PWM – Servo Control

Preparation

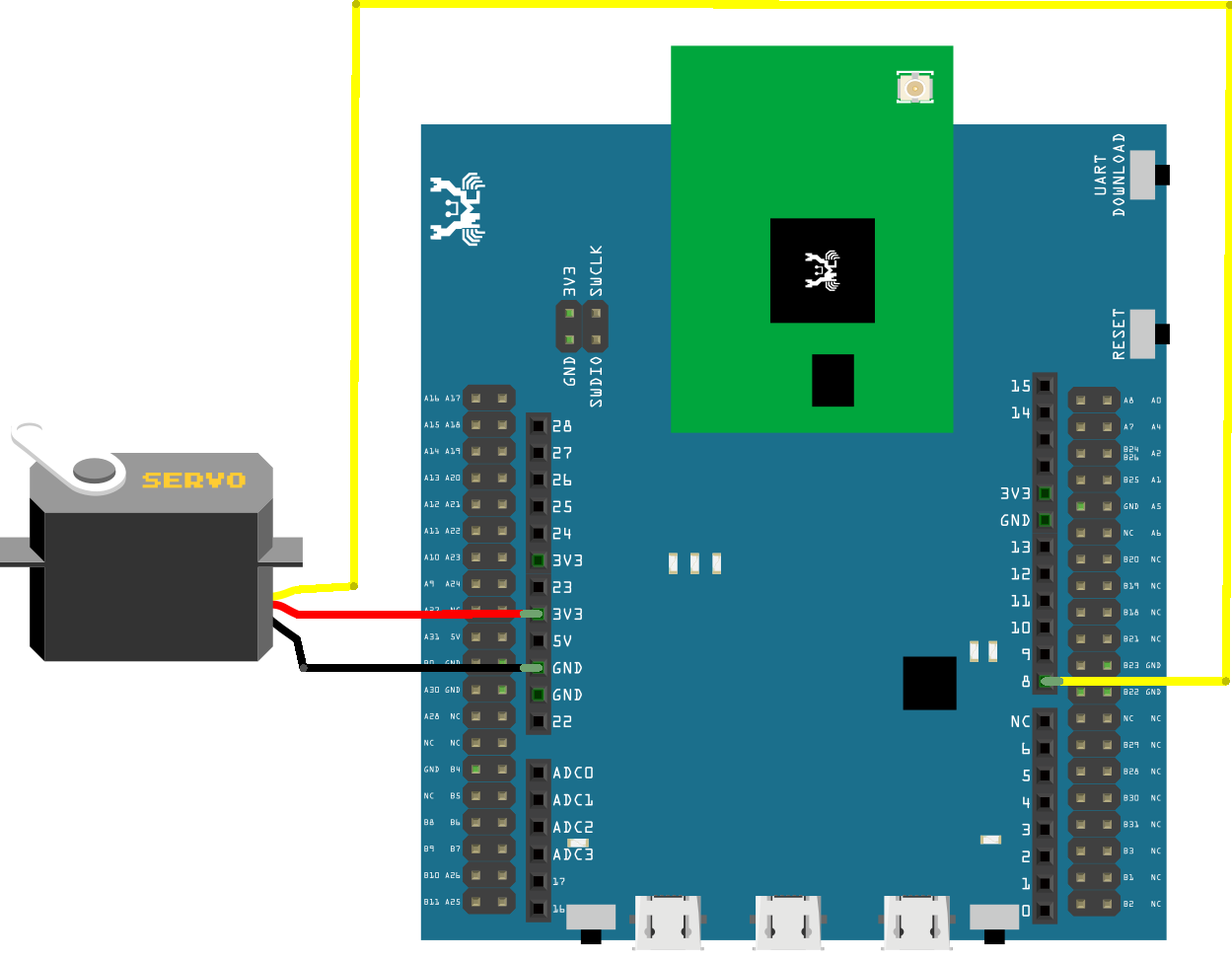
* AmebaD [ [AMB23](https://www.amebaiot.com/amebad/#rtk_amb23) / [AMB21](https://www.amebaiot.com/amebad/#rtk_amb21) / [AMB22](https://www.amebaiot.com/amebad/#rtk_amb22) / [BW16](https://www.amebaiot.com/amebad/#partner_bw16) ] x 1
* Servo x 1 (Ex. Tower Pro SG90)

Example

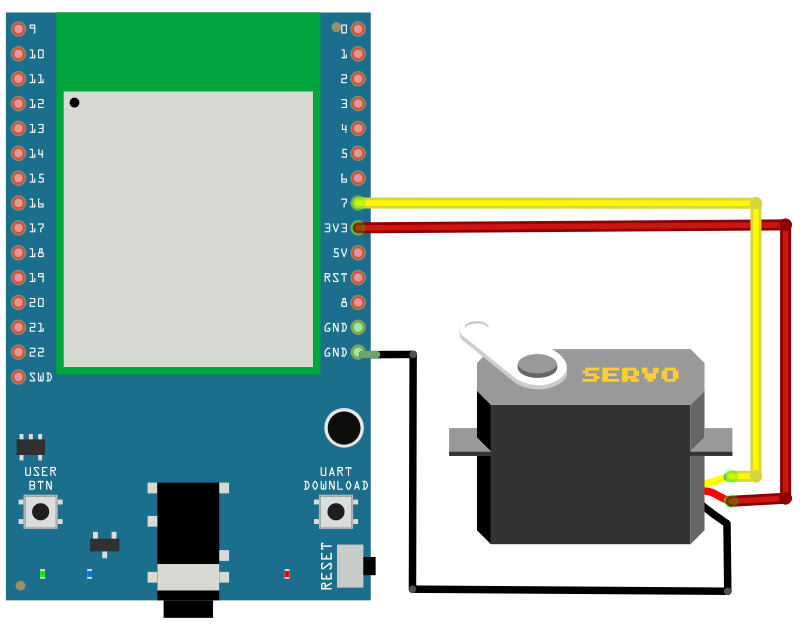
A typical servo has 3 wires, the red wire is for power, black or brown one should be connected to GND, and the other one is for signal data. We use PWM signal to control the rotation angle of the axis of the servo. The frequency of the signal is 50Hz, that is length 20ms. Each servo defines its pulse bandwidth, which is usually 1ms~2ms.

