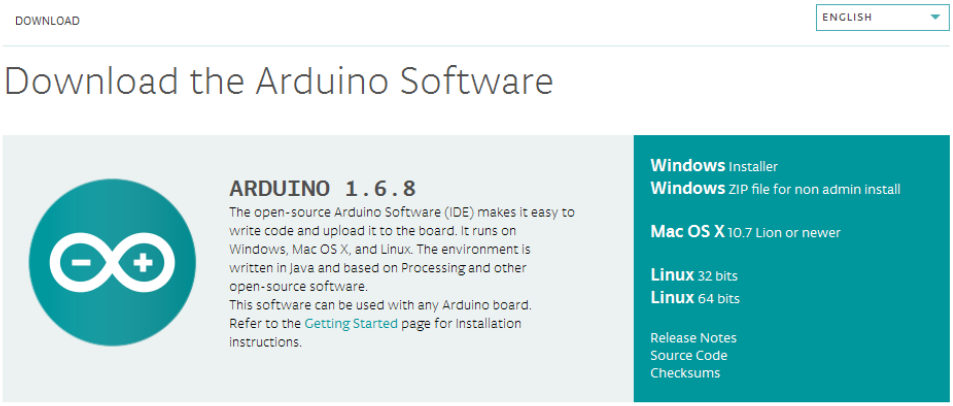
Getting Started with 7Bot - - for Arduino v1.0

1. Downloading Arduino IDE

The 7Bot Arm is based on Arduino Due, so you need try a little bit of Arduino. If Arduino is new to you, you can download the Arduino IDE here:

https://www.arduino.cc/en/Main/Software



You can see more about "Getting Started with Arduino" for the Arduino official website:

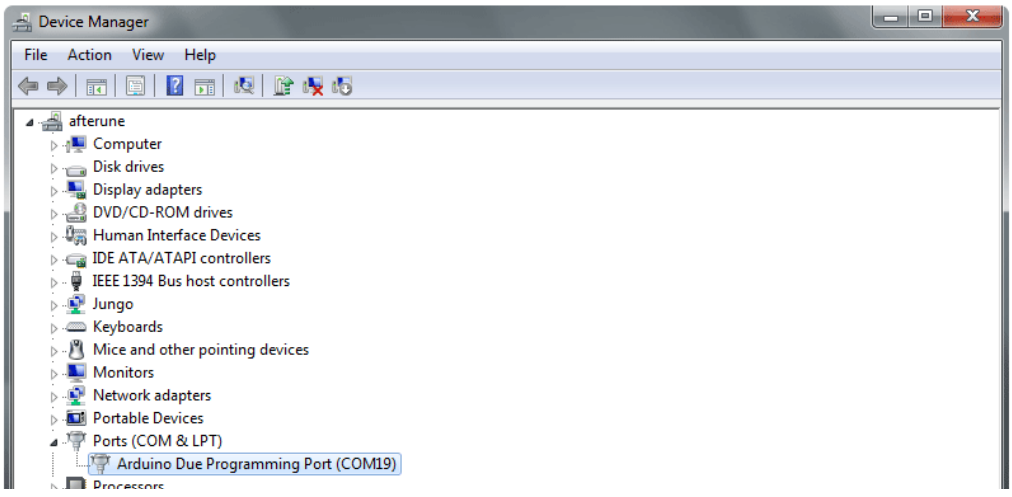
https://www.arduino.cc/en/Guide/HomePage

2. Installing Drivers for the Due

For Windows users, if this is the first time you connect your computer with the Arduino Due, you need install drivers for the Due. You can see more details of how to install the drivers here on Arduino official website:

https://www.arduino.cc/en/Guide/ArduinoDue#toc8

After you install the driver on your computer by connecting 7Bot with USB cable, you can see a port listing similar to "Arduino Due Programming Port (COM19)" in the Device Manager.



3. Installing the Arduino Sam Boards core

As Arduino Due is a powerful and special board, you need to install the core that supports the Arduino Due. Please follow the official guide to install the new core here:

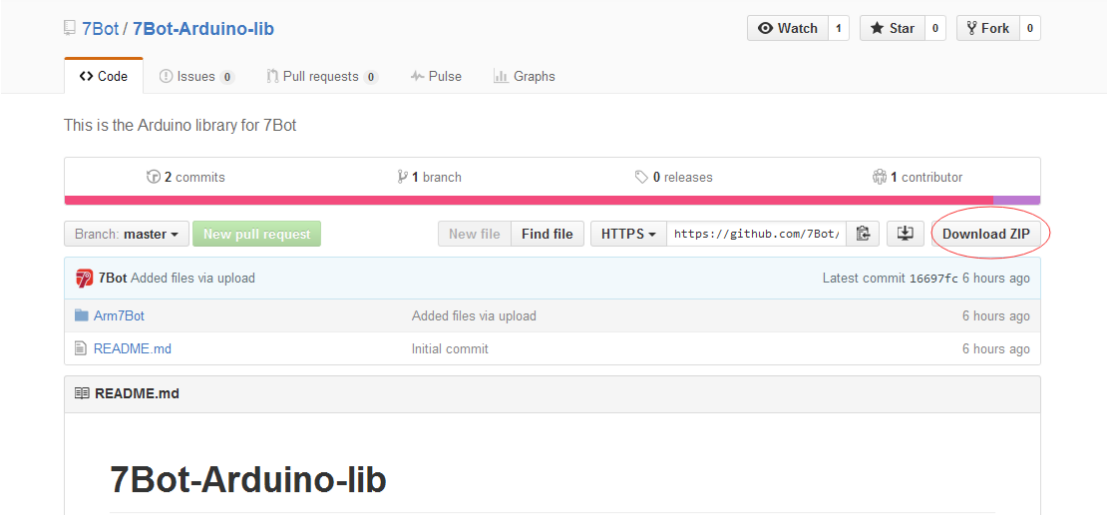
https://www.arduino.cc/en/Guide/Cores

4. Downloading 7Bot Arduino Library

7Bot Arduino Library can be downloaded from this Github repository:

https://github.com/7Bot/7Bot-Arduino-lib

Click the “Download ZIP” button:



In this library, controlling methods are in the Arm7Bot.cpp file of Src floder. If you have programming background and would like to develop Arduino, you can change it to meet your own needs. But if you are not focus on developing the underlying code of Arduino, you’d better leave them alone.

5. Downloading DueFlashStorage Library

In order to use our library, you will also need to download and install a DueFlashStorage library:

https://github.com/sebnil/DueFlashStorage

Download the DueFlashStorage library by clicking the “Download ZIP” button

6. Installing Arduino Libraries

You can learn more details about how to install a library here:

https://www.arduino.cc/en/Guide/Libraries

With Step 4, you have downloaded 7Bot Arduino Library as a ZIP file which named “7Bot-Arduino-lib-master”.

Unzip this file; you will get a “7Bot-Arduino-lib-master” folder.

