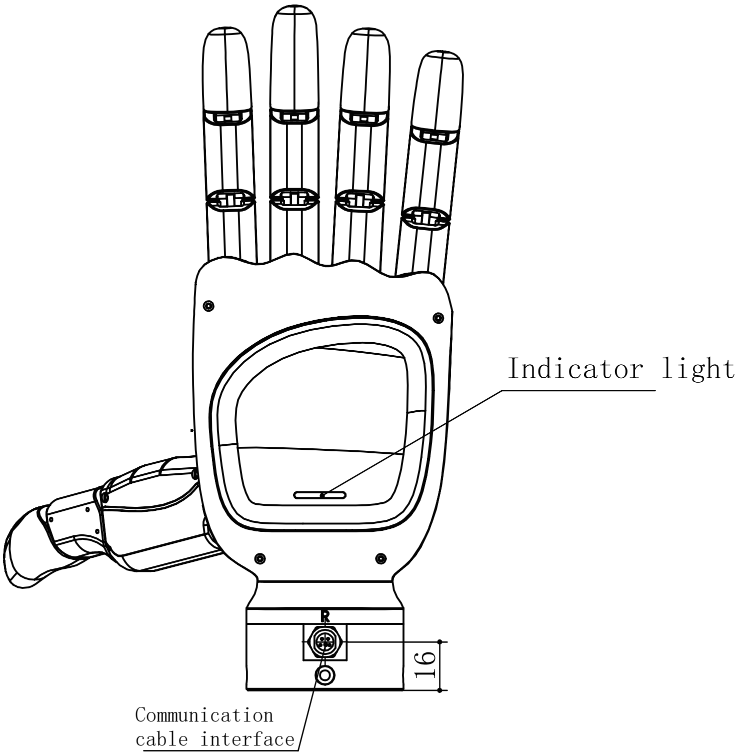
# Right Hand Dimension



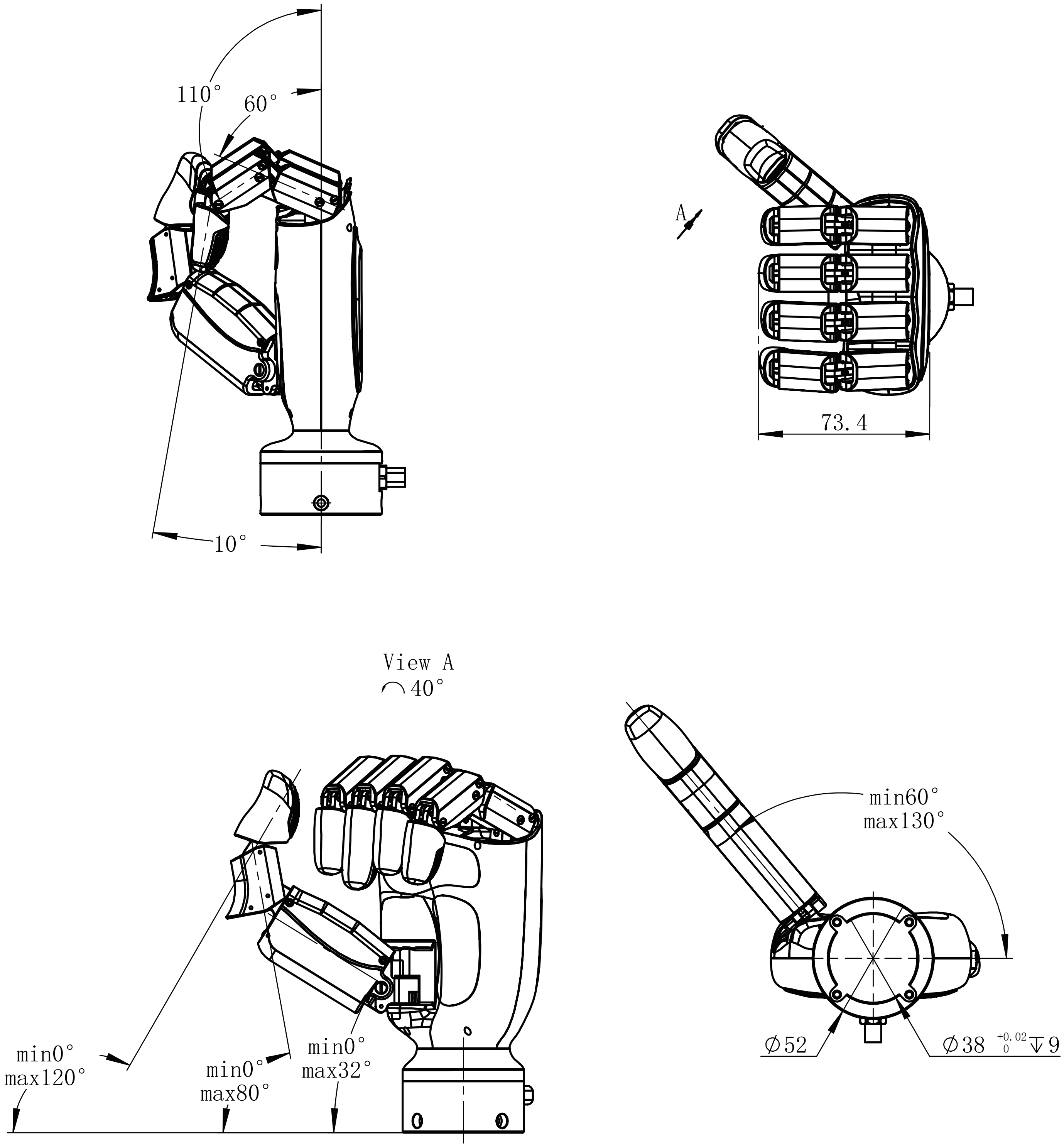
## Right Hand Dimension



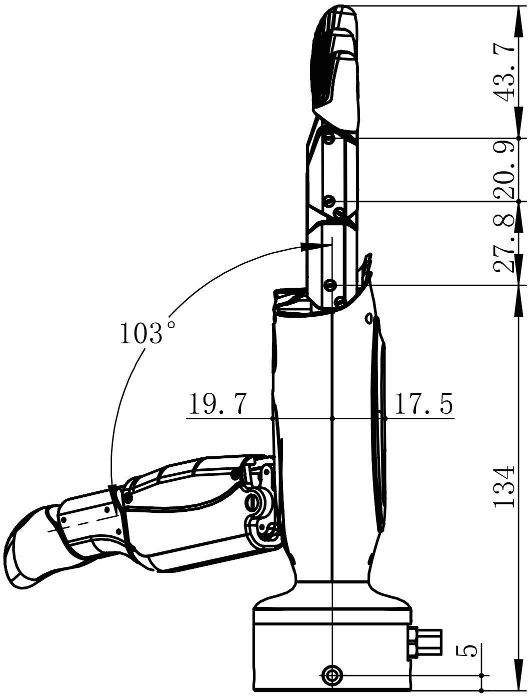


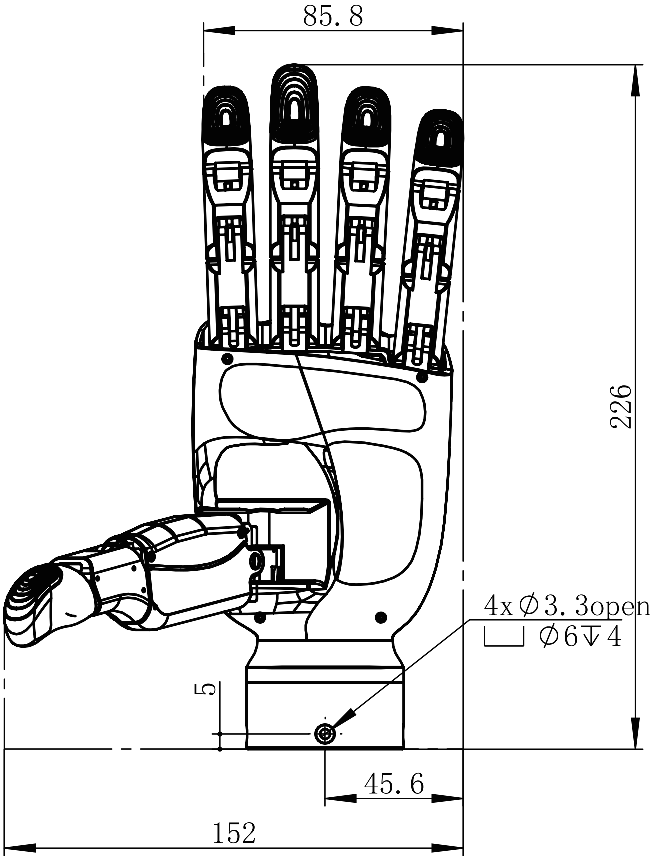


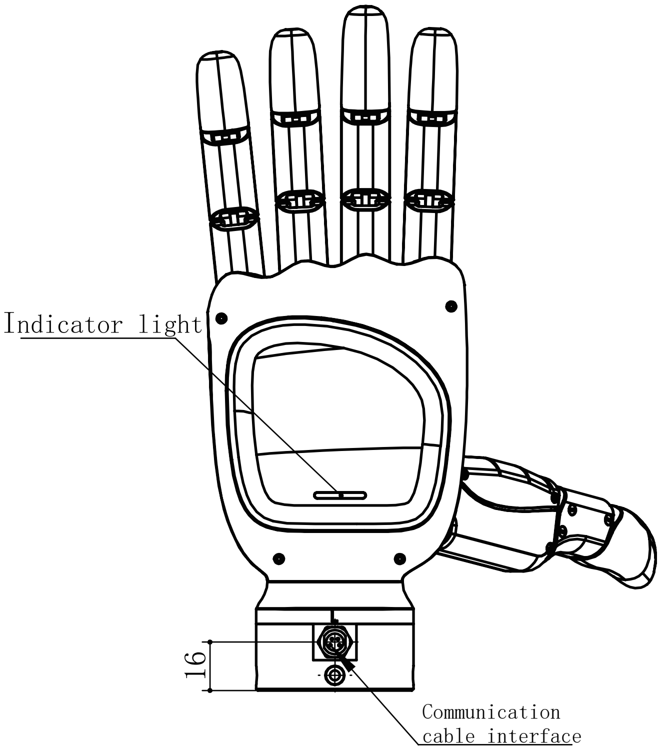
# Left Hand Dimension



## Left Hand Dimension







# External Communication Protocol

## 1. Protocol

This protocol defines the communication specification of the motor control system based on the FDCAN bus. It uses the FDCAN data frame with a maximum length of 64 bytes. The local application layer protocol defines the effective length as 32 bytes, the arbitration segment rate is 1Mbps, the data segment rate is 5Mbps, the arbitration segment sampling point is 80%, and the data segment sampling point is 75%.
It contains two message types: control instructions and status feedback. The protocol design is compact and efficient, suitable for real-time motor control scenarios.

## 2. Communication Parameter

| Parameter | Description |
|---|---|
| Communication mode | FDCAN bus communication，right hand ID 0x11, left hand ID 0x12 |
| Data frame format | FDCAN support max data frame(64 bits),The local application control protocol valid data frame is 32 bytes. |
| Communication speed | Arbitration segment rate is 1Mbps,data segment rate is 5Mbps |
| Message type | Control command frame/Status feedback frame |

## 3. Control command frame format

### 3.1 Byte definition and fill-in rule

The first two bytes of the control instruction frame are command and control word, which are used to select instruction type, operation object and control mode.
The last 30 bytes are control parameters, which are divided into six groups, each with 5 bytes. The last two bytes of each group are reserved and undefined.

