



Smart Home Kit for Micro:bit

(Python)

Contents

Smart Home Kit for Micro:bit1
(Python)1
1.Introduction:9
2.Description:
3.Preparations:
3.1Background Information about Micro:bit
(1)What is Micro:bit?11
(2)Layout13
(3) Pinout14
(4)Notes for the application of Micro:bit main board15
3.2.Install Micro:bit driver17
4.Python
4.1.Python
4.2.Mu19
5. Projects:
Project 1: Heartbeat29
(1)Project Introduction29
(2)Preparations:29





	(3)Test Code:	
	(4)Test Results:	40
	(5)Code Explanation:	40
Pro	ject 2: Light A Single LED	41
	(1) Project Introduction	41
	(2)Preparations:	41
	(3)Test Code:	42
	(4)Test Results:	45
	(5)Code Explanation:	46
	(6)Reference	47
Pro	oject 3: LED Dot Matrix	47
	(1) Project Introduction	47
	(2)Preparations:	48
	(3)Test Code:	48
	(4)Test Results:	51
	(5)Code Explanation:	52
	(6) Reference:	53
Pro	ject 4: Programmable Buttons	53
	(1) Project Introduction	53
	(2)Preparations:	54
	(3)Test Code1:	54
	(4)Test Results1:	57





(5)Test Code2:57
(6)Test Results2:61
(7)Code Explanation:62
Project 5: Temperature Detection64
(1) Project Introduction64
(2)Preparations:64
(3)Test Code1:65
(4)Test Results1:68
(5)Test Code2:69
(6)Test Results2:72
(7)Code Explanation:73
Project 6: Geomagnetic Sensor74
(1) Project Introduction74
(2)Preparations:75
(2) Test code1::75
(4)Test Result1:78
(5)Test code2:79
(6)Test Results2:84
Project 7: Accelerometer87
(1) Project Introduction87
(2)Preparations:88
(3)Test Code1:





(4)Test Results1:91
(5)Test code2:93
(6)Test Results2:97
(7)Code Explanation:98
Project 8: Light Detection
(1) Project Introduction100
(2)Preparations:100
(3)Test Code:
(4)Test Results:
(5)Code Explanation:104
Project 9: Speaker
(1) Project Introduction105
(2)Preparations:105
(3)Test Code1:
(4)Test Results1:
(5)Test Code2:108
(6) Test Results2:
(7)Code Explanation:113
Project 10: Touch-sensitive Logo 115
(1) Project Introduction115
(2)Preparations:116
(3)Test Code:116





(4)Test Results:
Project 11: Microphone121
(1) Project Introduction121
(2)Preparations:121
(3)Test Code:121
(4)Test Results1:
(5)Test Code2:124
(6)Test Results2:
(7)Code Explanation:128
Project 12: Touch-sensitive Logo Controlled Speaker
(1) Project Introduction130
(2)Components Needed:130
(3)Connection Diagram:130
(4)Test Code:
(5)Test Results:
(6)Code Explanation:134
Project 13: Bluetooth Wireless Communication
7.Expansion Projects:
Project 1: LED Blinks136
(1)Project Introduction136
(2)About the Yellow LED:138
(3)Test Code





	(4)Test Results:
	(5)Code Explanation:143
Pro	ject 2: Breathing LED144
	(1) Project Introduction144
	(2)About the Yellow LED:
	(3)Test Code
	(4)Test Results:
	(5)Code Explanation:152
Pro	ject 3:6812 2x2 Full Color RGB152
	(1)Project Introduction152
	(2)About the 6812 2x2 Full-color RGB:153
	(3)Test Code1 154
	(4)Test Results1:
	(5)Test Code2:
	(6)Test Results2:164
	(7)Test Code3:
	(8)Test Results3:
	(9)Code Explanation:168
Pro	ject 4: PIR Motion Sensor170
	(1)Project Introduction170
	(2)About PIR Motion Sensor:
	(3)Test Code:





(4)Test Results:
(5)Code Explanation:177
Project 5: Induction Lamp
(1)Project Introduction178
(2)Test Code:
(3)Test Results:
(4)Code Explanation:182
Project 6: Servo
(1) Project Introduction184
(2)Working Principle of Servo:184
(3)About the Servo:186
(4)Test Code:
(5)Test Results:
Project 7: 130 Motor 191
(1)Project Introduction191
(2)Parameters:
(3)Test Code 1: (high/low level control)
(4)Test Code2: (PWM control)196
(5)Test Results:
Project 8: Lithium Battery Power Module
(1)Project Introduction199
(2) Parameters:





(3)Schematic Diagram:201
(4)Features:
Project 9: 1602 LCD
(1)Project Introduction204
(3) About 1602 I2C:
(3)Test Code:
(4)Test Results:
Project 10: Steam Sensor
(1)Project Introduction214
(2)About the Stream Sensor:215
(3)Test Code:
(4)Test Results:
Project 11: Rains Alarm 220
(1)Project Introduction220
(2)Test Code:
(3)Test Results:
Project 12: Analog Gas (MQ-2) Sensor 225
(1)Project Introduction225
(2)About Analog Gas Sensor (MQ-2):226
(3)Test Code:
(4)Test Results:
Project 13: Gas Leakage Detector232





(1) Project Description:	232
(2)Test Code:	233
(3)Test Results:	241
Project 14: Multiple Functions	242
(1)Project Description:	242
(2)Test Code:	242
(3)Test Results:	255
Resources:	256

1.Introduction:

8.

Fueled by the rapid development of technology, smart homes automatically controlled remotely by smart phones and other devices have become more common. For the same reason, they have increasingly gained closer attention and caught people 's fancy.

Bearing the aim to make improvements in household living conditions, the smart home system has been integrated with technologies including computer science, telecommunication, automatic control and others and emerged as a comprehensive and smart system featuring safety, convenience, coziness, services, utility and environmental consciousness.





2.Description:

Launched by Keyestudio, this smart home kit is based on the open-source hardware of Micro:bit and designed for those who dream of living a more comfortable life with the help of technologies.

This smart home system, with Micro:bit as its control board, is equipped with a 1602 LCD, a DHT11 temperature and humidity sensor, an analog gas sensor(MQ_2), a PIR motion sensor , a 6812 RGB module, a servo, a steam sensor, a Micro:bit BT and other sensors.

With the help of these sensors, this kit can be applied to detect temperature, humidity and the concentration of flammable gases in your home and open and close doors. Furthermore, all the information detected can display on 1602 LCD in real time available for you to check and monitor via smart phones or iPad. By the way, it supports powering by solar energy or via USB cable.

This tutorial programs in MicroPython language which is the Micro:bit version of Python language. It will guide you to use software Mu to write MicroPython language for Micro:bit main board to control the smart home system. In this process, not only can you enhance your ability to make stuffs but also learn the skills of programming.





Python is one of the most popular programming language especially in machine learning for its availability and accessibility have brought huge convenience to this field. However, MicroPython is a lean and efficient implementation of the Python programming language for microcontrollers and embedded systems.

This tutorial is a Python tutorial for micro:bit smart home. If you haven't learned the basic tutorial (Makecode version of Tutorial), we strongly recommend you to learn it first. Because the basic one is programmed using graphical blocks, which is easier to understand and start.

3.Preparations:

3.1Background Information about Micro:bit

(1) What is Micro:bit?

Micro:bit is an open source hardware platform based on the ARM architecture launched by British Broadcasting Corporation (BBC) together with ARM, Barclays, element14, Microsoft and other institutions. The core device is a 32-bit Arm Cortex-M4 with FPU micro-processing.

Though it is just the size of a credit card, the Micro:bit main board is equipped with loads of components, including a 5*5 LED dot matrix, 2





programmable buttons, an accelerometer, a compass, a thermometer, a touch-sensitive logo and a MEMS microphone, a Bluetooth module of low energy, and a buzzer and others. Thus, it also boasts multiple functions.

The buzzer built in the other side of the board makes playing all kinds of sound possible without any external equipment. The golden fingers and gears added provide a better fixing of crocodile clips. Moreover, this board has a sleeping mode to lower power consumption of batteries and it can be entered if users long press the Reset & Power button on the back of it. It is capable of reading the data of sensors, controlling servos and RGB lights and attaching with a shield so as to connect with various sensors. It also supports a variety of codes and graphical programming platforms, and is compatible with almost all PCs and mobile devices. It has no need to install drivers. It is of high integration of electronic modules, and has a serial port monitoring function for easy debugging.

The board has found wild applications. It can be applied in programming video games, making interactions between light and sound, controlling a robot, conducting scientific experiments, developing wearable devices and make some cool inventions like robots and musical instruments, basically everything imaginable.





(2)Layout



For the Micro: Bit main board, pressing the Reset & Power button , it will

reset the Micro: Bit and rerun the program.

For more information, please resort to following links:

https://tech.microbit.org/hardware/

https://microbit.org/new-microbit/

https://www.microbit.org/get-started/user-guide/overview/

https://microbit.org/get-started/user-guide/features-in-depth/





(3) Pinout



The functions of pins:

	P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12,			
GPIO	P13, P14, P15, P16, P19, P20			
ADC/DAC P0, P1, P2, P3, P4, P10				
IIC	P19 (SCL) , P20 (SDA)			
SPI	P13 (SCK) , P14 (MISO) , P15 (MOSI)			
PWM (used	DO D1 D2 D4 D10			
frequently)	PU, PI, PZ, P3, P4, PIU			





PWM (not frequently used)	P5、P6、P7、P8、P9、P11、P12、P13、P14、P15、P16、P19、 P20
Occupied	P3(LED Col3), P4(LED Col1), P5(Button A), P6(LED Col4), P7(LED Col2), P10(LED Col5), P11(Button B)

Browse the official website for more details:

https://tech.microbit.org/hardware/edgeconnector/ https://microbit.org/guide/hardware/pins/

(4) Notes for the application of Micro:bit main board

a. It is recommended to cover it with a silicone protector to prevent short circuit for it has a lot of sophisticated electronic components.

b. Its IO port is very weak in driving since it can merely handle current less
than 300mA. Therefore, do not connect it with devices operating in large
current, such as servo MG995 and DC motor or it will get burnt.
Furthermore, you must figure out the current requirements of the devices
before you use them and it is generally recommended to use the board
together with a Micro:bit shield.





c. It is recommended to power the main board via the USB interface or via the battery of 3V. The IO port of this board is 3V, so it does not support sensors of 5V. If you need to connect sensors of 5 V, a Micro: Bit expansion board is required.

d. When using pins(P3、P4、P6、P7、P10)shared with the LED dot matrix, blocking them from the matrix or the LEDs may display randomly and the data about sensors connected maybe wrong.

e. Pin 19 and 20 can not be used as IO ports though the Makecode shows they can. They can only be used as I2C communication.

f. The battery port of 3V cannot be connected with battery more than 3.3V or the main board will be damaged.

g. Forbid to operate it on metal products to avoid short circuit.

To put it simple, Micro:bit V2 main board is like a microcomputer which has made programming at our fingertips and enhanced digital innovation. And as for programming environment, BBC provides a website: <u>https://microbit.org/code/</u>, which has a graphical MakeCode program easy for use.





3.2.Install Micro:bit driver

Micro:bit is free of driver installation. However, in case your computer fail to recognize the main board, you can install the diver too.

Just enter the link <u>https://fs.keyestudio.com/KS4027-4028</u> to download the driver file **e** mbed_usb_2020_x64_1212.exe of micro:bit in file folder **1**. Install Microbit Driver .

<mark>4.Python</mark>

The following instructions are applied for Windows system but can also serve as a reference if you are using a different system.

This tutorial is written for Python language. If you want to use graphical code programming, please refer to the manual "Makecode Tutorial.pdf". In the root directory of the resource you downloaded, there is a folder named "Python tutorial", which stores all the Python code of Microbit smart home. The Python code file is a file ending with ".py".





4.1.Python

Python is a scripting language. It has embraced a huge ecology after years of development evidenced by the fact that many of hot artificial intelligence are written in it. It is worth learning.

Micro:bit can be programmed in Python language. While since the micro:bit main board is a microcontroller, the hardware difference makes the micro:bit unable to fully support Python. Thus here comes the MicroPython, which is specially designed for micro:bit and can be regarded as the micro:bit version of Python.

MicroPython is a lean and efficient implementation of the Python 3 programming language that includes a small subset of the Python standard library and is optimised to run on microcontrollers and in constrained environments. It is very suitable for those who want to continue to learn programming in depth, with a series of code snippets, various images and music to help you program.

More details about it please log in official micro:bit website: <u>https://microbit-micropython.readthedocs.io/en/latest/tutorials/introducti</u> <u>on.html</u>





Python has two types of editors (web version and offline version).

Web version: https://python.microbit.org/v/1.1



The other one is the offline compiler tool -----Mu

(Download Mu: https://codewith.mu/en/download)

4.2.Mu

The official website for Mu is:

https://codewith.mu/.

Mu is a Python code editor for beginner programmers based on extensive feedback given by teachers and learners.Mu doesn' t support 32-bit Windows. The latest version is Mu 1.1.0-beta 2.

Follow steps below to install Mu:

Download Mu

Click "This PC" and right- click to select Properties to check the version of your computer.





📃 🛃 📕 🖛 This PC						
File Computer View						
$\leftarrow \rightarrow \checkmark \uparrow \blacksquare$ > This PC						
> 📌 Ouick access		∼ Fol	lders (7)			
> 👩 Creative Cloud Files				3D Objects		
> 🐔 OneDrive			~~			
> 💻 This PC			4-	Documents	1	
> 💣 Network	Expand					
😴 Manage						
	Pin to Sta	irt				
	Map netv	vork driv	/e			
	Open in r	new win	dow			
	Pin to Qu	iick acce	255			
	Disconnect					
-	Add a network location					
Delete					of 226 GB	
	Rename					
	Propertie	s				
l						

Below is shown system type of your computer.





	System					
~	← → ✓ ↑ 🗹 > Control Panel > System and Security > System					
	Control Panel Home	View basic information	about your computer			
•	Device Manager	Windows edition				
9	Remote settings	Windows 10 Home				
9	System protection	© 2018 Microsoft Corporation. All rights reserved.				
9	Advanced system settings					
		System				
		Processor:	Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz			
		Installed memory (RAM):	8.00 GB (7.73 GB usable)			
		System type:	64-bit Op rating System, x64-based processor			
		Pen and Touch:	No Pen or Touch Input is available for this Display			
l						

Enter link: <u>https://codewith.mu/en/download</u> to download the corresponding version of Mu.



Run and Install Mu

Find out the folder where Mu is downloaded and double-click file to install



Installation for Mac OSX please refer to : https://codewith.mu/en/howto/1.1/install_macos .

The installation method is similar and we will demonstrate how to download Mu on Windows 10.

Windows 10

You will view the page pop up, then click More info;





×

Don't run

Windows protected your PC

Windows Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk. More info

Then click "Run anyway";





Windows protected your PC				
Windows Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.				
App:	mu_2018-06-19_10_25_master_a132d40_64bit.e xe			
Publisher:	Unknown publisher			
	Run anyway Don't run			

Check the license agreement and tick to agree and tap" Install";





🔀 Mu Editor 1.1.0b2 Setup		Х
	Please read the Mu Editor 1.1.0b2 License Agreement	
S	GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007	^
	Copyright (C) 2007 Free Software Foundation, Inc. <http: fsf.org=""></http:> Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.	
	Preamble	
	The GNU General Public License is a free,	~
	☑ accept the terms in the License Agreement	
Print	Back Install Cance	1

Just wait for a few seconds until the installation is finished;





🖟 Mu Editor 1.1.0b2 Setup	-	- 🗆	×
Installing Mu Editor 1.1.0b2			Ð
Please wait while the Setup Wizard installs Mu Editor 1.1	.0b2.		
Status: Copying new files			
Back	Next		Cancel

Finally, click "Finish" .





🙀 Mu Editor 1.1.0b2 Setup	- 🗆 ×
S	Completed the Mu Editor 1.1.0b2 Setup Wizard
	Click the Finish button to exit the Setup Wizard.
and the second second	
	Back Einish Cancel

Start Mu:

Click Mu icon to get started. It may take a while for the first time;





=	Recently added	Productivity
	Mu Editor	
	*	
	3D Viewer	Office S Mail
	A	
	Alarms & Clock	◎ 🔼 🗸
	Audacity	Microsoft Edge Photos Microsoft To
	с	Explore
	Calculator	Partly Sunny
	Calendar	9° ^{12°}
	Camera	Microsoft Store London
(Candy Crush Friends	
L N	Connect	NETFLIX 🚱 🌣 🔛
	E	Solitaire Play
	eLicenser	
ŝ	F	
d۵.	Farm Heroes Saga	
	Feedback Hub	
	P Type here to search	O 🛱 🔚

Mu's main interface is shown below:





🕐 Mu 1.1.0.beta.2 - untitled			×
Image: Control of the second secon	Tidy Help	Quit	
<pre># Write your code here :-) 2 </pre>			
	BBC micro:bit	iii (ŀ

5. Projects:

Project 1: Heartbeat

(1)Project Introduction

This project is easy to conduct with a micro:bit main board, a Micro USB cable and a computer. This experiment serves as a starter for your entry to the magical programming world of Micro:bit.

(2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;B.Open the offline version of Mu.





(3)Test Code:

Open Mu software, click Mode, then click "BBC micro: bit" and "OK"



Tap "Load", select "Project 1: Heartbeat.py" file and click "open":

File	Route		File Name		
Туре					
Python	KS4027	folder/Python	Project	1	•
file	Tutorial/Python		Heartbeat	.ру	
	Code/Project Code/Project				
	1: Heartbeat				





🕐 Mu 1.1.0.beta.2 - untitled			×
Image: Save Image: Save	y Relp	Quit	
1 # Write your code here :-) 2			
BBC n	icro:bit		¢

🕝 Open file			×
$\leftarrow \rightarrow \checkmark \uparrow$ s with micro	robit Basic > Project 1: Heart beat	✓ ♂ ✓ ✓ Ø Search	n Project 1: Heart beat
Organize 🔻 New folder			::: • 🔟 ?
📃 Desktop 🛛 🖈 🐴	Name	Date modified	Туре
🕂 Downloads 🖈	Project 1: Heart beat	5/25/2021 11:34 AM	Python Source File
🔮 Documents 🖈			
📰 Pictures 🛛 🖈			
OneDrive			
💻 This PC			
File nan	ne:	~ *.py *.p w	*.hex *.css *.html *.P ∨
		Open	Cancel

There is another way to import code. Open Mu software and drag file" Project1:Heartbeat.py" into it.





🕐 Mu 1.1.0.beta.2 - untitled	– 🗆 X
Mode Hew Load Save Flash Files REFL Flotter	QQCImage: Common termImage: Common termIm
1 # Write your code here :-) 2	► Copy
	drag it to here
	drag it to here
Project 1: Heart beat	BBC micro: bit
Project 1: Heart beat ← → ▼ ↑ G ≪ microbit Basic → Project	BBC mioro:bit 🗰 🔕 - 🗆 × I: Heart beat 🗸 O 🖓 Search Project 1:
← → ✓ ↑ G < microbit Basic → Project Name	BBC micro:bit BBC micro:bit - X Heart beat Date modified Type
 Project 1: Heart beat ← → ~ ↑	BBC mioro: bit BBC mioro: bit
Project 1: Heart beat ← → ~ ↑	BBC mioro: bit BBC mioro: bit BBC mioro: bit C C C C C C C C C C C C C C C C C C C
 Project 1: Heart beat ← → ~ ↑	BBC mi or o: bit BBC mi or o: bit BBC mi or o: bit BBC mi or o: bit - - X - X - X - X - X - X - X - X - X - X - X - X - - X - - X - - X - - - X - - - - - - - - - -
 Project 1: Heart beat ← → ~ ↑	BBC micro: bit BBC micro: bit BBC micro: bit BBC micro: bit C Search Project 1: Date modified Type beat 5/25/2021 11:34 AM Pythoi

You can also input code in the edit window yourself.

(note:all English words and symbols must be written in English.)





(M	u 1.1.0.beta.2 - Project 1: Heart beat.py		×
Mode	+ +	Quit	
untit	led 💥 Project 1: Heart beat.py 🗶		
1	from microbit import *		
2	Adda Tarana		
3	display show(Image HEART)		
5	sleep(500)		
6	display.show(Image.HEART_SMALL)		
7	sleep(500)		
8			
			alla i
	BBC micro:bit		

The following is a list of built-in images.

- Image.HEART
- Image.HEART_SMALL
- Image.HAPPY
- Image.SMILE
- Image.SAD
- Image.CONFUSED
- Image.ANGRY
- Image.ASLEEP
- Image.SURPRISED
- Image.SILLY





- Image.FABULOUS
- Image.MEH
- Image.YES
- Image.NO
- Image.CLOCK12, Image.CLOCK11, Image.CLOCK10, Image.CLOCK9, Image.CLOCK8, Image.CLOCK7, Image.CLOCK6, Image.CLOCK5, Image.CLOCK4, Image.CLOCK3, Image.CLOCK2, Image.CLOCK1
- Image.ARROW_N, Image.ARROW_NE, Image.ARROW_E,
 Image.ARROW_SE, Image.ARROW_S, Image.ARROW_SW,

Image.ARROW_W, Image.ARROW_NW

- Image.TRIANGLE
- Image.TRIANGLE_LEFT
- Image.CHESSBOARD
- Image.DIAMOND
- Image.DIAMOND_SMALL
- Image.SQUARE
- Image.SQUARE_SMALL
- Image.RABBIT
- Image.COW
- Image.MUSIC_CROTCHET
- Image.MUSIC_QUAVER
- Image.MUSIC_QUAVERS





- Image.PITCHFORK
- Image.PACMAN
- Image.TARGET
- Image.TSHIRT
- Image.ROLLERSKATE
- Image.DUCK
- Image.HOUSE
- Image.TORTOISE
- Image.BUTTERFLY
- Image.STICKFIGURE
- Image.GHOST
- Image.SWORD
- Image.GIRAFFE
- Image.SKULL
- Image.UMBRELLA
- Image.SNAKE, Image.ALL_CLOCKS, Image.ALL_ARROWS

Connect micro:bit board to computer with USB cable, click "Flash" to download code to micro:bit board.



The code, even it is wrong, can be downloaded to micro:bit board successfully, yet not working on micro:bit board.

Click "Flash" to download code to micro:bit.




P Mu 1.1.0.beta.2 - Project 1: Heart beat.py	_		\times
Image: Save Image: Save	?	Quit	
1 from microbit import *			
2 . while True:			
display.show(Image.HEART)			
sleep(500)			
<pre>6 display.show(Image.HEART_SMALL)</pre>			
7 sleeps(500)			
8			

Click "REPL" and press the reset button on micro:bit, the error information will be displayed on REPL window, as shown below:





🕐 Mu 1.1.0.beta.2 - Project 1: Heart beat.py	_		×
Mode Here Image: Save Imag	? Help	Quit	
1 from microbit import *			
2 while True:			
4 display.show(Image.HEART)			
s sleep(500)			
<pre>6 display.show(Image.HEART_SMALL)</pre>			
7 sleeps(500)			
8			
BBC micro:bit REPL			
MicroPython v1.13 on 2021-02-19; micro:bit v2.0.0-beta.4 with nRF52833			
>>>			
>>> Traceback (most recent call last):			
File "main.py", line 7, in <module> NameError: name 'sleeps' isn't defined</module>			
BBC mid	ro:bit		Ċ-

Click "REPL" again to turn off REPL mode, then you could refresh new code.

To make sure code correct, you only need to tap "Check". The errors will be

shown on the window.





С м	u 1.1.0.beta.2 - Project 1: Heart beat.py	_		×
Mode	+ + <td>? Help</td> <td>(U) Quit</td> <td></td>	? Help	(U) Quit	
untit	led 🗶 Project 1: Heart beat.py 🗶			
1	from microbit import *			
2	while True:			
3	display.show(Image.HEART)			
5	sleep(500)			
6	display.show(Image.HEART_SMALL)			
7	sleeps(500)			
	↑ undefined name 'sleeps'			
8				
	BBC mic	ro:bit		¢.

Modify the code according to the prompt and click "Check" .







More tutorials, log in website please: https://codewith.mu/en/tutorials/.

(4)Test Results:

After uploading test code to micro:bit main board, clicking "Flash" again and keeping the connection with the computer to power the main board, the LED dot matrix shows pattern "" and then "" alternatively.

(5)Code Explanation:

from microbit import *	Import the library file of micro: bit		
while True:	This is a permanent loop that makes		
	micro:bit execute the code of it.		
display.show(Image.HEART)	micro: bit shows "♡"		
sleep(500)	Delay in 500ms		
display.show(Image.HEART_SMALL)	micro: bit displays "🔡"		





Project 2: Light A Single LED



(1) Project Introduction

The LED dot matrix consists of 25 LEDs arranged in a 5 by 5 square. In order to locate these LEDs quickly, as the figure shown below, we can regarded this matrix as a coordinate system and create two aces by marking those in rows from 0 to 4 from top to bottom, and the ones in columns from 0 to 4 from the left to the right. Therefore, the LED sat in the second of the first line is (1,0) and the LED positioned in the fifth of the fourth column is (3,4) and others likewise.



(2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;







B.Open the offline version of Mu.

(3)Test Code:

Enter Mu software and open the file "Project 2: Light A Single LED.py" to

import code:

Туре	Route		File Name			
Python	KS4027	folder/Python	Project 2:	Light A Single		
file	Tutorial/Python		LED.py			
	Code/Project Code/Project					
	2: Light A	Single LED				

You can also input code in the editing window yourself.

(Note:all English words and symbols must be written in English)





С м	lu 1.1.0.beta.2 - Project 2: Single LED flashing.py		×
Mode Proje	Image: Save Image: Save	Quit	
1	from microbit import *		
2			
3	val1 = Image("09000:""00000:""00000:""00000:""00000:")		
4	val2 = Image("00000:""00000:""00000:""00000:""00090:")		
5	val3 = Image("00000:""00000:""00000:""00000:""00000:")		
6			
7	while True:		
8	display.show(vall)		
9	sleep(500)		
10	display.show(val3)		
- 11	sleep(500)		
12	display.show(val2)		
13	steep(500)		
14	display.snow(val3)		
15	steep(500)		
16			
	BBC micro:bit		¢.

