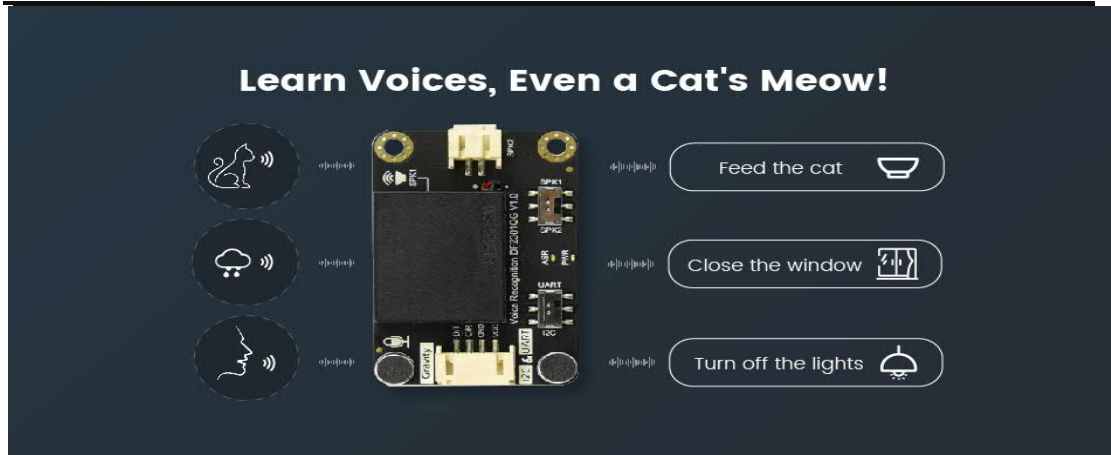


CAP019-EN

CAPTEUR DE RECONNAISSANCE VOCALE POUR ARDUINO/RASPBERRY PI/PYTHON/ESP32 - I2C ET UART



SKU:SEN0539-EN

Introduction

What is voice recognition?

Voice recognition is a computer technology that recognizes and converts speech signals into editable text or operational commands through analysis. It allows people to interact with computers by speaking without using a mouse, keyboard, or other input devices. Voice recognition technology has been widely used in applications such as voice assistants, smart homes, voice search, and voice recognition notebooks.

This Gravity: Offline Voice Recognition Sensor is built around an offline voice recognition chip, which can be directly used without an internet connection. It comes with 121 built-in fixed command words and supports the addition of 17 custom command words. Meanwhile, this voice recognition module compatibility with multiple common controllers enables it to provide a flexible solution for makers and electronics enthusiasts in terms of voice interaction. It can be applied to any application that requires voice control or interaction, such as various smart home appliances, toys, lighting fixtures, and robotics projects, among others.

Self-Learning Function

The Voice Recognition module is equipped with a self-learning function and supports the addition of 17 custom command words. Any sound could be trained as a command, such as whistling, snapping, or even cat meows, which brings great flexibility to interactive audio projects.

For instance automatic pet feeder. When a cat emits a meow, the offline voice recognition module can recognize the meow and trigger the feeder to automatically provide food for the cat. This innovative design ensures that the owner can promptly meet the cat's dietary needs. Moreover, the product is equipped with excellent noise resistance and long-distance recognition capabilities, allowing for precise identification of the cat's meows even in noisy environments.

User-Friendly

The offline voice recognition module boasts a user-friendly design. It comes with 121 built-in fixed command words, eliminating the need for users to record their own voices. For instance, in an intelligent window system, when it starts to rain or thunder, there's no need for manual window operation. The offline voice recognition module can recognize the pre-set command word "close the window," triggering the automatic closing of the window to cope with sudden weather changes.

Real-Time Voice Feedback

The module features a dual microphone design with better noise resistance and a longer recognition distance, making it relatively accurate and reliable even in noisy environments. It comes with a built-in speaker and an external speaker interface for real-time voice feedback of recognition results.

For instance, one can wake up a voice assistant using a wake-up word, and the assistant promptly responds and begins learning or utilizing command words. When learning or deleting command words, the voice assistant also provides immediate feedback on the success of the operation. This greatly enhances the user's experience and convenience.

High Compatibility

The module boasts high compatibility, supporting both I2C and UART communication methods, while also being compatible with various 3.3V or 5V controllers, such as micro:bit, Arduino (Arduino UNO, Arduino Leonardo, Arduino MEGA), Raspberry Pi, FireBeetle series, and more. This provides a flexible solution for building smart home systems, robotics projects, and more.

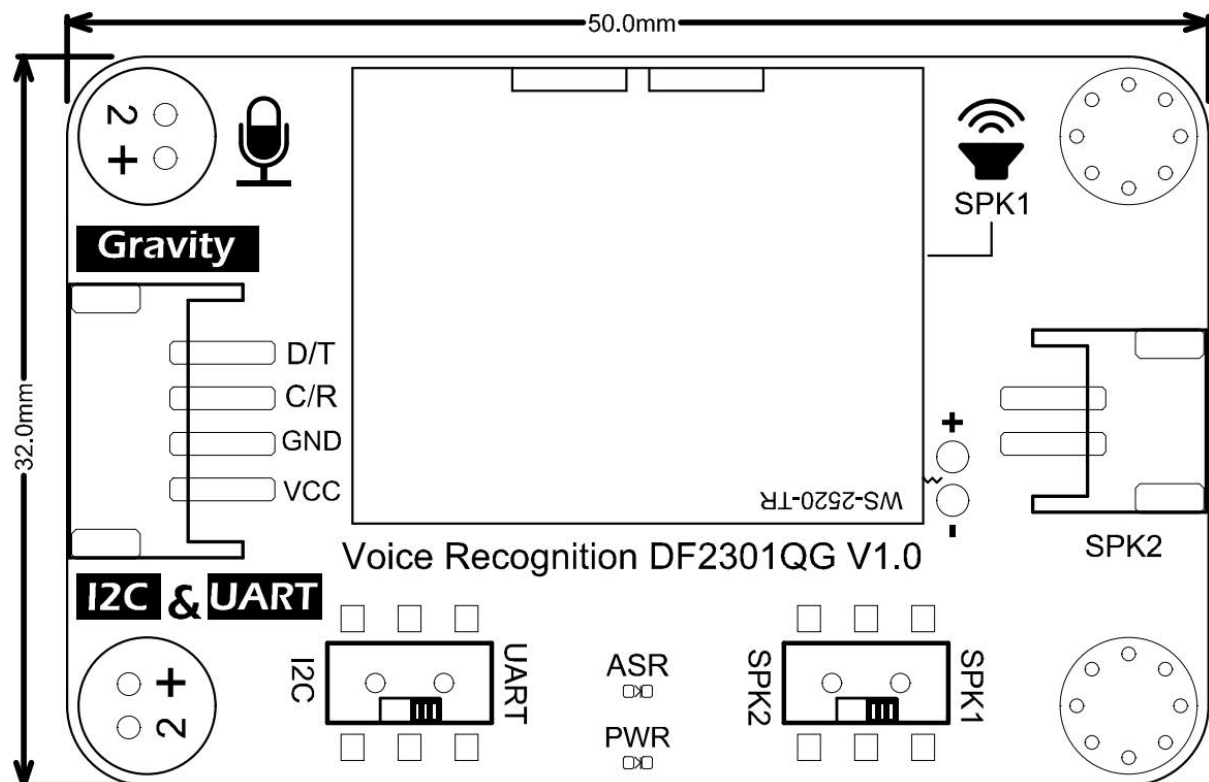
Features

- Self-learning function: Control the module to learn command words by the voice, and any audio can be trained as a command
- Support I2C and UART, with a Gravity interface
- Compatible with 3.3V/5V
- Built-in with 121 commonly-used fixed command words
- The module has a built-in speaker and an interface for an external speaker, which can provide real-time voice feedback on recognition results
- Equipped with power indicator (red) and recognition status indicator (blue)
- Dual microphones provide better noise resistance and longer recognition distance