| Byte No. | Item/Definition | Description | Example |
|---|---|---|---|
| 1 | Command + ID<br>uint8_t command :2<br>uint8_t MotorID :6 | Lower 2bit：0 Read 1 Write ;<br>Upper 6bit：ID locked bit2-bit7:<br>Corresponding to motor 1-6;<br>0x3F: Select all motors. | 0x05,Select motor 1 and write command;<br>0x41,Select motor 5 and write command;<br>0x45,Select motor 1 and 5，and write command;<br>0xFC,Select all motors and get status, the following bytes of reading state is not needed to fill in. |

| | | | |
|---|---|---|---|
| 2 | Control word<br>uint8_t controlword | Lower 4 bits：Control mode selection<br>0 Disabled<br>1 Position mode<br>2 Finger open<br>3 Finger close<br>4 Finger zeroing<br>5 Write and save point<br>6 Execute point | 0x01：Position mode,run to the designed position according to position/speed/torque of selected motors<br>0x02：Fingers open,open according to speed/torque of selected motor<br>0x03：Fingers close，close according to speed/torque of selected motor<br>0x04 :Fingers zeroing，zeroing according to selected motors<br>0x15：Take the selected motor parameters as point parameter and save to point 1,motors not selected do not move<br>0x16：Run point 1 |
| 3 | Thumb horizontal target position uint8_t position1 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 4 | Thumb horizontal target speed uint8_t speed1 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half of the max speed |
| 5 | Thumb horizontal target torque<br>uint8_t torque1 | 0-255<br>Corresponding to 0-100% | 0xCC Run at 80% of the torque |
| 6 | Reserved 1<br>uint8_t reserved1 | - | - |
| 7 | Reserved 2<br>uint8_t reserved2 | - | - |
| 8 | Thumb vertical target position uint8_t position2 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 9 | Thumb vertical target speed uint8_t speed2 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the max speed |
| 10 | Thumb vertical target torque uint8_t torque2 | 0-255<br>Corresponding to 0-100% | 0xCC Run at 80% of the torque |
| 11 | Reserved 3<br>uint8_t reserved3 | - | - |

| | | | |
|---|---|---|---|
| 12 | Reserved 4<br>uint8_t reserved4 | - | - |
| 13 | Index finger target position<br>uint8_t position3 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 14 | Index finger target speed<br>uint8_t speed3 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half full speed |
| 15 | Index finger target torque<br>uint8_t torque3 | 0-255<br>Corresponding to 0-100% | 0xCC Run at 80% torque |
| 16 | Reserved 5<br>uint8_t reserved5 | - | - |
| 17 | Reserved 6<br>uint8_t reserved6 | - | - |
| 18 | Middle finger target position<br>uint8_t position4 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 19 | Middle finger target speed<br>uint8_t speed4 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the max speed |
| 20 | Middle finger target torque<br>uint8_t torque4 | 0-255<br>Corresponding to 0-100% | 0xCC Run at 80% of the torque |
| 21 | Reserved 7<br>uint8_t reserved7 | - | - |
| 22 | Reserved 8<br>uint8_t reserved8 | - | - |
| 23 | Ring finger target position<br>uint8_t position5 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |

| | | | |
|---|---|---|---|
| 24 | Ring finger<br>target speed<br>uint8_t speed5 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the max speed |
| 25 | Ring finger<br>target torque<br>uint8_t torque5 | 0-255<br>Corresponding to 0-100% | 0xCC Run at 80% of the torque |
| 26 | Reserved 9<br>uint8_t reserved9 | - | - |
| 27 | Reserved 10<br>uint8_t reserved10 | - | - |
| 28 | Little finger<br>target position<br>uint8_t position6 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 29 | Little finger<br>target speed<br>uint8_t speed6 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the max speed |
| 30 | Little finger<br>target torque<br>uint8_t torque6 | 0-255<br>Corresponding to 0-100% | 0xCC Run at 80% of the torque |
| 31 | Reserved 11<br>uint8_t reserved11 | - | - |
| 32 | Reserved 12<br>uint8_t reserved12 | - | - |

## 3.2 Description of control parameter

### Position/Speed/Torque

| Data type | Range | Description |
|---|---|---|
| No symbol 8 integer<br>uint8_t | 0-255<br>Corresponding to 0-100%<br>Accuracy: 1% | Data value 0 corresponding to 0%;<br>1-255 corresponding to 1%-100%.<br>When converting an integer value to a percentage, it needs to be rounded up. For example, if you need to write 90%, then 255*0.9 = 229.5,to be rounded up to 230. |

