Project 2: Adjust LED Brightness

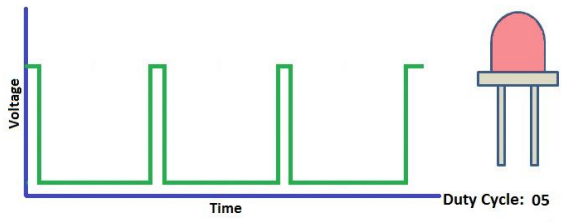
1.Description

In previous lesson, we control LED on and off and make it blink.

In this project, we will control LED’ s brightness through PWM simulating breathing effect.

PWM is a means of controlling the analog output via digital means. Digital control is used to generate square waves with different duty cycles (a signal that constantly switches between high and low levels) to control the analog output. In general, the input voltages of ports are 0V and 5V.

What if the 3V is required? Or a switch among 1V, 3V and 3.5V? We cannot change resistors constantly. For this reason, we resort to PWM.

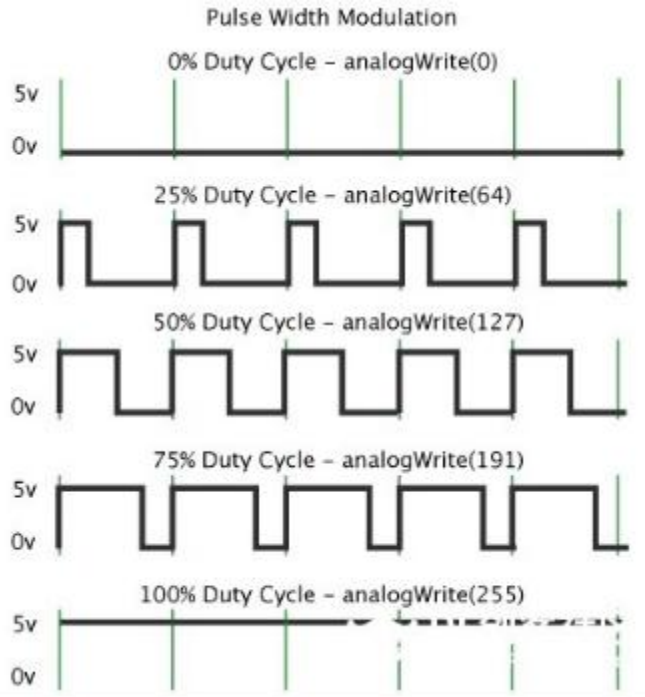


For the Arduino digital port voltage output, there are only LOW and HIGH, which correspond to the voltage output of 0V and 5V. You can define

LOW as 0 and HIGH as 1, and let the Arduino output five hundred 0 or 1 signals within 1s.

If all of the output five hundred are 1, that is 5V; if all of which are 0, that is 0V. If output 010101010101 in this way then the output port is 2.5V, which is like showing movie. The movie we watch are not completely continuous. It actually outputs 25 pictures per second. In this case, the human can’ t see it, neither does PWM. If we want different voltage, we need to control the ratio of 0 and 1. The more 0,1 signals output per unit time, the more accurate the control.

PWM is a technology that uses digital methods to obtain analog quantities. Digital control enables to form a square wave, the square wave signal only has on and off two states (high and low). A voltage ranging from 0 to 5V can be simulated by controlling the ratio of on to off duration. The time spent on (technically called high level) is called pulse width, so PWM is also called pulse width modulation.



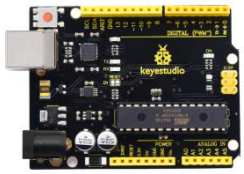
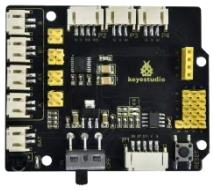
The green vertical bars represent one period of the square wave. The value written in each analogWrite(value) corresponds to a percentage, which is also called Duty Cycle. This percentage refers to the ratio of time occupied by the high level in a cycle, that is, duty cycle = high level time/cycle time.