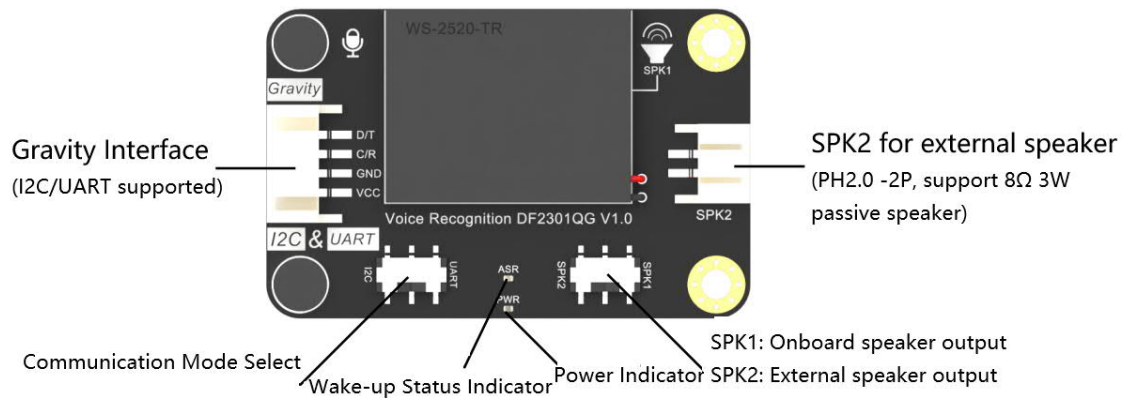
- Compatible with Arduino controllers: Arduino UNO, Arduino Leonardo, Arduino MEGA, FireBeetle series controllers, Raspberry Pi, ESP32

Specification

- Operating Voltage: 3.3 - 5V
- Maximum Operating Current: ≤ 370 mA (5V)
- Communication: I2C/UART
- I2C Address: 0x64
- Fixed Command: 121
- Fixed Wake-up Command: 1
- Custom Command: 17
- Learning Activation Command: 1
- Onboard Microphone Sensitivity: -28db
- Module Size: 49×32 mm/1.93×1.26"



Board Overview



Command Words

Wake-up word

The wake-up word refers to the word that switches a product from standby mode to operational mode. It serves as the initial point of interaction between users and voice-enabled devices.

Learning wake-up word:

- Initiate the voice assistant by employing the default wake-up word, then utter "Learning wake word" Follow the prompts to learn the new wake-up word (prior to each learning command, it is necessary to remove the previously learned wake-up word; kindly refer to the instructions for wake-up word and command phrase deletion)
- Indication: Learning now, be quiet, please say the wake word to be learned!
- The designated wake-up word to be acquired (taking "hello, there" as an example): "hello, there"
- Indication: Learning successful, please say it again!
- The designated wake-up word to be acquired : "hello, there"
- Prompt: Learning successful, please say it again!
- The designated wake-up word to be acquired : "hello, there"
- Prompt: Ok, learning completed!

Once the learning process is accomplished, you will be able to utilize the phrase "hello, there" to awaken the voice assistant!

Fixed command words:

Fixed command words refers to the designated vocabulary used by users to issue specific instructions to voice interactive products, enabling effective communication with them.

Learning command words:

Use a wake-up word (default or learned) to wake up the voice assistant, and then say "**Learning command word**" to initiate the process of learning command phrases, following

the provided prompts. Before each session of learning command phrases, it is necessary to delete the previously learned command phrases. Please refer to instructions on how to delete wake-up words and command phrases.

- Indication: Learning now, be quiet, please learn the command word according to the prompt! Please say the first command to be learned!
- Command phrase to be learned (using "Turn on red light" as an example): "Turn on red light"
- Indication: Learning successful, please say it again!
- Command phrase to be learned : "Turn on red light"
- Indication: Learning successful, please say it again!
- Command phrase to be learned : "Turn on red light"
- Indication: OK, learned the first command successfully! Please say the second command to be learned!

... (Continue learning)

Command phrase to exit learning mode: **"Exit learning"**

After the completion of the learning process, an ID will be automatically generated. Please refer to the subsequent document titled "Command Words/Wake-up Words & ID Table" By utilizing this unique ID, you can author programs to exercise control accordingly.

Delete Wake Words and Command Words:

Summon the voice assistant using the awakening word (default or customized), and articulate the phrase **"I want to delete"** Follow the prompts to eliminate the specified command phrase as instructed.

- Indication: Do you want to delete the learned wake word or command word?
- Delete command word: Remove the previously acquired command phrases.
- Delete wake word: Erase the learned awakening words from the system.
- Delete all: Eliminate the assimilated awakening utterances and command phrases from memory.
- **Exit deleting.**

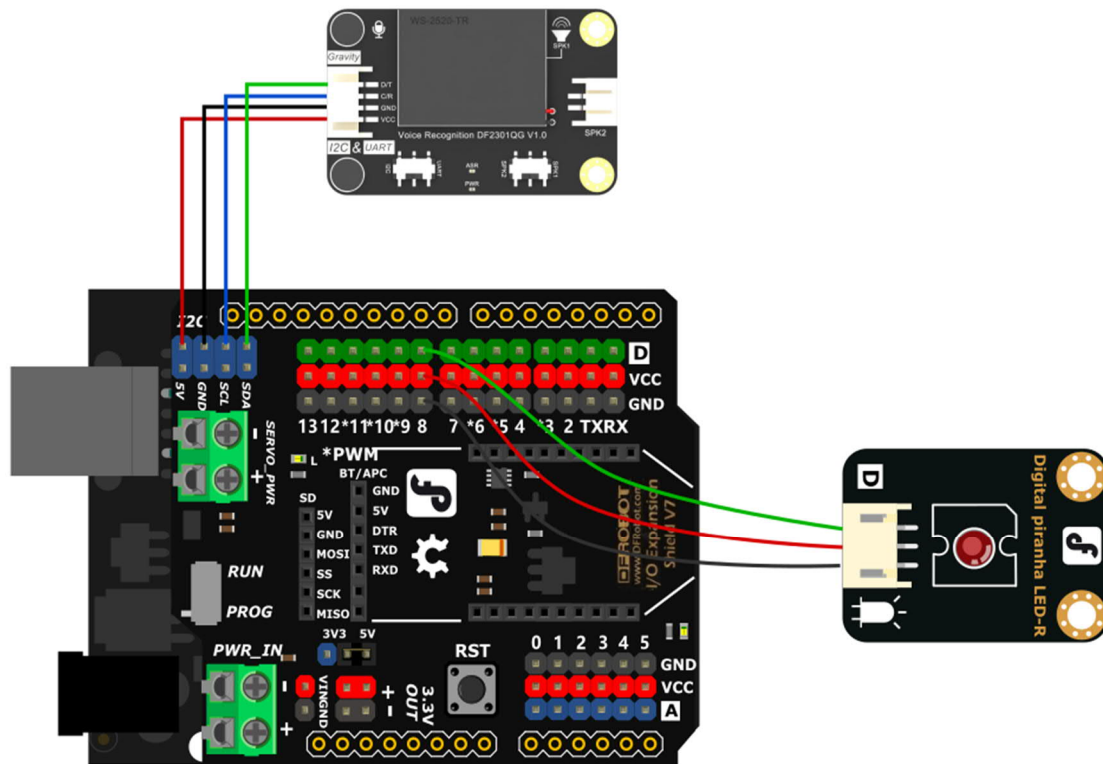
Tutorial

Requirements

- Hardware
 - [DFRduino UNO R3](#) (or similar) x 1
 - SEN0539 Gravity: Voice Recognition Module x I2C & UART
 - [Gravity: Digital RED LED Light Module](#)
- Software
 - [Arduino IDE](#)