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





С м	lu 1.1.0.beta.2 - Project 2: Single LED flashing.py	_		×
Mode Proje	Image: Save Image: Save	? Help	Quit	
1	from microbit import *			
2	N			
3	vall = Image("09000:""00000:""00000:""00000:""00000:")			
4	val2 = Image("00000:""00000:""00000:""00000:""00090:")			
5	val3 = Image("00000:""00000:""00000:""00000:""00000:")			
6				
7	while True:			
8	display.show(vall)			
9	sleep(500)			
10	display.show(val3)			
- 11	sleep(500)			
12	display.snow(val2)			
13	steep(500)			
14	clean (FOO)			
15	steep(500)			
16				
	BBC micr	o:bit		¢.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





С м	u 1.1.0.beta.2 - Project 2: Single LED flashing.py – 🗆 🗙
Mode Proje	Image: Save Image: Save
1	from microbit import * 🔪
2	
3	vall = Image("09000:""00000:""00000:""00000:")
4	val2 = Image("00000:""00000:""00000:""00000:""00090:")
5	val3 = 1mage("00000:""00000:""00000:""00000:")
6	while True:
7	display show(vall)
	sleen(500)
10	display.show(val3)
	sleep(500)
12	display.show(val2)
13	sleep(500)
14	display.show(val3)
15	sleep(500)
16	
	BBC micro:bit 🗰 🔅

(4)Test Results:

After uploading test code to micro:bit main board and powering the main board via the USB cable, the LED in (1,0) lights up for 0.5s and the one in (3,4) shines for 0.5s and repeat this sequence.





(5)Code Explanation:

from microbit import *	Import the library file
	of micro: bit
val1 =	Set Image() to val1
Image("09000:""00000:""00000:""00000:""00000:")	Set pixel of LED on
	micro:bit to the value
	in 0~9
	Pixel of each LED
	on micro:bit can be set
val2 =	in one of ten values
Image("00000:""00000:""00000:""00000:""00090:")	If set pixel to 0
val3 =	(zero) , which means
Image("00000:""00000:""00000:""00000:""00000:")	in close state, literally,
	0 is brightness, 9 is
	best brightness
	Set Image() to val2
	Set Image() to val3
while True:	This is a
	permanent loop that
	makes micro:bit
	execute the code of it.
display.show(val1)	





sleep(500)	LED at (1,0) blinks
display.show(val3)	for 0.5s
sleep(500)	
display.show(val2)	
sleep(500)	LED at (3,4) flashes
display.show(val3)	for 0.5s
sleep(500)	

(6)Reference

sleep(ms) : delay time

For more details about delay, please refer to:

https://microbit-micropython.readthedocs.io/en/latest/utime.html

Project 3: LED Dot Matrix



(1) Project Introduction

Dot matrices are very commonplace in daily life. They have found wide applications in LED advertisement screens, elevator floor display, bus stop





announcement and so on.

The LED dot matrix of Micro: Bit main board contains 25 LEDs in a grid. Previously, we have succeeded in controlling a certain LED to light by integrating its position value into the test code. Supported by the same theory, we can turn on many LEDs at the same time to showcase patterns, digits and characters.

What's more, we can also click" show icon "to choose the pattern we like to display. Last but not the least, we can design patterns by ourselves as well.

(2)Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

(3)Test Code:

You could open "Project 3: LED Dot Matrix.py "file to Import code (<u>How to</u>

load the project code?)

File	Route		File Nam	ne			
Туре							
Python	KS4027	folder/Python	Project	3	•	LED	Dot
file	Tutorial/Pytl	hon	Matrix.p	у			





Code/Project Code/Project

3: LED Dot Matrix

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English.)

() N	lu 1.1.0.beta.2 - Project 3: LED dot matrix display.py -		×	
Mode	Image: New Load Image: Save Image: Save </th <th>Quit</th> <th></th> <th></th>	Quit		
Proje	ect 3: LED dot matrix display.py 🛛 🗶			
1	from microbit import *			Δ
2	val = Image("00900:""00900:""90909:""09990:""00900")			
3	display.show('1')			
4	sleep(500)			
5	display.show('2')			
6	sleep(500)			
7	display.show('3')			
8	sleep(500)			
9	display.show('4')			
10	sleep(500)			
11	display.show('5')			
12	sleep(500)			
13	display.show(val)			
14	steep(500)			
15	display.scroll("nello!")			
16	steep(200)			
17	display.snow(image.HEARI)			
18	steep(500)			
19	alson (Foo)			
20	dieplaw shew(Image APPOW SE)			
21	cloop(E00)			
22	display show(Image APROW SW)			
23	sloon(500)			
24	display, show(Image ARROW NW)			
28	sleen(500)			
20	display.clear()		•	∇
	BBC micro:bit		¢	-





Click "Check" to examine error in the code. The program proves wrong if

underlines and cursors are shown.

(N	Iu 1.1.0.beta.2 - Project 3: LED dot matrix display.py	_		\times
Mode	Image: New Load Image: Save Image: Save	? Help	(U) Quit	
Proje	ect 3: LED dot matrix display.py 🛛 🗶			
1	from microbit import *			\triangle
2	val = Image("00900:""00900:""90909:""09990:""00900")			
3	display.show('1')			
4	sleep(500)			
5	display.show('2')			
6	sleep(500)			
7	display.show('3')			
8	sleep(500)			
9	display.show('4')			
10	sleep(500)			
- 11	display.show('5')			
12	steep(500)			
13	display.show(val)			
14	steep(500)			
15	display.scroll("nello!")			
16	steep(200)			
17	cloop(EQQ)			
18	display, shew(Image APPON NE)			
19	sloop(EQA)			
20	display show(Image ARROW SE)			
21	sleen(500)			
22	display, show (Image, ARROW, SW)			
23	sleen(500)			
25	display, show(Image, ARROW NW)			
26	sleep(500)			
27	display.clear()			∇
	BBC micr	ro:bit		0

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.







(4) Test Results:

After uploading test code to micro:bit main board and powering the main board via the USB cable, we find that the 5*5 dot matrix start to show numbers 1,2,3,4 and 5, and then it alternatively shows a downward arrow





northwest



5

, and then at





(5)Code Explanation:

from microbit import *	Import the library file
	of micro: bit
val =	
Image("09000:""00000:""00000:""00000:""00000:")	Set Image() to variable
	val
display.show(val)	micro:bit shows " \rightarrow "
display.show('1')	micro:bit shows "1"
sleep(500)	Delay in 500ms
display.scroll("hello!")	micro:bit scrolls to
	show "hello!"
display.show(Image.HEART)	micro:bit displays "🎔"
	pattern
display.show(Image.ARROW_NE)	micro:bit shows
display.show(Image.ARROW_SE)	"Northeast" arrow
display.show(Image.ARROW_SW)	micro:bit displays
display.show(Image.ARROW_NW)	"Southeast" arrow
	micro:bit shows
	"Southwest" arrow
	micro:bit displays
	"Northwest" arrow





display.clear()

The LED dot matrix of

micro:bit clears

(6) Reference:

display.scroll() :

The display scrolls to show the values, if it is integer or float, we use str ()

to transfer into character strings.

More details, please refer to

https://microbit-micropython.readthedocs.io/en/latest/utime.html

Project 4: Programmable Buttons



(1) Project Introduction

Buttons can be used to control circuits. In an integrated circuit with a push button, the circuit is connected when pressing the button and it is open the

other way around.

Both ends of button

river in between.



are like two mountains. There is a

The internal metal piece connect the two sides to let the current pass, just like building a bridge to connect two mountains.





Micro: Bit main board boasts three push buttons, two are programmable buttons(marked with A and B), and the one on the other side is a reset button. By pressing the two programmable buttons can input three different signals. We can press button A or B alone or press them together and the LED dot matrix shows A,B and AB respectively. Let's get started.

(2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;

B. Open the offline version of Mu.

(3)Test Code1:

Enter Mu software and open the file "Project 4: Code-1.py" to import code:

(How to load the project code?)
--------------------------------	---

File Type	Route		File Name
Python	KS4027	folder/Python	Project 4: Code-1.py
file	Tutorial/Pyth	non	
	Code/Projec	t Code/Project	
	4: Program	mable Buttons	





You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)

() M	u 1.1.0.beta.2 - Project 4: Programmable Buttons-1.py -		×
Mode	Image: Save Image: Save	Quit	
frojec	from microbit import *		
2			
3	while True:		
4	<pre>if button_a.is_pressed():</pre>		
5	<pre>display.snow("A") elif button a is pressed() and button b is pressed():</pre>		
7	display.scroll("AB")		
8	<pre>elif button_b.is_pressed():</pre>		
9	display.show("B")		
10			
	BBC micro:bit		¢

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





() M	lu 1.1.0.beta.2 - Project 4: Programmable Buttons-1.py	_		×
Mode Proje	Image: Save Image: Save	? Help	Quit	
1	from microbit import *			
2	ubile Truce			
3	if button a.is pressed():			
5	display.show("A")			
6	<pre>elif button_a.is_pressed() and button_b.is_pressed():</pre>			
7	display.scroll("AB")			
8	<pre>elif button_b.is_pressed():</pre>			
9	display.show("B")			
10				
	BBC mi	ro:bit		Q.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





С м	lu 1.1.0.beta.2 - Project 4: Programmable Buttons-1.py	_		\times
Mode	Image: New Load Image: Save Image: Save	idy Help	Quit	
Proje	ct 4: Programmable Buttons-1. 🛒 🗶			
1	from microbit import *			
3	while True:			
4	<pre>if button_a.is_pressed():</pre>			
5	display.show("A")			
6	<pre>elif button_a.is_pressed() and button_b.is_pressed():</pre>			
7	display.scroll("AB")			
8	elif button_b.is_pressed():			
9	display.snow("B")			
10				
	BBC	micro:bit		¢.

(4)Test Results1:

After uploading test code to micro:bit main board and powering the main board via the USB cable, the 5*5 LED dot matrix shows "A" if button A is pressed and then released, "B" if button B pressed and released, and "AB" if button A and B pressed together and then released.

(5)Test Code2:

Enter Mu software and open the file "Project 4: Code-2.py

" to import code: (<u>How to load the project code</u>?)

File Type	Route	File Name
Python	KS4027 folder/Python	Project 4: Code-2.py





file	Tutorial/Python	
	Code/Project Code/Project	
	4: Programmable Buttons	

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)

(M	lu 1.1.0.beta.2 - Project 4: Programmable Buttons-2.py — 🗌	×
(F) Mode	Image: New Load Image: Save Image: Save </th <th></th>	
Proje	ect 4: Programmable Buttons=2.py 🛛 💥	
1	from microbit import *	\bigtriangleup
2	a = 0	
3	b = 0	
4	vall = Image("00000:""00000:""00000:""00900")	
5	val2 = Image("00000:""00000:""00900:""99999")	
6	val3 = Image("00000:""00000:""00900:""999999:""999999")	
7	val4 = Image("00000:""00900:""999999:""999999:""999999")	
8	val5 = Image("00900:""999999:""999999:""999999")	
9	val6 = Image("999999:""999999:""999999:""999999")	
10	display.show(vall)	
11		
12	while True:	
13	while button_a.is_pressed() == True:	
14	steep(10)	
15	<pre>if button_a.is_pressed() == False:</pre>	
16	a = a + 1	
17	$1T(a \ge 5)$:	
18	a = 5	
19	break	







Click "Check" to examine error in the code. The program proves wrong if

underlines and cursors are shown.

(M)	1u 1.1.0.beta.2 - Project 4: Programmable Buttons-2.py —	
Mode	Image: series of the series	(U) Quit
Proje	ect 4: Programmable Buttons-2.py 🗶	
1	from microbit import *	
2	a = 0	
3	$b = \Theta$	
4	vall = Image("00000:""00000:""00000:""00000:""00900")	
Б	val2 = Image("00000:""00000:""00000:""00900:""99999")	
6	val3 = Image("00000:""00000:""00900:""999999:""99999")	
7	val4 = Image("00000:""00900:""99999:""99999:""99999")	
8	val5 = Image("00900:""999999:""999999:""999999:""999999")	
9	val6 = Image("99999:""99999:""99999:""999999:""999999	
10	display.show(vall)	
11		
12	while True:	
13	while button_a.is_pressed() == True:	
14	steep(10)	
15	1 Dutton_a.is_pressed() == False:	
16	a = a + 1	
17	$1T(a \ge 5);$	
18	d = 5	
19	Dreak	





20	<pre>while button_b.is_pressed() == True:</pre>
21	sleep(10)
22	<pre>if button_b.is_pressed() == False:</pre>
23	a = a - 1
24	if (a <= 0):
25	a = 0
26	break
27	if a == 0:
28	display.show(val1)
29	if a == 1:
30	display.show(val2)
31	if a == 2:
32	display.show(val3)
33	if a == 3:
34	display.show(val4)
35	if a == 4:
36	display.show(val5)
37	if a == 5:
38	display.show(val6)
	BBC micro:bit 🗰 🛟

Please notice that the following sentences are just for warning so the

presence of them doesn' t mean the code is wrong.

↑ Comparison	to	true	should	be	'íf	cond	is :	true:'	or	'if	cond:'	
↑ Comparison	to	false	should	t be	'if	cond	is	false:	' (or !-	íf not	cond:'

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





Г м	1u 1.1.0.beta.2 -	Project 4	: Program	mable Butt	ons-2.py						_		×
Mode	New Load	Save	Flash F	iles REPL	Plotter	Zoom-in	Zoom-out	Theme	Check	Tidy	Help	Quit	
Proje	ct 4: Program	mable Bu	ttons-2.p.	×									
1	from micro	obit in	nport *										\triangle
2	a = 0 b = 0												
4	vall = Ima	age("00	0000:""0	0000:""	90000:""	00000:"	"00900")					
5	val2 = Ima	age("0(0000:""0	0000:""	00000:""	00900:"	"999999")					
6	val3 = Ima	age("00	0000:""0	0000:""	00900:""	999999:"	"999999")					
7	val4 = Ima val5 = Ima	age("00 age("00	9000:""0 9966:""9	0900:"" 9999:""	99999:""		"999999" "999999")					
9	val6 = Ima	age("99	9999:""9	9999:""	99999:""	'999999:"	"999999")					
10	display.sh	now(va)	l1)										
11													
12	while Irue while	e: buttor	na is p	ressed() == Tru	le :							
14	sl	Leep(1	9)	cooca (,								
15	if	f butto	on_a.is_	pressed	() == F a	lse:							
16		a =	a + 1										
17		11(8	$a \ge 5$: a = 5										
19		brea	ak										
20	while	butto	n_b.is_p	ressed() == Tru	ue:							
21	s	leep(1	0)										
22	i	f butt	on_b.is_	pressed	() == F a	alse:							
23		a =	a - 1										
24		110	a <= 0). a = 0										
26		bre	ak										
27	if a	== 0:											
28	d	ísplay	.show(va	11)									
29	l a a	== ⊥: (splav	.show(va	12)									
31	if a == 2:												
32	display.show(val3)												
33	if a :	= 3:	-	1.0									
34	if a	rsptay == ⊿∙	.show(va	L4)									
36	d	ísplav	.show(va	15)									
37	if a	= 5:											
38	d	ísplay	.show(va	16)									\bigtriangledown
										BBC mi	cro:bit		Ċ.

(6)Test Results2:

After uploading test code to micro:bit main board and powering the main





board via the USB cable, when the button A is pressed, the LEDs turning red increase while when the button B pressed, the LEDs turning red reduce.

(7)Code Explanation:

from microbit import *	Import the library file of micro:				
	bit				
while True:	This is a permanent loop that				
	makes micro:bit execute the				
	code of it.				
if button_a.is_pressed():	If button A is pressed				
display.show("A")	micro:bit shows "A"				
elif button_a.is_pressed() and	If button A and B are pressed at				
button_b.is_pressed():	same time				
display.scroll("AB")	micro:bit displays "AB"				
elif button_b.is_pressed():	If button B is pressed				
display.show("B")	micro:bit shows "B"				
while button_a.is_pressed() == True:	When the button A is pressed				
sleep(10)	Delay in 10ms to eliminate the				
if button_a.is_pressed() == False:	shaking of button A				
a = a + 1	when button A is released,				
if(a >= 5):	Variable a adds 1				
a = 5	If variable a≥5				



I



break	Variable a=5
while button_b.is_pressed() == True:	exit the loop
sleep(10)	when button B is pressed
if button_b.is_pressed() == False:	Delay in 10ms to eliminate the
a = a - 1	shaking of button B
if(a <= 0):	When the button B is released
a = 0	Variable a reduces 1 gradually
break	When a≤0
if a == 0:	Variable a=0
display.show(val1)	exit the loop
if a == 1:	When a=0
display.show(val2)	micro:bit shows pattern val1
if a == 2:	When a=1
display.show(val3)	micro:bit displays pattern val2
if a == 3:	When a=2
display.show(val4)	micro:bit shows pattern val3
if a == 4:	If a=3
display.show(val5)	micro:bit displays pattern val4
if a == 5:	If a=4
display.show(val6)	micro:bit shows pattern val5
	If a=5
	micro:bit displays pattern val6





Project 5: Temperature Detection (1) Project Introduction

The Micro:bit main board is not equipped with a temperature sensor, but uses the temperature sensor built into NFR52833 chip for temperature detection. Therefore, the detected temperature is more closer to the temperature of the chip, and there maybe deviation from the ambient temperature. The sensor can detect temperature of external environment with the range of $40^{\circ}C \sim 105^{\circ}C$.

In this project, we use the sensor to test the temperature in the current environment, and display the test results in the display data (device). And then control the LED dot matrix to display different patterns by setting the temperature range detected by the sensor.

Note: the temperature sensor of Micro:bit main board is shown below:



(2)Preparations:





- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

(3)Test Code1:

Enter Mu software and open the file "Project 5: Code-1.py" to import code:

File Type	Route		File Name
Python file	KS4027	folder/Python	Project 5: Code-1.py
	Tutorial/P	ython	
	Code/Pro	ject	
	Code/Pro	ject 5 :	
	Temperat	ure Detection	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)





🕐 м	lu 1.1.0.beta.2 - Project 5: Temperature Measurement-1.py	_		\times				
Node	Image: New Load Save Image: S	? Help	Quit					
Proje	ct 5: Temperature Measurement-1.py 🗶							
1	from microbit import *							
2	while True:							
4	winte mue.							
5	Temperature = temperature()							
6	· - · · · · · · · · · · · · · · · · · ·							
7	print("Temperature:", Temperature, "C")							
8								
9	steep(500)							
10								
12								
13								
14								
15								
16								
	BBC micr	o:bit	# 1	¢.				

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





С М	u 1.1.0.beta.2 - Project 5: Temperature Measurement-1.py	_		×				
Node	Image: Weight of the second	? Help	Quit					
Proje	ct 5: Temperature Measurement-1.py 🗙							
1	from microbit import *							
2								
3	while True:							
4	Temperature = temperature()							
6	remperature - cemperature()							
7	print("Temperature:", Temperature, "C")							
8								
9	sleep(500)							
10								
11								
12								
13								
14								
16								
	BBC mici	ro:bit		¢.				

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





(4)Test Results1:

After downloading test code 1 to micro:bit board, keep USB connected and click "REPL" and press the reset button on micro:bit. Then REPL window will show the ambient temperature value, as shown below:(C stands for temperature unit)





Mu 1.1.0.beta.2 - Project 5: Temperature Measurement-1.py —		×						
Image: NodeImage: SaveImage: Sav	Quit							
Project 5: Temperature Measurement-1.py 🗶 🔨								
from microbit import *		\triangle						
2								
3 while True:								
4								
<pre>5 Temperature = temperature()</pre>								
6								
<pre>7 print("Temperature:", Temperature, "C")</pre>								
8								
sleep(500)		∇						
BBC micro:bit REPL								
Temperature: 30 C		\bigtriangleup						
Temperature: 30 C								
Temperature: 30 C								
Temperature: 30 C								
Temperature: 31 C								
Temperature: 31 C								
Temperature: 31 C								
Temperature: 31 C		∇						
	ar							
BBC micro:bit								

(5)Test Code2:

Enter Mu software and open the file "Project 5: Code-2.py" to import code:

File	Route		File Name
Туре			
Python	KS4027	folder/Python	Project 5: Code-2.py
file	Tutorial/Pyth	non	
	Code/Projec	t Code/Project	
	5: Temperat	ture Detection	





You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)

The temperature value can be set in compliance with the real temperature.



Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.







If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.







(6)Test Results2:

After uploading the code 2 to the board, when the ambient temperature is






(7)Code Explanation:

from microbit import *	Import the library file of micro:
	bit
while True:	This is a permanent loop that
	makes micro:bit execute the
	code of it.
Temperature = temperature()	Set temperature() to
	Temperature
<pre>print("Temperature:", Temperature, "C")</pre>	BBC micro:bit REPL prints
	temperature value
sleep(500)	Delay in 500ms
if temperature() >= 35:	If temperature value ≥35℃
display.show(Image.HEART)	micro:bit shows "♥"
else:	If temperature value < 35°C
display.show(Image.HEART_SMALL)	micro:bit displays "😐"





Project 6: Geomagnetic Sensor



(1) Project Introduction

This project aims to explain the use of the Micro: bit geomagnetic sensor, which can not only detect the strength of the geomagnetic field, but also be used as a compass to find bearings. It is also an important part of the Attitude and Heading Reference System (AHRS).

Micro: Bit main board uses LSM303AGR geomagnetic sensor, which supports four modes namely 100 kHz,400 kHz,1 MHz and 3.4 MHz and the dynamic range of magnetic field is ± 50 gauss.

In the board, the magnetometer module is used in both magnetic detection and compass. In this experiment, the compass will be introduced first, and then the original data of the magnetometer will be checked. The main component of a common compass is a magnetic needle, which can be rotated by the geomagnetic field and point toward the geomagnetic North Pole (which is near the geographic South Pole) to determine direction.

Attention: this geomagnetic sensor built in the board can help us determine bearings by showing readings in the value from 0 to 360. And the system will ask us to calibrate it the first time it is put into operation by





rotating the board.Please note that metal materials around may attenuate the accuracy of the reading and calibration.

(2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;

B.Open the offline version of Mu.

(2) Test code1::

Enter Mu software and open the file "Project 6: Code-1.py" to import code:

File Type	Route		File Name
Python	KS4027	folder/Python	Project 6: Code-1.py
file	Tutorial/Pyt	hon	
	Code/Proje	ct	
	Code/Proje	ct 6	
	Geomagnet	ic Sensor	

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





С м	lu 1.1.0.beta.2 - Project 6: Microbit's Compass-1.py	_		\times
Node	Image: New Load Save Image: Save	? Help	Quit	
Proje	ct 6: Microbit's Compass-1.py			
1	trom microbit import *			
3	compass.calibrate()			
4				
5	while True:			
6	is but here a de managed () :			
7	<pre>if Dutton_a.is_pressed(): display_scroll(compass_beading())</pre>			
9	i dispitaly ison of concentration and the graph			
10				
п				
	BBC mid	ro:bit		¢.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.







Note: We need to calibrate micro: bit due to different magnetic field in different areas. Micro:bit will prompt you to calibrate when you use it first time.

(4)Test Result1:

After uploading test code1 to micro:bit main board and powering the board via the USB cable, and pressing the button A, the board asks us to calibrate compass and the LED dot matrix shows "TILT TO FILL SCREEN".





Then enter the calibration page. Rotate the board until all 25 red LEDs are

on as shown below.



After that, a smile pattern appears, which implies the calibration is done. When the calibration process is completed, pressing the button A will make the magnetometer reading display directly on the screen. And the direction north, east, south and west correspond to 0°, 90°, 180° and 270° respectively.

(5)Test code2:





For the below picture, the arrow pointing to the upper right when the value ranges from 292.5 to 337.5. Because 0.5 can't be input in the code, the values we get are 293 and 338.

Then add other statements to make a set of complete code.



Enter Mu software and open the file "Project 6: Code-2.py" to import code:





File Type	Route		File Name
Python	KS4027	folder/Python	Project 6: Code-2.py
file	Tutorial/Py	thon	
	Code/Proje	ect	
	Code/Proje	ect 6	
	Geomagne	tic Sensor	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)





(N	lu 1.1.0.beta.2 - Project 6: Microbit's Compass-2.py –	×		
Mode	Image: New Load Image: Save Image: Save </th <th></th>			
Proje	ect 6: Microbit's Compass-2.py 🗙			
1	from microbit import *	\triangle		
2	compass.calibrate()			
3	$\mathbf{x} = \mathbf{\Theta}$			
4	while True:			
5	x = compass.heading()			
6	if $x \ge 293$ and $x < 338$:			
7	display.show(Image("00999:""00099:""00909:""09000:""90000"))			
8	elif $x \ge 23$ and $x \le 68$:			
9	display.show(Image("99900:""99000:""90900:""00090:""00009"))			
10	elif $x \ge 68$ and $x \le 113$:			
п	display.show(Image("00900:""09000:""999999:""09000:""009000"))			
12	elif $x \ge 113$ and $x \le 158$:			
13	display.show(Image("00009:""00090:""90900:""999000:""99900"))			
14	elif $x \ge 158$ and $x \le 203$:			
15	display.show(Image("00900:""00900:""90909:""09990:""00900"))			
16	<pre>16 elif x >= 203 and x < 248:</pre>			
17	display.show(Image("90000:""09000:""00909:""00099:""00999"))			
18	elif $x \ge 248$ and $x < 293$:			
19	display.snow(image("00900:""00090:""099999:""00090:""00900"))			
20	else:			
21	dishrah'zuom(twaše(.00000:00000:00000:000000:000000	∇		
22		-		
	BBC micro:bit			

Click "Check" to examine error in the code. The program proves wrong if

underlines and cursors are shown.