In the figure, from top to bottom, the duty cycle of the first square wave is 0%, and the corresponding value is 0, and the LED brightness is the lowest, that is, off state. The longer the high level lasts, the brighter it will be. Therefore, the value of the last duty cycle of 100% is 255, and the LED is the brightest. 50% is the brightest half, and 25% is darker.

PWM is more used to adjust the brightness of LED lights or the rotation speed of the motors, and the wheels speed driven by the motors can be

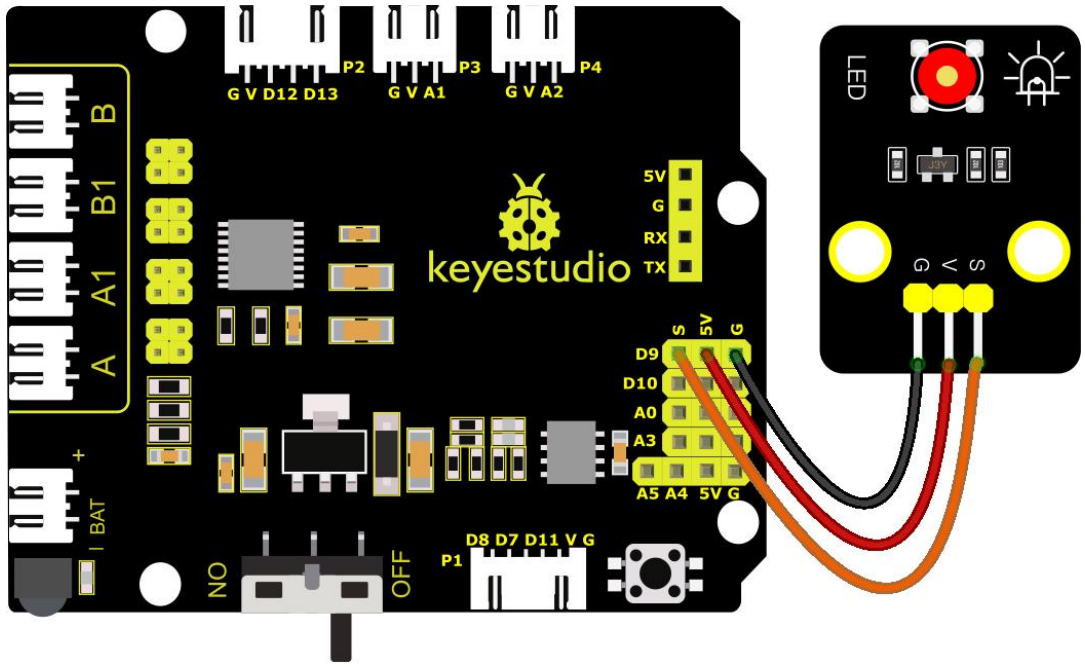
easily controlled. When playing with some Arduino robots, the benefits of PWM can be better reflected.

2.Components

Keyestudio 4.0 Development Board *1	Keyestudio 8833 Motor Driver Expansion Board *1	Red LED Module*1
		
3P F-F Dupont Wire*1	USB Cable*1	
		

3.Wiring Diagram

Keep the wiring-up unchanged.