- Download and install the [DFRobot_DF2301Q Library](#) ([About how to install the library?](#))

Connection Diagram - I2C



Sample Code

Switch the communication mode to the I2C orientation and include the necessary library file, [DFRobot_DF2301Q](#), for the code.

```

/*!
 * @file i2c.ino
 * @brief Control the voice recognition module via I2C
 * @n Get the recognized command ID and play the corresponding reply audio
 according to the ID;
 * @n Get and set the wake-up state duration
 * @copyright Copyright (c) 2010 DFRobot Co. Ltd (http://www.dfrobot.com)
 * @licence The MIT License (MIT)
 * @author [qsjhyy](yi huan. huang@dfrobot.com)
 * @version V1.0
 * @date 2022-04-02
 * @url https://github.com/DFRobot/DFRobot_DF2301Q
 */
#include "DFRobot_DF2301Q.h"

#define Led 8

```

```

//I2C communication
DFRobot_DF2301Q_I2C asr;

void setup() {
    Serial.begin(115200);

    pinMode(Led, OUTPUT);    //Init LED pin to output mode
    digitalWrite(Led, LOW);  //Set LED pin to low

    // Init the sensor
    while (! (asr.begin())) {
        Serial.println("Communication with device failed, please check connection");
        delay(3000);
    }
    Serial.println("Begin ok!");

    /**
     * @brief Set voice volume
     * @param voc - Volume value(1-7)
     */
    asr.setVolume(4);

    /**
     * @brief Set mute mode
     * @param mode - Mute mode; set value 1: mute, 0: unmute
     */
    asr.setMuteMode(0);

    /**
     * @brief Set wake-up duration
     * @param wakeTime - Wake-up duration (0-255)
     */
    asr.setWakeTime(20);

    /**
     * @brief Get wake-up duration
     * @return The currently-set wake-up period
     */
    uint8_t wakeTime = 0;
    wakeTime = asr.getWakeTime();
    Serial.print("wakeTime = ");
    Serial.println(wakeTime);

    // asr.playByCMDID(1);    // Wake-up command

    /**
     * @brief Play the corresponding reply audio according to the ID
     * @param CMDID - command word ID
     */
    //asr.playByCMDID(23);    // Command word ID
}

void loop() {
    /**
     * @brief Get the ID corresponding to the command word
     * @return Return the obtained command word ID, returning 0 means no valid ID is
     obtained
     */
    uint8_t CMDID = asr.getCMDID();

```

```

switch (CMDID) {
    case 103:                                     //If the command is
    "Turn on the light"
        digitalWrite(Led, HIGH);                 //Turn on the LED
        Serial.println("received' Turn on the light', command flag' 103' "); //Serial
transmits "received'Turn on the light", command flag"103"
        break;

    case 104:                                     //If the command is
    "Turn off the light"
        digitalWrite(Led, LOW);                  //Turn off the LED
        Serial.println("received' Turn off the light', command flag' 104' "); //The
serial transmits "received'Turn off the light", command flag"104""
        break;

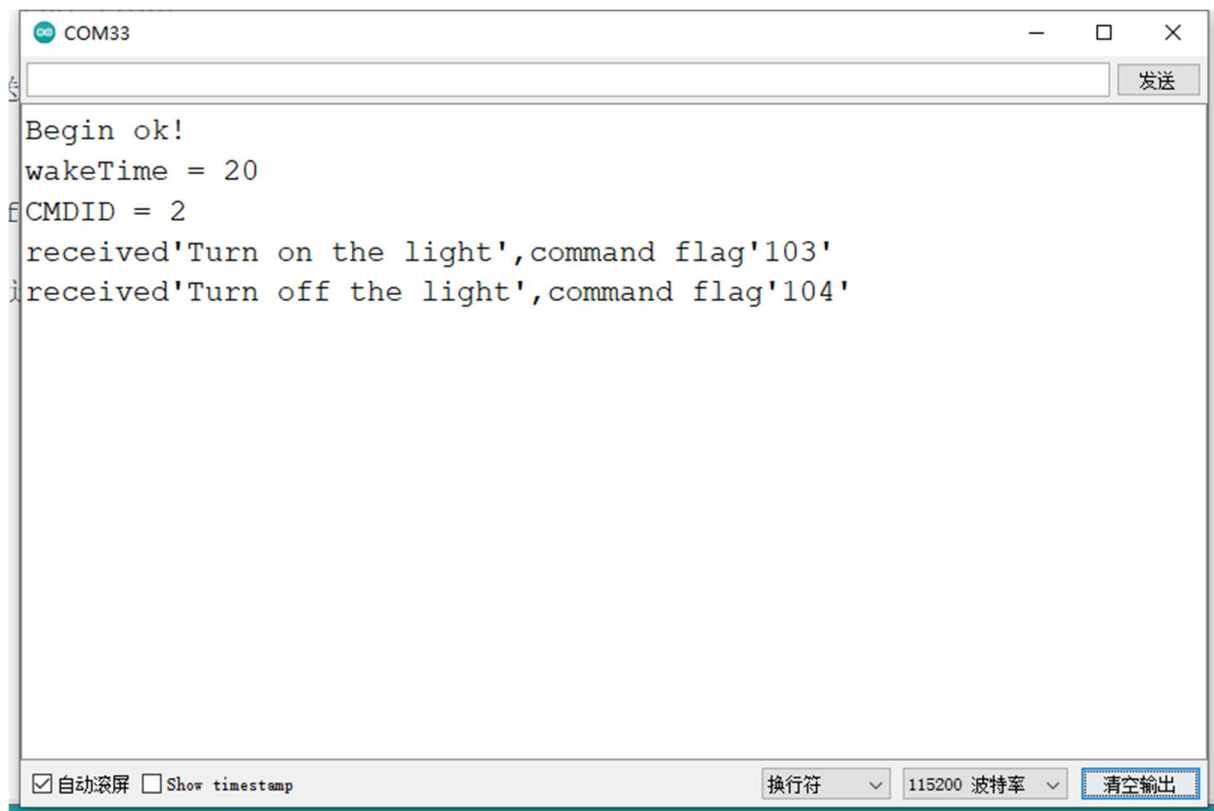
    default:
        if (CMDID != 0) {
            Serial.print("CMDID = "); //Printing command ID
            Serial.println(CMDID);
        }
    }
    delay(300);
}