# 4. Status feedback frame format

## 4.1 Byte feedback definition

The first two bytes of the status feedback frame are command and control words, which are used to transmit feedback type, feedback object and control mode.
The last 30 bytes are real-time feedback parameters, divided into six groups, each with 5 bytes, and the last two bytes of each group are reserved and undefined

| Byte No. | Item | Description | Example |
|---|---|---|---|
| 1 | Command + ID<br>uint8_t command :2<br>uint8_t MotorID :6 | Lower 2 bits：<br>0 Frame format error;<br>1 Fault uploading;<br>2 Read command response；<br>3 Write command response;<br>Upper 6 bits：message ID, corresponding to motor 1-6. | 0x07,receive motor 1 command，write command to response the returned control parameters of the 3rd to 32nd bytes received in the original path.<br>0x41,motor 5 fault，upload feedback parameter;<br>0x44, receive motor 1 and 5 command，reply frame format fault.<br>0xFE,receive hand command，read state，response. |
| 2 | Hand status+<br>Hand fault code<br>uint8_t state :4<br>uint8_t fault_code：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU initializing:<br>0x01:Standby<br>0x02:Zeroing<br>0x03:In position mode<br>0x56:Stall fault |
| 3 | Thumb horizontal status+<br>Thumb horizontal fault code<br>uint8_t state1 :4<br>uint8_t<br>fault_code1：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU initializing:<br>0x01:Standby<br>0x02:Zeroing<br>0x03:In position mode<br>0x56:Stall fault |
| 4 | Thumb horizontal position<br>uint8_t position1 | 0-255 Corresponding to 0-100% | 0x80 Run to the center position |
| 5 | Thumb horizontal speed<br>uint8_t speed1 | 0-255 Corresponding to 0-100% | 0x80 Run at half the max speed |
| 6 | Reserved 1<br>uint8_t reserved1 | - | - |
| 7 | Reserved 2<br>uint8_t reserved2 | - | - |

| | | | |
|---|---|---|---|
| 8 | Thumb vertical status+<br>Thumb vertical fault code<br>uint8_t state 2:4<br>uint8_t<br>fault_code2：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU initializing:<br>0x01:Standby<br>0x02:Zeroing<br>0x03:In position mode<br>0x56:Stall fault |
| 9 | Thumb vertical position<br>uint8_t position2 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 10 | Thumb vertical speed<br>uint8_t speed3 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the max speed |
| 11 | Reserved 3<br>uint8_t reserved3 | - | - |
| 12 | Reserved 4<br>uint8_t reserved4 | - | - |
| 13 | Index finger status+<br>Index finger fault code<br>uint8_t state3 :4<br>uint8_t<br>fault_code3：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU initializing:<br>0x01:Standby<br>0x02:Zeroing<br>0x03:In position mode<br>0x56:Stall fault |
| 14 | Index finger position<br>uint8_t position3 | 0-255<br>Corresponding to 0-100% | 0x80,rounding down to 128/255 = 0.5019 About 50%, 0.5*255 = 127.5 rounding up to 128 |
| 15 | Index finger speed<br>uint8_t speed3 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the max speed |
| 16 | Reserved 5<br>uint8_t reserved5 | - | - |
| 17 | Reserved 6<br>uint8_t reserved6 | - | - |

|  |  |  |  |
|---|---|---|---|
| 18 | Middle finger status+<br>Middle finger fault code<br>uint8_t state4 :4<br>uint8_t<br>fault_code4：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU power-on initializing<br>0x01:Standby<br>0x02:Zeroing<br>0x03:In position mode<br>0x56:Stall fault |
| 19 | Middle finger position<br>uint8_t position4 | 0-255<br>Correspongding to 0-100% | 0x80 Run to the center position |
| 20 | Middle finger speed<br>uint8_t speed4 | 0-255<br>Correspongding to 0-100% | 0x80 Run at half the full speed |
| 21 | Reserved 7<br>uint8_t reserved7 | - | - |
| 22 | Reserved 8<br>uint8_t reserved8 | - | - |
| 23 | Ring finger status+<br>Ring finger fault code<br>uint8_t state5 :4<br>uint8_t<br>fault_code5：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU power-on initializing:<br>0x01:Standby<br>0x02:Zeroing<br>0x03:In position mode<br>0x56:Stall fault |
| 24 | Ring finger position<br>uint8_t position5 | 0-255<br>Corresponding to 0-100% | 0x80 Run to the center position |
| 25 | Ring finger speed<br>uint8_t speed5 | 0-255<br>Corresponding to 0-100% | 0x80 Run at half the full speed |
| 26 | Reserved 9<br>uint8_t reserved9 | - | - |
| 27 | Reserved 10<br>uint8_t reserved10 | - | - |