P N	/u 110 heta 2 - Project 6: Microbit's Compass-2 nv — —	×	
•		~	
Mode	Image: sevent		
Proje	ect 6: Microbit's Compass-2.py 🗙		
1	from microbit import *	\bigtriangleup	
2	compass.calibrate()		
3	$\mathbf{x} = \mathbf{\Theta}$		
4	while True:		
5	x = compass.heading()		
6	if $x \ge 293$ and $x \le 338$:		
7	display.show(Image("00999:""00099:""00909:""09000:""90000"))		
8	elif $x \ge 23$ and $x \le 68$:		
9	display.show(Image("99900:""99000:""90900:""00090:""00009"))		
10	elif $x \ge 68$ and $x \le 113$:		
11	dísplay.show(Image("00900:""09000:""99999:""09000:""009000"))		
12	elif x >= 113 and x < 158:		
13	dísplay.show(Image("00009:""00090:""90900:""99000:""99900"))		
14	elif x >= 158 and x < 203:		
15	dísplay.show(Image("00900:""00900:""90909:""09990:""00900"))		
16	elif $x \ge 203$ and $x \le 248$:		
17	display.show(Image("90000:""09000:""00909:""00099:""00999"))		
18	elif $x \ge 248$ and $x \le 293$:		
19	display.show(Image("00900:""00090:""99999:""00090:""00900"))		
20	else:		
21	dísplay.show(Image("00900:""09990:""90909:""00900:""00900"))	∇	
22		-	
	BBC micro:bit	Q	

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





СМ	u 1.1.0.beta.2 - Project 6: Microbit's Compass-2.py –	×
Mode	Image: series of the series	
Proje	ct 6: Microbit's Compass-2. p. 🗶	
1	from microbit import *	\triangle
2	compass.calibrate()	
3	$\mathbf{x} = \mathbf{\Theta}$	
4	while True:	
5	x = compass.heading()	
6	if x >= 293 and x < 338:	
7	display.show(Image("00999:""00099:""00909:""09000:""90000"))	
8	elif $x \ge 23$ and $x \le 68$:	
9	display.show(Image("99900:""99000:""90900:""00090:""00009"))	
10	elif $x \ge 68$ and $x \le 113$:	
11	display.show(Image("00900:""09000:""99999:""09000:""00900"))	
12	elif $x \ge 113$ and $x \le 158$:	
13	display.show(Image("00009:""00090:""90900:""99000:""999000"))	
14	elif $x \ge 158$ and $x \le 203$:	
15	display.show(Image("00900:""00900:""90909:""09990:""00900"))	
16	elif $x \ge 203$ and $x \le 248$:	
17	display.show(Image("90000:""09000:""00909:""00099:""00999"))	
18	elif x >= 248 and x < 293:	
19	display.show(Image("00900:""00090:""99999:""00090:""00900"))	
20	else:	
21	display.show(Image("00900:""09990:""90909:""00900:""00900"))	
22		
	BBC micro:bit	2

(6)Test Results2:

Upload code 2 and plug micro:bit into power. After calibration, tilt micro:bit board, and the LED dot matrix displays the direction signs.

(6)Code Explanation:

from microbit import *	Import the
	library file of
	micro: bit
compass.calibrate()	Compass
	calibration





while True:	This is a
	permanent
	loop that
	makes
	micro:bit
	execute the
	code of it.
if button_a.is_pressed():	When the
display.scroll(compass.heading())	button A is
	pressed
	Micro:bit
	scrolls to show
	the value of
	compass
$\mathbf{x} = 0$	Set variable
	x=0
x = compass.heading()	Set the value of
	compass to
	variable x
ifelifelse	Condition
	judgement
	statement:ifel





	se ifelse
display.show(Image("00999:""00099:""00909:""09000:""9	Micro:bit
0000"))	shows the
display.show(Image("99900:""99000:""90900:""00090:""0	Northeast
0009"))	arrow sign
display.show(Image("00900:""09000:""999999:""09000:""0	Micro:bit
0900"))	shows the
display.show(Image("00009:""00090:""90900:""99000:""9	Northwest
9900"))	arrow sign
display.show(Image("00900:""00900:""90909:""09990:""0	Micro:bit
0900"))	shows the west
display.show(Image("90000:""09000:""00909:""00099:""0	arrow sign
0999"))	Micro:bit
display.show(Image("00900:""00090:""999999:""00090:""0	shows the
0900"))	Southwest
display.show(Image("00900:""09990:""90909:""00900:""0	arrow sign
0900"))	Micro:bit
	shows the
	South arrow
	sign
	Micro:bit
	shows the





South arrow
sign
Micro:bit
shows the East
arrow sign
Micro:bit
shows the
North arrow
sign

Project 7: Accelerometer



(1) Project Introduction

The Micro: Bit main board V2 has a built-in LSM303AGR gravity acceleration sensor, also known as accelerometer, with a resolution of 8/10/12 bits. The code section sets the range to 1g, 2g, 4g, and 8g.





We often use accelerometer to detect the status of machines.

In this project, we will introduce how to measure the position of the board with the accelerometer. And then have a look at the original three-axis data output by the accelerometer.

(2) Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

(3)Test Code1:

Enter Mu software and open the file "Project 7: Accelerometer-1.py" to import code:

(How to load the project code?)

File Type	Route	File Name
Python	KS4027 folder/Python	Project 7 :
file	Tutorial/Python	Accelerometer-1.py
	Code/Project	
	Code/Project 7 :	
	Accelerometer	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)





(C)	Mu 1.1.0.beta.2 - Project 7: Accelerometer-1.py —		\times
Mode	Image: Save Imag	p Quit	
Proj	ect /: Accelerometer=1.py		
1	from microbit import *		\bigtriangleup
2	shile Town		
3	while frue:		
4	gesture = acceterometer.current_gesture()		
5	<pre>if gesture == "snake"; dicplay_shake";</pre>		
6	if gesture == "up":		
7	display_show("2")		
8	if gesture == "dowp":		
9	display_show("3")		
10	if gesture == "face up":		
12	display.show("4")		
13	if gesture == "face down":		
14	display, show("5")		
15	<pre>if gesture == "left":</pre>		
16	display.show("6")		
17	if gesture == "right":		
18	display.show("7")		
19	if gesture == "freefall":		
20	display.show("8")		\bigtriangledown
	BBC micro:b	it 🗰	O

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





С м	lu 1.1.0.beta.2 - Project 7: Accelerometer-1.py	_		×
Mode Proje	Image: Second	? Help	Quit	
1	from microbit import *			\triangle
2				
3	while True:			
4	gesture = accelerometer.current_gesture()			
5	if gesture == "shake":			
6	display.show("1")			
7	if gesture == "up":			
8	display.show("2")			
9	if gesture == "down":			
10	display.show("3")			
11	if gesture == "face up":			
12	display.show("4")			
13	if gesture == "face down":			
14	display.show("5")			
15	<pre>nf gesture == "left":</pre>			
16	display.snow("6")			
17	<pre>if gesture == "right": diaplay_show("7")</pre>			
18	display.snow("/")			
19	display_show("8")			
20	display.snow("8")			\sim
	BBC mi	cro:bit		Q.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





🕐 м	lu 1.1.0.beta.2 - Project 7: Accelerometer-1.py —		\times
Mode	Image: Save Imag	Quit	
froje	ct /: Accelerometer-1.py		
1	from microbit import *		\bigtriangleup
2			
3	while True:		
4	gesture = accelerometer.current_gesture()		
5	if gesture == "shake":		
6	display.show("1")		
7	if gesture == "up":		
8	display.show("2")		
9	if gesture == "down":		
10	display.show("3")		
- 11	<pre>if gesture == "face up":</pre>		
12	display.show("4")		
13	<pre>if gesture == "face down":</pre>		
14	display.show("5")		
15	<pre>if gesture == "left":</pre>		
16	display.show("6")		
17	if gesture == "right":		
18	display.show("7")		
19	<pre>if gesture == "freefall":</pre>		
20	display.show("8")		\bigtriangledown
	BBC micro:bit		¢.

(4)Test Results1:

After uploading the test code 1 to micro:bit main board and powering the board via the USB cable, if we shake the Micro: Bit main board, no matter at any direction, the LED dot matrix displays the digit "1".

When it is kept upright (make its logo above the LED dot matrix), the number 2 shows.







When it is kept upside down(make its logo below the LED dot matrix) , it shows as below.



When it is placed still on the desk, showing its front side, the number 4 appears.

When it is placed still on the desk, showing its back side, the number 5 exhibits.

When the board is tilted to the left, the LED dot matrix shows the number 6 as shown below.



When the board is tilted to the right , the LED dot matrix displays the





number 7 as shown below:



When the board is knocked to the floor, this process can be considered as a free fall and the LED dot matrix shows the number 8. (Please note that this test is not recommended for it may damage the main board.) Attention: if you' d like to try this function, you can also set the acceleration to 3g, 6g or 8g. But still ,we do not recommend.

(5)Test code2:

Enter Mu software and open the file "Project 7: Accelerometer-2.py" to import code:

File	Route		File Name		
Туре					
Python	KS4027	folder/Python	Project	7	••
file	Tutorial/Python		Acceleromet	er-2.py	
	Code/Project Code/Project				
	7: Accelerometer				





You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)

С м	lu 1.1.0.beta.2 - Project 7: Accelerometer-2.py	_		×
Mode	Image: New Load Image: Save Image: Save	? Help	(U) Quit	
Proje	ct 7: Accelerometer-2.py 🛛 🗶			
1	from microbit import *			
2	while True:			
4	<pre>x = accelerometer.get_x()</pre>			
7	y = accelerometer.get_y()			
9	z = accelerometer.get_z()			
11	print("x, y, z:", x, y, z)			
13	sleep(100)			
	BBC micr	o:bit	#H	¢

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





С м	u 1.1.0.beta.2 - Project 7: Accelerometer-2.py	_		\times
Mode	Image: Save Imag	Help	Quit	
froje	from microbit import *			
2				
3	while True:			
4				
5	<pre>x = accelerometer.get_x()</pre>			
7	y = accelerometer.get_y()			
8				
9	z = accelerometer.get_z()			
10	print("x, v, z:", x, v, z)			
12				
13	sleep(100)			
14				
	BBC m	icro:bit		Q.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



After referring to the MMA8653FC data manual and the hardware schematic diagram of the Micro: Bit main board, the accelerometer coordinate of the Micro: Bit are shown in the figure below:







(6)Test Results2:

Upload the test code 1 to micro:bit main board and power the board via the USB cable.

Click "REPL" and press the reset button. The value of acceleration on X axis,

Y axis and Z axis are shown below:

r 🖉	lu 1.1.0.beta.2 - Project 7: Accelerometer-2.py	—		\times
Mode	Image: New Load Save Image: Save	? Help	Quit	
Proje	ct 7: Accelerometer-2.py 🗶			
1	from microbit import *			\bigtriangleup
2	while True:			
4				
5	<pre>x = accelerometer.get_x()</pre>			
6				
7	y = accelerometer.get_y()			
8				
9	z = accelerometer.get_z()			
10	print("x y z:" x y z)			
12	princ(x, y, z, , x, y, z)			
13	sleep(100)			\bigtriangledown
BBC	microshit REDI			
X. V.	z: -788 -376 576			
х, у,	z: -772 -396 548			
х, у,	Z: -836 -392 644			
х, у,	Z: -808 -436 260			
х, у,	z: -1024 -524 336			
х, у,	z: -676 -540 -828			
х, у,	z: -484 -580 -1364			
х, у,	z: -1012 104 284			
х, у,	z: -1412 580 504			
х, у,	z: -1368 468 148			
х, у,	z: -992 32 48			
х, у,	Z: 12 -808 -480			
х, у,	Z: -160 -584 364			
х, у,	Z: -656 -368 844			
х, у,	Z: -1244 20 1988			∇
				*
	BBC M10	10.01T		μF





(7)Code Explanation:

from microbit import *	Import the library file of micro: bit
gesture =	Set accelerometer.current_gesture()
accelerometer.current_gesture()	to gesture
while True:	This is a permanent loop that makes
	micro:bit execute the code of it.
if gesture == "shake":	Shaking micro:bit board, number 1
display.show("1")	will appear
if gesture == "up":	When log points to the North,
display.show("2")	number 2 will show up.
if gesture == "down":	When log points to the South,
display.show("3")	number 3 will be shown
if gesture == "face up":	When the LED dot matrix is upward,
display.show("4")	the number 4 is shown.
if gesture == "face down":	the number 5 is displayed when the
display.show("5")	LED dot matrix is downward.
if gesture == "left":	When Micro:bit board is tilt to the
display.show("6")	left, number 6 is shown.
if gesture == "right":	When micro:bit is tilt to the right
display.show("7")	When Micro:bit board is inclined to
if gesture == "freefall":	the right, number 7 is displayed.





display.show("8") When it is free fall(accidentally	
	making it fall), number 8 appears on
	dot matrix.
x = accelerometer.get_x()	Read the acceleration value on x
y = accelerometer.get_y()	axis, the return value is integer, and
z = accelerometer.get_z()	set x= the read value on x axis
	Read the acceleration value on y
	axis, the return value is integer, and
	set y= the read value on y axis
	Read the acceleration value on z
	axis, the return value is integer, and
	set z= the read value on z axis
print("x, y, z:", x, y, z)	The value of acceleration will be
	shown
sleep(100)	Delay in 100ms





Project 8: Light Detection



(1) Project Introduction

In this project, we focus on the light detection function of the Micro: Bit main board. It is achieved by the LED dot matrix since the main board is not equipped with a photoresistor.

(2) Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

(3)Test Code:

Enter Mu software and open the file "project 8: Light Detection.py" to import code:

(How to load the project code?)

File Type	Route		File Name	
Python file	KS4027 folder/Python		project 8: Light Detection.py	
	Tutorial/Python			
	Code/Project	Code/Project		





8: Light Detection

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.







If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.





(M	1u 1.1.0.beta.2 - project 8: Light Intensity Detection.py — 🗆 🗙	
Mode	Image: Save set of the same set	
proje	from microbit import *	_
2		
3	while True:	
4		
5	Lightintensity = display.read_light_level()	
7	print("Light intensity:", Lightintensity)	
8		
9	sleep(50)	
10	if display read light level() <= 20:	
11	display.show(Image("00990:""09900:""09900:""09900:""09900))	
13	else:	
14	display.show(Image("90909:""09990:""99999:""09990:""90909"))	
15		
	BBC micro:bit 🗰 🔅	

(4)Test Results:

Upload the test code to micro:bit main board, power the board via the USB cable and click "Show console Device".

Download code onto micro:bit board, don' t plug off USB cable. Click "REPL" and press the reset buttons, the light intensity value will be displayed, as shown below.

When the LED dot matrix is covered by hand, the light intensity showed is approximately 0; when the LED dot matrix is exposed to light, the light intensity displayed gets stronger with the light.

The 20 in the code is an arbitrary value of light intensity. If the current light





level is less than or equal to 20, the icon moon will appear on the LED dot matrix. If it's bigger than 20, the sun will appear.

(M	u 1.1.0.beta.2 - project 8: Light Intensity Detection.py –	\times
Mode	+ New Load Save Flash Files EFL Plotter Coom-in Zoom-out Theme Check Tidy Help Quit	
projec	ot 8: Light Intensity Detection.py 🛛 🗶 🥄	
1	from microbit import *	\triangle
2		
3	while True:	
5	Lightintensity = display.read light level()	
6	2.8	
7	<pre>print("Light intensity:", Lightintensity)</pre>	
8		
9	sleep(50)	
10	if display read light lovel() <= 20:	
11	display.show(Image("00990:""09900:""09900:""09900:""09900:""09900"))	
13	else:	
14	display.show(Image("90909:""09990:""99999:""09990:""90909"))	\bigtriangledown
BBC n	nicro:bit REPL	
Light	intensity: 14	\bigtriangleup
Light	intensity: 16	
Light Light	intensity: 16 intensity: 25	
Light	intensity: 21	
Light	intensity: 22	
Light	intensity: 25	
Light	intensity: 28	
Light Light	intensity: 34 intensity: 48	
Light	intensity: 64	-
	BBC micro:bit	Q

(5)Code Explanation:

from microbit import *	obit import * Import the library file of micro: bit		
gesture =	Set accelerometer.current_gesture()		
accelerometer.current_gesture()	to gesture		
while True:	This is a permanent loop that makes		





	micro:bit execute the code of it.
Lightintensity =	Set display.read_light_level() to
display.read_light_level()	Lightintensity
<pre>print("Light intensity:",</pre>	BBC microbit REPL prints the
Lightintensity)	detected light intensity value
sleep(100)	Delay in 100ms

Project 9: Speaker



(1) Project Introduction

Micro: Bit main board has an built-in speaker, which makes adding sound to the programs easier. It can also be programmed to air all kinds of tones, like playing the song *Ode to Joy*.

(2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;B.Open the offline version of Mu.

(3)Test Code1:





Enter Mu software and open the file "Project 9: Speaker-1.py" to import code:

(How to load the project code?)

File Type	Route		File Name		
Python file	KS4027	folder/Python	Project	9	:
	Tutorial/Python		Speaker-1.py		
	Code/Project				
	Code/Proje	ct 9: Speaker			

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)







Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.

С м	lu 1.1.0.beta.2 - Project 9: Speaker-1.py	_		×
Mode Proje	Image: Speaker 1. py Image: Speaker 1. py <td< th=""><th>? Help</th><th>Quit</th><th></th></td<>	? Help	Quit	
1	<pre>from microbit import *</pre>			
2				
3	import audio			
4				
5	display.show(Image.MUSIC_QUAVER)			
6				
7	while True:			
8	audio.play(Sound.GIGGLE)			
9	steep(1000)			
10	sleen(1000)			
12	audio_nlav(Sound_HELLO)			
13	sleep(1000)			
14	audio.play(Sound.YAWN)			
15	sleep(1000)			
16				
	BBC micr	o:bit		0

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





С м	u 1.1.0.beta.2 - Project 9: Speaker-1.py	_		×
Mode	Image: New Load Image: Save Image: Save	? Help	Quit	
Proje	ct 9: Speaker-1.py 🗶 🥄			
1	from microbit import *			
2				
3	import audio			
4				
Б	display.show(Image.MUSIC_QUAVER)			
6				
7	while True:			
8	audio.play(Sound.GIGGLE)			
9	steep(1000)			
10	audio.play(Sound.HAPPY)			
11	steep(1000) audio play (Sound HELLO)			
12	sleen(1000)			
13	audio_play (Sound_YAWN)			
15	sleep(1000)			
16				
				-
	BBC micr	o:bit		Q.

(4)Test Results1:

After uploading the test code1 to micro:bit main board and powering the board via the USB cable, the speaker utters sound and the LED dot matrix shows the logo of music.

(5)Test Code2:

Enter Mu software and open the file "Project 9: Speaker-2.py" to import code:

```
(How to load the project code?)
```




File Type	Route		File Name		
Python file	KS4027 folder/Python		Project	9	•
	Tutorial/Python		Speaker-2	.ру	
	Code/Project Code/Project				
	9: Speaker				

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)

N	lu 1.1.0.beta.2 - Project 9: Speaker-2.py — 🗌	\times
Mode	Image: New Load Image: Save Image: Save </th <th></th>	
Proje	ect 9: Speaker-2. py 🛛 🗶	
1	from microbit import *	\triangle
2	import music	
3	display.show(Image.MUSIC_QUAVER)	
4	tune = ["E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4",	
5	"C5:4", "C5:4", "D5:4", "E5:4", "E5:4", "D5:4", "D5:4", "E5:4",	
6	"E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4", "C5:4",	
7	"C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4", "D5:4", "D5:4",	
8	"E5:4", "C5:4", "D5:4", "E5:2", "F5:2", "E5:4", "C5:4", "D5:4",	
9	"E5:2", "F5:2", "E5:4", "D5:4", "C5:4", "D5:4", "G4:4", "E5:4",	
10	"E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4",	
п	"C5:4", "C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4", "D5:4",	
12	"D5:4", "E5:4", "C5:4", "D5:4", "E5:2", "F5:2", "E5:4", "C5:4",	
13	"D5:4", "E5:2", "F5:2", "E5:4", "D5:4", "C5:4", "D5:4", "G4:4",	
14	"E5:4", "E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4",	
15	"C5:4", "C5:4", "C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4",	
16	"D5:4", "C5:2", "C5:4", "G5:4", "F5:4", "E5:2", "E5:4", "C5:4",	
17	"B5:4", "A5:2", "A5:4", "F5:2", "D5:2", "C5:2", "B4:2", "D5:2",	
18	"B4:2", "A4:2", "G4:2", "A4:2", "B4:2", "C5:2", "E5:2", "D5:2",	
19	"B4:2", "C5:4", "C5:2", "C5:1", "C5:4"]	
20		
21	while True:	
22	music.play(tune)	\bigtriangledown
	BBC micro:bit 🗰 🕻	¥

Click "Check" to examine error in the code. The underlines and cursors





signal that the program is wrong.

N	Mu 1.1.0.beta.2 - Project 9: Speaker-2.py -	×
Mode	Image: state	
Proje	ect 9: Speaker-2. py 🗙	
1	from microbit import *	\triangle
2	import music	
3	display.show(Image.MUSIC_QUAVER)	
4	tune = ["E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4",	
5	"C5:4", "C5:4", "D5:4", "E5:4", "E5:4", "D5:4", "D5:4", "E5:4",	
6	"E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4", "C5:4",	
7	"C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4", "D5:4", "D5:4",	
8	"E5:4", "C5:4", "D5:4", "E5:2", "F5:2", "E5:4", "C5:4", "D5:4",	
9	"E5:2", "F5:2", "E5:4", "D5:4", "C5:4", "D5:4", "G4:4", "E5:4",	
10	"E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4",	
11	"C5:4", "C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4", "D5:4",	
12	"D5:4", "E5:4", "C5:4", "D5:4", "E5:2", "F5:2", "E5:4", "C5:4",	
13	"D5:4", "E5:2", "F5:2", "E5:4", "D5:4", "C5:4", "D5:4", "G4:4",	
14	"E5:4", "E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4",	
15	"C5:4", "C5:4", "C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4",	
16	"D5:4", "C5:2", "C5:4", "G5:4", "F5:4", "E5:2", "E5:4", "C5:4",	
17	"B5:4", "A5:2", "A5:4", "F5:2", "D5:2", "C5:2", "B4:2", "D5:2",	
18	"B4:2", "A4:2", "G4:2", "A4:2", "B4:2", "C5:2", "E5:2", "D5:2",	
19	"B4:2", "C5:4", "C5:2", "C5:1", "C5:4"]	
20		
21	while True:	
22	music.play(tune)	\bigtriangledown
	BBC micro:bit 🗰 🕻	ŀ

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





(N	Nu 1.1.0.beta.2 - Project 9: Speaker-2.py —	×
Mode	+ ++ + + +	
Proje	ect 9: Speaker-2.py 🗙 🥄	
1	from microbit import *	\triangle
2	import music	
3	display.show(Image.MUSIC_QUAVER)	
4	tune = ["E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4",	
5	"C5:4", "C5:4", "D5:4", "E5:4", "E5:4", "D5:4", "D5:4", "E5:4",	
6	"E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4", "C5:4",	
7	"C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4", "D5:4", "D5:4",	
8	"E5:4", "C5:4", "D5:4", "E5:2", "F5:2", "E5:4", "C5:4", "D5:4",	
9	"E5:2", "F5:2", "E5:4", "D5:4", "C5:4", "D5:4", "G4:4", "E5:4",	
10	"E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4", "D5:4",	
п	"C5:4", "C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4", "D5:4",	
12	"D5:4", "E5:4", "C5:4", "D5:4", "E5:2", "F5:2", "E5:4", "C5:4",	
13	"D5:4", "E5:2", "F5:2", "E5:4", "D5:4", "C5:4", "D5:4", "G4:4",	
14	"E5:4", "E5:4", "E5:4", "F5:4", "G5:4", "G5:4", "F5:4", "E5:4",	
15	"C5:4", "C5:4", "C5:4", "D5:4", "E5:4", "D5:4", "C5:2", "C5:4",	
16	"D5:4", "C5:2", "C5:4", "G5:4", "F5:4", "E5:2", "E5:4", "C5:4",	
17	"B5:4", "A5:2", "A5:4", "F5:2", "D5:2", "C5:2", "B4:2", "D5:2",	
18	"B4:2", "A4:2", "G4:2", "A4:2", "B4:2", "C5:2", "E5:2", "D5:2",	
19	"B4:2", "C5:4", "C5:2", "C5:1", "C5:4"]	
20		
21	while frue:	
22	music.play(tune)	\triangleleft
	BBC micro:bit 🗮 🐔	5

The musical score of *Ode to Joy* is attached below:





Ode To Joy
$1 = B \frac{2}{4} = 120$ Beethoven
$ \begin{array}{c} \begin{smallmatrix} 1 \\ 3 \\ f \end{smallmatrix} \\ \begin{array}{c} 3 \\ f \end{smallmatrix} \\ \begin{array}{c} 3 \\ 5 \end{smallmatrix} \\ \begin{array}{c} 4 \\ 5 \end{smallmatrix} \\ \begin{array}{c} 5 \\ 4 \\ 5 \\ \end{array} \\ \begin{array}{c} 2 \\ 2 \\ 1 \\ 1 \\ \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ 1 \\ \end{array} \\ \begin{array}{c} 2 \\ 2 \\ 2 \\ \end{array} \\ \begin{array}{c} 2 \\ 2 \\ 2 \\ \end{array} \\ \begin{array}{c} 2 \\ 2 \\ 3 \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ 3 \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ 3 \\ \end{array} \\ \end{array} \\ \end{array} \\ \begin{array}{c} 2 \\ \end{array} \\$
^[2] 5 4 3 2 1 1 2 3 2 · <u>1</u> 1 0 <u>2</u> 2 3 1 <i>mp</i> crese
$\dot{\underline{34}}\dot{\underline{34}}\dot{\underline{31}}\underline{$
$\dot{5}$ $\dot{4}$ $\dot{3}$ $\dot{2}$ $\dot{1}$ $\dot{1}$ $\dot{2}$ $\dot{3}$ $\dot{2}$ · $\underline{\dot{1}}$ $\dot{1}$ 0 $\dot{\underline{\dot{2}}}$ $\dot{\underline{\dot{2}}}$ $\dot{3}$ $\dot{1}$ <i>mp</i> crese
$\dot{2}$ $\dot{\underline{34}}$ $\ddot{3}$ $\dot{1}$ $\dot{2}$ $\dot{\underline{34}}$ $\ddot{3}$ $\dot{2}$ $\dot{1}$ $\dot{2}$ 5 $\overset{\vee}{3}$ $\dot{3}$ $\dot{3}$ $\dot{3}$ $\dot{4}$ $\dot{5}$ \dot{f}
$5\dot{4}\dot{3}\dot{2} \dot{1}\dot{1}\dot{2}\dot{3} \dot{2}\cdot\underline{\dot{1}}\dot{1}0: \dot{2}\cdot\underline{\dot{1}}\dot{1}\overset{>}{\dot{5}} $
$\dot{\underline{3}} \cdot \underline{3} \dot{\underline{3}} \dot{\underline{1}} ^{\flat} \dot{\overline{7}} \cdot \underline{6} \dot{\underline{6}} \dot{\underline{42}} \underline{\underline{17276567}} \underline{\underline{1327}} \dot{\underline{1327}} \dot{\underline{1}} \underline{\underline{1}} \underline{\underline{1}} $
i o o o

Find more information about musical notations via this link: https://en.wikipedia.org/wiki/Numbered_musical_notation

(6) Test Results2:

After uploading the test code2 to micro:bit main board and powering the board via the USB cable, the speaker on built-in the Micro:bit board plays the sound *Ode to Joy* and the LED dot matrix shows the logo of music.