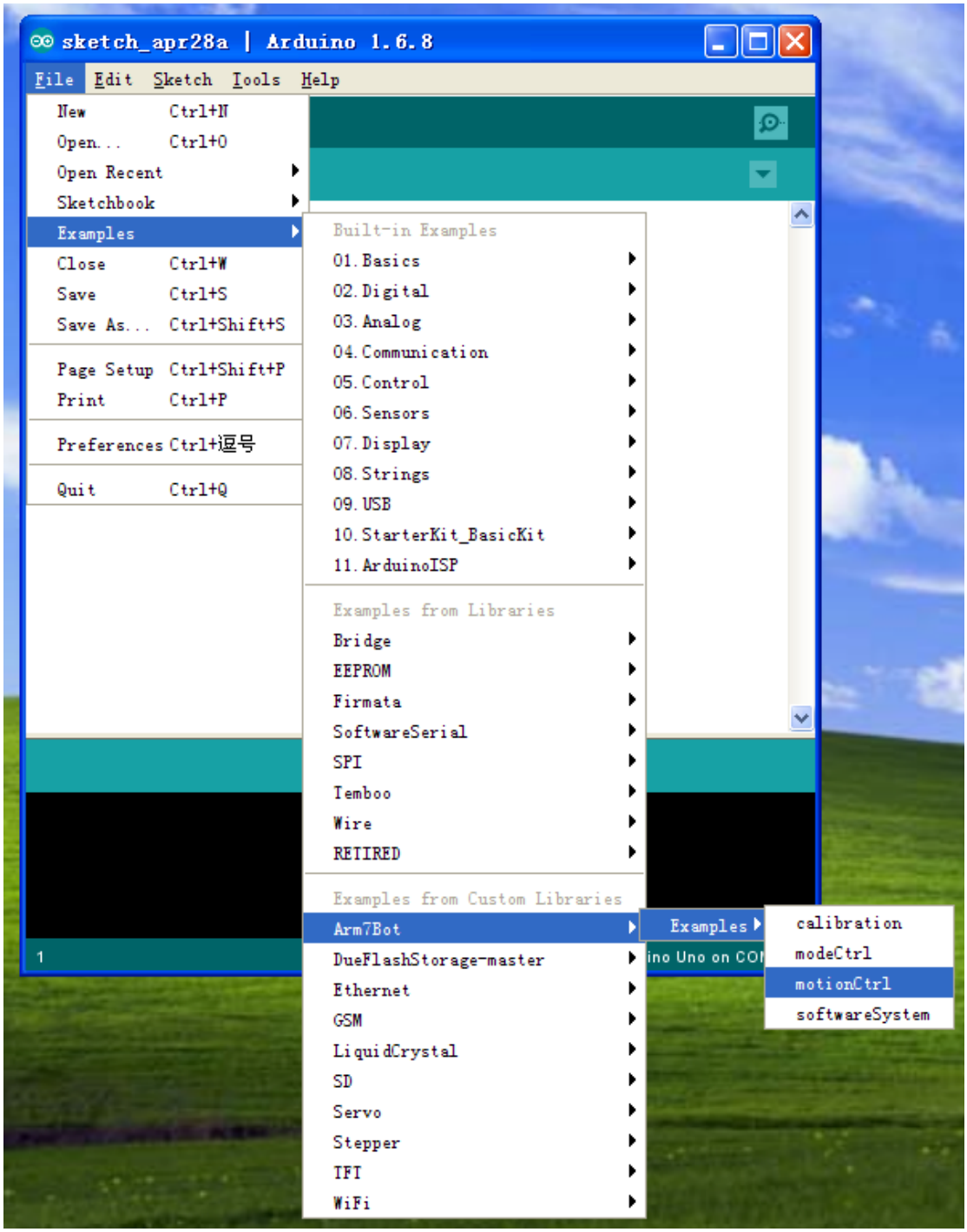
Open the “7Bot-Arduino-lib-master” folder; you can see a folder named “Arm7Bot”.

Create a ZIP-file called Arm7Bot.zip that contains files from "Arm7Bot" directory.

Then open the Arduino IDE and click to the "Sketch" menu and then *Include Library* > *Add .ZIP Library* and import “Arm7Bot.zip” library.

For the DueFlashStorage Library you downloaded from Step 5, you can import the “DueFlashStorage-master.zip” file directly by the Library Manger.

Return to the *Sketch* > *Import Library* menu. You should now see the libraries at the drop-down menu. They are ready to be used in your sketch. The zip files have been expanded in the libraries folder in your Arduino sketches directory.

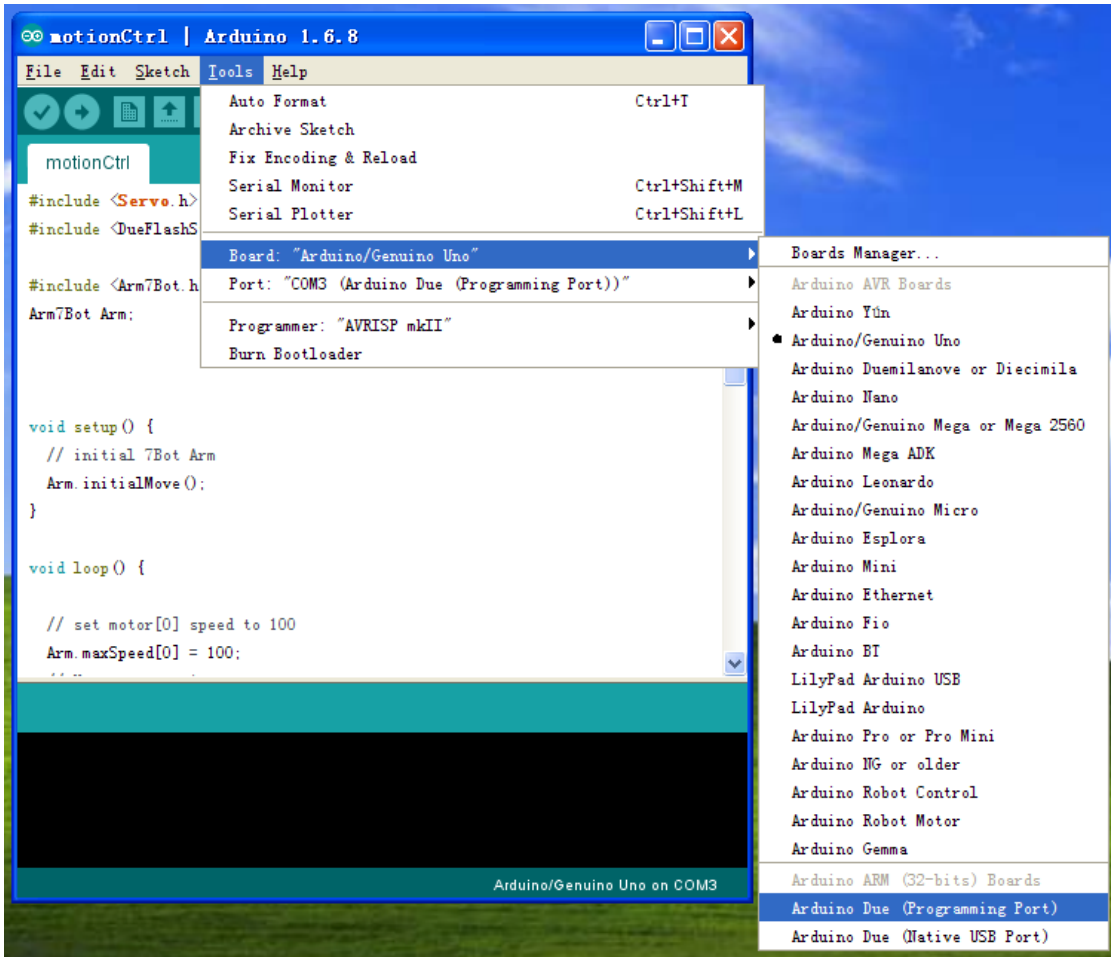


7. Uploading example codes

Connect 7Bot to your computer by the USB cable. You'd better not connect the power supply adapter to the robot while uploading.