| | | | |
|---|---|---|---|
| 28 | Little finger status+<br>Little finger fault code<br>uint8_t  state5 :4<br>uint8_t  fault_code5：4 | For details on the corresponding status of the value, please refer to "Field detailed description" | 0x00:MCU power-on initializing:<br>0x01:Standby<br>0x02:Zeroing<br>x03:In position mode<br>0x56:Stall fault |
| 29 | Little finger position<br>uint8_t reserved11 | 0-255<br>Correspongding to 0-100% | 0x80 Run to the center position |
| 30 | Little finger speed<br>uint8_t reserved12 | 0-255<br>Correspongding to 0-100% | 0x80  Run at half full speed |
| 31 | Reserved 11 | - | - |
| 32 | Reserved 12 | - | - |

## 4.2  Field Detailed Description

### State (uint8_t state)

| Value | State description |
|---|---|
| 0x0 | Initializing |
| 0x1 | Standby state |
| 0x2 | Calibration state |
| 0x3 | Position mode |
| 0x4 | Reserved |

Single hand state signal (uint8_t state1-6)

| Value | State |
|---|---|
| 0x0 | Initializing |
| 0x1 | Standby mode |
| 0x2 | Calibration mode |
| 0x3 | Position mode |
| 0x4 | Reserved |

| Value | State |
|---|---|
| 0x5 | Aging mode |
| 0x6 | Fault state |
| 0x7 | Command response waiting |
| 0x8-0xF | Reserved |

Fault code (uint8_t fault_code)

| Code | Fault type |
|---|---|
| 0x0 | No fault |
| 0x1 | Overcurrent protection |
| 0x2 | Overvoltage protection |
| 0x3 | Undervoltage protection |
| 0x4 | Overheat protection |

| Code | Fault type |
|---|---|
| 0x5 | Stall protection |
| 0x6 | Communication overtime |
| 0x7 | Hardware fault |
| 0x8-0xF | Reserved |

## 4.3 Detailed description of status feedback value

### Position/Speed

| Data type | Range | Description |
|---|---|---|
| No symbol 8 integer<br>uint8_t | 0-255<br>Corresponding to 0-100%<br>Accuracy 1% | Data 0 corresponding to 0%<br>1-255 corresponding to 1%-100%<br>The read value will truncate the decimal places. For example, the actual calculation is 234.245, and the value obtained after communication is 234. |

# 5. Communication frame example

## 5.1 Control command frame

Main：

```
FD 01 80 FF FF 00 00 33 CC E6 00 00 4D C0 B3 00 00 66 D9 F3 00 00 99 A6 8D 00 00 5F 76 8D 00 00
```

Slave：

```
FF 03 80 FF FF 00 00 33 CC E6 00 00 4D C0 B3 00 00 66 D9 F3 00 00 99 A6 8D 00 00 5F 76 8D 00 00
```

FD： Binary 11111101 The upper 6 bits are all valid, all motors are selected, and the lower 2 bits are 1.Write command operation, the lower 2 bits of the slave response frame are 3, write response, and the remaining bytes are consistent with the command frame.