To control the rotation angle, for example if 1ms-length pulse rotates the axis to degree 0, then 1.5 ms pulse rotates the axis to 90 degrees, and 2 ms pulse rotates the axis to 180 degrees. Furthermore, a servo defines the “dead bandwidth”, which stands for the required minimum difference of the length of two consecutive pulse for the servo to work.

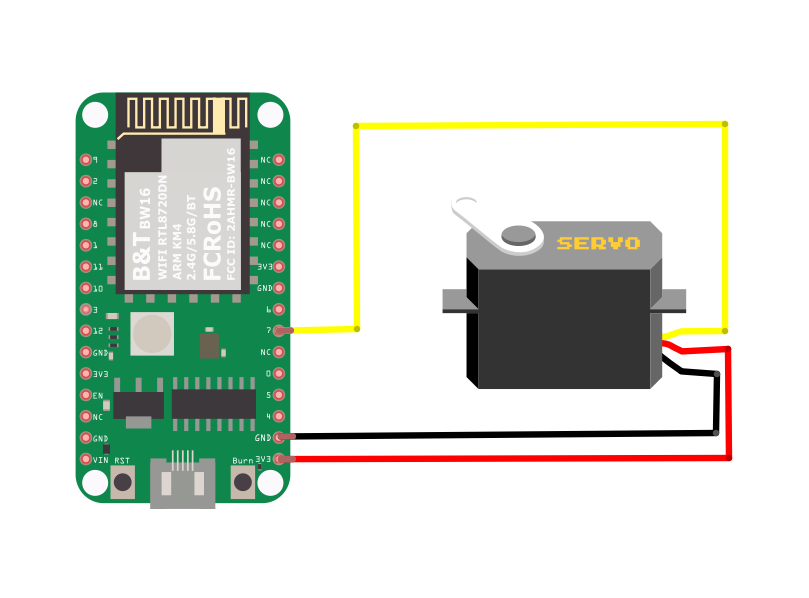
**AMB21 / AMB22**wiring diagram:



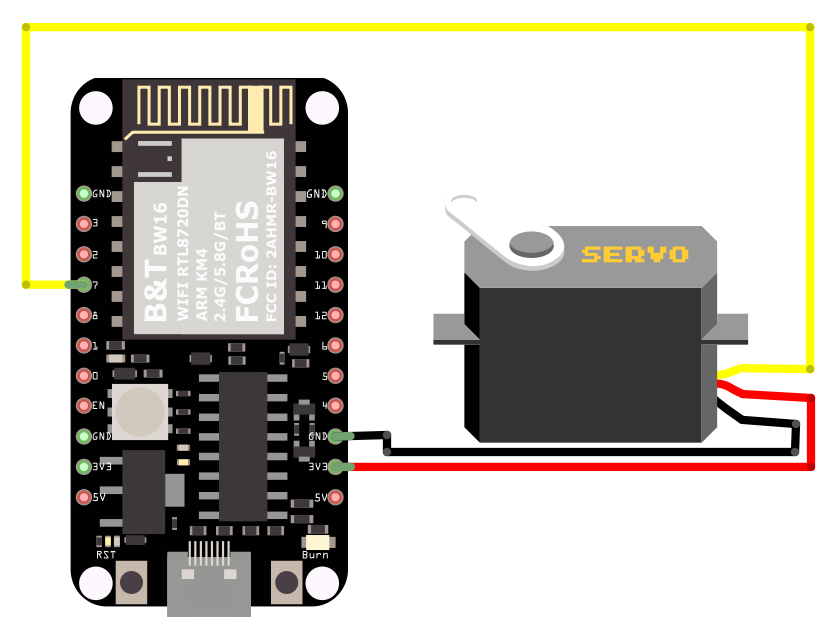
**AMB23**wiring diagram:



**BW16** wiring diagram:



**BW16-TypeC** wiring diagram:



Open the example, “File” -> “Examples” -> “AmebaAnalog” -> “ServoSweep”  
This example makes the servo to rotate from degree 0 to 180, and then rotate back to degree 0.

Code Reference

The Servo API of Ameba is similar to the API of Arduino. To distinguish from the original API of Arduino, we name the header file “AmebaServo.h” and the Class “AmebaServo”, the usage is identical to the Arduino API.

The default pulse bandwidth of Arduino Servo is 0.5ms~2.4ms, which is the same as Tower Pro SG90. Therefore, we set the attached pin directly:

myservo.attach(9);

Next, rotate the axis to desired position:

myservo.write(pos);