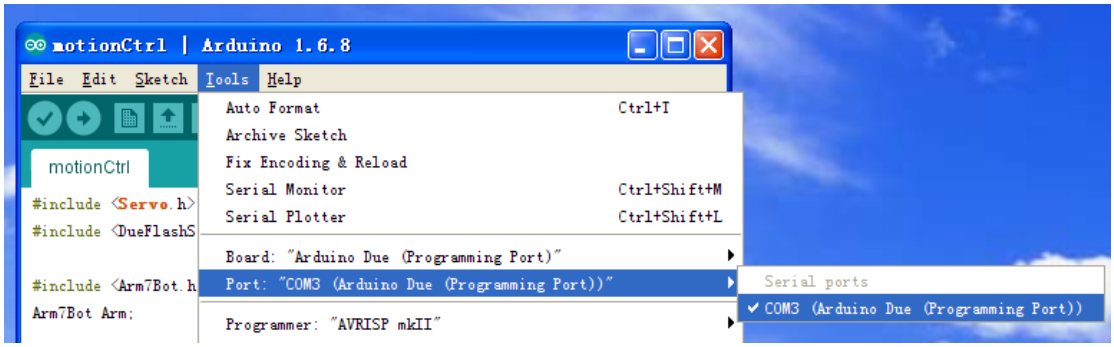
In the Arduino IDE, click 'File' > 'Examples' > 'Arm7Bot' > 'Examples' to open one of the four examples we offered such as 'motionCtrl'

In the Arduino IDE, click ‘Tools’ > ‘Board:…’ > ‘Arduino Due (Programming Port)’ to select the right Arduino board.



Go back to the main menu, click ‘Tools’ > ‘Port:…’ > ‘COM3 (Arduino Due (Programming Port))’ to choose the programming port.

(There might be more than one port in the Port selection, but only one Arduino Due board. The COM number will also be different in different computers.)



Then click ‘Sketch’ > ‘Upload’ and wait while the new firmware is compiled and uploaded to the robot. You can power the robot on, it comes to life.

Now the Arduino IDE and 7Bot libraries are installed. You can rewrite the codes by calling the functions from 7Bot libraries to program 7Bot using Arduino.

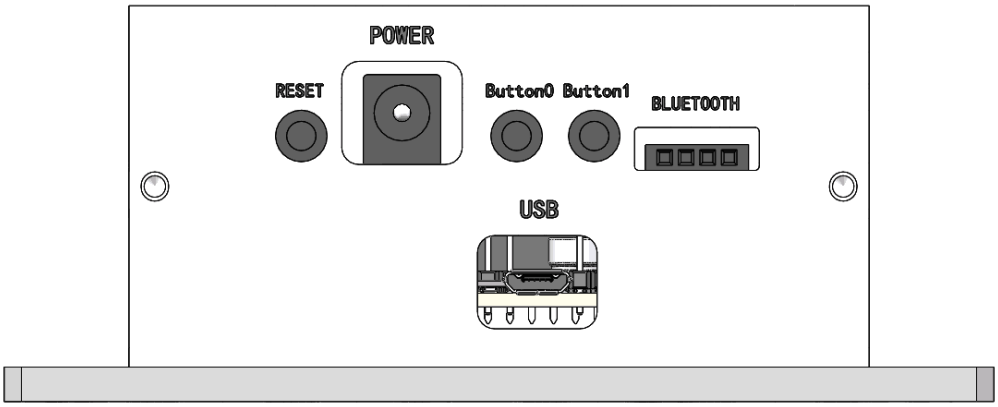
8. Mounting

There are 4 Suction Cup 45-M5 in the package for mounting the robot on the smooth surface. You can use 4 Nut M5 to install the Suction Cup 45-M5 on the base of 7Bot.

9. Panel

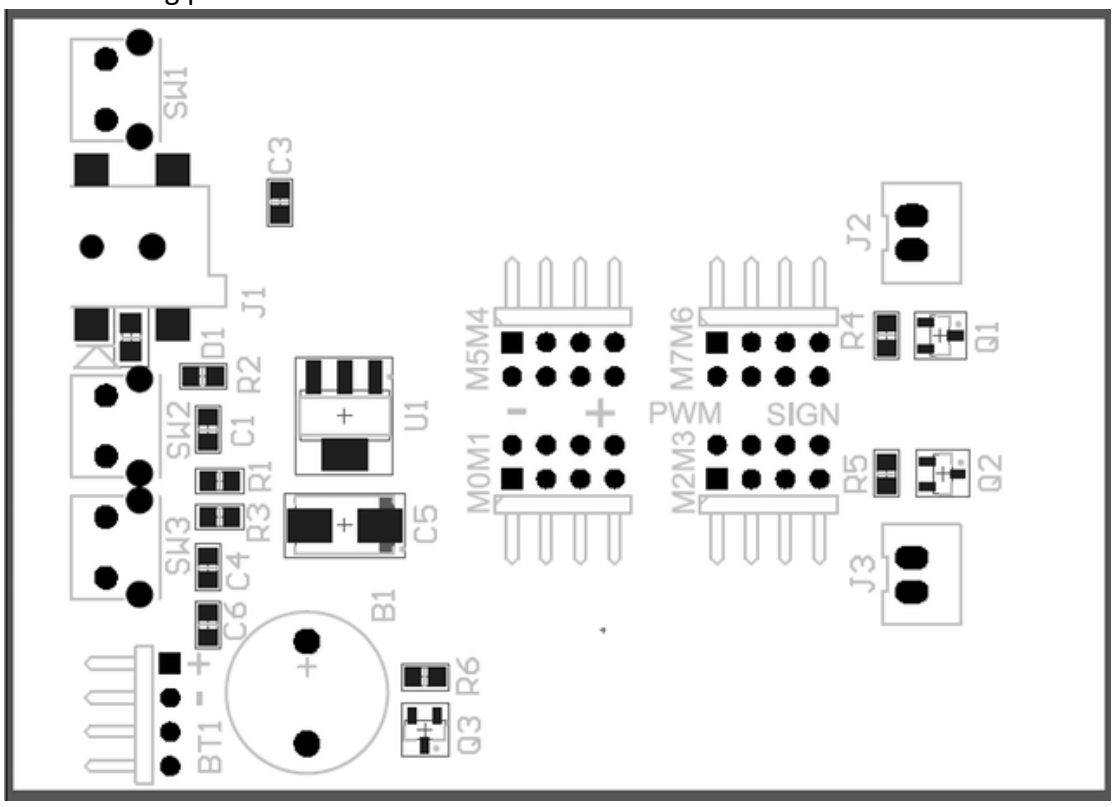
The panel on the base is an Acrylic board with protective paper, so you can remove the protective paper to make the panel look better.