(7)Code Explanation:

from microbit import *	Import the library of micro: bit
import audio	Audio library
while True:	This is a permanent loop that
	makes micro:bit execute the code
	of it.
audio.play(Sound.GIGGLE)	Emit the "giggle" sound
display.show (Image.MUSIC_QUAVER)	Music logo shows on the LED dot
	matrix on the micro:bit
import music	Import music library controlling
	sounds
tune = ["E5:4" , "E5:4" , "F5:4" ,	Create variable" tune" to save
"G5:4", "G5:4", "F5:4",	notes
"E5:4", "D5:4", "C5:4",	
"C5:4", "D5:4", "E5:4",	
"E5:4", "D5:4","D5:4",	
"E5:4", "E5:4", "F5:4",	
"G5:4", "G5:4", "F5:4",	
"E5:4", "D5:4", "C5:4",	
"C5:4", "D5:4", "E5:4",	





"D5:4", "C5:2", "C5:4",				
"D5:4" ,	"D5:4" ,	"E5:4",		
"C5:4" ,	"D5:4", "	E5:2",		
"F5:2",	"E5:4",	"C5:4" ,		
"D5:4" ,	"E5:2",	"F5:2",		
"E5:4" ,	"D5:4" ,	"C5:4" ,		
"D5:4" ,	"G4:4" ,	"E5:4",		
"E5:4", "I	E5:4", "F	5:4",		
"G5:4" ,	"G5:4",	"F5:4",		
"E5:4" ,	"D5:4",	"C5:4" ,		
"C5:4" ,	"D5:4",	"E5:4",		
"D5:4" ,	"C5:2",	"C5:4",		
"D5:4" ,	"D5:4",	"E5:4",		
"C5:4" ,	"D5:4",	"E5:2",		
"F5:2", "I	E5:4" , "(25:4",		
"D5:4" ,	"E5:2",	"F5:2",		
"E5:4" ,	"D5:4", "	C5:4" ,		
"D5:4" ,	"G4:4" ,	"E5:4",		
"E5:4" ,	"E5:4" ,	"F5:4",		
"G5:4" ,	"G5:4" ,	"F5:4",		
"E5:4" ,	"C5:4" ,	"C5:4" ,		
"C5:4", "	D5:4" , "	E5:4",		



"D5:4", "C5:2", "C5:4",	
"D5:4", "C5:2", "C5:4",	
"G5:4", "F5:4", "E5:2",	
"E5:4", "C5:4", "B5:4",	
"A5:2", "A5:4", "F5:2",	
"D5:2", "C5:2", "B4:2",	
"D5:2", "B4:2", "A4:2",	
"G4:2", "A4:2", "B4:2",	
"C5:2", "E5:2", "D5:2",	
"B4:2", "C5:4", "C5:2",	
"C5:1", "C5:4"]	
music.play(tune)	Use the function play () to play the
	notes reserved in "tune"
sleep(1000)	delay in 1000ms

Project 10: Touch-sensitive Logo



(1) Project Introduction





The Micro: Bit main board V2 is equipped with a golden touch-sensitive logo, which can act as an input component and function like an extra button.

It contains a capacitive touch sensor that senses small changes in the electric field when pressed (or touched), just like your phone or tablet screen do.When you press it, you can activate the program.

(2)Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;



B.Open the offline version of Mu.

(3)Test Code:

Enter Mu software and open the file "Project 10: Touch-sensitive Logo.py"

to import code:

```
(How to load the project code?)
```





File Type	Route		File Name
Python file	KS4027	folder/Python	Project 10: Touch-sensitive
	Tutorial/Python		Logo.py
	Code/Project Code/Project		
	10: Touch-sensitive Logo		

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)

🕐 М	u 1.1.0.beta.2 - Project 10: Touch Sensitive Logo.py -		\times		
Mode	Image: sevent	Quit			
Projec	ot 10: Touch Sensitive Logo.py 🛛 🕱				
1	from microbit import *		\bigtriangleup		
2	time = 0				
3	start = 0				
4	running = Fatse				
6 6	while True:				
7					
8	<pre>if button_a.was_pressed():</pre>				
9	running = True				
10	<pre>start = running_time()</pre>				
11	<pre>if button_b.was_pressed():</pre>				
12	if running:				
13	time += running_time() - start				
14	running = False				
15	<pre>if pat rupping:</pre>				
16	if not running: display_scroll(int(time/1000))				
18	if running:				
19	display.show(Image.HEART)				
20	sleep(300)				
21	display.show(Image.HEART_SMALL)				
22	sleep(300)				
23	else:				
24	display.show(Image.ASLEEP)		\bigtriangledown		
	BBC micro:bit	#	¢		

How Micro:bit works?

A. The runtime is recorded in milliseconds(ms).





- B. When you press button A, a variable named start is set to the current running time.
- C. When you press button B, the start time will be subtracted from the new running time to calculate how much time has passed since you started the stopwatch. This difference is added to the total time, which is stored in a variable named time.
- D. If you press the golden logo, the program will display the total elapsed time on the LED display. It converts time from milliseconds (thousandths of a second) to seconds by dividing by 1000. It uses the integer division operator to give an integer (integer) result.
- E. The program is also controlled by a Boolean variable named running. Boolean variable can only have two values: true or false. If "running" is "true", it means that the stopwatch has started. If "running" is false, it means that the stopwatch has not started or has stopped.
- F. If "running" is true, the beating heart pattern is displayed on the LED dot matrix screen.
- G. (7) If the stopwatch has stopped and the "running" is false, when you press the golden logo, it will only display the time.
- H. If the stopwatch has been started and "running" is true, it only need to ensure that the time variable will only change when button B is pressed, and the code can also prevent false readings.





Click "Check" to examine error in the code. The underlines and cursors

signal that the program is wrong.



If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





С м	u 1.1.0.beta.2 - Project 10: Touch Sensitive Logo.py –	×					
Mode Rusia	Image: Save Image: Save						
Project 10: Touch Sensitive Logo.py							
1	time = 0						
2	start = 0						
4	running = False						
5							
6	while True:						
7							
8	<pre>if button_a.was_pressed():</pre>						
9	running = True						
10	start = running_time()						
11	<pre>if button_b.was_pressed():</pre>						
12	if running:						
13	time += running_time() - start						
14	running = False						
15	if path supping:						
16	<pre>if not running: display scroll(int(time(1000)))</pre>						
17	aisplay.scroll(int(time/1000))						
19	display.show(Image.HEART)						
20	sleep(300)						
21	display.show(Image.HEART_SMALL)						
22	sleep(300)						
23	else:						
24	display.show(Image.ASLEEP)	∇					
	BBC micro:bit	۲¢					

(4)Test Results:

Upload the test code to micro:bit main board and power the board via the USB cable, and press button A to start the stopwatch. When timing, the beating heart pattern will be displayed on the LED dot matrix screen. Press button B to stop it and you can start and stop it at any time. It will keep recording time, just like a real stopwatch. Press the golden logo on the front of the micro:bit to display the measured time in seconds. And time can be reset to zero by pressing the reset button on the back of it.





Project 11: Microphone



(1) Project Introduction

The Micro: Bit main board is built with a microphone which can test the volume of ambient environment. When you clap, the microphone LED indicator turns on. Since it can measure the intensity of sound, you can make a noise scale or disco lighting changing with music. The microphone is placed on the opposite side of the microphone LED indicator and in proximity with holes that lets sound pass. When the board detects sound, the LED indicator lights up.

(2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;B.Open the offline version of Mu.

(3)Test Code:

Enter Mu software and open the file "Project 11: Microphone-1.py" to import code:





(How to load the project code?)

File Type	Route		File Name		
Python file	KS4027 folder/Python		Project	11	•
	Tutorial/Python		Microphone	e-1.py	
	Code/Project Code/Project				
	11: Microphone				

You can also input code in the editing window yourself. (note:all English

words and symbols must be written in English)



Click "Check" to examine error in the code. The underlines and cursors





signal that the program is wrong.

PMu 1.1.0.beta.2 - Project 11: Microphone-1.py	_		\times
Image: Mode Imag	? Help	Quit	
Project 11: Microphone-1.py			
from microbit import *			
while True:			
<pre>if microphone.current_event() == SoundEvent.LOUD: display.show(Image.HEART) sleep(200) if microphone.current_event() == SoundEvent.QUIET: display.show(Image.HEART_SMALL)</pre>			
BBC mi	cro:bit		¢.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.







(4)Test Results1:

After uploading test code to micro:bit main board and powering the board via the USB cable, the LED dot matrix displays pattern "

(5)Test Code2:

Enter Mu software and open the file "Project 11: Microphone-2.py" to import code:

```
(How to load the project code?)
```





File	Route		File Name		
Туре					
Python	KS4027	folder/Python	Project	11	:
file	Tutorial/Python		Microphone	-2.py	
	Code/Project Code/Project				
	11: Microphone				

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



Click "Check" to examine error in the code. The underlines and cursors





signal that the program is wrong.

Mu 1.1.0.beta.2 - Project 11: Microphone-2.py	_		\times
Image: Mode Image: Mode	? Help	Quit	
Project 11: Microphone-2. py 🗙			
<pre>1 from microbit import *</pre>			
$_{2}$ maxSound = 0			
<pre>3 Lights = Image("11111:""11111:""11111:""11111")</pre>			
<pre>4 # ignore first sound level reading</pre>			
<pre>soundLevel = mfcrophone.sound_level()</pre>			
6 SLeep(200)			
7			
<pre>% while True:</pre>			
<pre>if button_a.is_pressed():</pre>			
10 display.scroll(maxSound)			
n else:			
<pre>12 soundLevel = mfcrophone.sound_level()</pre>			
<pre>13 display.show(lights * soundLevel)</pre>			
14 if soundLevel > maxSound:			
maxSound = soundLevel			
16			
BBC mic	ro:bit		Ö

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.

Upload test code to micro:bit main board, power the board via the USB cable







(6)Test Results2:

Upload test code to micro:bit main board and power the board via the USB cable. When the button A is pressed, the LED dot matrix displays the value of the biggest volume(please note that the biggest volume can be reset via the Reset button on the other side of the board) while when clapping, the LED dot matrix shows the pattern of the sound.





(7)Code Explanation:

from microbit import *	Import the library of
	micro: bit
while True:	This is a permanent loop
	that makes micro:bit
	execute the code of it.
<pre>if microphone.current_event() ==</pre>	If there is a sound
SoundEvent.LOUD:	LED shows 💙
display.show(Image.HEART)	Delay in 200ms
sleep(200)	if no sound is detected
if microphone.current_event() ==	LED lights show
SoundEvent.QUIET:	
display.show(Image.HEART_SMALL)	
<pre>print("Light intensity:", Lightintensity)</pre>	BBC microbit REPL prints
	the detected light
	intensity value
maxSound = 0	The initial value of
	maxSound is 0
lights =	Assign Image() to
Image("11111:""11111:""11111:""11111:""11111	variable lights
")	
soundLevel = microphone.sound_level()	Assign





	microphone.sound_level
	() to the variable
	soundLevel
if button_a.is_pressed():	if the button A is pressed
display.scroll(maxSound)	LED lights show the
else:	sound value
soundLevel = microphone.sound_level()	lf not
display.show(lights * soundLevel)	Assign
if soundLevel > maxSound:	microphone.sound_level
maxSound = soundLevel	() to the variable
	soundLevel
	As the sound changes,
	the micro:bit will display
	the breathing light effect
	If the sound value is
	higher than its maximum
	value
	the maximum sound
	value is equal to sound
	level value





Project 12: Touch-sensitive Logo Controlled Speaker

(1) Project Introduction

In the previous projects, we have learned about the touch-sensitive logo and the speaker respectively. In the project, we will combine these two components to play music. That' s the logo will be applied to control the speaker to sing songs.

(2)Components Needed:



(3)Connection Diagram:

Attach the Micro:bit main board to your computer via the USB cable.



(4)Test Code:





Enter Mu software and open the file "Project 12: Touch-sensitive Logo Controlled Speaker.py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 12: Touch-sensitive Logo
file	Tutorial/Python	Controlled Speaker.py
	Code/Project Code/Project	
	12 : Touch-sensitive Logo	
	Controlled Speaker	

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)





() M	lu 1.1.0.beta.2 - Project 12: Touch the Logo to control the speaker.py			×
Mode Proje	Image: Save Image: Save	dy Help	Quit	
1	from microbit import *			
2				
3	import music			
4				
5	display.show(image.MUSIC_QUAVER)			
7	while True:			
8				
9	<pre>if pin_logo.is_touched():</pre>			
10	music.play(music.BIRTHDAY)			
- 11				
	BBC	micro:bit	# 1	¢

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.







If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





(M	lu 1.1.0.beta.2 - Project 12: Touch the Logo to control the speaker.py —		×
Mode Proje	Image: New Load Image: Save Image: Save </td <td>p Quit</td> <td></td>	p Quit	
1	from microbit import *		
2			
3	import music		
4	display show(Image MUSIC OUAVER)		
6	arspeay (show(image, hosic_(ovariv)		
7	while True:		
8			
9	<pre>if pin_logo.is_touched():</pre>		
10	music.play(music.BIRTHDAY)		
11			
	BBC micro:b	it 🗰	0

(5)Test Results:

After uploading test code to micro:bit main board and powering the board via the USB cable, the speaker plays the song *Happy Birthday to You* when the logo is touched.

(6)Code Explanation:

from microbit import *	Import the library of micro: bit
while True:	This is a permanent loop that
	makes micro:bit execute the
	code of it.





display.show (Image.MUSIC_QUAVER)	Music logo shows on the LED
	dot matrix on the micro:bit
if pin_logo. is_touched():	When the logo is touched, it
	executes the following
	command
music.play (music.BIRTHDAY)	The speaker plays the
	song" Happy Birthday to You"

Project 13: Bluetooth Wireless Communication

With 16k RAM, micro:bit owns a low-consumption Bluetooth module and support Bluetooth communication. However, BLE heap stack occupies 12K RAM, which implies that there is no enough space to run microPython. At present, microPython doesn't support Bluetooth.

https://microbit-micropython.readthedocs.io/en/latest/ble.html





7.Expansion Projects:

The former 14 projects are the introduction of sensors and modules. The further lessons are challenging for new starters.

Note: (G), marked on each sensor and module, is the negative pole and connected to "G", "-" or "GND" on the sensor shield or control board; (V) is the positive pole and linked with V, VCC, + or 5V on the sensor shield or control board. And you need to connect a power in case that power supply is weak.

Project 1: LED Blinks



(1)Project Introduction

We' ve set up the micro:bit smart home. Now let' s get started from the most simple experiment---LED blinks.

LED is a type of semiconductor called "Light Emitting Diode "which is an electronic device made of semiconductor materials (silicon, selenium,





germanium, etc.). It features unidirectional conductivity, that is, the positive voltage is applied to the anode (long leg) and the cathode (short leg) of the diode. when the voltage of its anode is higher than the voltage of its cathode, thus, the diode is turned on(LED is on). When a reverse voltage is applied to the anode and cathode, the diode is disconnected(that is, the LED is off). Therefore, the disconnection and connection of the diode is equivalent to turning on and off LED. Light-emitting diodes have an anode (+) and a cathode (-), and they can only allow current to flow from one anode to the cathode. The components will be damaged if LED is directly connected in series in the LED circuit.

This LED module can be controlled by a basic code to turn on and off the light alternatively, simulating the breathing effect. And the time gap can be changed in the code. When the signal end S is at high level, the LED lights up while when it is at low level the LED reminds off.





(2)About the Yellow LED:



(3)Test Code

Micro: bit	Vellow LED
Expansion	Modulo
Board	would





GND	G
5V	V
S (16)	S

Enter Mu software and open the file "Project 1: LED Blinks.py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 1: LED
file	Tutorial/Python	Blinks.py
	Code/Expansion Project	
	Code/Project 1: LED Blinks	

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)





n 🕞	u 1.1.0.beta.2 - Project 1: LED flashing.py —		\times			
Mode Projec	Image: New Load Image: Save Image: Save </td <td>Quit</td> <td></td>	Quit				
1 from microbit import *						
2						
3	display.show(Image.HAPPY)					
4						
5	pin16.write_digital(0)					
6						
7	while True:					
8	pin16.write_digital(1)					
9	sleep(1000)					
10	pin16.write_digital(0)					
11	sleep(1000)					
12						
	BBC micro:bi	t 🗰	¢			

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.







If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





P Mu	lu 1.1.0.beta.2 - Project 1: LED flashing.py —		\times			
Mode Projec	+ +	Quit				
1 from microbit import *						
2						
3	display.show(Image.HAPPY)					
4						
5	pin16.write_digital(0)					
6						
7	while True:					
8	<pre>s pin16.write_digital(1)</pre>					
9	sleep(1000)					
10	pin16.write_digital(0)					
11	sleep(1000)					
12						
	BBC micro:bit		3			

(4)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

The micro:bit will show smile expression, and a yellow LED will flash with an interval of 1000ms.





(5)Code Explanation:

from microbit import *	Import the library file of micro:	
	bit	
display.show(Image.HAPPY)	The LED dot matrix on the	
	microbit displays a "smiley face"	
	pattern	
while True:	This is a permanent loop that	
	makes micro:bit execute the	
	code of it.	
Pin16.write_digital(1)	Control pin 16 to output high	
	level to light up the LED	
Pin16.write_digital(0)	Control pin 16 to output low	
	level, turn off the LED	
sleep(1000)	Delay in 1000 ms	





Project 2: Breathing LED



(1) Project Introduction



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. Similarly, you can change the step length and delay time in the code so as to demonstrate different breathing effects.

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 3V. What if the 1.5V is required? Or a switch among 1V, 1.5V and 3V? We cannot change resistors constantly. For this reason, we resort to PWM.




For Micro:bit digital port voltage outputs, there are only LOW and HIGH levels, which correspond to the voltage outputs of 0V and 3V respectively. You can define LOW as "0" and HIGH as "1", and let Micro:bit output five hundred "0" or '1' within 1 second. If output five hundred '1", that is 3V; if all of which is '0', that is 0V; if output 250 01 pattern, that is 1.5V. This process can be likened to showing a movie. The movie we watch are not completely continuous. Actually, it generates 25 pictures per second, which cannot be told by human eyes. Therefore, we mistake it as a continuous process. PWM works in the same way. To output different voltages, we need to control the ratio of 0 and 1. The more '0' or '1' output per unit time, the more accurate the control.

In the graphic below, the green lines represent a regular time period. This duration or period is the inverse of the PWM frequency. In other words, with Micro:bit's PWM frequency at about 500Hz, the green lines would measure 2 milliseconds each. A call to analogWrite() is on a scale of 0-255, such that analogWrite(255) requests a 100% duty cycle (always on), and analogWrite(127) is a 50% duty cycle (on half the time).





PWM is applied to light brightness adjustment, speed adjustment of motor and sound emitting

Parameters of PWM:







A.pulse width (minimum / max)

B.Pulse cycle (insertion of pulse frequency within 1 second)

C.Voltage level (0V-3V)

D.There are commonly used PWM ports, namely P0, P1, P2, P3, P4 and P10.

And there are other rarely used ports, namely P5, P6, P7, P8, P9, P11, P12,

P13, P14, P15, P16, P19 and P20.

In the experiment, we connect the port S of yellow LED Module to the port S (16) of the expansion board. And P16 can also be used as a PWM interface.

(2)About the Yellow LED:

Working	DC 3.3-5V	VCC
Voltage:		R3
Working	< 20mA	YELLOW LED
Current:		$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
Max	0.1W	2 VCC 0603 1k R2 1 0603 10k C1
Power:		GND GND
Control	digital port	
Port:	(digital	





	input)
Working	-10 ° C ~
Temperat	+50°C
ure:	
Display	Yellow
Color:	

(3)Test Code

Micro:bit	Yellow	
Expansion	LED	
Board	Module	
GND	G	
5V	V	
S (16)	S	

Enter Mu software and open the file "Project 2: Breathing LED .py" to import code:

(How to load the project code?)

File		Route	File Name
Туре			
Python	KS4027	folder/Python	Project 2: Breathing





file	Tutorial/Python	LED .py
	Code/Expansion Project	
	Code/Project 2: Breathing	
	LED	

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)

P Mu	u 1.1.0.beta.2 - Project 2: Breathing lamp.py	_		×	
Mode	Image: New Load Image: Save Image: Save	? Help	Quit		
Projec	ot 2: Breathing lamp.py 🛛 🗶				
1	<pre>from microbit import *</pre>				
2					
3	display.show(Image.HAPPY)				
4					
5	pin16.write_digital(0)				
6					
7	while True:				
8	for index in range (0, 255):				
9	pin16.write_analog(index)				
10	sleep(10)				
11	for index in range (0, 255):				
12	pin16.write_analog(255-index)				
13	sleep(10)				
14					
15					
		1			
	BBC micro	o:bit	# I	Q.	

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





n 🕞	u 1.1.0.beta.2 - Project 2: Breathing lamp.py – 🗆 🗙
Mode Node	Image: New Load Image: Save Image: Save
Projec	t 2: Breathing lamp.py 🗙
1	from microbit import *
2	
3	display.show(Image.HAPPY)
4	
5	pin16.write_digital(0)
6	
7	while True:
8	for index in range (0, 255):
9	pin16.write_analog(index)
10	sleep(10)
11	for index in range (0, 255):
12	pin16.write_analog(255-index)
13	sleep(10)
14	
15	
	BBC micro:bit 🇰 🔅

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





P M	u 1.1.0.beta.2 - Project 2: Breathing lamp.py — 🗆 🗙
Mode	Image: Save Image: Save
frojec	t 2: Dreathing lamp. py
1	from microbit import *
2	
3	display.show(Image.HAPPY)
4	
5	pin16.write_digital(0)
6	
7	while True:
8	for index in range (0, 255):
9	pin16.write_analog(index)
10	sleep(10)
11	for index in range (0, 255):
12	pin16.write_analog(255-index)
13	sleep(10)
14	
15	
	BBC micro:bit

(4)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

The micro:bit will show a smile expression, and LED smoothly changes its brightness from light to dark and back to light, continuing to do so, which is similar to a lung breathing in and out.





(5)Code Explanation:

from microbit import *	Import the library file of micro: bit		
display.show(Image.HAPPY)	The LED dot matrix on the microbit displays		
	a "smiley face" pattern		
while True:	This is a permanent loop that makes		
	micro:bit execute the code of it.		
for index in range (0, 255):	range() is a function; for index in range(0,		
	255) is to assign 0~255 to index		
pin16.write_analog(index)	Control pin 16 output analog index		
sleep(10)	Delay in 10 ms		

Project 3: 6812 2x2 Full Color RGB



(1)Project Introduction

6812 2X2 full-color RGB module integrates the controlling circuit and the illuminating circuit. Each LED is the same as a 5050 LED lamp bead, and





each component is a pixel point. The inner pixel point includes a amplify driving circuit that latch signal from digital ports shapes, a high-precision internal oscillator and and a 12V high voltage programmable current control portion, which effectively ensures that the color of the pixel point.

The data protocol uses a single-line zero code communication method. After the pixel point is reset, the S-terminal receives the data transmitted from the controller. First, the 24bit data sent by the first pixel is extracted by the first pixel point, and sent to the internal portion of the pixel point. It has the advantages of low-voltage driving, environmental protection, high brightness, large scattering angle, good consistency, ultra-low power, long life expectancy.

Working	DC 3.3-5V	Max Working	200mA	Max Power:	1W
Voltage:		Current:			
Working	-10 °C	Source of	SMD 5050	ІС Туре:	4 pcs/WS2811
Temperature:	~+50°C	light:	RGB		





Gray Scale:	256	Illuminating	180°	Illuminating	Red, yellow,
		Angle:		Color:	blue,green and
					white
GND I USS DOUT 4 S 2 DIN VDD 3 WS2812B-4P		S DOUT S DOUT	VSS DOUT DIN VDD VS2812B-4P GND	GND 1 1 VSS DOUT 4 2 DIN VDD 3 03 100NF WS2812B-4P	$ \begin{array}{c} $

(3)Test Code1

Micro:bit	6812 2x2
Expansion	Full-color RGB
Board	Module
GND	G
5V	V
S (14)	S

Enter Mu software and open the file "Project 3: 6812 2x2 full color RGB-1.py" to import code:

(How to load the project code?)