```

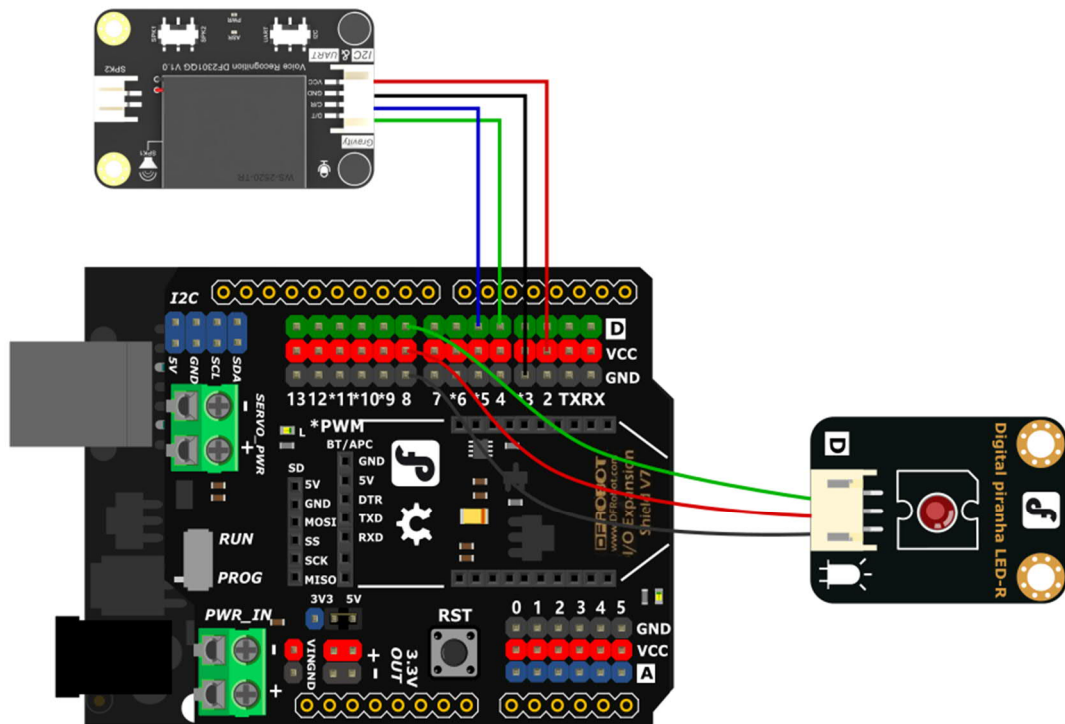
Copy

Expected Results

Awaken the speech recognition module using a predefined or learned awakening phrase. Subsequently, command the illumination module with the phrases "Turn on the light" or "Turn off the light" for controlling its functionality.



Connection Diagram - UART



Sample Code

Please switch the communication mode switch to the UART and download the required library file DFRobot_DF2301Q library for the code.

```
/*!
 * @file uart.ino
 * @brief Control the voice recognition module via UART
 * @n Get the recognized command ID and play the corresponding reply audio
according to the ID;
 * @n Set the wake-up state duration, set mute mode, set volume, enter the wake-
up state, and reset the module
 * @copyright Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @licence The MIT License (MIT)
 * @author [qsjhyy](yi.huan.huang@dfrobot.com)
 * @version V1.0
 * @date 2022-04-02
 * @url https://github.com/DFRobot/DFRobot_DF2301Q
 */
#include "DFRobot_DF2301Q.h"

#define Led 8

/**
 @brief DFRobot_URM13_RTU constructor
 @param serial - serial ports for communication, supporting hard and soft serial
ports
 @param rx - UART The pin for receiving data
 @param tx - UART The pin for transmitting data
 */
#if defined(ARDUINO_AVR_UNO) || defined(ESP8266) // Use software serial
SoftwareSerial softSerial(/*rx=*/4, /*tx=*/5);
DFRobot_DF2301Q_UART asr(/*softSerial=*/&softSerial);
#elif defined(ESP32) // Use the hardware serial with remappable pin: Serial1
DFRobot_DF2301Q_UART asr(/*hardSerial=*/&Serial1, /*rx=*/D3, /*tx=*/D2);
#else // Use hardware serial: Serial1
DFRobot_DF2301Q_UART asr(/*hardSerial=*/&Serial1);
#endif

void setup() {
  Serial.begin(115200);

  pinMode(Led, OUTPUT); //Init LED pin to output mode
  digitalWrite(Led, LOW); //Set LED pin to low

  // Init the sensor
  while (!asr.begin()) {
    Serial.println("Communication with device failed, please check connection");
    delay(3000);
  }
  Serial.println("Begin ok!");

  /**
 @brief Reset module
 */
  // asr.resetModule();

  /**
 @brief Set commands of the module
 @param setType - Set type
 */
}
```

```

    @n      DF2301Q_UART_MSG_CMD_SET_VOLUME: Set volume, the set value range 1-7
    @n      DF2301Q_UART_MSG_CMD_SET_ENTERWAKEUP: Enter wake-up state; set value
0
    @n      DF2301Q_UART_MSG_CMD_SET_MUTE Mute mode; set value 1: mute, 0:
unmute
    @n      DF2301Q_UART_MSG_CMD_SET_WAKE_TIME ; Wake-up duration; the set value
range 0-255s
    @param setValue - Set value, refer to the set type above for the range
    */
    asr.setti ngCMD(DF2301Q_UART_MSG_CMD_SET_MUTE, 0);
    asr.setti ngCMD(DF2301Q_UART_MSG_CMD_SET_VOLUME, 7);
    asr.setti ngCMD(DF2301Q_UART_MSG_CMD_SET_WAKE_TIME, 20);
    //asr.setti ngCMD(DF2301Q_UART_MSG_CMD_SET_ENTERWAKEUP, 0);

    /**
    @brief Play the corresponding reply audio according to the command word ID
    @param CMDID - Command word ID
    */
    asr.pl ayByCMDID(23);
}

void loop() {
    /**
    @brief Get the ID corresponding to the command word
    @return Return the obtained command word ID, returning 0 means no valid ID is
obtained
    */
    uint8_t CMDID = asr.getCMDID();
    swi tch (CMDID) {
        case 103:                                     //If the command is
"Turn on the light"
            digi talWri te(Led, HIGH);                 //Turn on the LED
            Serial.println("received'Turn on the light',command flag'103'"); //Serial
transmits "received'Turn on the light',command flag'103'"
            break;

        case 104:                                     //If the command is
"Turn off the light"
            digi talWri te(Led, LOW);                   //Turn off the LED
            Serial.println("received'Turn off the light',command flag'104'"); //The
serial transmits "received'Turn off the light',command flag'104'"
            break;

        default:
            if (CMDID != 0) {
                Serial.print("CMDID = "); //Print command ID
                Serial.println(CMDID);
            }
    }
    del ay(300);
}

```

Copy

Expected Result

Awaken the speech recognition module using a predefined or learned awakening phrase. Subsequently, command the illumination module with the phrases "Turn on the light" or "Turn off the light" for controlling its functionality.

The screenshot shows a serial terminal window titled 'COM33'. The text area contains the following log entries:

```
Begin Begin ok!
CMDID = 2
received'Turn on the light',command flag'103'
received'Turn off the light',command flag'104'
```

At the bottom of the window, there is a status bar with the following controls:

- ☒ 自动滚屏 (Auto scroll)
- ☐ Show timestamp
- 换行符 (Line feed) dropdown menu
- 115200 波特率 (Baud rate) dropdown menu
- 清空输出 (Clear output) button

Command Words/Wake-up Words & ID Table

Wake-up words				ID	
Wake-up words for learning				1	
Hello robot				2	
Commands for learning	ID	Commands for learning	ID	Commands for learning	ID
The first custom command	5	The second custom command	6	The third custom command	7
The fourth custom command	8	The fifth custom command	9	The sixth custom command	10

Commands for learning	ID	Commands for learning	ID	Commands for learning	ID
The seventh custom command	11	The eighth custom command	12	The ninth custom command	13
The tenth custom command	14	The eleventh custom command	15	The twelfth custom command	16
The thirteenth custom command	17	The fourteenth custom command	18	The fifteenth custom command	19
The sixteenth custom command	20	The seventeenth custom command	21		
Fixed Command Words	ID	Fixed Command Words	ID	Fixed Command Words	ID
Go forward	22	Retreat	23	Park a car	24
Turn left ninety degrees	25	Turn left forty-five degrees	26	Turn left thirty degrees	27
Turn right forty-five degrees	29	Turn right thirty degrees	30	Shift down a gear	31
Line tracking mode	32	Light tracking mode	33	Bluetooth mode	34
Obstacle avoidance mode	35	Face recognition	36	Object tracking	37
Object recognition	38	Line tracking	39	Color recognition	40

Fixed Command Words	ID	Fixed Command Words	ID	Fixed Command Words	ID
Tag recognition	41	Object sorting	42	Or code recognition	43
General settings	44	Clear screen	45	Learn once	46
Forget	47	Load model	48	Save model	49
Take photos and save them	50	Save and return	51	Display number zero	52
Display number one	53	Display number two	54	Display number three	55
Display number four	56	Display number five	57	Display number six	58
Display number seven	59	Display number eight	60	Display number nine	61
Display smiley face	62	Display crying face	63	Display heart	64
Turn off dot matrix	65	Read current posture	66	Read ambient light	67
Read compass	68	Read temperature	69	Read acceleration	70
Reading sound intensity	71	Calibrate electronic gyroscope	72	Turn on the camera	73