There is a Reset button of Arduino, a Power supply interface, two function buttons, a Bluetooth interface and a USB port on the panel.

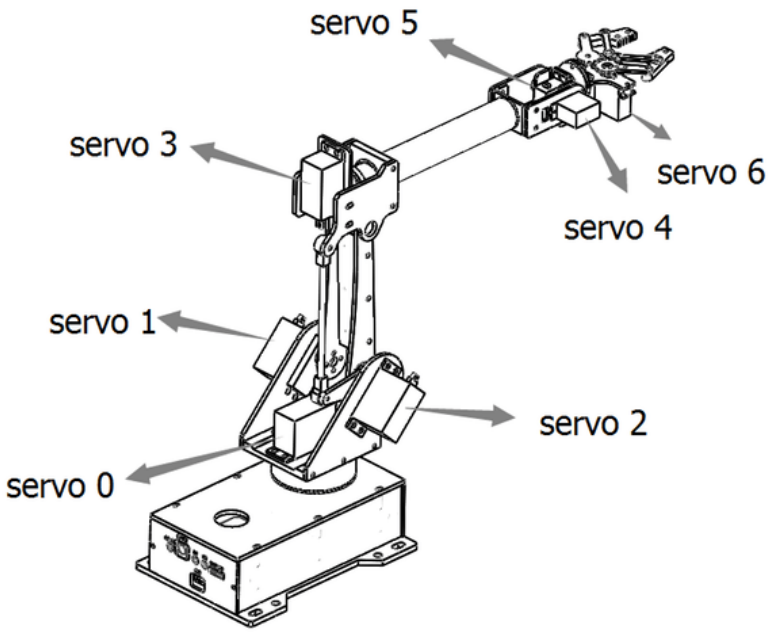


10. Pinout of Arduino Due

The following picture shows the connectors on the Arduino shield.



There are 7 servos of 7Bot from servo0 to servo6 which connected from M0 to M6.



For more details, you can download the pdf file of the Schematic here:

https://www.dropbox.com/s/djbb91gptd7nndk/Arduino_shield_V2.pdf?dl=0

Pinout of Arduino Due we used is:

Axis 1: Servo 0; M0; Readout = A0; ServoControl = D2

Axis 2: Servo 1; M1; Readout = A1; ServoControl = D3

Axis 3: Servo 2; M2; Readout = A2; ServoControl = D4

Axis 4: Servo 3; M3; Readout = A3; ServoControl = D5

Axis 5: Servo 4; M4; Readout = A4; ServoControl = D6

Axis 6: Servo 5; M5; Readout = A5; ServoControl = D7

Axis 7: Servo 6; M6; Readout = A6; ServoControl = D8

Valve: J2; Control = D10; Low is open (suction); High is closed (no suction)

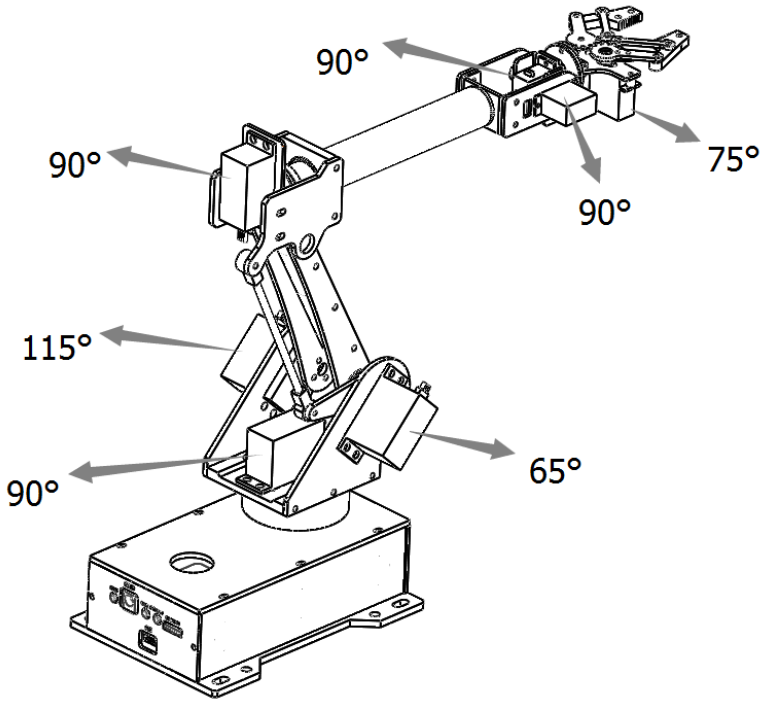
Pump: J3; Control = D11; High is on; Low is off

Beep: Control = D12; High is on; Low is off

Button0: SW2; Readout = D71

Button1: SW3; Readout = D70

The assembling angles of the servos is list below:

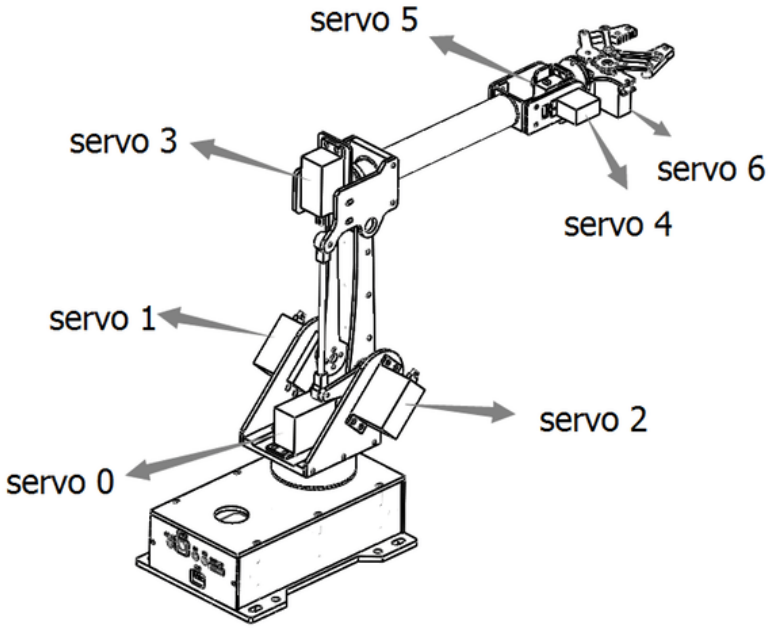


11. End-Effector

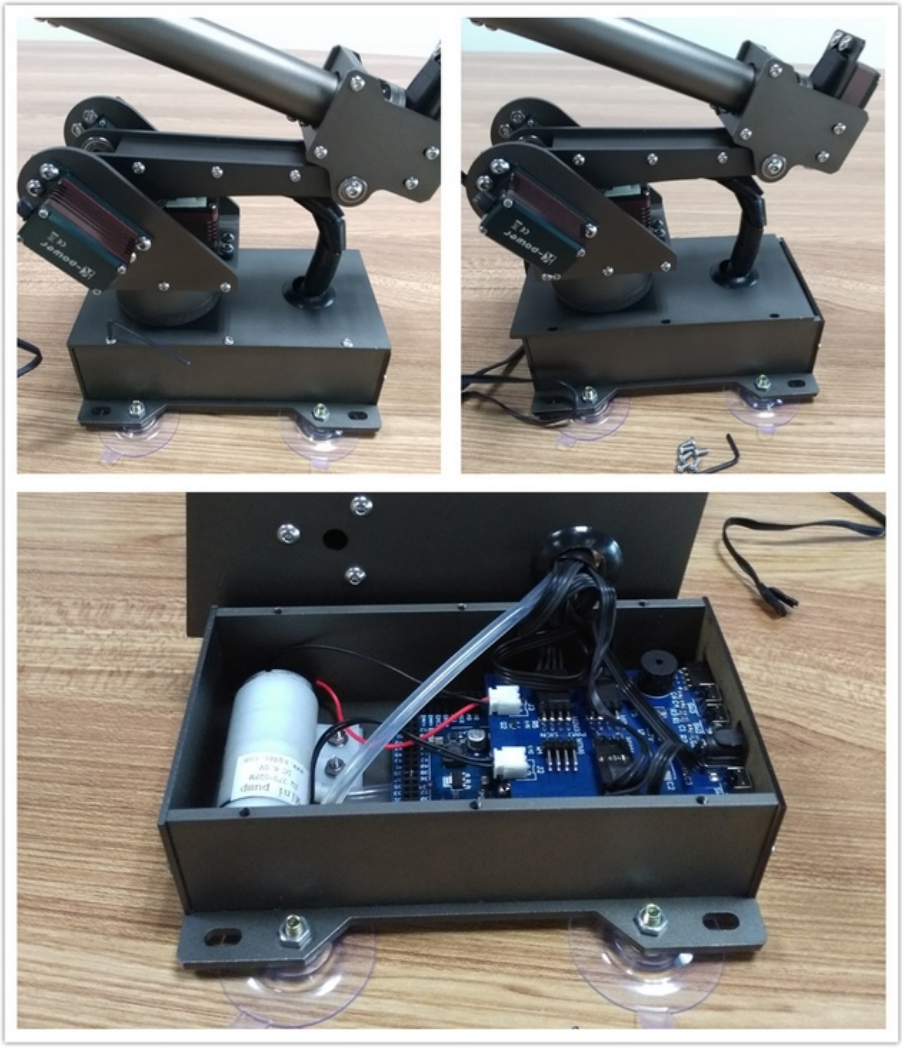
The 7Bot Arm comes with pump and valve inside, so you can easily assemble the Vacuum Cup End-Effector and connect it to the pump to make the Vacuum Cup Gripper work. We offered two different kinds of Vacuum Cup. Choose one and assemble it to the Vacuum Cup Holder. Attach the Vacuum Cup Holder Connector to the end of the arm by using 4 screw M3-7 (there are 5 in most of the kits) with M3 Screw Driver (the small black one). Then install the Vacuum Cup Holder to the Connector and connect the Vacuum Cup to the pump by the silicone tube.

If you want use the Two-Finger Gripper, uninstall the Vacuum Cup Gripper and then install the Two-Finger Gripper with 2 screw M3-7.

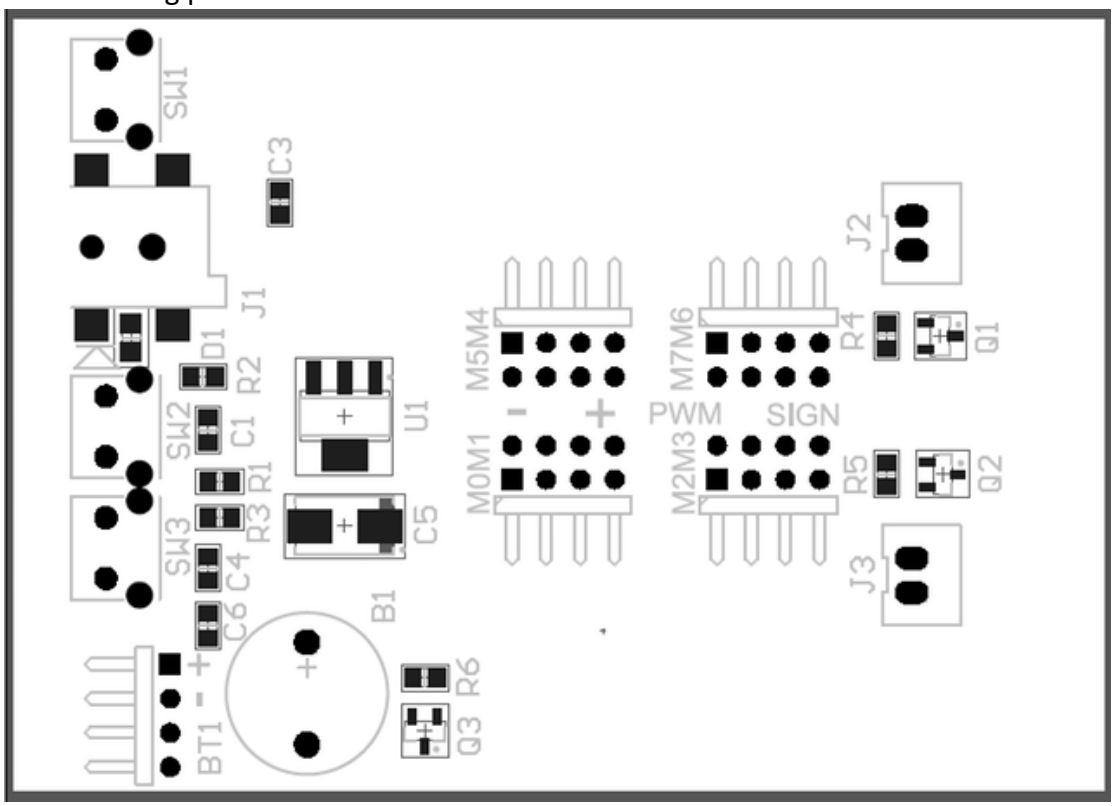
Then you need connect the servo of the gripper to the Arduino shield. There are 7 servos of 7Bot from servo0 to servo6 and the servo on the gripper is servo6.



Open the base of 7Bot by using Screw Driver M3.