4.Test Code

You can drag blocks to edit. Blocks listed below are for your reference

（1）



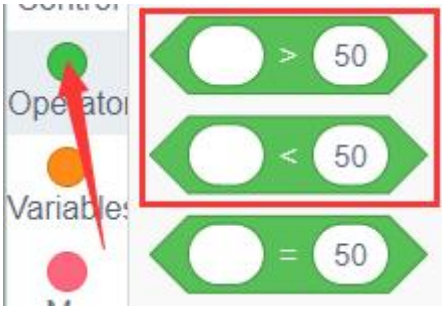
（2）



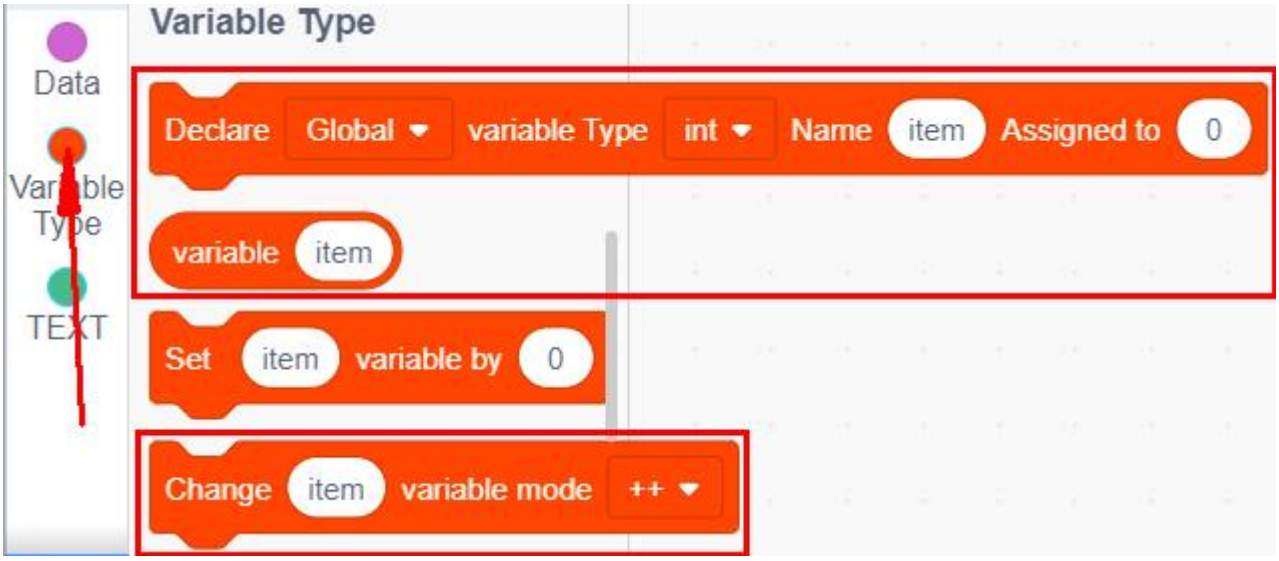
（3）

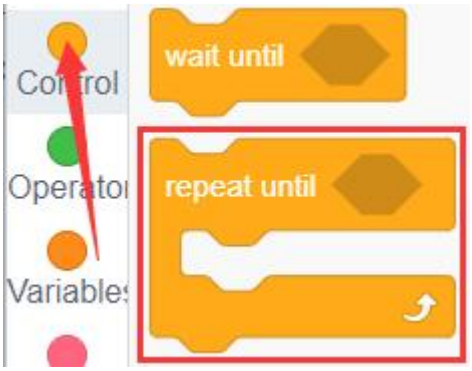

Control
Events
Control
Operator
Variables
My Blocks
Pins
wait 1 seconds
repeat 10
forever