Fixed Command Words	ID	Fixed Command Words	ID	Fixed Command Words	ID
Turn off the camera	74	Turn on the fan	75	Turn off the fan	76
Turn fan speed to gear one	77	Turn fan speed to gear two	78	Turn fan speed to gear three	79
Start oscillating	80	Stop oscillating	81	Reset	82
Set servo to ten degrees	83	Set servo to thirty degrees	84	Set servo to forty-five degrees	85
Set servo to sixty degrees	86	Set servo to ninety degrees	87	Turn on the buzzer	88
Turn off the buzzer	89	Turn on the speaker	90	Turn off the speaker	91
Play music	92	Stop playing	93	The last track	94
The next track	95	Repeat this track	96	Volume up	97
Volume down	98	Change volume to maximum	99	Change volume to minimum	100
Change volume to medium	101	Play poem	102	Turn on the light	103
Turn off the light	104	Brighten the light	105	Dim the light	106

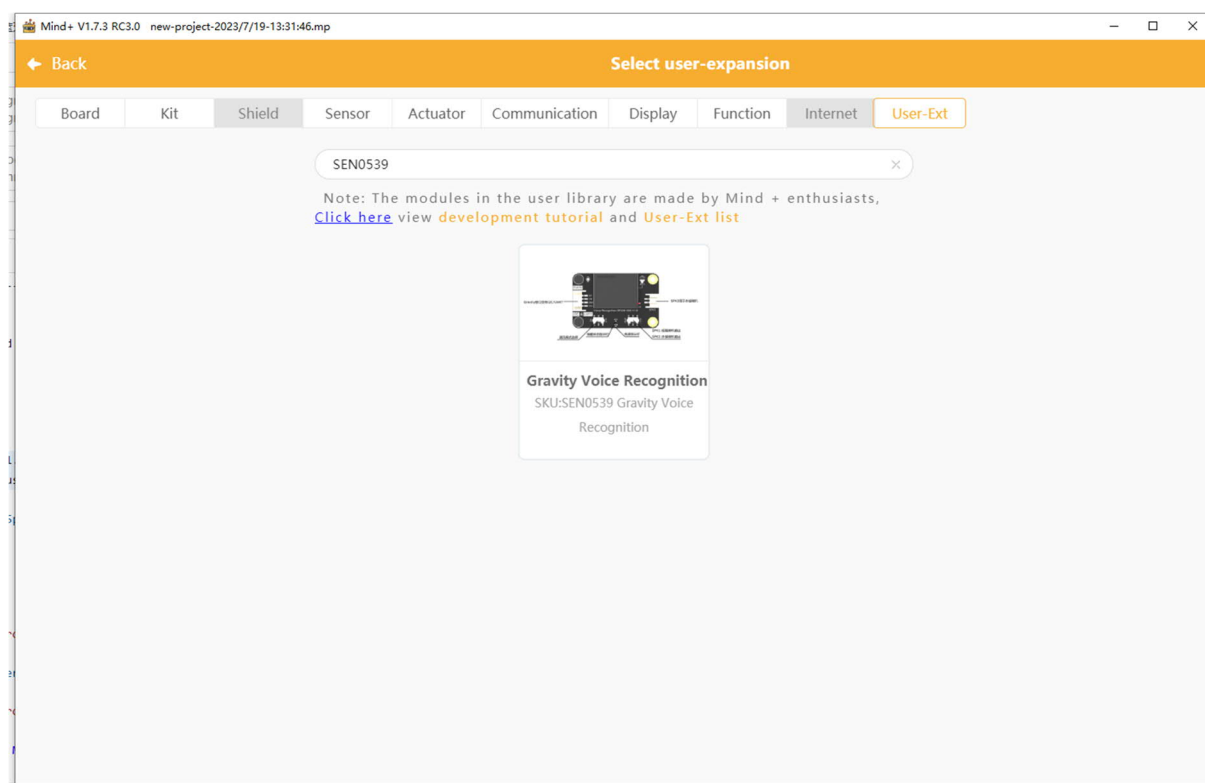
Fixed Command Words	ID	Fixed Command Words	ID	Fixed Command Words	ID
Adjust brightness to maximum	107	Adjust brightness to minimum	108	Increase color temperature	109
Decrease color temperature	110	Adjust color temperature to maximum	111	Adjust color temperature to minimum	112
Daylight mode	113	Moonlight mode	114	Color mode	115
Set to red	116	Set to orange	117	Set to yellow	118
Set to green	119	Set to cyan	120	Set to blue	121
Set to purple	122	Set to white	123	Turn on ac	124
Turn off ac	125	Increase temperature	126	Decrease temperature	127
Cool mode	128	Heat mode	129	Auto mode	130
Dry mode	131	Fan mode	132	Enable blowing up & down	133
Disable blowing up & down	134	Enable blowing right & left	135	Disable blowing right & left	136
Open the window	137	Close the window	138	Open curtain	139
Close curtain	140	Open the door	141	Close the door	142

Learning-related commands	ID	Learning-related commands	ID	Learning-re
Learning wake word	200	Learning command word	201	Re-learn
Exit learning	203	I want to delete	204	Delete wake
Delete command word	206	Exit deleting	207	Delete all

Mind+ User Guide

Explanation

Utilizing Mind+ version 1.7.3 or a more recent iteration, kindly navigate to the **Extensions** section and conduct a search within the user library for **SEN0539**.



Example program

- Arduino Uno mode:

The image shows a Scratch script for an Arduino Uno. The script is as follows:

```

    [Uno starts]
    serial output [ ] in string , Wrap
    Voice Recognition setup I2C mode address 0x64
    set volume 8
    set mute mode 0
    set wake time 20
    serial output [get wake time] in string , Wrap
    play word 2 ID
    serial output [ ] in string , Wrap
    forever loop:
        identify once and save the results
        if recognize it? then
            serial output [get the result] in string , Wrap
            if [get the result = word 103 ID] then
                digital pin 13 output HIGH
            if [get the result = word 104 ID] then
                digital pin 13 output LOW
    
```

Two callout boxes provide additional information:

- Serial transmits "received"Turn on the light",command flag"103""
- The serial transmits "received"Turn off the light",command flag"104""