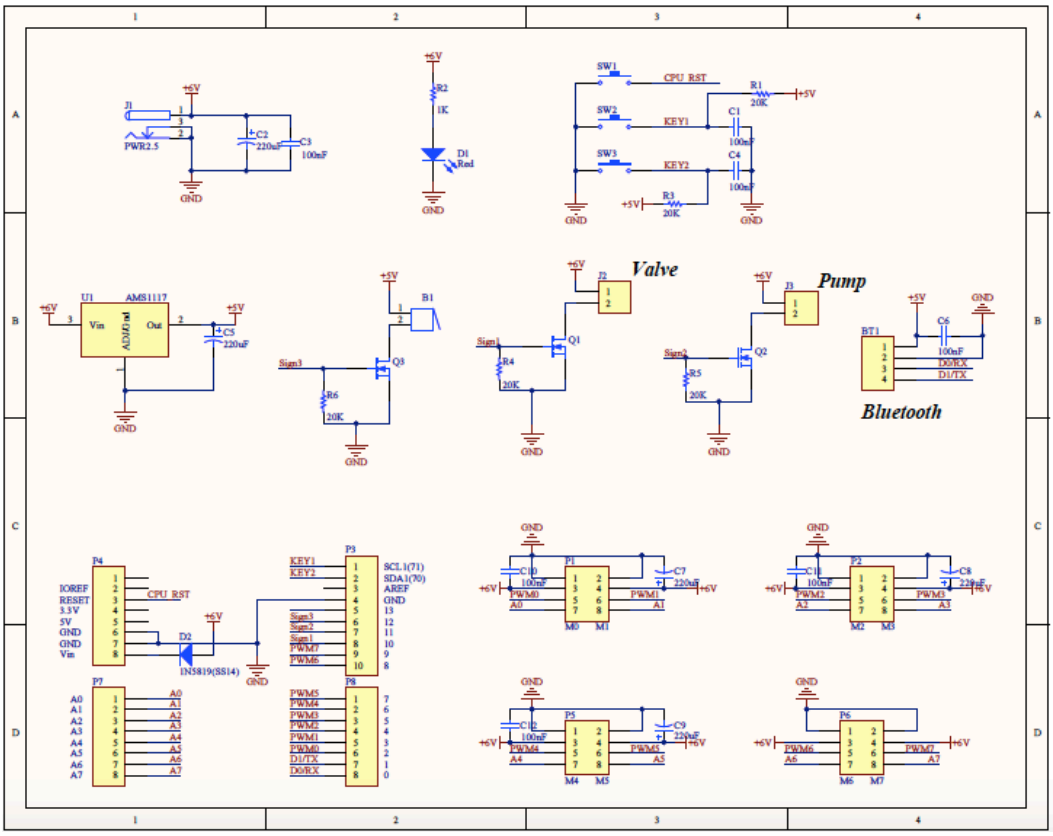


The following picture shows the connectors on the Arduino shield.



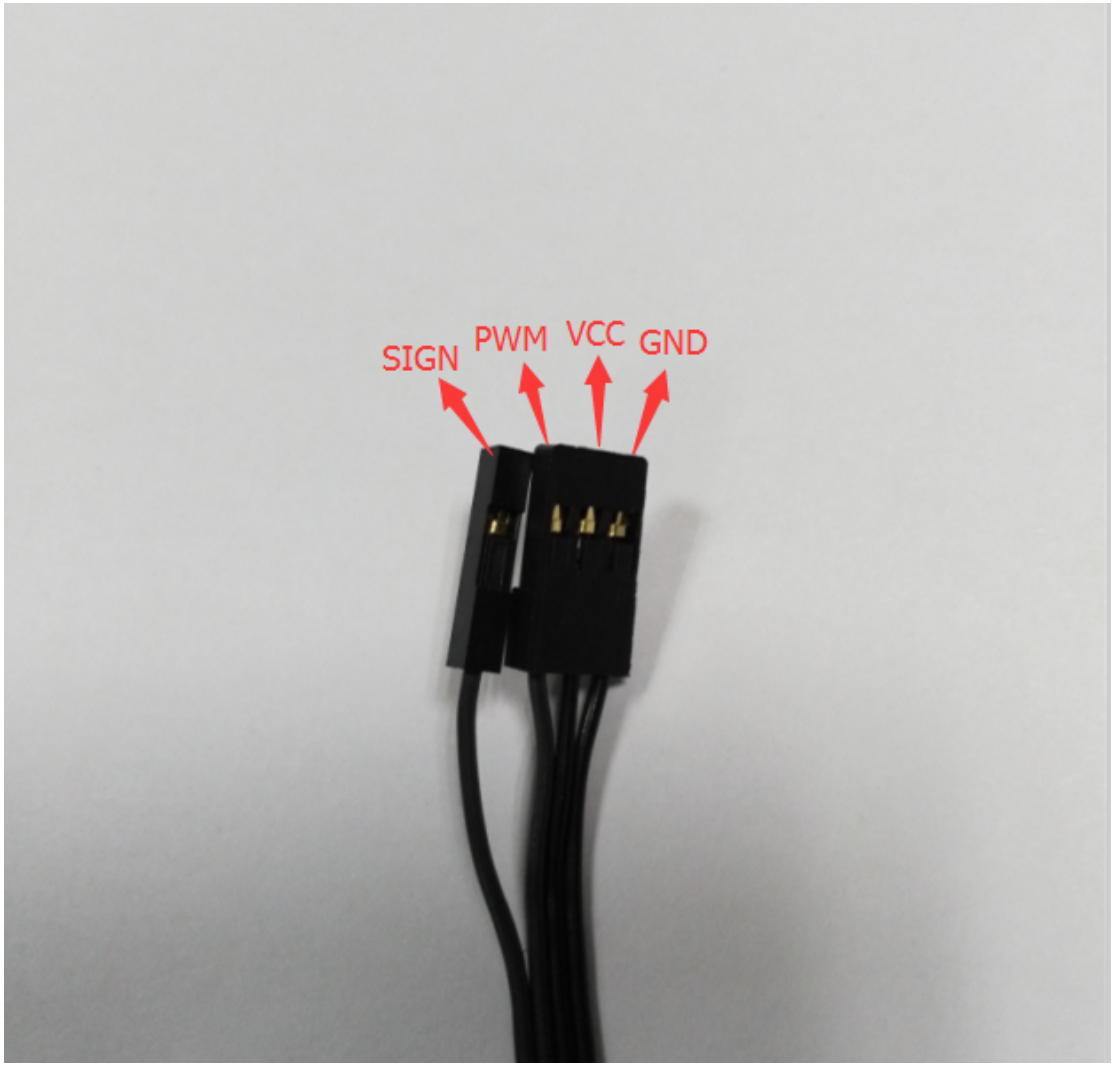
Servo0 connects to M0, servo1 connects to M1 and so on servo6 connects to M6. Pump connects to J3 and Valve connects to J2.