File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 3: 6812 2x2 full color
file	Tutorial/Python	RGB-1.py
	Code/Expansion Project	
	Code/Project 3: 6812 2x2 Full	
	Color RGB	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)

```
P Mu 1.1.0.beta.2 - Project 3: 6812 2x2 full color RGB-1.py
                                                                                                      \times
P
                                                      Ð
                                                              Q
                                                                     C
                                                                                         ?
                                                                                               +
             t.
                   ÷
                                             -∿-
                          ð
                                      (2000)
(2000)
                                                                             .
                                                                                   Ξ
                                o
Mode
       New
            Load
                  Save
                         Flash
                               Files
                                      REPL
                                           Plotter
                                                    Zoom-in
                                                            Zoom-out
                                                                    Theme
                                                                            Check
                                                                                  Tidy
                                                                                        Help
                                                                                               Quit
Project 3: 6812 2x2 full color RGB-1.py
                                       ×
     from microbit import *
 1
                                                                                                        \Delta 
    import neopixel
    np = neopixel.NeoPixel(pin14, 4)
  a.
  5
    while True:
  6
          for pixel_id1 in range(0, len(np)):
  7
              np[pixel_id1] = (255, 0, 0)
  8
              np.show()
  9
         sleep(1000)
 10
         for pixel_id2 in range(0, len(np)):
 n
              np[pixel_id2] = (255, 165, 0)
 12
              np.show()
 13
         sleep(1000)
 14
          for pixel_id3 in range(0, len(np)):
 15
              np[pixel_id3] = (255, 255, 0)
 16
 17
              np.show()
         sleep(1000)
 18
          for pixel_id4 in range(0, len(np)):
 19
              np[pixel_id4] = (0, 255, 0)
 20
              np.show()
 21
         sleep(1000)
 22
```





23	<pre>for pixel_id5 in range(0, len(np)):</pre>	
24	np[pixel_id5] = (0, 0, 255)	
25	np.show()	
26	sleep(1000)	
27	<pre>for pixel_id6 in range(0, len(np)):</pre>	
28	np[pixel_id6] = (75, 0, 130)	
29	np.show()	
30	sleep(1000)	
31	<pre>for pixel_id7 in range(0, len(np)):</pre>	
32	np[pixel_id7] = (238, 130, 238)	
33	np.show()	
34	sleep(1000)	
35	<pre>for pixel_id8 in range(0, len(np)):</pre>	
36	np[pixel_id8] = (160, 32, 240)	
37	np.show()	
38	sleep(1000)	
39	<pre>for pixel_id9 in range(0, len(np)):</pre>	
40	np[pixel_id9] = (255, 255, 255)	
41	sleep(1000)	
42		\bigtriangledown
	BBC micro:bit	*

Click "Check" to examine error in the code. The underlines and cursors

signal that the program is wrong.

(M	lu 1.1.0.beta.2 - Project 3: 6812 2x2 full color RGB-1.py –	×
Mode	Image: New Load Save Image: Save	
Proje	ot 3: 6812 2x2 full color RGB-1.py 🗙	
1	from microbit import *	\triangle
2	import heopixel	
3		
4	np = neopixel.NeoPixel(pin14, 4)	
5		
6	while frue:	
7	for pixel_idl in range(0, ten(np)):	
8	$np[p1xel_1d1] = (255, 0, 0)$	
9	np.snow()	
10	for pixel id2 in range(A lep(pp)):	
	pn[pixe] id21 = (255 - 165 - 0)	
12	np_prxet_rdz] = (235, 105, 0)	
14	sleep(1000)	
15	for pixel id3 in range(0, len(np)):	
16	np[pixel id3] = (255, 255, 0)	
17	np,show()	
18	sleep(1000)	
19	<pre>for pixel_id4 in range(0, len(np)):</pre>	
20	$np[pixel_id4] = (0, 255, 0)$	
21	np.show()	
22	sleep(1000)	





23	<pre>for pixel_id5 in range(0, len(np)):</pre>		
24	np[pixel_id5] = (0, 0, 255)		
25	np.show()		
26	sleep(1000)		
27	<pre>for pixel_id6 in range(0, len(np)):</pre>		
28	np[pixel_id6] = (75, 0, 130)		
29	np.show()		
30	sleep(1000)		
31	<pre>for pixel_id7 in range(0, len(np)):</pre>		
32	np[pixel_id7] = (238, 130, 238)		
33	np.show()		
34	sleep(1000)		
35	<pre>for pixel_id8 in range(0, len(np)):</pre>		
36	np[pixel_id8] = (160, 32, 240)		
37	np.show()		
38	sleep(1000)		
39	<pre>for pixel_id9 in range(0, len(np)):</pre>		
40	np[pixel_id9] = (255, 255, 255)		
41	sleep(1000)		
42			∇
	BBC	micro:bit	*

If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.

```
P Mu 1.1.0.beta.2 - Project 3: 6812 2x2 full color RGB-1.py
                                                                                             \times
                                                             Q
                                                                                              P
       +
             t.
                  ÷
                          ð
                               o
                                     2003
                                            -----
                                                     Q
                                                                    C
                                                                                        ?
                                                                            .
                                                                                  Ξ
            Load
                        Flash
                               Files
                                                           Zoom-out
                  Save
                                     REPL
                                           Plotter
                                                   Zoom-in
                                                                                 Tidy
                                                                                             Quit
Mode
       New
                                                                   Theme
                                                                           Check
                                                                                       Help
Project 3: 6812 2x2 full color RGB py
                                      ×
    from microbit import *
  1
                                                                                                      \Delta
    import neopixel
  2
  3
  4 np = neopixel.NeoPixel(pin14, 4)
  5
    while True:
  6
         for pixel_id1 in range(0, len(np)):
  7
              np[pixel_id1] = (255, 0, 0)
  8
              np.show()
  9
         sleep(1000)
 10
         for pixel_id2 in range(0, len(np)):
 11
              np[pixel_id2] = (255, 165, 0)
 12
              np.show()
 13
         sleep(1000)
 14
         for pixel_id3 in range(0, len(np)):
 15
              np[pixel_id3] = (255, 255, 0)
 16
              np.show()
 17
         sleep(1000)
 18
         for pixel_id4 in range(0, len(np)):
 19
              np[pixel_id4] = (0, 255, 0)
 20
              np.show()
 21
         sleep(1000)
 22
```







(4)Test Results1:

Upload the test code1 to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch. You will view the 6812 RGB module display red, orange, yellow, green, blue, Indigo, violet, purple and white, in loop way.





(5)Test Code2:

Enter Mu software and open the file "Project 3: 6812 2x2 full color

RGB-2.py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 3: 6812 2x2 full color
file	Tutorial/Python	RGB-2.py
	Code/Expansion Project	
	Code/Project 3: 6812 2x2 Full	
	Color RGB	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)





P 🕅	lu 1.1.0.beta	.2 - Project 3: 681	2 2x2 full co	lor RGB-	-2.py						_		\times
Mode	+ C	ad Save Flas	h Files	REPL	Plotter	Q Zoom-in	Q Zoom-out	C Theme	Check	Tidy	? Help	(U) Quit	
Proje	ct 3: 6812	2x2 full color H	GB-2.ру	×									
1	from Mi	crobit import	*										\triangle
2	import	neopixel											
3	np = neo	pixel.NeoPix	cel(pinl	4, 4)									
4	while Tu	ue:											
5	for	index in rar	nge(0, 4)):									
6		np.clear()											
7		np[index] =	(255, 0	, 0)									
8		np.show()											
9		sleep(100)											
10	for	index1 in ra	nge(0, 4	4):									
11		np.clear()											
12		np[index1] =	(255, .	165, 0	9)								
13		np.show()											
14		sleep(100)											
15	for	index2 in ra	nge(0, 4	4):									
16		np.clear()											
17		np[index2] =	(255, 2	255, 0	3)								
18		np.show()											
19		sleep(100)											
20	for	index3 in ra	nge(0, 4	4):									
21		np.clear()											
22		np[index3] =	(0, 25	5, 0)									
23		np.show()											
24		sleep(100)											





		BBC micro:bit 🗰 🔅	ł
50			$\overline{\nabla}$
49		sleep(100)	_
48		np.show()	
47		np[index8] = (255, 255, 255)	
46		np.clear()	
45	for	index8 in range(0, 4):	
44		sleep(100)	
43		np.show()	
42		np[index7] = (160, 32, 240)	
40	TOP	nn.clear()	
39	for	index7 in range(0, 4):	
38		np.snow()	
37		np[index6] = (238, 130, 238)	
36		np.clear()	
35	for	index6 in range(0, 4):	
34		sleep(100)	
33		np.show()	
32		np[index5] = (75, 0, 130)	
31		np.clear()	
30	for	index5 in range(0, 4):	
29		sleep(100)	
28		np.show()	
27		np[index4] = (0, 0, 255)	
26		np.clear()	
25	for	index4 in range(0, 4):	

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





() N	1u 1.1.0.beta	a.2 - Proje	ect 3: 6	5812 2	x2 full (color RG	B-2.py						_		×
P	+	<u>t) (</u>		ځ	6		(-v-)	(@)	Q	C			?	(එ)	
Mode	New 1	load Sav	re F	lash	Files	REPL	Plotter	Zoom-in	Zoom-out	Theme	Check	Tidy	Help	Quit	
Proje	ect 3: 6812	2x2 ful	l colo	r RGB	-2. ру	×									
1	from Mi	crobit	impo ol	ort *	c										\bigtriangleup
2	np = ne	neopix opixel	eι .NeoΡ	rixel	(pin	14.4)								
4	while T	rue:			- \F	,	/								
5	for	index	in r	ange	e(0,	4):									
6		np.cl	ear() dovi	- (3	EE	0 0)									
7		np.n	ow()	- (2	,	0, 0)									
9		sleep	(100)												
10	for	index	1 in	rang	ge(0,	4):									
11		np.cl	ear() day 11		OFF	1.05	0)								
12		np_in np.sh	ow()	= (255,	165,	0)								
14		sleep	(100)												
15	for	index	2 in	rang	ge(0,	4):									
16		np.cl	ear()												
17		nptin npish	aexzj ow()	=	255,	255,	0)								
19		sleep	(100)												
20	for	index	3 in	rang	ge(0,	4):									
21		np.cl	ear()												
22		np_in sh	dex3] ow()	= ((0, 2	55, 0)								
2.0		sleep	(100)												
					1.5										
25	for	index	4 in ear()	rang	;e(0,	4):									
26		np[in	dex4]	= (0.0	, 255)								
28		np.sh	ow()												
29		sleep	(100)		(0										
30	for	Index	5 in ear()	rang	;e(0,	4):									
31		np[in	dex5]	= (75,	0, 13	0)								
33		np.sh	ow()		. ,	<i>,</i>									
34		sleep	(100)												
35	for	Index	6 111 ear ()	rang	;e(0,	4):									
36		np[in	dex6]	= (238.	130,	238)								
38		np.sh	ow()		,	,	,								
39		sleep	(100)												
40	for	index	7 in	rang	;e(0,	4):									
41		np.ct	dex71	= (160.	32.	240)								
43		np.sh	ow()	`	,,	,	,								
44		sleep	(100)												
45	for	Index	8 in	rang	;e(0,	4):									
46		npict	dex81	= (255.	255.	255)								
48		np.sh	ow()	`	,	,	,								
49		sleep	(100)												
50															
											1	BBC mic	ro:bit		Q





If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.

🕐 м	u 1.1.0.beta	a.2 - Project 3: 6812 2x2 full color RGB-2.py	_		\times
Mode		Image: Save Imag	? Help	Quit	
froje	et 3: 6012	crobit import t			
2	import	neopixel			
3	np = ne	opixel.NeoPixel(pin14, 4)			
4	while T	rue:			
5	for	index in range(0, 4):			
6		np.clear()			
7		np[index] = (255, 0, 0)			
8		np.show()			
9	_	sleep(100)			
10	for	Index1 in range(0, 4):			
11		np.clear()			
12		np[mdex1] = (255, 165, 0)			
14		sleen(100)			
15	for	index2 in range(0, 4);			
16		np.clear()			
17		np[index2] = (255, 255, 0)			
18		np.show()			
19		sleep(100)			
20	for	index3 in range(0, 4):			
21		np.clear()			
22		np[index3] = (0, 255, 0)			
23		np.show()			
24		steep(100)			





	25	for	index4 in range(0, 4):	
l	26		np.clear()	
l	27		np[index4] = (0, 0, 255)	
l	28		np.show()	
l	29		sleep(100)	
l	30	for	index5 in range(0, 4):	
l	31		np.clear()	
l	32		np[index5] = (75, 0, 130)	
l	33		np.show()	
l	34		sleep(100)	
l	35	for	index6 in range(0, 4):	
l	36		np.clear()	
l	37		np[index6] = (238, 130, 238)	
l	38		np.show()	
l	39		sleep(100)	
l	40	for	index7 in range(0, 4):	
l	41		np.clear()	
l	42		np[index7] = (160, 32, 240)	
l	43		np.show()	
l	44		sleep(100)	
l	45	for	index8 in range(0, 4):	
l	46		np.clear()	
l	47		np[index8] = (255, 255, 255)	
l	48		np.show()	
l	49		sleep(100)	
	50			∇
			BBC micro; bit 🗰 🐔	
1				

(6)Test Results2:

Upload the test code 2 to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

You can view four WS2812RGB lights light up, like a flowing light.

(How to download? How to quick download?)

(7)Test Code3:

Enter Mu software and open the file "Project 3: 6812 2x2 full color





RGB-3.py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 3: 6812 2x2 full color
file	Tutorial/Python	RGB-3.py
	Code/Expansion Project	
	Code/Project 3 : 6812 2x2	
	Full Color RGB	

You can also input code in the editing window yourself.

(note:all	words	and	symbols	must	be	written	in	English)
(011101	5,					





(M	1u 1.1.0.beta.2 - Project 3: 6812 2x2 full color RGB-3.py	_		×
Mode	Image: New Load Image: Save Image: Save	? Help	Quit	
Proje	st 3: 6812 2x2 full color RGB-3.py 🛛 🗙			
1	from microbit import *			
2	import neopixel			
3	np = neopixel.NeoPixel(pin14, 4)			
4	from random import randint			
5	$\mathbf{R} = \mathbf{\Theta}$			
6				
7	B = 0			
8	for index in range(0, 4):			
9	R = randint(10, 255)			
10	6 = randint(10, 255)			
12	B = randint(10, 255)			
13	np.clear()			
14	np[index] = (R, G, B)			
15	np.show()			
16	sleep(500)			
17				
	BBC mi	ro:bit		¢

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





P Mu 1.1.0.beta.	2 - Project 3: 6812 2x2	full color RGE	3-3.py					_		×
Mode New Lo	ad Save Flash F 2x2 full color RGB-3	iles REPL	Plotter	Q Zoom-in	Q Zoom-out	Theme Ch	teck	? Help	Quit	
1 from mic 2 import n 3 np = neo 4 from ran 5 R = 0 6 G = 0 7 B = 0 8 while Tr 9 for 1 10 1 12 13 14 15 16 17	<pre>zx2 full color Nob-3 robit import * eopixel pixel.NeoPixel(dom import rand ue: ndex in range(0 R = randint(10, G = randint(10, B = randint(10, np.clear() np[index] = (R, np.show() sleep(500)</pre>	pin14, 4) int , 4): 255) 255) 255) 6, B)								
							BBC micr	o:bit		¢

If the code is correct, connect micro:bit to computer and click "Flash" todownloadcodetomicro:bitboard.



(8)Test Results3:

Upload the test code 3 to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

Then you will see 4 WS2812RGB lights light up with random colors, like a flowing light.

(How to download? How to quick download?)

(9)Code Explanation:

from microbit import *	Import the library file of micro: bit
import neopixel	Import the library file of neopixel





np = neopixel.NeoPixel(pin14, 4)	Set Neopixel as pin P14 to initialize				
	the light with 4 LEDs				
np.clear()	RGB on Neopixel are all off				
while True:	This is a permanent loop that makes				
	micro:bit execute the code of it				
for pixel_id1 in range(0, len(np)):	Set pixel of RGB in (0, len (np)) to				
	pixel_id1				
for index in range(0, 4):	Set pixel of RGB in (0,4) to index				
np.show()	Display current pixel on Neopixel				
np[pixel_id1] = (255, 0, 0)					
np[pixel_id2] = (255, 165, 0)	Set pixel_id1 to display red color				
np[pixel_id3] = (255, 255, 0)	Set pixel_id2 to display orange color				
np[pixel_id4] = (0, 255, 0)	Set pixel_id3 to display yellow color				
np[pixel_id5] = (0, 0, 255)	Set pixel_id4 to display green color				
np[pixel_id6] = (75, 0, 130)	Set pixel_id5 to display blue color				
np[pixel_id7] = (238, 130, 238)	Set pixel_id6 to display indigo color				
np[pixel_id8] = (160, 32, 240)	Set pixel_id7 to display violet color				
np[pixel_id9] = (255, 255, 255)	Set pixel_id8 to display purple color				
	Set pixel_id9 to display white color				
from random import randint	Import randint from random variables				
np[pixel_id] = (R, G, B)	Set pixel_id to display rainbow color				
R = 0	Set the initial value of R to 0				





G = 0	Set the initial value of G to 0
B = 0	Set the initial value of B to 0
R = randint(10, 255)	Set R=randint(10, 255)
G = randint(10, 255)	Set G=randint(10, 255)
B = randint(10, 255)	Set B=randint(10, 255)

Project 4: PIR Motion Sensor



(1)Project Introduction

The Pyroelectric infrared motion sensor can detect infrared signals from moving objects, and output switching signals. Applied to a variety of occasions, it can detect movement of human body.

Conventional pyroelectric infrared sensors are much more bigger, with complex circuit and lower reliability. Yet, this new pyroelectric infrared motion sensor, is more practical. It integrates a digital pyroelectric infrared sensor and connecting pins. It features higher sensibility and reliability,





lower power consumption, light weight, small size, lower voltage working mode and simpler peripheral circuit.

(2) About PIR Motion Sensor:

Working	DC	XC6206P332MR (662K) SOT-23
Voltage:	4.5-6.5V	
Max	50MA	GND GND 3V3
Working		TED-RED SV SV SV
Current:		$\begin{array}{cccccccccccccccccccccccccccccccccccc$
Static	<50uA	Q1 MOS C1 100NF Q1 MOS GND GND GND
Current:		GND GND
Control	Digital	
Port:	output	
	(high level	
	is 3.3V, low	
	level is	
	0V)	Keyestudio
Control	Digital	
Signals:	signal 1/0	





Working	-10 ~ 50 ℃
Tempera	
ture:	
Max	4m
detectio	
n	
distance	
Sensing	< 100°
Angle:	
Trigger	L doesn' t
Way:	repeatedly
	trigger/H
	trigger
	repeatedly
1	

Note:

1. The maximum distance is 4 meters during testing.

2. In the test, open the white lens to check rectangular sensing part. When the long line of the sensing part is parallel to the ground, the distance is the best.

3. In the test, covering the sensor with white lens can sense the distance





precisely.

4. The distance is best at 25°C, and the detection distance value will reduce when temperature exceeds 30°C.

- 5. After powering up and uploading the code, you can start testing after
- 5-10 seconds, otherwise the sensor is not sensitive.

(3)Test Code:

Micro:bit	DID Mation Concor			
Expansion Board	PIK MOTION Sensor			
GND	G			
5V	V			
S (15)	S			

Enter Mu software and open the file "Project 4: PIR motion sensor.py" to import code:

(How to load the project code?)

File		Route	File Name
Туре			
Python	KS4027	folder/Python	Project 4:





file	Tutorial/Python	PIR motion sensor.py
	Code/Expansion Project	
	Code/Project 4: PIR Motion	
	Sensor	

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





P Mu	u 1.1.0.beta.2 - Project 4: PIR motion sensor.py	_		\times		
Mode Projec	+ New Load Save Flash Files REPL Plotter Coom-in Zoom-out Theme Check Tidy t 4: FIR motion sensor.py	? Help	Quit			
1	<pre>from microbit import *</pre>					
2						
3	val = 0					
4						
5	display.show(Image.HAPPY)					
6						
7	while True:					
8	<pre>val = pin15.read_digital()</pre>					
9						
10	<pre>print("digital signals:", val)</pre>					
11						
12	sleep(100)					
13						
14						
	BBC mic	ro:bit		¢		

If the code is correct, connect micro:bit to computer and click "Flash" todownloadcodetomicro:bitboard.





P M	110 heta 2 - Project 4: PIP motion cencor ny
Mode	Image: Save Image: Save
Projec	t 4: FIR motion sensor.py
1	from microbit import
2	
з	val = 0
4	
5	display.show(Image.HAPPY)
6	
7	while True:
8	val = pin15.read_digital()
9	
10	print("digital signals:", val)
11	
12	sleep(100)
13	
14	
	BBC micro:bit 🗰 🔅

(4)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

Click "**REPL**" and then press the reset button on the back of the board.

If PIR motion sensor detects someone nearby, the serial monitor will display "1", and the indicator on the module will be off. If nobody is around, the serial monitor will show "0", the indicator will be on. As shown below:





(M	u 1.1.0.beta.2 - Project 4: PIR motion sensor.py —		\times						
Mode	Image: New Load Image: Save Image: Save	Quit							
Project 4: PIR motion sensor.py 🗶									
1	from microbit import *		\bigtriangleup						
2									
3	val = 0								
4	display show(Image HARDY)								
5	(TSptay, Show(Image, HAPPT)								
	while True:								
	val = pin15.read digital()								
9									
10	print("digital signals:", val)								
- 11									
12	sleep(100)		\bigtriangledown						
BBC	micro-bit REPI								
digit	al signals: 0		•						
digit	al signals: 0								
digit	al signals: 0								
digit	al signals: 0								
digit	at signats. e								
digit	al signals: 1								
digit	al signals: 1								
digit	al signals: 1								
digit	at signats: 0 al signals: 1								
digit	al signals: 1								
			\bigtriangledown						
	BBC micro:bit		¢.						

(5)Code Explanation:

from microbit import *	Import the library file of micro: bit	
display.show (Image.HAPPY)	The LED dot matrix on the microbit	
	displays a "smiley face" pattern	
val = 0	Set the initial value of the variable	
	val to 0	
while True:	This is a permanent loop that	
	makes micro:bit execute the code	





	of it.
val = pin15.read_digital()	Assign the digital signal read by
	the PIR sensor connected to pin 15
	to the variable val
<pre>print("digital signals:", val)</pre>	BBC microbit REPL window prints
	the digital signal read by the PIR
	sensor
sleep(100)	Delay in 100 ms

Project 5: Induction Lamp

(1)Project Introduction

In the previous project experiment, we have mastered the working principle of the PIR motion sensor and its control method. In this project, we combine it with a yellow LED to control LED' s brightness.

(2)Test Code:

Micro:bit	PIR Motion	Micro:bit	Yellow LED
Expansion Board	Sensor	Expansion Board	Module
GND	G	GND	G





5V	V	5V V	V
S (15)	S	S (16) S	S

Enter Mu software and open the file "PProject 5: Induction Lamp.py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 5:
file	Tutorial/Python	Induction Lamp.py
	Code/Expansion Project	
	Code/Project 5: Induction	
	Lamp	

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)





n 🕐	u 1.1.0.beta.2 - Project 5: sensor light.py —		×
Mode	Image: New Load Image: Save Image: Save </th <th>Quit</th> <th></th>	Quit	
Projec	t 5: sensor light.py 🗶		
1	from microbit import *		
2			
3	display.show(Image.HAPPY)		
4			
5	pin16.write_digital(0)		
6			
7	while True:		
8	<pre>if pin15.read_digital() == 1:</pre>		
9			
10	pin16.write_digital(1)		
11			
12	else:		
13	pini6.write_digital(0)		
14			
	BBC micro:bit		\$

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.




() Ми	u 1.1.0.beta.2 - Project 5: sensor light.py	_		\times
Mode	+ + <th>? Help</th> <th>Quit</th> <th></th>	? Help	Quit	
Projec	t 5: sensor light.py 🗶			
1	from microbit import *			
2				
3	display.show(Image.HAPPY)			
4				
5	pin16.write_digital(0)			
6				
7	while True:			
8	<pre>if pin15.read_digital() == 1:</pre>			
9				
10	pin16.write_digital(1)			
11				
12	else:			
13	pin16.write_digital(0)			
14				
	BBC micr	o:bit	#	ð-

If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit boa	ard.
--------------------------------	------





🕐 Ми	lu 1.1.0.beta.2 - Project 5: sensor light.py — [×
Mode	Image: New Load Save Image: Save	Quit	
frojec	ot 5: sensor light.py		
1	trom microbit import *		
2			
3	display.show(Image.HAPPY)		
4			
5	pin16.write_digital(0)		
6			
7	while True:		
8	<pre>if pin15.read_digital() == 1:</pre>		
9			
10	pin16.write_digital(1)		
11			
12	else:		
13	pin16.write_digital(0)		
14			
	BBC micro:bit		ŀ

(3)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch. The micro:bit will show a smile image.

When the PIR motion sensor detects people, the yellow LED will be on; otherwise, the LED will be off.

(How to download? How to quick download?)

(4)Code Explanation:

from microbit import *	Import the library file of micro: bit
Pin16.write_digital(0)	Set pin 16 to low level to turn off





	the LED	
while True:	This is a permanent loop that	
	makes micro:bit execute the code	
	of it.	
If pin15.read_digital() == 1:	If the PIR motion sensor connected	
Pin16.write_digital(1)	to pin 15 detects the movement of	
else:	nearby people:	
Pin16.write_digital(0)	Pin 16 is set to high level to light up	
	the LED	
	Otherwise,pin 16 is set to low level	
	to turn off the LED	

Project 6: Servo







(1) Project Introduction

The servo, window and door of this smart home have been fixed together so the servo can be used to drive the window and door to open or close, which is quite smart. In this project we will focus on the servo.

Servo is a position control rotary actuator. It mainly consists of a housing, a circuit board, a core-less motor, a gear and a position sensor. Unlike motor which is often applied to control rotating speed and direction servo is used to control angle. Generally, the angle range of servo rotation is 0° ~180°.

It has 3 wires which are marked in brown, red, and orange respectively. For different brands, its application may have slight difference. So it is recommended to refer to some documents before use. The servo we use is a very common one, wires in brown, red, and orange corresponding to "power negative, power positive, control signal" respectively.



(2)Working Principle of Servo:

The rotation angle of servo motor is controlled by regulating the duty cycle of PWM (Pulse-Width Modulation) signal. The standard cycle of PWM





signal is 20ms (50Hz). Theoretically, the width is distributed between 1ms-2ms, but in fact, it's between 0.5ms-2.5ms. The width corresponds the rotation angle from 0° to 180°. But note that for different brand motors, the same signal may have different rotation angles.