（4）

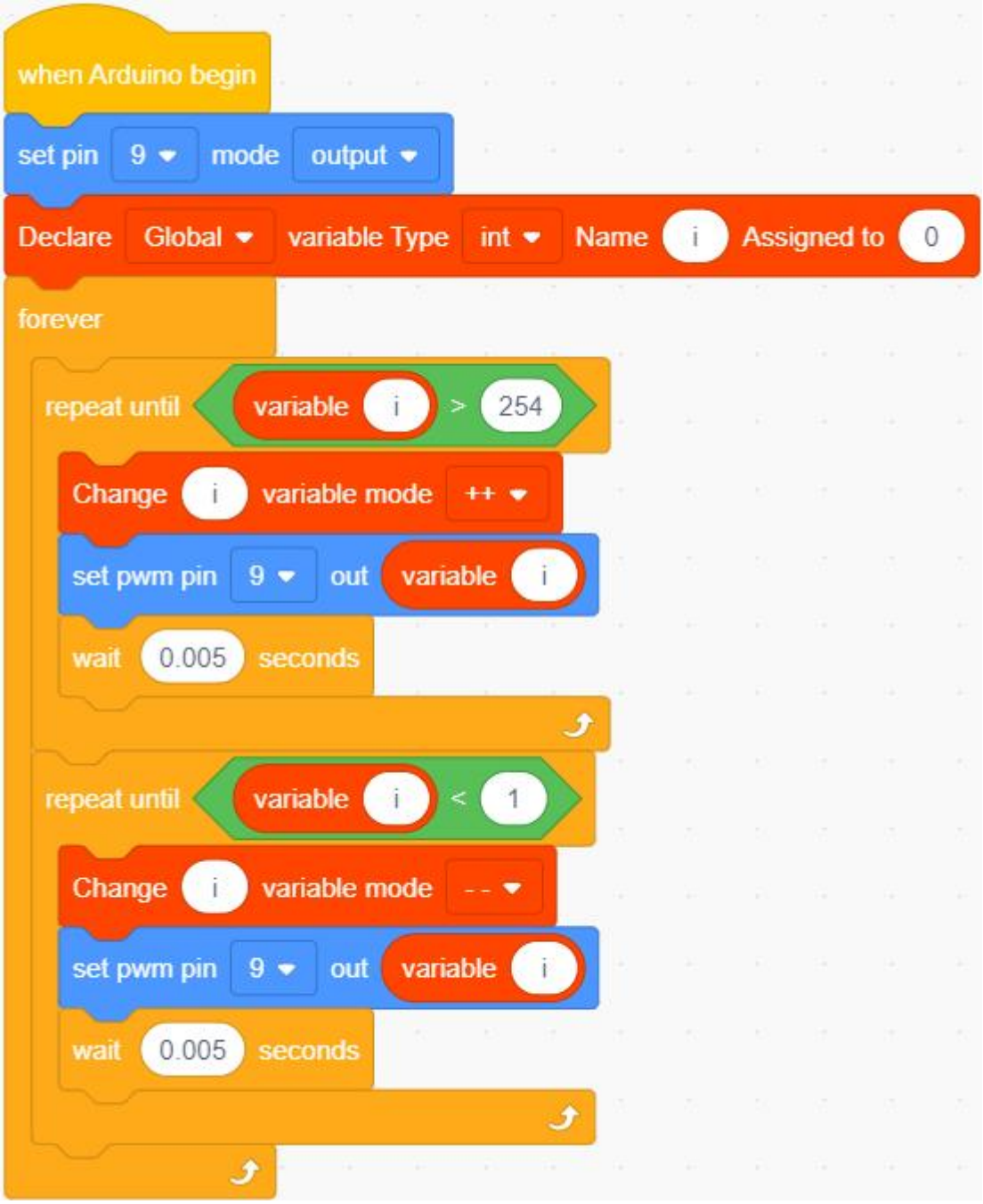

Operator
Variables
> 50
< 50
= 50

（5）


Variable Type
Data
Variable Type
TEXT
Declare Global variable Type int Name item Assigned to 0
variable item
Set item variable by 0
Change item variable mode ++