- Python mode Unihiker



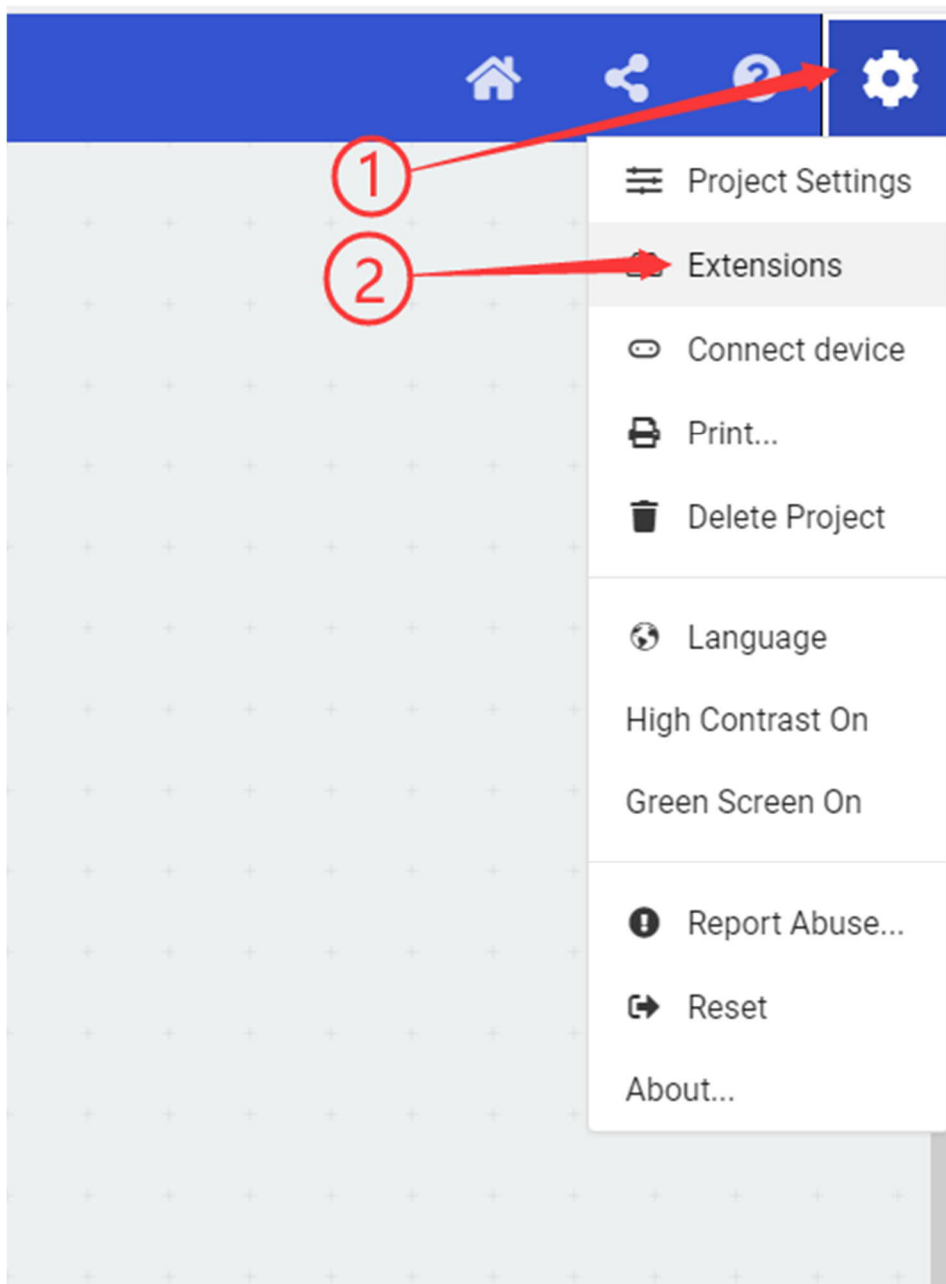
Tutorial on using MakeCode modules.

Introduction to MakeCode

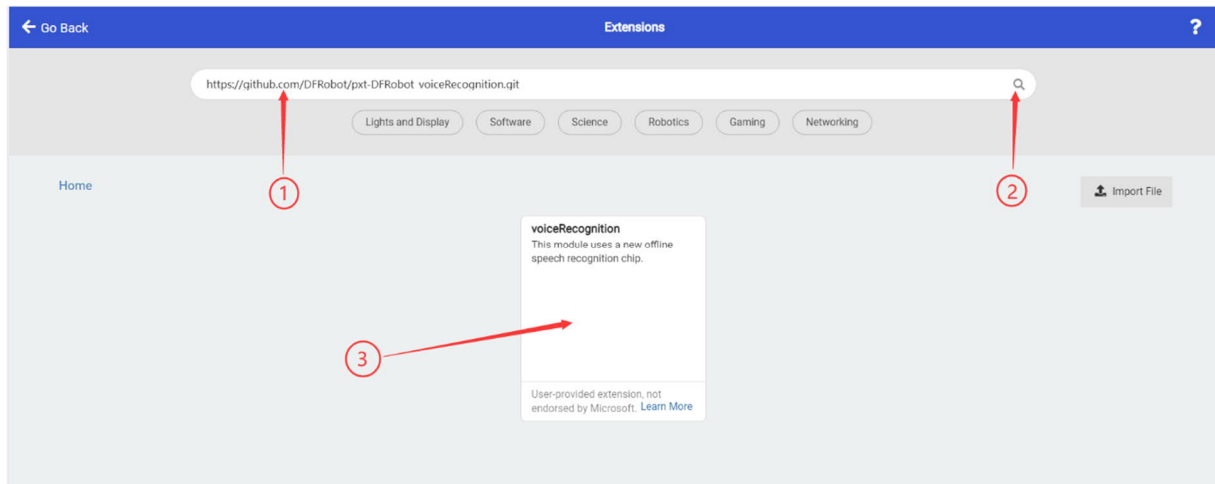
MakeCode is a free and open-source platform designed to create an engaging learning experience for computer science, as well as lay the foundation for actual programming. The web version allows for online programming and firmware downloads. [Click here to access the](#) MakeCode web version specifically designed for micro:bit.

Loading the voice recognition plugin on MakeCode

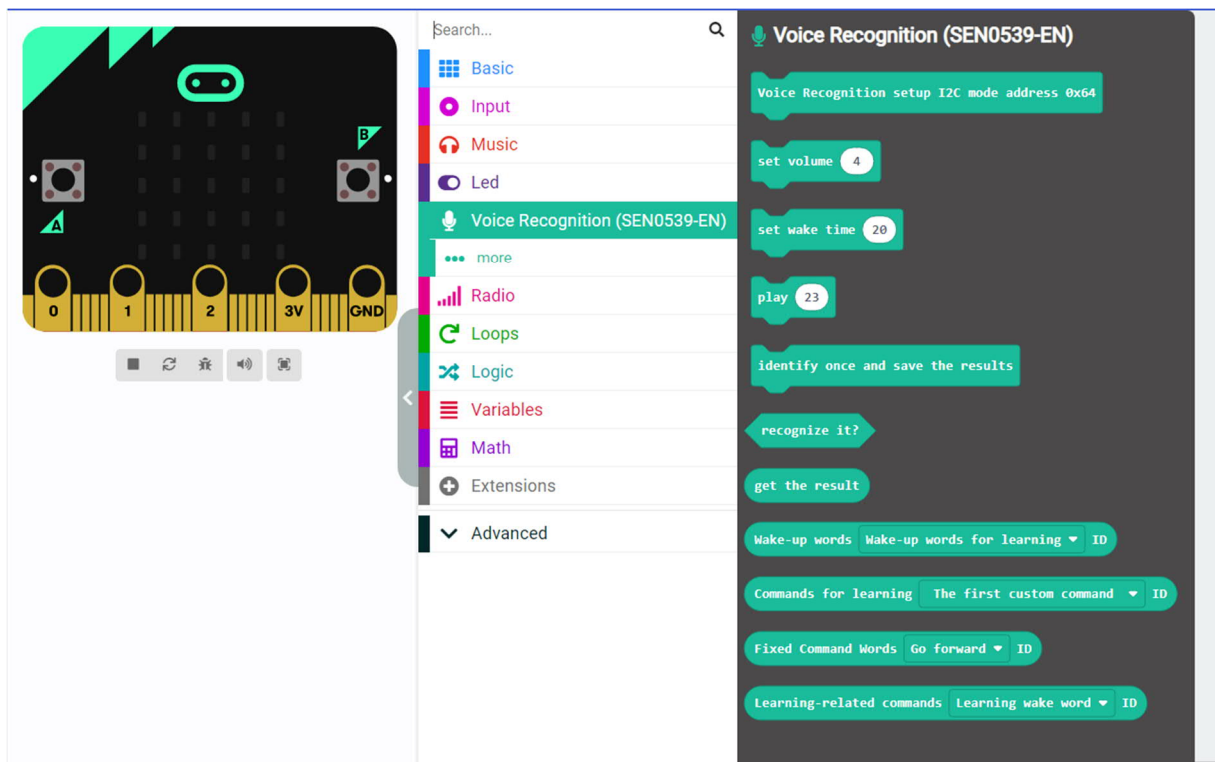
1. To create a new project in [MakeCode web version](#), click on the "more" button (gear icon) in the top right corner. From the drop-down menu, select "Extensions" to open the extension interface.