After measurement, it is found that the pulse range of the steering gear is 0.65ms~2.5ms. See more details in the table:

high level	Servo angle	Reference signal cycle time		
time		(20ms)		
0.65ms	0°	0.65ms high level+19.35mslow		
		level		
1.5ms	90°	1.5ms high level+18.5mslow		





		level
2.5ms	180°	2.5ms high level+17.5mslow
		level

(3)About the Servo:

Working	DC 4.8V ~	Operational	About 180 ° (500 →	
voltage:	6V	Angle:	2500µsec)	
Pulse width	500 → 2500	Size:	22.9*12.2*30mm	
range:	µsec			
No-load	0.12±0.01 sec/60° (DC 4.8V) 0.1±0.01 sec/60°			
speed:	(DC 6V)			
No-load	200±20mA (DC 4.8V) 220±20mA (DC 6V)			
current:				
Stop torque:	1.3±0.01kg·cm(DC 4.8V) 1.5±0.1kg·cm(DC 6V)			
Stop current:	\leq 850mA (DC 4.8V) \leq 1000mA (DC 6V)			
Standby	3±1mA (DC 4.8V) 4±1mA (DC 6V)			
Current:				
Weight:	9±1g (without servo horn)			
Working	-30°C~60°C			
temperature:				

Note: Supplying power via USB cable or computer may burn the servo;





thus, we recommend using batteries.

(4)Test Code:

Micro:bit	
Expansion	Servo
Board	
GND	Brown Wire
5V	Red Wire
C (9)	Orange
5 (0)	Wire

Enter Mu software and open the file "Project 6: Servo .py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 6: Servo .py
file	Tutorial/Python	
	Code/Expansion Project	
	Code/Project 6: Servo	

You can also input code in the editing window yourself.(note:all English





words and symbols must be written in English)

Image: Mu 1.1.0.beta.2 - Project 6: adjust the angle of a servo.py — …
Mode New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Tidy Help Quit
Project 6: adjust the angle of a servo.py 🗙
from microbit import *
2 class Servo:
<pre>def init (self, pin, freq=50, min us=600, max us=2400, angle=180);</pre>
<pre>self.min_us = min_us</pre>
6 self.max_us = max_us
7 self.us = 0
s self.freq = freq
sect: angle = angle
self.pin = pin
<pre>12 analog_period = round((1/self.freq) * 1000) # hertz to miliseconds</pre>
<pre>self.pin.set_analog_period(analog_period)</pre>
<pre>is def write_us(self, us):</pre>
duty = round(us * 1024 * self.freq // 1000000)
<pre>self.pin.write_analog(duty)</pre>
19 sleep(100)
20 self.pin.write_analog(0)
21
<pre>22 def write_angle(self, degrees=None):</pre>
23 if degrees is None:
degrees = math.degrees(radians)
total range = self.max us - self.min us
us = self.min_us + total_range * degrees // self.angle
28 self.write_us(us)
29
30 Servo(pin8).write_angle(0)
a urspray.snow(image.HAPPi)
33 while True:
34 Servo(pin8).write_angle(0)
35 sleep(1000)
36 Servo(pin8).write_angle(45)
37 Steep(1000) Serve(pip8) write apgle(00)
sleep(1000)
Servo(pin8).write_angle(135)
4 sleep(1000)
42 Servo(pin8).write_angle(180)
43 sleep(1000) ▼
BBC micro:bit 🗰 😨

Click "Check" to examine error in the code. The underlines and cursors





signal that the program is wrong.

🕐 м	lu 1.1.0.beta	2 - Project 6: adjust the angle of a servo.py	—		×
Mode	+ C	L & C C C C C C C C C C C C C C C C C C	Tidy Help	Quit	
Proje	ct 6: adju	st the angle of a servo.py 🗶			
1	from mi	crobit import *			
2			`		
3	class Se	ervo: ipit_ (1f pipfrom_E0min_uc=000mov_uc=0400op	al 190) -		
4	det	<pre>Init(setf, pin, ireq=50, min_us=600, max_us=2400, ang celf min_us = min_us</pre>	gte=180):		
6		self.max us = max us			
7		self.us = 0			
8		self.freq = freq			
9		self.angle = angle			
10		<pre>self.analog_period = 0</pre>			
11		self.pin = pin			
12		<pre>analog_period = round((1/self.freq) * 1000) # hertz to</pre>	miliseconds	5	
13		<pre>self.pin.set_analog_period(analog_period)</pre>			
14	4.6	write up(a-16 up).			
15	der	write_us(setf, us):			
16		duty = round(us * 1024 * self.fred // 1000000)			
18		self.pin.write analog(duty)			
19		sleep(100)			
20		<pre>self.pin.write_analog(0)</pre>			
21					
22	def	write_angle(self, degrees=None):			
23		if degrees is None:			
24		degrees = math.degrees(radians)			
25		degrees = degrees % 360			
26		us - celf min us + total range + degrees // celf angle			
27		<pre>self.write us(us)</pre>			
29		eerini ree_us(us)			
30	Servo(p	in8).write_angle(0)			
31	display	show(Image.HAPPY)			
32					
33	while Tu	ue:			
34		Servo(pin8).write_angle(0)			
35		sleep(1000)			
36		servo(pin8).write_angle(45)			
37		Steep(1000) Serve(pip8) write angle(90)			
38		sleen(1000)			
40		Servo(pin8).write angle(135)			
41		sleep(1000)			
42		Servo(pin8).write_angle(180)			
43		sleep(1000)			\bigtriangledown
				-	1
			DU MICRO:DIT		1 4





If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.

```
P Mu 1.1.0.beta.2 - Project 6: adjust the angle of a servo.py
                                                                                          \times
P
       +
             t
                  ÷
                         ð
                              ō
                                    23
                                           -∿-
                                                   Q
                                                           Q
                                                                  C
                                                                                     ?
                                                                                           Ξ
Node
       New
            Load
                 Save
                       Flash
                             Files
                                    REPL.
                                         Plotter
                                                                 Theme
                                                                        Check
                                                                              Tidy
                                                                                    Help
                                                                                          Ouit
                                                  Zoom-in
                                                         Zoom-out
Project 6: adjust the angle of 🍢
                                           ×
                               servo. pv
    from microbit import *
  1
                                                                                                   \Delta
  2
    class Servo:
  3
         def __init__(self, pin, freq=50, min_us=600, max_us=2400, angle=180):
  4
             self.min_us = min_us
             self.max_us = max_us
  6
             self.us = 0
  7
             self.freq = freq
             self.angle = angle
  9
             self.analog_period = 0
 10
             self.pin = pin
             analog_period = round((1/self.freq) * 1000) # hertz to miliseconds
             self.pin.set_analog_period(analog_period)
 14
         def write_us(self, us):
             us = min(self.max_us, max(self.min_us, us))
 16
             duty = round(us * 1024 * self.freq // 1000000)
             self.pin.write_analog(duty)
 18
             sleep(100)
 19
             self.pin.write_analog(0)
 20
 21
         def write_angle(self, degrees=None):
 22
             if degrees is None:
 23
                  degrees = math.degrees(radians)
 24
             degrees = degrees % 360
 25
             total_range = self.max_us - self.min_us
 26
             us = self.mín_us + total_range * degrees // self.angle
             self.write us(us)
 28
 29
    Servo(pin8).write_angle(0)
 30
    display.show(Image.HAPPY)
 31
    while True:
 33
             Servo(pin8).write_angle(0)
 34
             sleep(1000)
 35
             Servo(pin8).write_angle(45)
 36
             sleep(1000)
 37
             Servo(pin8).write_angle(90)
 38
             sleep(1000)
 39
             Servo(pin8).write_angle(135)
 40
             sleep(1000)
 41
             Servo(pin8).write_angle(180)
 42
             sleep(1000)
                                                                                                  \nabla
 43
                                                                            BBC micro:bit
```





(5)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch. The micro:bit will show smile expression, the servo will rotate 0°~45°~90°~135°~180°~0°, in loop way. (How to download? How to quick download?)

Project 7: 130 Motor



(1)Project Introduction

130 motor adopts the HR1124S chip which is applied to single-channel H-bridge drive chip in direct current motor.

H-bridge driving part uses the PMOS and NMOS power tubes of low on-resistance. In addition, the HR1124S chip has the low standby and static current.

This motor is compatible with all kinds of MCU control boards. It comes with 2.54mm anti-reverse white connectors. In the experiment, you can take advantage of the voltage direction of IN+和 IN- to control the rotation





of motor and alter its speed via PWM signals.

(2)Parameters:

Working	3.3-5V(DC)	Max Current:	200mA (DC5V)		
Voltage:					
Max Power:	1W	Control port:	Dual digital		
			port (digital		
			input)		
Working	-10°C ~+50°C	Environmental	ROHS		
Temperature:		Attribute:			
$\begin{array}{c c} & & & C5 \\ \hline & & & & \\ \hline \\ \hline$					

(3)Test Code 1: (high/low level control)

Micro:bit Expansion Board	Motor
GND	G
5V	V





S (13)	IN+
S (12)	IN-

Enter Mu software and open the file "Project 7: 130 Motor-1.py" to import code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 7: 130
file	Tutorial/Python	Motor-1.py
	Code/Expansion Project	
	Code/Project 7: 130 Motor	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)





СМ	u 1.1.0.beta.2 - Project 7: Small fan rotation-1.py —		\times
(P) Node	+ +	Quit	
Proje	et 7: Small fan rotation-1.py 🗙		
1	from microbit import *		\triangle
2			
3	pin12.write_digital(0)		
4	pinl3.write_digital(0)		
5			
6	while True:		
7	pin12.write_digital(1)		
8	pinl3.write_digital(0)		
9	sleep(5000)		
10	pin12.write_digital(0)		
11	pinl3.write_digital(0)		
12	steep(1000)		
13	pin12.write_digital(0)		
14	pini3.write_digital(1)		
15	steep(5000)		
16	pini2.write_digital(1)		
17	pinis.write_digital(1)		
18	Steep(1000)		
19			\bigtriangledown
-11	BBC micro:bit	₩ 4	X

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





С м	u 1.1.0.beta.2 - Project 7: Small fan rotation-1.py			\times
Mode	Image: Save Imag	y Help	Quit	
Proje	ct 7: Small fan rotation-1.py 🗙			
1	from microbit import *			\bigtriangleup
2				
3	pin12.write_digital(0)			
4	pin13.write_digital(0)			
5				
6	while True:			
7	pinl2.write_digital(1)			
8	pinl3.write_digital(0)			
9	sleep(5000)			
10	pin12.write_digital(0)			
11	pini3.write_digital(0)			
12	steep(1000)			
13	pini2.write_digital(0)			
14	pinis.write_digital(1)			
15	steep(5000)			
16	piniz.write_digital(1)			
17	place (1000)			
18	steep(1000)			
19				\bigtriangledown
	BBC	nicro:bit		¢.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





0			_	
🕐 М	u 1.1.0.beta.2 - Project 7: Small fan rotation-1.py —			×
Mode	+ + <th></th> <th>Quit</th> <th></th>		Quit	
Proje	ct 7: Small fan rotation-1.py			
1	from microbit import * 🔪			
2				
3	pin12.write_digital(0)			
4	pin13.write_digital(0)			
5				
6	while True:			
7	pin12.write_digital(1)			
8	pinl3.write_digital(0)			
9	sleep(5000)			
10	pin12.write_digital(0)			
11	pin13.write_digital(0)			
12	sleep(1000)			
13	pin12.write_digital(0)			
14	pinl3.write_digital(1)			
15	sleep(5000)			
16	pin12.write_digital(1)			
17	pinl3.write_digital(1)			
18	sleep(1000)			
19				∇
20.	BBC micro:	oit		¢

(4)Test Code2: (PWM control)

Enter Mu software and open the file "Project 7: 130 Motor-2.py" to import

code:

(How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 7: 130 Motor-2.py
file	Tutorial/Python	
	Code/Expansion Project	
	Code/Project 7 : 130	
	Motor	





You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)

🕐 м	u 1.1.0.beta.2 - Project 7: Small fan rotation-2.py -	_		\times
Mode	Image: New Load Image: Save Image: Save	?	Quit	
Proje	ct 7: Small fan rotation-2.py 🛛 🗶			
1	from microbit import *			
2				
3	pin12.write_digital(0)			
4	pin13.write_digital(0)			
5				
6	while True:			
7	pinl2.write_digital(1)			
8	pin13.write_analog(600)			
9	sleep(5000)			
10	pinl2.write_digital(0)			
11	pin13.write_analog(0)			
12	sleep(1000)			
13	pinl2.write_digital(0)			
14	pinl3.write_analog(400)			
15	sleep(5000)			
16	pinl2.write_digital(1)			
17	pinl3.write_analog(1023)			
18	sleep(1000)			
19				
	BBC micro	:bit		¢

Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





🕐 м	u 1.1.0.beta.2 - Project 7: Small fan rotation-2.py	_		×
Mode Proje	Image: Save Image: Save	? Help	Quit	
1	<pre>from microbit import *</pre>			
2				
3	pinl2.write_digital(0)			
4	pin13.write_digital(0)			
5				
6	while True:			
7	pinl2.write_digital(1)			
8	pinl3.write_analog(600)			
9	sleep(5000)			
10	pinl2.write_digital(0)			
11	pin13.write_analog(0)			
12	sleep(1000)			
13	pinl2.write_digital(0)			
14	pinl3.write_analog(400)			
15	sleep(5000)			
16	pinl2.write_digital(1)			
17	pinl3.write_analog(1023)			
18	sleep(1000)			
19				
	BBC mi	cro:bit		¢.

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





С м	u 1.1.0.beta.2 - Project 7: Small fan rotation-2.py	_		×
Mode Proje	+ New Load Save Elash Files REPL Plotter Zoom-in Zoom-out Theme Check	Tidy Help	Quit	
IIOje	et 1. Small fail fotation 2. py			
1	from microbit import *			
2				
3	pini2.write_digital(0)			
4	pinis.write_digital(0)			
5	and the Tourse			
6	while inue:			
7	pini2.write_digitat(1)			
8	plnis, write_analog(600)			
9	steep(5000)			
10	piniz.write_digitat(0)			
11	place (1000)			
12	steep(1000)			
13	piniz, write_urgitat(0)			
14	sloop(E000)			
15	pip12 write digital(1)			
16	pini2 write_onglog(1023)			
17	sleen(1000)			
18	steep(1000)			
19				
	BI	BC micro:bit		¢

(5)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch. The fan will rotate clockwise for 5s, stop 1, rotate anticlockwise for 5s and stop for 1s, in loop way. (<u>How to download?</u> <u>How to quick download?</u>)

Project 8: Lithium Battery Power Module

(1)Project Introduction





This module integrates a charging and discharging chip, which can be interfaced with an external rechargeable battery through the PH2.0MM interface. In the experiment, we use a single lithium battery.

It has a Micro USB port and a charging port for solar panels, which can supply power for an external lithium battery.

In addition, this module has a boost module which can increase the voltage of batteries to 6.6V. The DIP switch on the module is the OUTPUT switch of 6.6V. The pin G and V can output 6.6V and the pin S can read the battery voltage after the resistance 1/2 voltage.

(2) Parameters:

Charging Port	Micro USB, HP2.0MM port for solar
	panels
Input Voltage of ports of the	4.4-6V
solar panel	
constant-voltage charging	4.15-4.24V
Max Charging Current	800mA
Output Port	3 P 2.54mm Pins
Input Voltage	6.6V
Max Output Current	800mA
Batteries	Single-cell Lithium Battery





(3)Schematic Diagram:



(4)Features:







OLAR4.8-6.0V, the input port of power, is connected to polar panels. The solar energy is converted into electric energy via solar panels.



BAT, the output port of power, is interfaced with the lithium battery holder(rechargeable batteries) and saves the electric energy into batteries.







This is the switch. Slid to ON end, then the external lithium battery will be connected, supplying to the expansion board; on the contrary, slide to OFF, then the current of lithium battery will be disconnected.



You can charge the lithium battery via USB cable.

Test the solar battery panel:

We can connect the solar battery panel and an LED we provide together, as shown below.

Disconnect the power, after a while, you will see the LED light up.







Project 9: 1602 LCD



(1)Project Introduction

With I2C communication module, this is a display module that can show 2 lines with 16 characters per line.

It shows blue background and white word and connects to I2C interface of MCU, which highly save the MCU resources.

On the back of LCD display, there is a blue potentiometer for adjusting the backlight. The communication address defaults to 0x27.

The original 1602 LCD can start and run with 7 IO ports, but ours is built with Arduino IIC/I2C interface, saving 5 IO ports. Alternatively, the module comes with 4 positioning holes with a diameter of 3mm, which is convenient for you to fix on other devices.

Notice that when the screen gets brighter or darker, the characters will become more visible or less visible.







(3) About 1602 I2C:

Working	DC5V	I2C Address:	0x27	Control	12C
Voltage :				Port:	
Working	<	Working	0°C ~ 45°C	Driving	PCF8574T
Current:	130mA	Temperature:	(recommend)	Chip:	
GND: a	pin			SDA :	A pin that
connected	to the	VCC: A pin th	connects to analog		
ground		+5V power sup	port A4 for IIC		
			communication		
SCL: a	pin	Backlight		Adjustak	ole
interfaced v	vith SCL		contrast		
or A5, use	d for IIC				
communica	tion				

(3)Test Code:





Micro:bit				
Expansion Board	12C 1602 LCD MIOdule			
GND	GND			
5V	5V			
SDA	SDA			
SCL	SCL			

Enter Mu software and open the file "Project 9:1602 LCD.py" to import code:

(How to load the project code?)

File	Route	File Name		
Туре				
Python	KS4027 folder/Python	Project 9: 1602		
file	Tutorial/Python	LCD.py		
	Code/Expansion Project			
	Code/Project 9: 1602 LCD			

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)





() M	lu 1.1.0.bet	a.2 - Project 9: 1602	LCD.py							_		×
	+	1 (±)			0	Q				?	(b)	
Mode	New 1	Load Save Flash	Files REPL	Plotter 2	loom-in	Zoom-out	Theme	Check	Tidy	Help	Quit	
Proje	ct 9: 1602	2 LCD. py 🗶										
1	Trom M1	crobit import	*									
3	LCD_I2C	_ADDR=0x27										
4	class	CD1602():										
6	def	init(sel	f):									
7		<pre>self.buf = by self PK = 0x4</pre>	ytearray(1)									
8		self.RS = 0x0	98 90									
10		<pre>self.E = 0x04</pre>	4									
11		<pre>self.setcmd() sloop(E)</pre>	9x33)									
12		steep(s) self.send(0x)	30)									
14		sleep(5)										
15		<pre>self.send(0x) sleen(5)</pre>	20)									
16		self.setcmd(9x28)									
18		<pre>self.setcmd()</pre>	9x0C)									
19		<pre>self.setcmd() self.setcmd()</pre>	9x06) 9x01)									
20		self.version	='1.0'									
22			1-1.									
23	def	<pre>setReg(self, self.buf[0] :</pre>	dat): = dat									
25		i2c.write(LC	D_I2C_ADDR,	self.buf)							
26		sleep(1)										
27												
28	def	<pre>send(self, da d=dat&0xE0</pre>	at):									
30		d = self .BK										
31		d = self. RS										
32		self.setReg(d	1) 1[0x04]									
34		<pre>self.setReg(d)</pre>	i)									
35	ما م ا	cotomd/aalf	and) i									
36	der	self.RS=0	chid).									
38		self.send(cmd	d)									
39		self.send(cmd	1<<4)									
40	def	setdat(self ,	dat):									
42		<pre>self.RS=1</pre>										
43		<pre>self.send(dat self.send(dat</pre>	t) E<<4)									
45	I	See insena (da										
46	def	<pre>clear(self):</pre>										
47	l	self.setcmd(.	L)									
49	def	backlight(sel	lf, on):									
50		if on:										
51		else:	9X08									
53	Í	self.BK=0	9									







Click "Check" to examine error in the code. The underlines and cursors signal that the program is wrong.





M M M M M M M M M	lu 1.1.0.be	ta.2 - Project	9: 1602 LC	CD.py								_		\times
P	+	±	3	6		~	0	Q				?	(b)	
Mode	New	Load Save	Flash	Files R	EPL I	Plotter	Zoom-in	Zoom-out	Theme	Check	Tidy	Help	Quit	
Proje	ct 9: 160	02 LCD. py	×											
1	from M	icrobit i	mport *											
3	LCD_I2	C_ADDR= <mark>⊙</mark> x	27											
4	c1	LCD1602()												
6	de	finit_	_(self)	:										
7		self.bu	f = byt	earray	(1)									
8		self.BK self.RS	$= 0 \times 08$ $= 0 \times 00$											
10		self.E	= 0x04											
11		self.se	tcmd(⊖x ∖	33)										
12		self.se	/ nd(0x30)										
14		sleep(5)											
15		self.se sleep(5	nd(0x20))										
17		self.se	, tcmd(⊖x	28)										
18		self.se	tcmd(0x	0C) 0C)										
20		self.se	tcmd(0x	01)										
21		self.ve	rsion='	1.0'										
22	de	f setReg(self. d	at):										
24		self.bu	f[0] =	dat										
25		i2c.wri	te(LCD_	I2C_AD)R, ≲	self.bu	uf)							
26	1	steep(1)											
28	de	f send(se	elf, da	t):										
29		d=dat&	0×F0											
30		d =sel: d =sel:	F.BK											
31		self.se	etReg(d)										
33		self.Se	etReg(d	0x04)										
34		self.Se	etReg(d)										
36	de	f setcmd	(self,	cmd):										8
37		self.RS	5=0											
38		self.S	end(cmd end(cmd) << 4)										
40	1	50000	ena (ena											
41	de	f setdat	(self,	dat):										
42		self.R: self.S	s=1 end(dat)										
44		self.se	end(dat	/ <<4)										
45		6												
46	de	self.se	self): etcmd(1)										
48	1			, ,										
49	de	f backlig	ght(sel	f, on)										
50		nf on:	lf.BK=0	x08										
52		else:												
53		se	lf.BK=0											