（6）

Complete Test Code



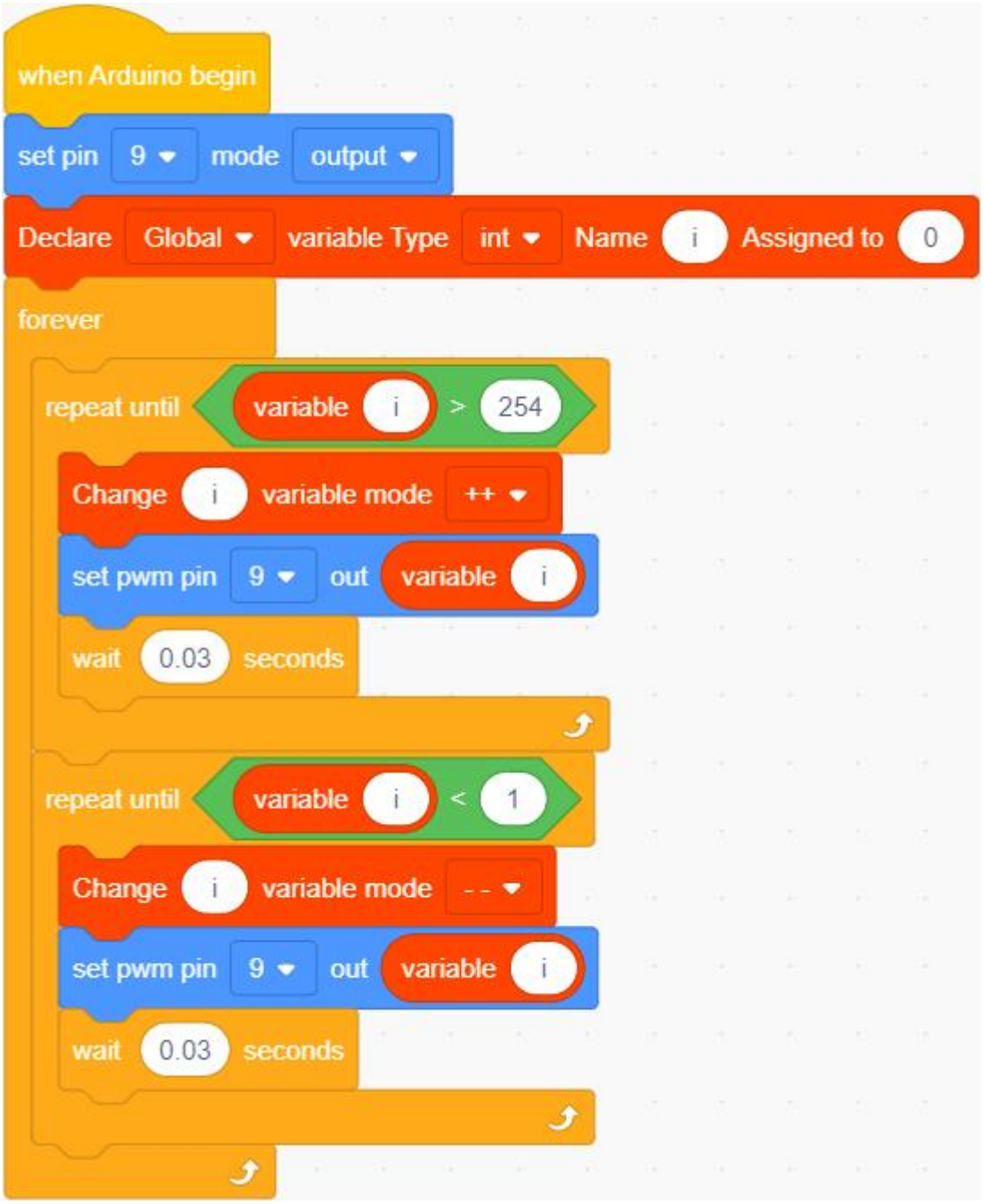
5.Test Result

After successfully uploading the code to the V4.0 board, connect the wirings according to the wiring diagram, and use a USB cable to connect the computer to power the board. After powering on, you will see that

the LED gradually changes from bright to dark, like human' s breath, rather than turning on and off immediately.

6.Extension Practice

Keep pins of the LED unchanged, then change code(values behind wait)



Upload the code to the development board, then the LED will blink more slowly.