2. Type "https://github.com/DFRobot/pxt-DFRobot_voiceRecognition.git" in the search bar and click on the magnifying glass icon to search. You will see the voiceRecognition plugin (as shown in the image below). Click on it to load the plugin into your project.



3. In the programming interface, you will see the Voice Recognition (SEN0539) module. Clicking on it will bring up the command blocks.



MakeCode Example: Controlling the Micro:bit LED Matrix

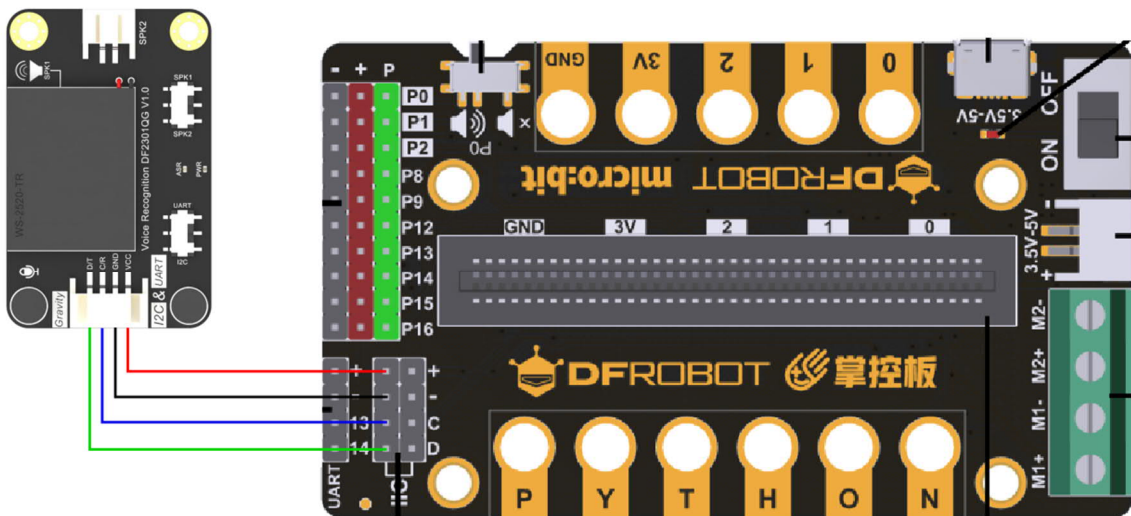
This example shows how to connect the Voice Recognition module to the Micro:bit board, and then read speech recognition results from the Voice Recognition module. The Micro:bit controls the LED matrix to display smiley faces, sad faces, and hearts. The communication interface between the Voice Recognition (SEN0539) and the Micro:bit uses the I2C interface.

Materials Needed

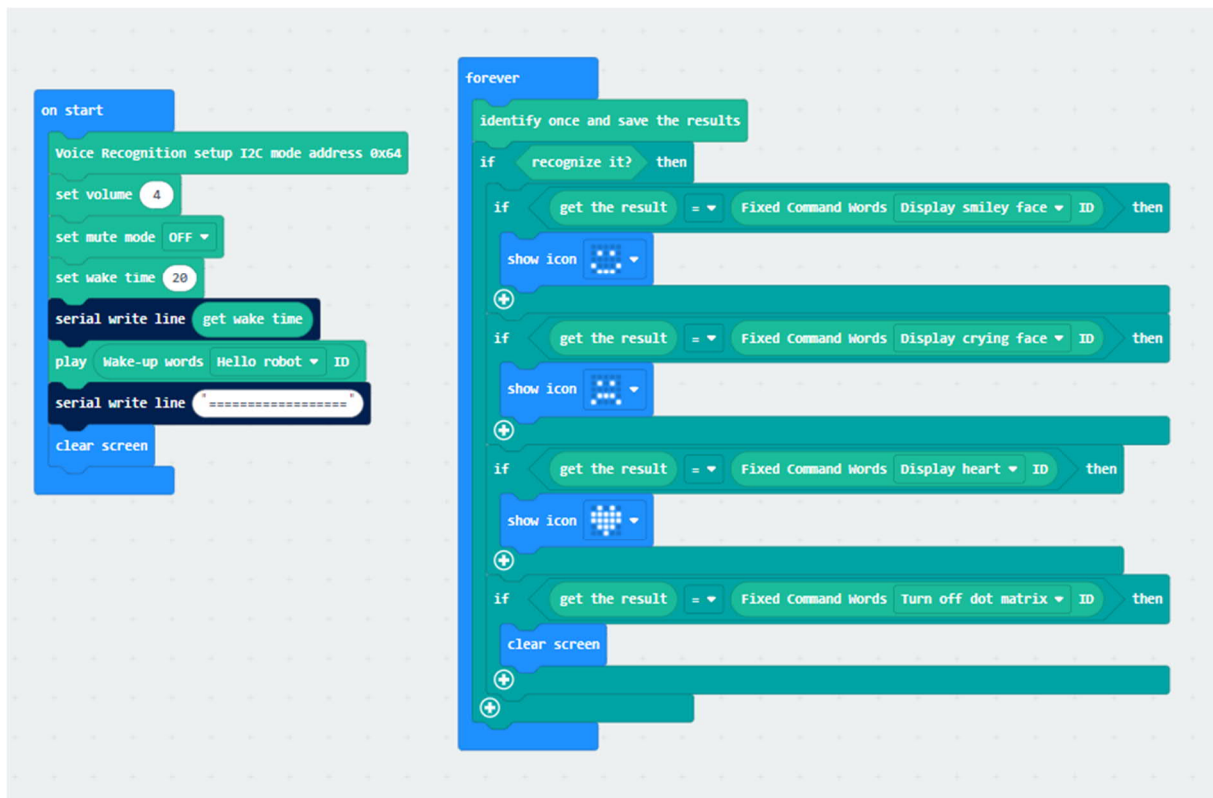
- Hardware
 - [micro:bit V2](#) x 1
 - [Micro:bit IO Expansion Board](#) x1
 - [Gravity:Voice Recognition Module - I2C & UART](#) x 1
 - 4-pin connection cable (or Dupont cable)
- 软件
 - [Microsoft MakeCode for micro:bit](#)
 - [Voice Recognition plugin](#)

Hardware wiring diagram

Properly connect the micro:bit motherboard and IO expansion board, and set the communication mode selection switch of the voice recognition module to I2C end according to the diagram below, then connect it to the expansion board.