<pre>self.setcmd(0) def on(self): self.setcmd(0x0C) def off(self): self.setcmd(0x08) def sht(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0; self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0; self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0; self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0; self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0; self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if s=0xC0 self.setcmd(0x1C) def char(self, s, x=0, y=0): if s=0xC0 self.setcmd(a) self.setcmd(a) self.setcmd(a) self.setcmd(a) self.setcmd(a) self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): i self.char(ord(s[1])) drisplay.show(Image.HAPPY) l = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) v KBC microbit</pre>						
<pre>def on(self): self.setcmd(0x0C) def off(self): self.setcmd(0x08) def shl(self): self.setcmd(0x18) def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0:</pre>	54		<pre>self.setcmd(0)</pre>			
<pre>def on(self):</pre>	55					
<pre>def off(self): self.setcmd(0x08) def shl(self): self.setcmd(0x18) def shr(self): self.setcmd(0x10) def char(self, ch, x=-1, y=0):</pre>	56	def	on(self):			
<pre>def off(self): self.setcmd(0x08) def shl(self): self.setcmd(0x18) def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0xC0 a=0xC0 a=+x self.setcmd(a) self.setcmd(a) self.setdat(ch)</pre>	57		self.setCMd(0x0C)			
<pre>def of (SetTy: self.setcmd(0x08) def shl(self): self.setcmd(0x18) def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0:</pre>	58	1.6	off(colf);			
<pre>def sht(setf): setf.setcmd(0x00) def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=bx00 a=tx self.setcmd(a) self.setcmd(a) self.setdat(ch)</pre>	59	def	off(setr):			
def shl(self): self.setcmd(θx18) def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=0x80 if y>0: a=0xC0 a+=x self.setcmd(a) self.setdat(ch) def puts(self, s, x=0, y=0): if len(s)>0: self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): i self.char(ord(s[1])) dfsplay.show(Image.HAPPY) l = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000)	60	1	Sect. Second (0x00)			
<pre>self.setcmd(0x18) def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=0x00 a+=x self.setcmd(a) self.setcmd(a) self.setdat(ch) def puts(self, s, x=0, y=0): if len(s)>0: self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): self.char(ord(s[1])) display.show(Image.HAPPY) t = LCDI602() t.puts("Hello microbit!") n = 0 while True: t.puts(str(n), 0, 1) n = n + 1 sleep(1000) </pre>	61	def	<pre>shl(self):</pre>			
<pre>def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0:</pre>	63		self.setcmd(0x18)			
<pre>def shr(self): self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=0xC0 a+=x self.setcmd(a) self.setdat(ch)</pre>	64	i				
<pre>self.setcmd(0x1C) def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=0xC0 a+=x self.setcmd(a) self.setcdat(ch) def puts(self, s, x=0, y=0): if len(s)>0: self.char(ord(s[0]),x,y) for f in range(1, len(s)): self.char(ord(s[1])) display.show(Image.HAPPY) l = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) BBC micro:bit</pre>	65	def	shr(self):			
<pre>def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0:</pre>	66		<pre>self.setcmd(0x1C)</pre>			
<pre>def char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=0x80 if y>0: a=exc0 a+=x self.setcmd(a) self.setdat(ch) self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): self.char(ord(s[1])) self.char(ord(s[1])) t = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) z BBC micro:bit</pre>	67					
<pre>if x>=0; a=0xC0 if y>0; a=0xC0 a+=x self.setcmd(a) self.setdat(ch)</pre>	68	def	char(self , ch, x=-1, y=0):			
<pre>20 a=0x80 if y>0: a=0xC0 a+=x self.setcmd(a) self.setdat(ch) 70 def puts(self, s, x=0, y=0): if len(s)>0: 79 self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): self.char(ord(s[1])) 80 display.show(Image.HAPPY) 1 = LCD1602() 81 l.puts("Hello microbit!") 80 n = 0 77 suble True: 9 l.puts(str(n), 0, 1) 9 n = n + 1 9 sleep(1000) 72 DBC micro:bit</pre>	69		if x>=0:			
if y>0: a=0xC0 a+=x self.setcmd(a) self.setdat(ch) if len(s)>0: if len(s)>0: self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): self.char(ord(s[1])) display.show(Image.HAPPY) self.char(ord(s[1])) while True: l.puts("Hello microbit!") n = 0 n = n + 1 sleep(1000) v BBC micro:bit iii iii iiii	70		a=0x80			
na=0xC0 a+=x self.set(a) self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): self.char(ord(s[1])) self.char(ord(s[1])) self.char(ord(s[1])) self.char(ord(s[1])) self.char(ord(s[1])) self.set(mage.HAPPY) self.char(ord(s[1])) self.set(mage.HAPPY) self.set(str(n), 0, 1) n = 0 self.set(str(n), 0, 1) n = n + 1 sleep(1000) self.set(str(n), 0, 1) set(str(a)) set(str(a)) set(str(a)) set(str(a)) set(str(a)) set(s	71		if y>0:			
ni a+=x self.setCmd(a) self.setdat(ch) def puts(self, s, x=0, y=0): if len(s)>0: self.char(ord(s[0]),x,y) for 1 in range(1, len(s)): self.char(ord(s[1])) display.show(Image.HAPPY) i 1 = LCD1602() s.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(10000) z BBC micro:bit	72		a=0xC0			
71 setf.setChm(a) 72 setf.setChm(a) 73 setf.setChm(a) 74 setf.setChm(a) 75 if len(s)>0: 75 setf.char(ord(s[0]),x,y) 76 for 1 in range(1, len(s)): 77 setf.char(ord(s[1])) 78 display.show(Image.HAPPY) 81 LCD1602() 82 l.puts("Hello microbit!") 86 n = 0 77 while True: 83 while True: 9 l.puts(str(n), 0, 1) 9 n = n + 1 9 steep(1000) 72 BBC micro:bit	73		a+=x			
75 setf.setual(ch) 76 def puts(self, s, x=0, y=0): 11 if len(s)>0: 78 if len(s)>0: 79 self.char(ord(s[0]),x,y) 80 for 1 in range(1, len(s)): 81 self.char(ord(s[i])) 82 display.show(Image.HAPPY) 84 l = LCD1602() 85 l.puts("Hello microbit!") 86 n = 0 87 while True: 88 while True: 89 l.puts(str(n), 0, 1) 91 sleep(1000) 92 BBC micro:bit	74		self.setdat(ch)			
77 def puts(self, s, x=0, y=0): 78 if len(s)>0: 79 self.char(ord(s[0]),x,y) 80 for i in range(1, len(s)): 81 self.char(ord(s[i])) 82 display.show(Image.HAPPY) 84 l = LCD1602() 85 l.puts("Hello microbit!") 86 n = 0 87 while True: 89 l.puts(str(n), 0, 1) 90 n = n + 1 91 sleep(1000) 92 BBC micro:bit	75		sett.setual(CD)			
177 def puts(self, s, x=0, y=0): 18 if len(s)>0: 19 self.char(ord(s[0]),x,y) 10 for 1 in range(1, len(s)): 11 self.char(ord(s[1])) 12 display.show(Image.HAPPY) 11 LCD1602() 12 l.puts("Hello microbit!") 13 n = 0 14 l.puts(str(n), 0, 1) 15 l.puts(str(n), 0, 1) 16 n = n + 1 17 sleep(1000) 12 BBC micro:bit	76					
<pre>if len(s)>0: self.char(ord(s[0]),x,y) for i in range(1, len(s)): self.char(ord(s[i])) display.show(Image.HAPPY) l = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) 22 BBC micro:bit</pre>	77	def	puts(self , s, x=0, y=0):			
<pre>self.char(ord(s[0]),x,y) for i in range(1, len(s)): self.char(ord(s[1])) display.show(Image.HAPPY) l = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) </pre>	78		if len(s)>0:			
<pre>set for i in range(1, len(s)): self.char(ord(s[i])) self.char(ord(</pre>	79		<pre>self.char(ord(s[0]),x,y)</pre>			
<pre>si self.char(ord(s[i])) display.show(Image.HAPPY) l = LCD1602() l.puts("Hello microbit!") n = 0 while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) BBC micro:bit</pre>	80		for i in range(1, len(s)):			
<pre>82 83 display.show(Image.HAPPY) 84 l = LCD1602() 85 l.puts("Hello microbit!") 86 n = 0 87 88 while True: 89 l.puts(str(n), 0, 1) 90 n = n + 1 91 sleep(1000) 92 BBC micro:bit</pre>	81		<pre>self.char(ord(s[i]))</pre>			
<pre>83 display.show(Image.HAPPY) 84 l = LCD1602() 85 l.puts("Hello microbit!") 86 n = 0 87 88 while True: 89 l.puts(str(n), 0, 1) 90 n = n + 1 91 sleep(1000) 92 BBC micro:bit ↓ </pre>	82					
<pre>s4 t = LCD1602() s5 l.puts("Hello microbit!") s6 n = 0 s7 s8 while True:</pre>	83	display	.show(Image.HAPPY)			
<pre>ss t.puts("Hetto microDitt") ss n = 0 sr ss while True: ss l.puts(str(n), 0, 1) n = n + 1 sleep(1000) g2 BBC micro:bit ↓↓ </pre>	84	L = LCD.	1602() WWelle microbit(")			
86 H = 0 87 88 while True: 89 l.puts(str(n), 0, 1) 90 n = n + 1 91 sleep(1000) 92 BBC micro:bit # ℃	85	r.puts("Hello microdit!")			
<pre>while True: l.puts(str(n), 0, 1) n = n + 1 sleep(1000) BBC micro:bit</pre>	86	= ⊎				
BBC micro:bit BBC micro:bit	87	while T	rue'			
90 n = n + 1 91 sleep(1000) 92 BBC micro:bit ₩ 🔅	88	1. ni	uts(str(n), 0, 1)			
91 Sleep(1000) 92 BBC micro:bit ₩ 0	90	n =	n + 1			
92 BBC micro:bit	91	sle	ep(1000)			
BBC micro:bit 🗰 🔅	92					∇
BBC micro:bit 🌉 💽				سىر	مقدر الم	-
			BBC micro	:bit 🚛	F Q	

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





С м	u 1.1.0.be	ta.2 - Project 9: 1602 LCD.py		—	\times
Mode	+ New	Load Save Flash Files REFL Plotter Zoom-in Zoo	Q C Theme	idy Help Quit	
Proje	ct 9: 16	02 LCD. py 🗙 🥄			
1	from M	icrobit import *			\triangle
2		N			
3	LCD_I2	C_ADDR=0x27			
4					
5	class	LCD1602():			
6	de	<pre>finit(self):</pre>			
7		<pre>self.buf = bytearray(1)</pre>			
8		self.BK = 0x08			
9		self.RS = 0x00			
10		self.E = 0x04			
11		self.setcmd(0x33)			
12		sleep(5)			
13		self.send(0x30)			
14		sleep(5)			
15		self.send(0x20)			
16		sleep(5)			
17		self.setcmd(0x28)			
18		self.setcmd(0x0C)			
19		self.setcmd(0x06)			
20		self.setcmd(0x01)			
21		self.version='1.0'			
22					
23	de	<pre>f setKeg(self, dat): l6 buf[0] = dat</pre>			
24		sett.pur(0) = dat			
25		<pre>iden(1) </pre>			
26		steep(1)			
27					





	de f	sound (colf_dot);	
28	aet	d=dat%0xE0	
29			
30		d =self.BK	
31		d =self.RS	
32		self.setReg(d)	
33		<pre>self.setReg(d 0x04)</pre>	
34		<pre>self.setReg(d)</pre>	
35			
36	def	setcmd(self, cmd):	
37		self.RS=0	
38		self.send(cmd)	
39		<pre>self.send(cmd<<4)</pre>	
40			
41	def	setdat(self , dat):	
42		self.RS=1	
43		self.send(dat)	
44		<pre>self.send(dat<<4)</pre>	
45			
46	def	clear(self):	
47		self.setcmd(1)	
48			
49	def	backlight(self, on):	
50		if on:	
51		self.BK=0x08	
52		else:	
53		self.BK=0	
54		self.setcmd(0)	
55			
56	def	on(self):	
57		self.setcmd(0x0C)	
58			
59	def	off(self):	
60		self.setcma(0x08)	
61			
62	def	Sht(self):	
63		self.setcmd(0x18)	
64			
65	def	shr(self):	
66		self.setcmd(0x1C)	
67			
68	def	char(self , ch, x=-1, y=0):	
69		1f X>=0:	
70		a=0x80	
71		1f y>0:	
72		a=0xC0	
73		a+=x	
74		self.setCMd(a)	
75		self.setdat(Ch)	
76			





(4)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

The micro:bit board will show a smile image. Then rotate the knob of the potentiometer at the back of the LCD module, you will see "Keyestudio" at one row and numbers at the second row. In addition, the number increases by 1 with an interval of 0.5s.

Note: When the display doesn' t show characters, you can adjust the potentiometer behind the 1602LCD and backlight to make the 1602LCD display the corresponding character string.





Project 10: Steam Sensor



(1)Project Introduction

This is a commonly used steam sensor. Its principle is to detect the amount of water by bare printed parallel lines on the circuit board. The more the water content is, the more wires will be connected. As the conductive contact coverage increases, the output voltage will gradually rise. It can detect water vapor in the air as well. The steam sensor can be used as a rain water detector and level switch. When the humidity on the sensor surface surges, the output voltage will increase.

The sensor is compatible with various microcontroller control boards, such as Arduino series microcontrollers. When using it, connect the sensor to the analog port of the Micro:bit microcontroller, and display the corresponding analog value on the serial monitor.

Note: the connection part is not waterproof. Therefore, don't immerse it in the water please.





(2) About the Stream Sensor:

Working	DC 3.3-5V	C1 0805 100NF
Voltage:		R1 0805 1M
Working	- 10°C ~ + 70°C	2 R2 0805 470R J1
Temperatu		$\begin{array}{c c} 1 + & V & 1\\ \hline & G & 3\\ \hline & & & \\ \hline \\ \hline$
re Range:		GND
Max	5uA (DC5V, when	
Working	the two pins of the	
Current:	steam sensor are	
	in short circuit.	
Control	Analog output	
Port:		

(3)Test Code:

Micro:bit Expansion	Stoom Sonsor			
Board	Steam Sensor			
GND	G			
3V3	V			
S(0)	S			





Enter Mu software and open the file "Project 10: Steam Sensor.py" to import

code:

(How to load the project code?)

File	Route	File Name		
Туре				
Python	KS4027 folder/Python	roject 10: Steam		
file	Tutorial/Python	Sensor.py		
	Code/Expansion Project			
	Code/Project 10 : Steam			
	Sensor			

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)




PMu 1.1.0.beta.2 - Project 10: Drop sensor.py			\times
Image: Mode Image: Mode	y Help	Quit	
<pre>froject 10: prop sensor.py from microbit import * val = 0 display.show(Image.HAPPY) while True: val = pin0.read_analog() print("value:", val) sleep(100) </pre>			
			-
BBC n	nicro:bit		





PMu 1.1.0.beta.2 - Project 10: Drop sensor.py			×
Image: Node Image: New Load Image: Save Image: Save </td <td>Tidy Help</td> <td>Quit</td> <td></td>	Tidy Help	Quit	
<pre>1 from microbit import *</pre>			
a^{2} val = 0			
display.show(Image.HAPPY)			
5			
val = pin0.read analog()			
<pre>print("value:", val)</pre>			
<pre> sleep(100) </pre>			
	BBC micro:bit		¢







(4)Test Results:

Upload the test code, and plug in power with micro USB cable. Then the micro:bit will show " \bigcirc ". Click "REPL" and then press the reset button on the back of the board. The serial monitor will show the output data, and the steam sensor will read the analog signals at the signal end. The more the immersed area of the module, the larger the analog value.

As shown below;





С М	u 1.1.0.beta.2 - Project 10: Drop sensor.py	_		×
Mode Projec	+ 10: Drop sensor py	Tidy Help	Quit	
1	from microbit import *			
2				
3	val = 0			
4	display.show(Image.HAPPY)			
5				
6	while True:			
7	val = pin0.read_analog()			
8	print("value:", val)			
9	sleep(100)			
BBC r	nicro:bit REPI			
vacue				•
value	: 853			
value	: 859			
value	: 859			
value	: 861			
value	: 860			
vatue	: 861 : 862			
value	: 865			
				\bigtriangledown
	BB	C micro:bi	t 🇰 i	¢

Project 11: Rains Alarm

(1)Project Introduction

Steam Sensor is a wide range of applications, such as raining alarm, automotive automatic scraping system, intelligent lighting system, and smart sunroof system. In the previous project experiment, we already know the working principle of Steam Sensor, then in this project experiment, we combine Steam Sensor, Micro:bit, and yellow LEDs, making a simple rain





alarm.

(2)Test Code:

Micro:bit	Stoom	Micro:bit	Yellow
Expansion	Steam	Expansion	LED
Board	Sensor	Board	Module
GND	G	GND	G
3V3	V	5V	V
S (0)	S	S (16)	S

Enter Mu software and open the file "Project 11: Rains Alarm.py" to import

code: (How to load the project code

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 11: Rains
file	Tutorial/Python	Alarm.py
	Code/Expansion Project	
	Code/Project 11 : Rains	
	Alarm	

You can also input code in the editing window yourself.





(note:all words and symbols must be written in English)

(M	lu 1.1.0.beta.2 - Project 11: The rain warning.py -		\times
Mode	Image: New Load Image: Save Image: Save </td <td>Quit</td> <td></td>	Quit	
Proje	ct 11: The rain warning py 🛛 💥		
1	from microbit import *		\bigtriangleup
2	import music		
3	display.show(Image.HAPPY)		
4	ndn10 vyndta ddadta1(0)		
5	pinie.write_digital(0)		
5	while True:		
8	if pin0.read analog() > 500:		
9	music.play("C5:4")		
10	pin16.write_digital(1)		
п	sleep(100)		
12	music.reset()		
13	pin16.write_digital(0)		
14	sleep(100)		
15	music.play("C5:4")		
16	pin16.write_digital(1)		
17	sleep(100)		
18	music.reset()		
19	pini6.write_digital(0)		
20	steep(100)		
21	else:		
22	ninite digital(0)		
23	i buite all car(a)		∇
			V
	BBC micro:bit		





С м	u 1.1.0.beta.2 - Project 11: The rain warning.py	_		×
Mode	Image: New Load Image: Save Image: Save	? Help	Quit	
Proje	ct 11: The rain warning py 🗙			
1	from microbit import *			\triangle
2	import music			
3	dísplay.show(Image.HAPPY)			
4				
5	pini6.write_digital(0)			
6	while True:			
6	if pipe read analog() > 500.			
9	music.play("C5:4")			
10	pinl6.write digital(1)			
11	sleep(100)			
12	music.reset()			
13	pin16.write_digital(0)			
14	sleep(100)			
15	music.play("C5:4")			
16	pin16.write_digital(1)			
17	sleep(100)			
18	music.reset()			
19	pin16.write_digital(0)			
20	sleep(100)			
21	else:			
22	music.reset()			
23	pinte.write_digitat(0)			
24				
	BBC mic:	ro:bit	₩ 3	2





С м	u 1.1.0.beta.2 - Project 11: The rain warning.py —	×
Mode	••	
froje	ct 11: The rain warning py	
1	import music	
2	display.show(Image.HAPPY)	
4	and cay for out (indge find fir)	
5	pin16.write digital(0)	
6		
7	while True:	
8	<pre>if pin0.read_analog() > 500:</pre>	
9	music.play("C5:4")	
10	pin16.write_digital(1)	
- 11	sleep(100)	
12	musíc.reset()	
13	pin16.write_digital(0)	
14	sleep(100)	
15	music.play("C5:4")	
16	pinl6.write_digital(1)	
17	sleep(100)	
18	music.reset()	
19	pinl6.write_digital(0)	
20	steep(100)	
21	else:	
22	music.reset()	
23	phite.write_digitat(0)	
24		\mathbf{v}
	BBC micro:bit 🗰 🕻	ł

(3)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.The micro:bit will show smile expression. When the detected analog signals are more than 500, the micro:bit will emit "tick, tick" and the yellow LED will flash. Otherwise, no sound and LED is off.





Project 12: Analog Gas (MQ-2) Sensor



(1)Project Introduction

This gas sensor is used for household gas leak alarms, industrial combustible gas alarms and portable gas detection instruments. Also, it is suitable for the detection of liquefied gas, benzene, alkane, alcohol, hydrogen, etc.,

The MQ-2 smoke sensor can be accurately a multi-gas detector, with the advantages of high sensitivity, fast response, good stability, long life, and simple drive circuit.

It can detect the concentration of flammable gas and smoke in the range of 300~10000ppm. Meanwhile, it has high sensitivity to natural gas, liquefied petroleum gas and other smoke, especially to alkanes smoke.

It must be heated for a period of time before using the smoke sensor, otherwise the output resistance and voltage are not accurate. However, the heating voltage should not be too high, otherwise it will cause internal signal line to blow.





It belongs to the tin dioxide semiconductor gas-sensitive material. At a certain temperature, tin dioxide adsorbs oxygen in the air and forms negative ion adsorption of oxygen, reducing the electron density in the semiconductor, thereby increasing its resistance value.

When in contact with flammable gas in the air and smog, and the potential barrier at the grain boundary is adjusted by the smog, it will cause the surface conductivity to change. With this, information about the presence of smoke or flammable gas can be obtained. The greater the concentration of smoke or flammable gas in the air, the greater the conductivity, and the lower the output resistance, the larger the analog signal output. In addition, the sensitivity can be adjusted by rotating the potentiometer.



(2)About Analog Gas Sensor (MQ-2):

Working	3.3-5V		VCC - VR1 Adjustable pote	ntiometer 10K R1
Voltage:				$\frac{D0}{40} = \frac{1}{2}$
Working	160mA (DC5V)	J? <u>A0</u> 0603 Red	C1	3
Current:		PJ4		U BA10393F \$
Working	0°C ~ 40°C	GND		VCC
Temperature:		VCC T	<u>A0</u>	2





Control Port:	Digital and analog
	output
Detection	300-10000ppm
concentration:	(combustible gas)
Rake Ratio:	≤
	0.6(R3000ppm/R10
	00ppm C3H8)
Sensitivity:	Rs(in
	air)/Rs(1000ppmiso
	butane)≥5
Sensitive	2K Ω -20K Ω (in
Resistance (Rs)	2000ppm C3H8)

Features:

- (1) Have a signal output instruction.
- (2) Dual-channel signal output (analog output and TTL level output)
- (3) TTL output effective signal is Low Level. (When the Low Level is output,

the signal light will be on)

(4) The analog output is $0 \sim 5V$ voltage. The higher the concentration, the higher the voltage.

- (5) a good sensitivity to liquefied gas, natural gas and urban gas.
- (6) Have long-term life expectancy and reliable stability





(7) Fast response recovery.

(3)Test Code:

Micro:bit	Analog Gas
Expansion	(MO_{-2}) Sensor
Board	
GND	G
5V	V
S (1)	D

Enter Mu software and open the file "Project 12: Analog Gas (MQ-2)

Sensor.py" to import code: (How to load the project code?)

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Project 12: Analog Gas (MQ-2)
file	Tutorial/Python	Sensor.py
	Code/Expansion Project	
	Code/Project 12: Analog Gas	
	(MQ-2) Sensor	

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)







(4)Test Results:

Upload the test code to the micro:bit, plug in power and dial the DIP switch to ON. Then the micro:bit will show smile expression.Click "REPL" and then press the reset button on the back of the board.

(How to download? How to quick download?)

When the gas sensor detects no flammable gases, the serial monitor prints 1. While when you turn on a fire lighter near it, it detects gases, the serial monitor prints 0 and the red indicator on the module lights up as shown below:

(By the way, the sensitivity of this sensor can be adjusted by rotating the





blue potentiometer on it.)

С м	lu 1.1.0.beta.2 - Project 12: Analog gas (MQ-2) sensor.py	_		\times
Mode	Image: New Load Image: Save Image: Save	? Help	Quit	
Proje	ct 12: Analog gas (MQ-2) sensor.py 🛛 🗙 🥄			
1	from microbit import *			\bigtriangleup
2				
3	display.show(Image.HAPPY)			
4				
5	while True:			
6	val = pinl.read_digital()			
7	print("digital signal:", val)			
8	sleep(100)			\bigtriangledown
BBC	micro:bit REPI			
41615	at official t			
digit	al signal: 1			
digit	al signal: 1			
digit	at signal: 1			
digit	at signal. 0			
digit	al signal: 0			
digit	al signal: 0			
digit	al signal: 0			
digit	al signal: 0			
digit	al signal: 0			
digit	al signal: 0			
digit	al signa			~
	BBC micz	o:bit		0

Project 13: Gas Leakage Detector

(1) **Project Description**:

This MQ-2 gas sensor is used for household gas leak alarms, industrial combustible gas alarms and portable gas detection instruments. And it is suitable for the detection of liquefied gas, benzene, alkane, alcohol, hydrogen, etc., and widely used in various fire alarm systems. It can be





accurately a multi-gas detector, and has the advantages of high sensitivity, fast response, good stability, long life, and simple drive circuit.

It can detect the concentration of flammable gas and smoke in the range of 300~10000ppm.Meanwhile, it has high sensitivity to natural gas, liquefied petroleum gas and other smoke, especially to alkanes smoke.

We will make a gas leakage detector with a MQ-2 gas sensor, a yellow LED and a 1602 LCD.

(2)Test Code:

Micro:bit			Micro:bit	Yellow	
Expansion	Analog Gas		Expansion	LED	
Board	(MQ-2) Sensor		Board	Module	
GND	GND G		GND	G	
5V	V		5V	V	
S (1)	D		S (16)	S	

Enter Mu software and open the file "Project 13: Gas Leakage Detector.py" to import code:

(How to load the project code?)

File Route	File Name
------------	-----------





Туре		
Python	KS4027 folder/Python	Project 13: Gas Leakage
file	Tutorial/Python	Detector.py
	Code/Expansion Project	
	Code/Project 13 : Gas	
	Leakage Detector	

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)

-							
🕐 М	/u 1.1.0.beta.2 - Project 13: Gas leak detector.py				—		×
Mode	Image: New Load Save Image:	tter Zoom-in	Q Zoom-out Theme	Check Tidy	? Help	Quit	
Proje	ect 13: Gas leak detector.py 🛛 🗶						
1	from microbit import *						\bigtriangleup
2	import Music						
3	LCD_I2C_ADDR=0x27						
4							
5	class LCD1602():						
6	<pre>definit(self):</pre>						
7	<pre>self.buf = bytearray(1)</pre>						
8	self .BK = 0x08						
9	self .RS = 0x00						
10	self .E = 0x04						
- 11	<pre>self.setcmd(0x33)</pre>						
12	sleep(5)						
13	<pre>self.send(0x30)</pre>						
14	sleep(5)						
15	<pre>self.send(0x20)</pre>						
16	sleep(5)						
17	<pre>self.setcmd(0x28)</pre>						
18	<pre>self.setcmd(0x0C)</pre>						
19	<pre>self.setcmd(0x06)</pre>						
20	<pre>self.setcmd(0x01)</pre>						
21	<pre>self.version='1.0'</pre>						
22							
23	<pre>def setReg(self, dat):</pre>						
24	<pre>self.buf[0] = dat</pre>						
25	i2c.write(LCD_I2C_ADDR, se	lf .buf)					
26	sleep(1)						
27							





29 30	def	<pre>send(self, dat): d=dat&0xF0 d =self.BK dl=self RS</pre>	1
32		self.setReg(d)	
33		self.setReg(d 0x04)	
34		<pre>self.setReg(d)</pre>	
35			
36	def	<pre>setcmd(self, cmd):</pre>	
37		self.RS=0	
38		self.send(cmd(<4)	
40	i	acer. Seria (elia (elia))	
41	def	setdat(self, dat):	
42		self.RS=1	
43		<pre>self.send(dat)</pre>	
44		<pre>self.send(dat<<4)</pre>	
45			
46	aer	clear(self):	
48	i	Sect. Second(1)	
49	def	backlight(self , on):	
50		if on:	
51		self.BK=0x08	
52		else:	
53		self.BK=0	
54		self.setcmd(0)	ł.
55			a 🗉 🗉
56	def	on(self):	
56 57 58	def	on(self): self.setcmd(0x0C)	
56 57 58 59	def def	<pre>on(self): self.setcmd(0x0C) off(self):</pre>	
56 57 58 59 60	def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08)</pre>	
56 57 58 59 60 61	def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08)</pre>	
56 57 58 59 60 61 62	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): celf.setcmd(0x18)</pre>	
56 57 58 60 61 62 63 63	def def def	on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18)	
56 57 59 60 61 62 63 64 65	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self):</pre>	
56 57 59 60 61 62 63 64 65 66	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C)</pre>	
56 57 58 59 60 61 62 63 64 65 66 66 67	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C)</pre>	
56 57 58 60 61 62 63 64 65 66 65 66 68	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0):</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76	def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0: a=0x80 if y>0: a=0xC0 a+=x self.setcmd(a) self.setdat(ch) putc(self, c, w=0, w=0);</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79	def def def def	<pre>on(self): self.setcmd(0x0C) off(self): self.setcmd(0x08) shl(self): self.setcmd(0x18) shr(self): self.setcmd(0x1C) char(self, ch, x=-1, y=0): if x>=0:</pre>	





80 81	<pre>for i in range(1, len(s)): self.char(ord(s[i]))</pre>	
82		
83	display.show(Image.HAPPY)	
84	pinl6.write_digital(0)	
85	l = LCD1602()	
86	l.clear()	
87	while True:	
88		
89	<pre>if pin1.read_digital() == 0;</pre>	
90	l.puts("MQ-2:", 1, 0)	
91	l.puts("gas leakage", 1, 1)	
92	music.play("C4:4")	
93	pin16.write_digital(1)	
94	sleep(100)	
95	music.reset()	
96	pin16.write_digital(0)	
97	sleep(100)	
98	else:	
99	l.clear()	
100	music.reset()	
101	pin16.write_digital(0)	
102		_
		∇
	BBC micro:bit 🗰 🔅	