The Schematic of the Arduino shield list below:

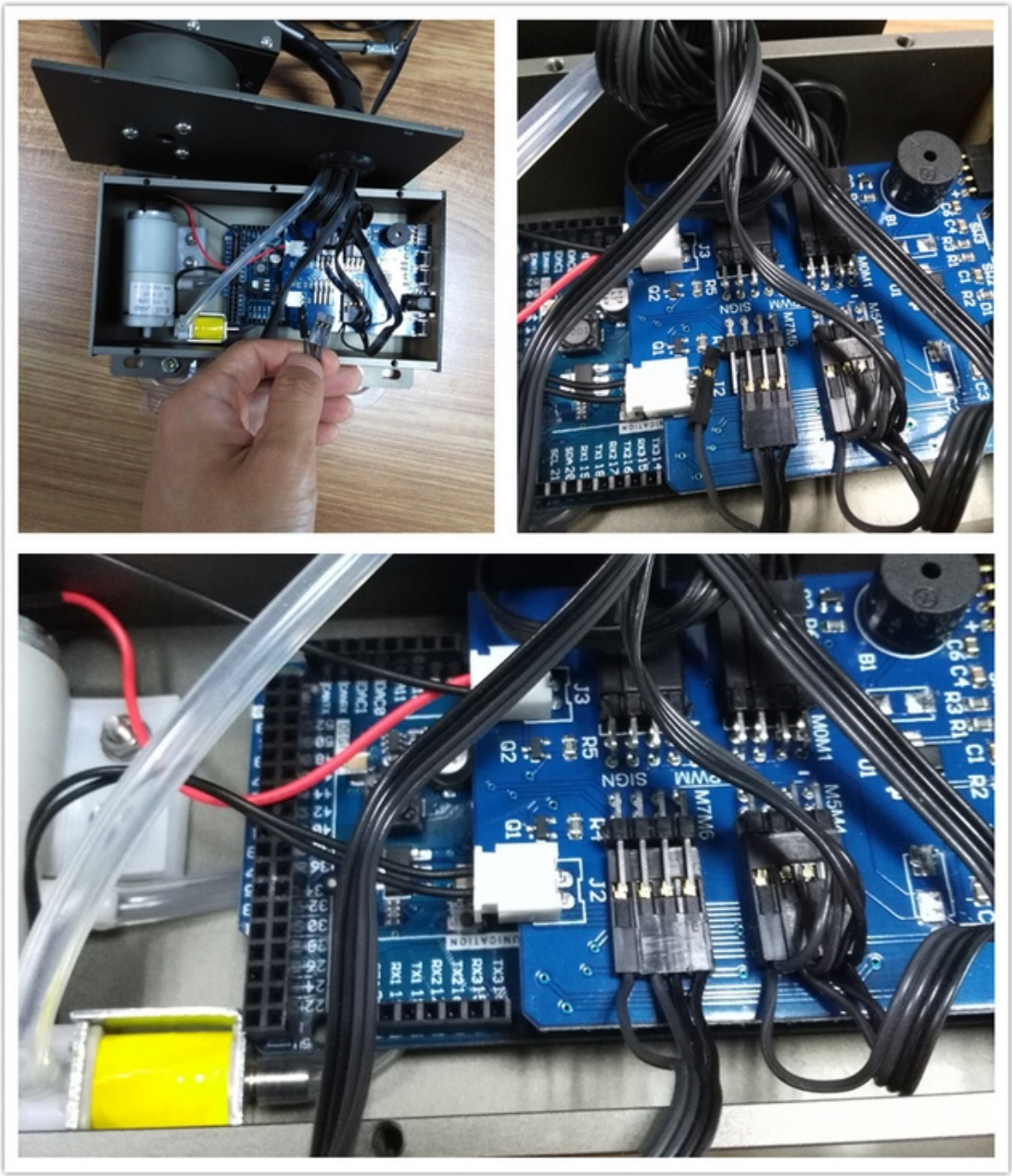


And you can download the pdf file of the Schematic here:
https://www.dropbox.com/s/djbb91gptd7nndk/Arduino_shield_V2.pdf?dl=0

As you can see the servo of on the robot has 4Pin with feedback signal. The divided Pin is the signal Pin, the next is the PWM Pin, then the VCC Pin and the GND Pin.

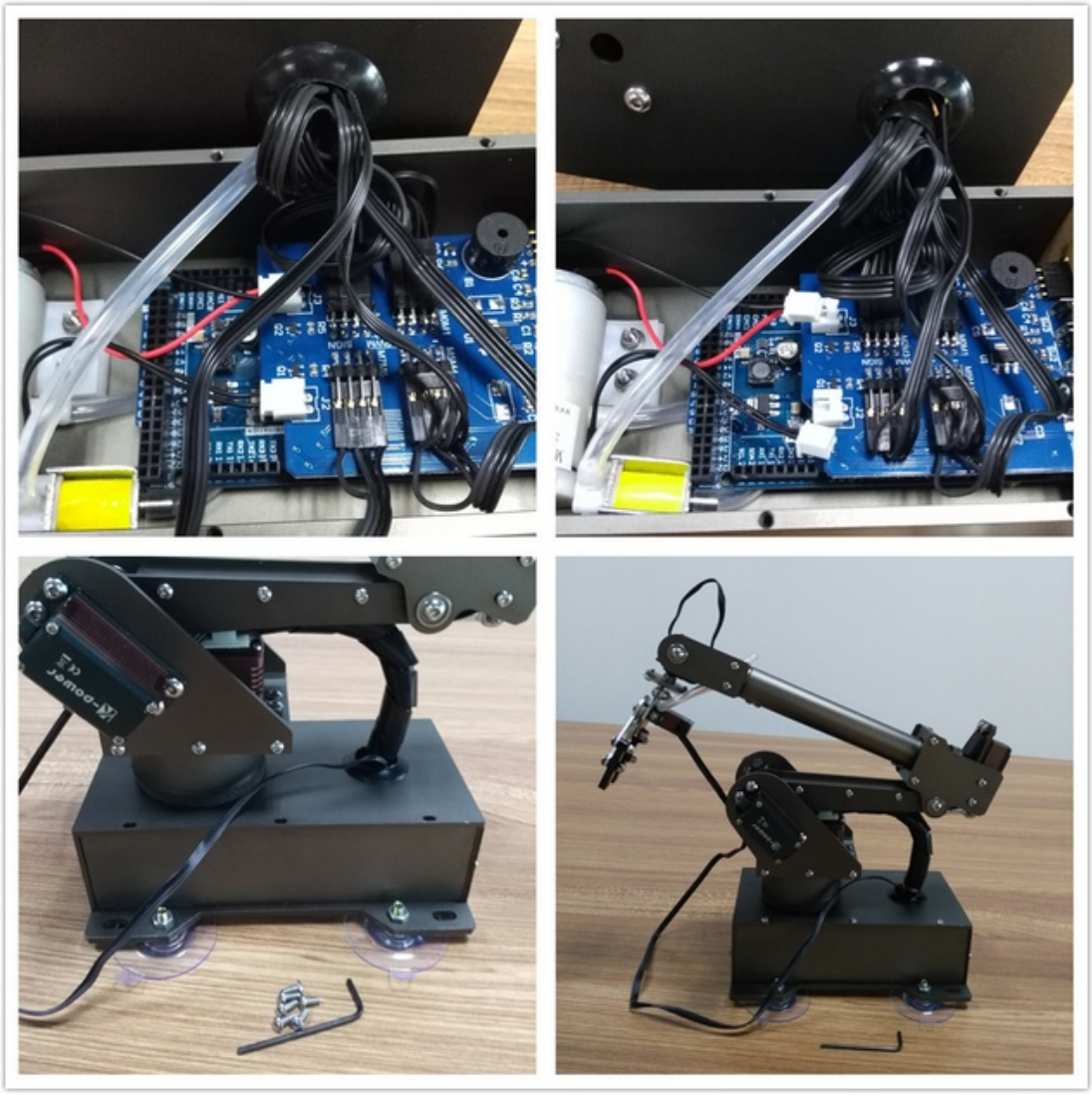


Connect servo6 of the Gripper to the Arduino shield with right Pins.