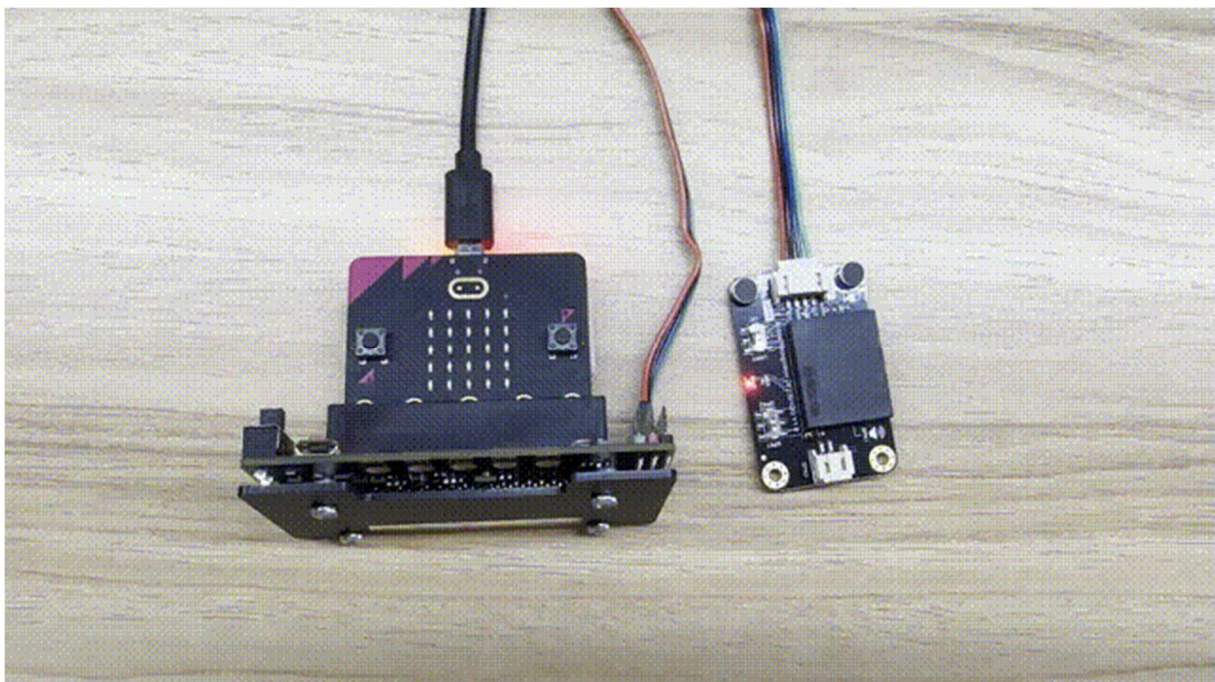


Sample code



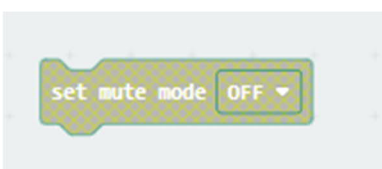
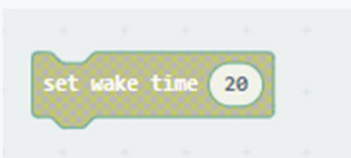
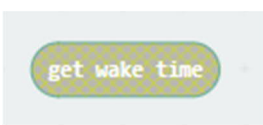
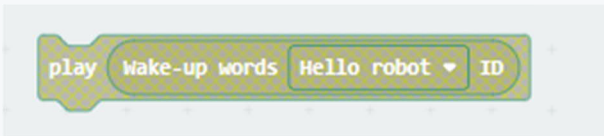
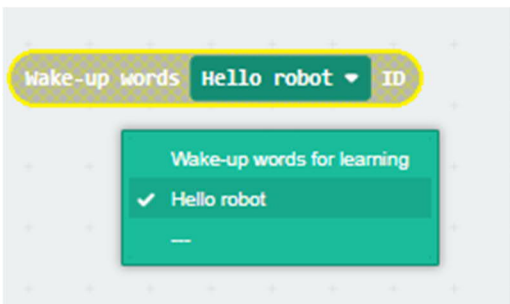




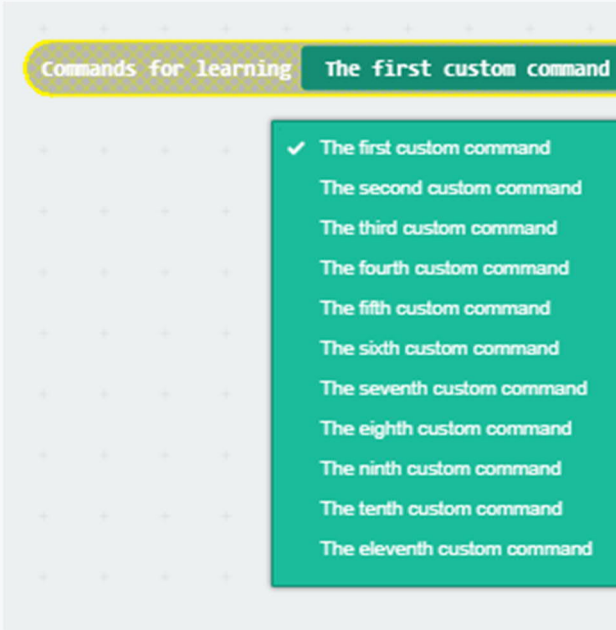
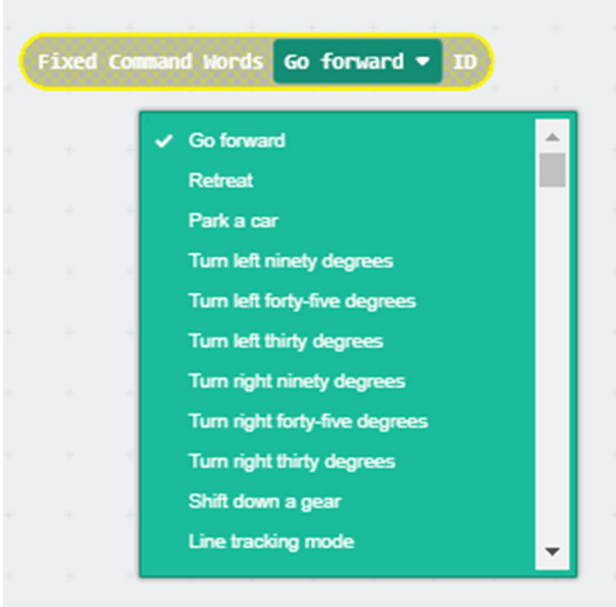
Expected result

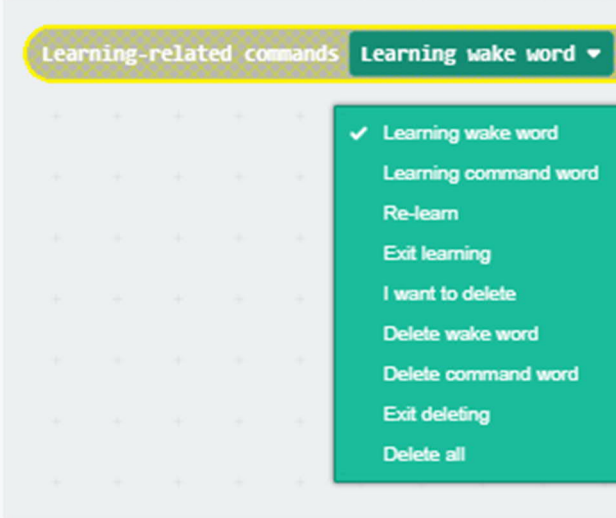
To wake up the voice recognition module, say either the fixed wake-up word or the learned wake-up word. To display a smiley face on the LED matrix of the micro:bit, say "**Display smiley face**". To display a sad face on the micro:bit, say "**Display crying face**". To display a heart on the micro:bit, say "**Display heart**". To turn off all LED matrix displays on the micro:bit, say "**Turn off dot matrix**".



MakeCode模块说明

Module	Explanation
	Initialization. It only needs to be executed once and should be placed at the beginning of the main program. The communication mode selection switch for the speech recognition module needs to be set to I2C end.
	Adjust the volume of the module. The range of the volume is from 0 to 7, with a larger number representing a higher volume.
	Enable or disable the mute mode of the module.
	Configure the time in seconds for the module to return to sleep mode after awakening. This time will be refreshed after each successful command recognition.
	Retrieve the duration in seconds for the module to return to sleep mode after awakening.
	Play the corresponding response for the recognized command word.
	Select the Wake-up Word ID.

Module	Explanation
	<p>Retrieve the recognition result from the speech recognition and save it to the recognition result module.</p>
	<p>Determine if the speech recognition has recognized the command.</p>
	<p>Select the ID of the command word to be learned.</p>
	<p>Select a fixed command word ID.</p>

Module	Explanation
	Select the ID of the word to be learned.

FAQ