01： Control word is in position mode

80： Thumb horizontal target position-50%

FF： Thumb vertical target speed - 100%

FF： Thumb vertical target torque - 100%

00： Reserved

00： Reserved

33： Thumb vertical target position -20%

CC： Thumb vertical target speed - 80%

E6： Thumb vertical target torque - 90%

00： Reserved

00： Reserved

4D： Index finger target position - 30%

C0： Index finger target speed - 75%

B3： Index finger target torque - 70%

00： Reserved

00： Reserved

66： Middle finger target position -40%

D9： Middle finger target speed - 85%

F3： Middle finger target torque - 95%

00： Reserved

00： Reserved

5F： Little finger target position -37%

76： Little finger target speed - 46%

8D： Little finger target torque - 77%

00： Reserved

00： Reserved

## 5.2 State feedback frame

Main：

FC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Slave：

FE 02 03 80 FF 00 00 03 33 CC 00 00 03 4D C0 00 00 03 66 D9 00 00 00 03 99 A6 00 00 03 5F 76 00 00

FE： Binary 111111110 The upper 6 bits are all valid, the command selects all motors, the lower 2 bits are 1: read reply, return all motors state.
The read status instruction operation only needs to fill the first byte of the read response (2) + ID part, and the following 31 bytes do not need to be filled.

02： Machine state -running, no fault

03: Thumb horizontal state- position mode

80： Thumb horizontal current position -50%

FF： Thumb vertical current speed -100%

03: Thumb vertical state-position mode

00： Reserved

00： Reserved

33： Thumb vertical current position -20%

CC：Thumb vertical current speed -80%

03： Index finger state -position mode -

00： Reserved

00： Reserved

4D：Index finger current position-30%

C0： Index finger current speed - 75%

B3： Index finger current torque - 70%

03: Middle finger state- position mode

00： Reserved

00： Reserved

66： Middle finger current position -40%

D9： Middle finger current speed - 85%

F3： Middle finger current torque- 95%

00： Reserved

00： Reserved

03： Ring finger current state -position mode

99 ：Ring finger current speed - 60%

A6：Ring finger current torque- 65%

00： Reserved

00： Reserved

5F： Little finger current position - 37%

76： Little finger current speed - 46%

8D： Little finger state-position mode - no fault

00： Reserved

00： Reserved

## 6. Wiring method

The dexterous hand uses a 4-core M8 aviation plug for external connection.

| Cable color | Sequence |
|---|---|
| Brown | 24V+ |
| Blue | 0V（24V-、GND） |
| White | CAN_L |
| Black | CAN_H |

Note:

1. When wiring, please ensure that the positive and negative poles of the power cord are connected correctly, and that the CAN communication cable and power cable are correctly connected. Damage caused by incorrect wiring is not covered by the warranty.

2. The CAN line and 24V power line are not physically isolated inside the hand. If isolation is required, use other equipment.

# 7. Communication process

## 7.1 Control command sending process

①The control software constructs the control command frame, selects the write command and selects the corresponding joint.
②Send to the palm master via FDCAN to specify the motor node to execute action, point or save point.
③After receiving the control frame, the host node returns the carrier data as a feedback frame.

## 7.2 Cycle status access process

①The control program periodically (recommended 1-100ms) accesses the host status, and the host sends the corresponding finger status or whole machine feedback frame.
②In case of emergency (such as fault), the host should immediately send a status feedback frame.
③The main controller receives the read command and returns the feedback data of each node.

## 7.3 Error handling mechanism

①Communication timeout: The motor enters a safe state after the timeout.

②Fault status: The motor stops immediately and sends a fault status frame.

③Illegal parameter: Ignore the parameter, keep the original value, and send an error frame format feedback frame.

# 8. Typical application scenarios

## 8.1 Motor starting process

①Position Mode

Write 1 for the control word
Set target speed, position and torque parameters

Frame send command

②Speed Mode

Write 2(extend) or 3(shrink) for the control word

Set target speed and torque

Frame send command

③Imitating drag teaching through “① position mode” , after finding the appropriate position, run the saved points.
Or obtain the stall clamping position information through “② speed mode”, save to point number, run point.

④Monitor real-time status: send the read commands to obtain the status of the entire hand.

## 8.2 Emergency stop process

Send a broadcast stop command(Select the corresponding motor ID,controlword = 0)

All motors stop immediately and report their status.

# 9. Protocol extension

①Reserved fields can be used for future functionality extensions.

②More control functions can be achieved by extending the command word.
(a maximum of 64 bytes per FDCAN frame)

HITBOT 慧灵科技

Making automation easier

Huiling-tech Robotic Co., Ltd.
Phone：+86 15220259858
Email：marketing@hitbot.cc
Website：www.hitbotrobot.com

HITBOT Web