Mu 1.1.0.beta.2 - Project 13: Gas leak detector.py Image: Mu 1.1.0.beta.2 - Project 13: Gas leak detector.py Image: Mode
Image: Second
Project 13: Gas leak detector.py X from microbit import * import music LCD_I2C_ADDR=0x27 class LCD1602(): def_init_(celf):
<pre>1 from microbit import * 2 import music 3 LCD_I2C_ADDR=0x27 4 5 class LCD1602(): 6 definit(celf):</pre>
<pre>import music LCD_I2C_ADDR=0x27 class LCD1602(): defipit(celf):</pre>
<pre>a LCD_I2C_ADDR=0x27 4 5 class LCD1602(): 6</pre>
<pre>4 s class LCD1602(): def init (celf):</pre>
<pre>s class LCD1602(): defipit(celf):</pre>
def init (celf)
s derintc(seci).
<pre>self.buf = bytearray(1)</pre>
s self.BK = 0x08
9 self.RS = 0x00
10 self. $E = 0 \times 04$
n self.setcmd(0x33)
12 SLeep (5)
13 set f. send ($0X_{3}0$)
4 SLEEP(5)
is sett.send(0x20)
16 Steep(S)
17 sett.setcmd(0x28)
is set.setcimd(0x0C)
is set.setcimd(0x05)
20 setf.setCim((0x01)
sect.version=1.0
a def setReg(self (at):
elf buf[0] = dat
i2c.write(ICD_T2C_ADDR_self_buf)
sleen(1)
27





28	def	send(self , dat):	
29		d=dat&0xF0	
30		d = self. BK	
31		d = self .RS	
32		<pre>self.setReg(d)</pre>	
33		self.setReg(d 0x04)	
34		<pre>self.setReg(d)</pre>	
35	م م ا	cotomd(colf, ord);	
36	der	setting(sett, chid).	
30		self.send(cmd)	
39		self.send(cmd<<4)	
40	:		
41	def	setdat(self, dat):	
42		self.RS=1	
43		<pre>self.send(dat)</pre>	
44		<pre>self.send(dat<<4)</pre>	
45			
46	det	clear(self):	
47	l	sett.setCmd(1)	
48	def	backlight(self, on):	
50		if on:	
51		self.BK=0x08	
52		else:	
53		self.BK=0	
54		self.setcmd(0)	
55	i		
56	def	on(self):	
57		<pre>self.setcmd(0x0C)</pre>	
58			
59	def	off(self):	
60		self.setcmd(0x08)	
61	daf	shl(colf);	
62	der	self.setomd(0x18)	
64	i	active cond (0x10)	
65	def	shr(self):	
66		<pre>self.setcmd(0x1C)</pre>	
67			
68	def	char(self , ch, x=-1, y=0):	
69		if x>=0:	
70		a=0x80	
71		if y>0:	
72			
73		elf setemd(a)	
74		self.setdat(ch)	
76	I	a a constant of the second s	
77	def	puts(self , s, x=0, y=0):	
78		<pre>if len(s)>0:</pre>	
79		<pre>self.char(ord(s[0]),x,y)</pre>	





80	<pre>for 1 in range(1, len(s)):</pre>	
81	<pre>self.char(ord(s[1]))</pre>	
82		
83	display.show(Image.HAPPY)	
84	pinl6.write_digital(0)	
85	l = LCD1602()	
86	l.clear()	
87	while True:	
88		
89	<pre>if pin1.read_digital() == 0:</pre>	
90	l.puts("MQ-2:", 1, 0)	
91	l.puts("gas leakage", 1, 1)	
92	music.play("C4:4")	
93	pin16.write_digital(1)	
94	sleep(100)	
95	music.reset()	
96	pin16.write_digital(0)	
97	sleep(100)	
98	else:	
99	l.clear()	
100	music.reset()	
101	pin16.write_digital(0)	
102		-
		\sim
	BBC micro:bit 🗰 🛟	





(M	lu 1.1.0.beta	.2 - Project 13: Gas leak detector.	ру						_		×
	•	t 🛃 👌 🖨 🖷		۹	Q				?	(0)	
Mode	New L	oad Save Flash Files REP	L Plotter	Zoom-in	Zoom-out	Theme	Check	Tidy	Help	Quit	
Proje	ct 13: Gas	leak detector.py									
1	import	music									
3	LCD_I2C	_ADDR=0x27									
4	class 1	CD1602():									
6	def	init(self):									
7		<pre>self.buf = bytearray() lf pk = owop</pre>)								
8		self.BK = 0X08 self.RS = 0X00									
10		self. E = 0x04									
11		<pre>self.setcmd(0x33) close(5)</pre>									
12		steep(5) self.send(0x30)									
14		sleep(5)									
15		<pre>self.send(0x20) </pre>									
16		<pre>steep(5) self.setcmd(0x28)</pre>									
18		<pre>self.setcmd(0x0C)</pre>									
19		<pre>self.setcmd(0x06)</pre>									
20		<pre>self.setcmd(0x01) self.version='1.0'</pre>									
22	1										
23	def	<pre>setReg(self, dat):</pre>									
24		<pre>self.buf[0] = dat i2c.write(ICD_T2C_ADDB</pre>	self.but	F)							
26		sleep(1)	.,	/							
27	·										
28	def	<pre>send(self, dat):</pre>									
29		d=dat&0xF0									
30		d =self.BK d =self.BS									
31		<pre>self.setReg(d)</pre>									
33		<pre>self.setReg(d 0x04)</pre>									
34		<pre>self.setReg(d)</pre>									
35	def	<pre>setcmd(self, cmd):</pre>									
37		<pre>self.RS=0</pre>									
38		<pre>self.send(cmd) self.send(cmd<<4)</pre>									
40	1	Sect: Send (clid ()4)									
41	def	setdat(self , dat):									
42		<pre>self.RS=1 celf send(dat)</pre>									
43		self.send(dat<<4)									
45											
46	def	<pre>clear(self): self setcmd(1)</pre>									
47	l	sect second(1)									
49	def	<pre>backlight(self, on):</pre>									
50		if on:									
51		else:									
53		self.BK=0									





```
self.setcmd(0)
54
55
        def on(self):
56
            self.setcmd(0x0C)
57
58
        def off(self):
59
            self.setcmd(0x08)
60
61
        def shl(self):
62
            self.setcmd(0x18)
63
64
        def shr(self):
65
            self.setcmd(0x1C)
66
67
        def char(self, ch, x=-1, y=0):
68
            if x>=0:
69
                 a=0x80
70
                 if y>0:
71
                     a=0xC0
72
                 .
a+=x
73
                 self.setcmd(a)
74
            self.setdat(ch)
75
76
        def puts(self, s, x=0, y=0):
77
            if len(s)>0:
78
                 self.char(ord(s[0]),x,y)
79
                 for i in range(1, len(s)):
80
                      self.char(ord(s[1]))
81
82
   display.show(Image.HAPPY)
83
   pin16.write_digital(0)
84
   l = LCD1602()
85
   l.clear()
86
   while True:
87
88
        if pinl.read_digital() == 0:
89
             l.puts("MQ-2:", 1, 0)
90
             l.puts("gas leakage", 1, 1)
91
             music.play("C4:4")
92
             pin16.write_digital(1)
93
             sleep(100)
94
             music.reset()
95
            pin16.write_digital(0)
96
            sleep(100)
97
        else:
98
             l.clear()
99
             music.reset()
100
             pin16.write_digital(0)
101
102
                                                                                                   \nabla
                                                                                          # O
                                                                             BBC micro:bit
```

(3)Test Results:





Upload the test code to the micro:bit, plug in power, dial the DIP switch to ON and press "1" on the rocket switch.

The micro:bit will show a smile image. Make a fire lighter close to the gas sensor, 1602 LCD will display "MQ-2" at the first row and show "gas leakage" at the second row. At same time, it will emit "tick,tick" sound and LED will flash.

Project 14: Multiple Functions

(1)**Project Description**:

The final lesson is the combination of all modules and sensors. It is an analog smart home.

(2)Test Code:

Enter Mu software and open the file "Project 14: Multiple Functions.py" to import code:

```
(How to load the project code?)
```

File	Route	File Name
Туре		
Python	KS4027 folder/Python	Multiple Functions
file	Tutorial/Python	
	Code/Expansion Project	





Code/Project 14: Multiple Functions

You can also input code in the editing window yourself.(note:all English

words and symbols must be written in English)

С м	u 1.1.0	0.beta.2 - Project 14: multi-function.py -	\times
(P) Mode	+ New	Image: Weight with the second seco	
Proje	ct 14:	: multi-function.py 🗶	
1	from	n microbit import *	\bigtriangleup
2	impo	ort music	
3	impo	ort neopixel	
4	LCD_	_I2C_ADDR=0x27	
5	dísp	play.show(Image.HAPPY)	
6	pini	l6.write_digital(0)	
7	np =	= neopixel.NeoPixel(pin14, 4)	
8	clas	ss LCD1602():	
9		definit(self):	
10		self.buf = bytearray(1)	
- 11		self.BK = 0x08	
12		self.KS = 0x00	
13		self.E = 0X04	
14		self.SetCmd(0x33)	
15		steep(s)	
16		sect.send(0x30)	
17		steep(s)	
18		sleen(5)	
19		steep(3)	
20		self_setcmd(0x20)	
21		self.setcmd(0x06)	
22		self.setcmd(0x01)	
24		self.version='1.0'	
25			





```
def setReg(self, dat):
26
            self.buf[0] = dat
27
            i2c.write(LCD_I2C_ADDR, self.buf)
28
            sleep(1)
29
30
        def send(self, dat):
31
            d=dat&0xF0
32
            d|=self.BK
33
            d|=self.RS
34
            self.setReg(d)
35
            self.setReg(d|0x04)
36
            self.setReg(d)
37
38
        def setcmd(self, cmd):
39
            self.RS=0
40
            self.send(cmd)
41
            self.send(cmd<<4)</pre>
42
43
44
        def setdat(self, dat):
            self.RS=1
45
            self.send(dat)
46
            self.send(dat<<4)</pre>
47
48
        def clear(self):
49
            self.setcmd(1)
50
51
        def backlight(self, on):
52
            if on:
53
54
                 self.BK=0x08
            else:
55
                 self.BK=0
56
            self.setcmd(0)
57
58
        def on(self):
59
            self.setcmd(0x0C)
60
61
        def off(self):
62
            self.setcmd(0x08)
63
64
        def shl(self):
65
            self.setcmd(0x18)
66
67
        def shr(self):
68
            self.setcmd(0x1C)
69
70
        def char(self, ch, x=-1, y=0):
71
            if x>=0:
72
                 a=0x80
73
                 if y>0:
74
                 a=0xC0
75
                 a+=x
76
```









127	np.show()
128	sleep(1000)
129	<pre>for pixel_id3 in range(0, len(np)):</pre>
130	np[pixel_1d3] = (255, 255, 0)
131	np.snow()
132	for pixel idd in sange(0, len(np));
133	For pixel_id41 = $(0, 255, 0)$
134	np_pixet_id4] = (0, 255, 0)
136	sleen(1000)
137	for pixel id5 in range(0, len(np)):
138	$np[pixel_id5] = (0, 0, 255)$
139	np.show()
140	sleep(1000)
141	<pre>for pixel_id6 in range(0, len(np)):</pre>
142	np[pixel_id6] = (75, 0, 130)
143	np.show()
144	steep(1000)
145	prime price [107 in range(0, ten(np));
146	np.show()
148	sleep(1000)
149	<pre>for pixel_id8 in range(0, len(np)):</pre>
150	np[pixel_id8] = (160, 32, 240)
151	np.show()
152	sleep(1000)
153	<pre>for pixel_id9 in range(0, len(np)):</pre>
154	np[p1xel_1d9] = (255, 255, 255)
155	steep(1000)
156	sleen(1000)
157	Servo(pin9), write angle(0)
159	sleep(3000)
160	Servo(pin8).write_angle(120)
161	sleep(3000)
162	pin12.write_digital(1)
163	pinl3.write_digital(0)
164	sleep(5000)
165	else:
166	music.reset()
167	pinl6.write_digital(0)
168	Servo(ping).write_angle(90)
169	Steep(200) Serve(pip8) write angle(0)
170	sleen(200)
171	pin12.write digital(0)
173	pin13.write digital(0)
174	np.clear()
175	<pre>if pinl.read_digital() == 0:</pre>
176	l.puts("MQ-2:", 1, 0)







(N	lu 1.1.0.	beta.	2 - Project 1	4: multi	-functi	on.py							—		\times	
Mode	+ New		ad Save	Flash	Files	REPL	Plotter	Zoom-in	Q Zoom-out	C Theme	Check	Tidy	? Help	Ouit		
Proje	ct 14:	mult	i-function	py 3	K			,								
1	from	m 1 c	robit in	port *	¢											
2	impor	t n	NUSIC													
4	LCD_1	2C_	ADDR=0x2	27												
5	displ	ay.	show(Ima	ge.HAP	PPY)											
6	pin10	nec nec	ite_digi nixel.Ne	tal(0)) (nin	14 4)									
8	class	E LO	D1602():	OFIAC	(pm	14, 4	/									
9	¢	lef	init	(self)	:											
10			self.but	= byt - ovos	tearr	ay (1)										
11			self.RS	$= 0 \times 0 0$))											
13			self.E =	0x04												
14			self.set	cmd(0)	(33)											
15			steep(s)	nd(0x30	3)											
17			<pre>sleep(5)</pre>		, 											
18			self.ser	nd(0x20	9)											
20			steep(5) self.set	cmd(0)	(28)											
21			self.set	cmd(0)	(0C)											
22			self.set	cmd(0)	(06)											
23			self.ver	sion='	1.0'											
25	i															
26	d	lef	setReg(s	elf, d	lat):											
27			self.buf	[0] =	dat Tac /		- 16 1	E \								
28			sleep(1)	e(LCD_	120_6	NUDR,	setti	jur)								l
30	:															l
31	c	lef	send(sel	f, dat	:):											l
32			d=data0x	BK												
34			d=self.	RS												
35			self.set	Reg(d) Reg(d)	0×04											l
36			self.set	Reg(d)	0,04,	,										
38																
39	c	lef	setcmd(s	elf, c	md):											
40			self.ser	d(cmd)												
42			self.ser	d(cmd<	<4)											
43		. f	cotdat(a	alf d	a+).											
44	G	ет	setuar(s self.RS=	1 1	ac):											
46			self.ser	d(dat)												
47			self.ser	d(dat<	<4)											
48	d	lef	clear(se	lf):												
50			self.set	cmd(1)												





```
51
        def backlight(self, on):
52
            if on:
53
                 self.BK=0x08
54
            else:
55
                 self.BK=0
56
            self.setcmd(0)
57
58
        def on(self):
59
            self.setcmd(0x0C)
60
61
        def off(self):
62
            self.setcmd(0x08)
63
64
        def shl(self):
65
            self.setcmd(0x18)
66
67
        def shr(self):
68
            self.setcmd(0x1C)
69
70
        def char(self, ch, x=-1, y=0):
71
             if x>=0:
72
                 a=0x80
73
                 if y>0:
74
                     a=0xC0
75
                 a + = x
76
                 self.setcmd(a)
77
            self.setdat(ch)
78
79
        def puts(self, s, x=0, y=0):
80
            if len(s)>0:
81
                 self.char(ord(s[0]),x,y)
82
                 for 1 in range(1, len(s)):
83
                     self.char(ord(s[1]))
84
   class Servo:
85
        def __init__(self, pin, freq=50, min_us=600, max_us=2400, angle=180):
86
            self.min_us = min_us
87
            self.max_us = max_us
88
            self.us = 0
89
            self.freq = freq
90
            self.angle = angle
91
            self.analog_period = 0
92
            self.pin = pin
93
            analog_period = round((1/self.freq) * 1000) # hertz to miliseconds
94
            self.pin.set_analog_period(analog_period)
95
96
        def write_us(self, us):
97
            us = min(self.max_us, max(self.min_us, us))
98
            duty = round(us * 1024 * self.freq // 1000000)
99
            self.pin.write_analog(duty)
100
            sleep(100)
101
```





102	<pre>self.pin.write_analog(0)</pre>
103	
104	<pre>def write_angle(self, degrees=None):</pre>
105	if degrees is None:
106	degrees = math.degrees(radians)
107	degrees = degrees % 360
108	total_range = self. max_us - self. min_us
109	us = self .min_us + total_range * degrees // self .angle
110	<pre>self.write_us(us)</pre>
111	l = LCD1602()
112	l.clear()
113	Servo(pin9).write_angle(90)
114	Servo(pin8).write_angle(0)
115	display.show(image.HAPPY)
116	pini2.write_digital(0)
117	pini3.write_digital(0)
118	np.ctear()
119	white frue: if pipe read analog() > 400:
120	for pixel idl in range(0, lep(pp)):
121	pr[rive] idll = (255, 0, 0)
122	$np[privec_[dir] = (235, 6, 6)$
123	sleen(1000)
125	for pixel id2 in range(0, len(np)):
126	$np[pixel_id2] = (255, 165, 0)$
	,,,,,,
127	np.show()
128	sleep(1000)
129	<pre>for pixel_id3 in range(0, len(np)):</pre>
130	np[p1xel_1d3] = (255, 255, 0)
131	np.snow()
132	steep(1000)
133	tor pixel_id41 = $(0, 255, 0)$
134	$np[p1xet_1d4] = (0, 255, 0)$
135	sloop(1000)
136	for pixel id5 in range(A lep(pp)):
137	$privel_idSi = (0, 0, 255)$
139	np.show()
140	sleep(1000)
141	<pre>for pixel id6 in range(0, len(np)):</pre>
142	np[pixel id6] = (75, 0, 130)
143	np.show()
144	sleep(1000)
145	<pre>for pixel_id7 in range(0, len(np)):</pre>
146	np[pixel_id7] = (238, 130, 238)
147	np.show()
148	sleep(1000)
149	<pre>for pixel_id8 in range(0, len(np)):</pre>
150	np[pixel_id8] = (160, 32, 240)
151	np.show()





152	steep(1000)	
153	Tor pixel_1d9 in range(0, ten(np)):	
154	sleen(1000)	
155	pp. clear()	
155	sleen(1000)	
158	Servo(pin9).write angle(0)	
159	sleep(3000)	
160	Servo(pin8).write angle(120)	
161	sleep(3000)	
162	pin12.write_digital(1)	
163	pin13.write_digital(0)	
164	sleep(5000)	
165	else:	
166	music.reset()	
167	pin16.write_digital(0)	
168	Servo(pin9).write_angle(90)	
169	steep (200)	
170	Servo(pin8).write_angle(0)	
171	steep(200)	
172	pin12.write_digital(0)	
173	np.clear()	
175	if pinl.read digital() == 0:	
176	l.puts("MO-2:", 1, 0)	
177	l.puts("gas leakage", 1, 1)	
178	music.play("C5:4")	
179	pini6.write_digital(1)	
180	steep(100)	
181	ninl6.write_digital(0)	
183	sleep(100)	
184	music.play("C5:4")	L
185	pin16.write_digital(1)	
186	sleep(100)	L
187	musíc.reset()	L
188	pin16.write_digital(0)	L
189	sleep(200)	
190	else:	L
191	l.clear()	
192	music.reset()	L
193	pinl6.write_digital(0)	
194		
	BBC migrathit 🚟 💏	
		L





🕐 М	u 1.1.0.beta.2 - Project 14: multi-function.py – 🗆 🗙	
Mode	+ 1	
Proje	ct 14: multi-function.py	
1	from microbit import *	Δ
2	import music	
3	import neopixel	
4	LCD_I2C_ADDR=0x27	
5	display.show(Image.HAPPY)	
6	pin16.write_digital(0)	
7	np = neopixel.NeoPixel(pin14, 4)	
8	class LCD1602():	
9	<pre>definit(self):</pre>	
10	<pre>self.buf = bytearray(1)</pre>	
11	self.BK = 0x08	
12	<pre>self.RS = 0x00</pre>	
13	self.E = 0x04	
14	<pre>self.setcmd(0x33)</pre>	
15	sleep(5)	
16	<pre>self.send(0x30)</pre>	
17	sleep(5)	
18	<pre>self.send(0x20)</pre>	
19	sleep(5)	
20	<pre>self.setcmd(0x28)</pre>	
21	<pre>self.setcmd(0x0C)</pre>	
22	<pre>self.setcmd(0x06)</pre>	
23	<pre>self.setcmd(0x01)</pre>	
24	<pre>self.version='1.0'</pre>	
25		

```
def setReg(self, dat):
26
            self.buf[0] = dat
27
            i2c.write(LCD_I2C_ADDR, self.buf)
28
            sleep(1)
29
30
        def send(self, dat):
31
            d=dat&0xF0
32
            d =self.BK
33
            d|=self.RS
34
            self.setReg(d)
35
            self.setReg(d|0x04)
36
            self.setReg(d)
37
38
       def setcmd(self, cmd):
39
            self.RS=0
40
            self.send(cmd)
41
            self.send(cmd<<4)</pre>
42
43
       def setdat(self, dat):
44
            self.RS=1
45
            self.send(dat)
46
            self.send(dat<<4)</pre>
47
48
        def clear(self):
49
            self.setcmd(1)
50
```




```
51
        def backlight(self, on):
52
            if on:
53
                self.BK=0x08
54
            else:
55
                self.BK=0
56
            self.setcmd(0)
57
58
        def on(self):
59
            self.setcmd(0x0C)
60
61
        def off(self):
62
            self.setcmd(0x08)
63
64
        def shl(self):
65
            self.setcmd(0x18)
66
67
        def shr(self):
68
            self.setcmd(0x1C)
69
70
        def char(self, ch, x=-1, y=0):
71
            if x>=0:
72
                 a=0x80
73
                 if y>0:
74
                     a=0xC0
75
                 a + = x
76
                 self.setcmd(a)
77
78
             self.setdat(ch)
79
        def puts(self, s, x=0, y=0):
80
             if len(s)>0:
81
                 self.char(ord(s[0]),x,y)
82
                 for i in range(1, len(s)):
83
                     self.char(ord(s[1]))
84
   class Servo:
85
        def __init__(self, pin, freq=50, min_us=600, max_us=2400, angle=180):
86
            self.min_us = min_us
87
            self.max_us = max_us
88
            self.us = 0
89
            self.freq = freq
90
            self.angle = angle
91
            self.analog_period = 0
92
            self.pin = pin
93
            analog_period = round((1/self.freq) * 1000) # hertz to miliseconds
94
            self.pin.set_analog_period(analog_period)
95
96
        def write_us(self, us):
97
            us = min(self.max_us, max(self.min_us, us))
98
            duty = round(us * 1024 * self.freq // 1000000)
99
            self.pin.write_analog(duty)
100
            sleep(100)
101
```





```
self.pin.write_analog(0)
102
103
        def write_angle(self, degrees=None):
104
            if degrees is None:
105
                 degrees = math.degrees(radians)
106
            degrees = degrees % 360
107
            total_range = self.max_us - self.min_us
108
            us = self.min us + total range * degrees // self.angle
109
            self.write_us(us)
110
   l = LCD1602()
111
   l.clear()
   Servo(pin9).write_angle(90)
113
   Servo(pin8).write_angle(0)
114
   display.show(Image.HAPPY)
115
   pin12.write_digital(0)
116
   pin13.write_digital(0)
117
   np.clear()
118
   while True:
119
        if pin0.read_analog() > 400:
120
            for pixel_id1 in range(0, len(np)):
121
                 np[pixel_id1] = (255, 0, 0)
                 np.show()
123
            sleep(1000)
124
             for pixel_id2 in range(0, len(np)):
125
                 np[pixel_id2] = (255, 165, 0)
126
                 np.show()
127
            sleep(1000)
128
            for pixel_id3 in range(0, len(np)):
129
                 np[pixel_id3] = (255, 255, 0)
130
                 np.show()
131
            sleep(1000)
132
            for pixel_id4 in range(0, len(np)):
133
                 np[pixel_id4] = (0, 255, 0)
134
                 np.show()
135
            sleep(1000)
136
            for pixel_id5 in range(0, len(np)):
                 np[pixel_id5] = (0, 0, 255)
138
                 np.show()
139
            sleep(1000)
140
            for pixel_id6 in range(0, len(np)):
141
                 np[pixel_id6] = (75, 0, 130)
142
                 np.show()
143
            sleep(1000)
144
            for pixel_id7 in range(0, len(np)):
145
                 np[pixel_id7] = (238, 130, 238)
146
                 np.show()
147
            sleep(1000)
148
             for pixel_id8 in range(0, len(np)):
149
                 np[pixel_id8] = (160, 32, 240)
150
                 np.show()
```





	BBC micro:bit 🗮 🚺	
194		٢
193	pin16.write_digital(0)	
192	music.reset()	
191	l.clear()	
190	else:	
189	sleep(200)	
188	pin16.write_digital(0)	
187	music.reset()	
186	sleep(100)	
185	pin16.write_digital(1)	
184	music.play("C5:4")	
183	sleep(100)	
182	pin16.write_digital(0)	
181	music.reset()	
180	sleep(100)	
179	pin16.write digital(1)	
179	music.play("C5:4")	
177	1.puts("gas leakage", 1, 1)	
176	l.puts("MQ-2:", 1, 0)	
175	<pre>if pinl.read_digital() == 0:</pre>	
174	np.clear()	
173	pin13.write_digital(0)	
172	pin12.write_digital(0)	
171	sleep(200)	
170	Servo(pin8).write_angle(0)	
169	sleep(200)	
168	Servo(pin9).write_angle(90)	
167	pin16.write_digital(0)	
166	music.reset()	
165	else:	
164	sleep(5000)	
163	pin13.write digital(0)	
162	pin12.write digital(1)	
161	sleep(3000)	
160	Servo(pin8).write_angle(120)	
158	sleep(3000)	
157	Servo(pip9) write angle(0)	
156	np.ctear()	
155	steep(1000)	
154	np[p1xel_1d9] = (255, 255, 255)	
153	for pixel_1d9 in range(0, len(np)):	
152	sleep(1000)	
		100

(3)Test Results:

Upload the test code to the micro:bit, plug in power, dial the DIP switch to





ON and press "1" on the rocket switch. The micro: bit will show a smile image.

When the analog value detected by the steam sensor is bigger than 400, the 5 WS2812 RGB lights on the 6812 RGB module are all on, displaying in red, orange,yellow, green, blue, Indigo, violet, purple and white, in loop way.

Then the window is closed, the door and the fan rotate;

Make a fire lighter close to the gas sensor, 1602 LCD will display "MQ-2" at the first row and show "gas leakage" at the second row. At same time, it will emit "tick,tick" sound and the yellow LED will flash. Otherwise,the speaker makes no sound, the LED reminds off and the 1602LCD displays no characters.

8.Resources:

https://fs.keyestudio.com/KS4027-4028