Then unplug the connectors of Pump and Valve and close the base.

After that, the installation of the Two-Finger Gripper is done.



12. Public Functions of 7Bot Arduino Library

In 7Bot Arduino Library, we offered some public functions for you to recall and program your 7Bot Arm.

1) double maxSpeed[SERVO_NUM];

Set running speed to proper value of each servo. The unit is degrees/second and the max speed is 250 degrees/second. Too high may cause unstable.

2) boolean isFluent[SERVO_NUM];

Set fluent movement of each servo. 'True' is moving to the target angle fluent. And 'False' is moving to the target angle directly.

3) double offset[SERVO_NUM];

Assembly offsets, set offset value of each servo. See more details from the 'calibration' example codes. We will release calibration software UI for this function in our next getting started version. This calibration one can compensate the offsets between servo angles and structure angles that come from assembly. Every individual 7Bot have a little bit different from each other. So your can change these offset values a little to make your 7Bot works much precise to its geometry posture.

4) boolean allConverge();

Check whether every servo finished moving to target angle. 'True' is done and 'False' is not.

5) void initialMove();

Read current position first, and then adjust to the initial position softly and gradually.

6) void forcelessMode();

Set 7Bot to forceless mode, which you can drag 7Bot by hand easily and also read stable pose feedbacks.

7) void stopMode();

This mode is useful for device protection, robot can be moved by external forces, but still have some resistance.

8) void move(double angles[SERVO_NUM]);

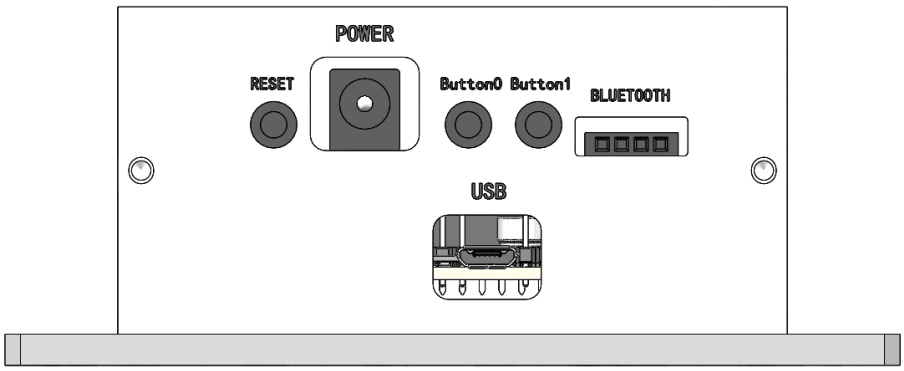
Given each axis angle(Unit:degrees). For axis 7, you both control the Two-Finger Gripper and Vacuum Cup Gripper. For Vacuum Cup Gripper, when the value is lower than 30, the pump and valve on to grab. When the value is higher than 30, the pump and valve off to release. The angle of each servo is from 0 to 180.

9) void softwareSystem();

This fuction is design for general usage and software application developments.

13. Firmware Usage

After you upload the 'softwareSystem' Arduino example to your 7Bot, you can use it by just power on. On its left side is the reset button, and two functional buttons on its right side, we call them Button0 & Button1.



If you want set up a new program without coding by recording postures and replaying. You can follow the steps below:

1) Power on
2) Click reset button, this robotic arm will move to its initial posture softly.
3) Long press Button0 (while holding arm) till you hear a long "beep" to enter record mode. Please hold the tube of the robot as it will be forceless after the long press on Button0.
4) Long press Button0 again till you hear a long "beep" to clear current list of poses you have recorded last time.
5) The arm is forceless, so you can easy move it and set all joins to next pose you need.
6) If this is a movement-only post, short press Button1 till you hear a short "beep". If grab/release pose, long press Button1 till you hear a long "beep". (The grab/release pose will both work for vacuum cup and two-finger gripper. When you first give a long press on Button1, it will grab, then second time, it will release. Grab and release change alternately.)
7) Repeat steps 4 & 5 for additional poses.
8) When complete, short press Button0 to enter replay mode. (Replay mode runs each recorded pose with 1 second pauses between poses and then repeats)
9) To add more poses, long press Button0.
10) To stop replay, press the reset.

If you want use this firmware record and play well. You need take some practice. The

time interval between two postures is 1 second. You can add more postures to make the motion slowly or stop somewhere for a few seconds until next step.

Even though you power off 7Bot, the recorded posture will not disappear. Power on whenever you want, the postures are still there. The only way to clear them is in forceless state and gives a long press on Button0.

14. Communication Instruction

The majority reason we develop 7Bot is to make AI & CV applications. So a lot of sophisticated algorithms should run on a powerful processor. Usually we use PC to do these stuffs. And after you upload the softwareSystem Arduino examples onto your device, you can use the communication protocol to control the robot.

For more details, you can download the Communication Instruction file here:

https://www.dropbox.com/s/pytci253ql5lmey/Communication%20Instruction%20%28v1.0.1%29.pdf?dl=0

And in this instruction, we add some explanation of 7Bot coordinate system, which will give a sense of how to use the Inverse Kinematics functions for this robot.

We have given out a Processing example to illustrate how to write communication protocols in Java. You can download the example code here:

https://github.com/7Bot/7Bot-Processing-Examples

If Processing is new to you, you can download the software here:

https://processing.org/download/

You can also learn more about Processing on the Processing official website. And remind one thing about processing IDE here, we found the new Processing3 is not stable right now. So we recommend you use Processing2 for a while.

For the example code, one thing you should take note here is the COM port ID on you computer may be different. Running this example you need change it according to your own situation.

```
void setup()
{
  // Open Serial Port: Your should change PORT_ID accoading to
  // your own situation.
  // Please refer to: https://www.processing.org/reference/libraries/serial/Serial.html
  int PORT_ID =  3;
  myPort = new Serial(this, Serial.list()[PORT_ID], BAUD_RATE);
```

If the Arduino Due Programming Port is the only one COM port in your computer, the port ID here should be 0. Then change the code to 'int PORT_ID = 0; '. If there are N COM ports in your computer, the port ID here should be N-1.