Mi:Node Kit User Manual

Make creative ‘things’ with micro:bit

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>V0.1.0</td>
<td>March 2017 by Paul</td>
<td>Initial version</td>
</tr>
</tbody>
</table>
Mi:Node Kit User Manual

Introduction

Features

What is the Kit?

What is it include?

How it works

E-brick Connector

E-brick Connector - Analog In/PWM Type

E-brick Connector - Digital IO Type

E-brick Connector - IIC Type

Conversation on Remaping Micro:bit Edge Pin

How to use it

Programming with the kit

Work with our library

Steps

Work without our library

Modules Usage & API Reference

Light Sensor

Block API

Get the light level

Parameters

Light Sensor event

Parameters

Example

Show the light level

Light level change event

DHT11 (Temperature and Humidity Sensor)

Block API

Get the temperature

Parameters

Get humidity

Parameters

Temperature change event

Parameters

Example

Use button to get the temperature and humidity

Temperature change event

Sound Sensor

Block API

Get sound level

Parameters

Sound Sensor event

Parameters

Example
Show the sound level
Sound level change event

Rotary Module
Block API
Get the percentage
Parameters
Light Sensor event
Parameters

Example
Show the percentage
Rotary control the LED show
Rotary change event

Mini Fan Module
Block API
Control the motor speed
Parameters

Example
Use button to control the motor's speed.

Speaker Module
Example
Let your speaker make a sound

PIR Module
Block API
Check the PIR status
Parameters
PIR event
Parameters

Example
Detecting the movement

RGB LED
Block API
Choose a RGB color
Parameters
Set a specific RGB color by setting rgb gray value
Parameters

Example
Use button to show different color.
Set specific color

Switch Module
Block API
Check the switch's status
Parameters
Switch event
Parameters

Example
Switch control the screen show
Show the switch module's status

Relay Module
Block API
Set relay status(open/close)
Parameters

Example
Control the relay by button

Appendix
Microsoft PXT
Support
Introduction

The Mi:Node kit is a modular, safe and easy to use group of items that allow kids to minimize the effort required to get started with IOT learning with Micro:bit.

For this kit, there is no need for soldering, plug then use, the construction of the working circuit can be done within one minute.

Features

- It is modular, safe and ready-to-use just like Lego
- 10 sensor modules include environmental monitoring, user interface and physical monitoring can be built a lot of cool projects include wearable device and smart home, etc.
- It is expandable and reusable
- Rich education guide document and many project stories

What is the Kit?

What is it include?

<table>
<thead>
<tr>
<th>Category</th>
<th>Module</th>
<th>Quantity</th>
<th>Conector Type</th>
<th>Description</th>
</tr>
</thead>
</table>

![Image of Mi:Node kit and Micro:bit](image-url)
<table>
<thead>
<tr>
<th>Component</th>
<th>Quantity</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro:bit</td>
<td>0</td>
<td>NA</td>
<td>The Micro:bit is not included in the kit</td>
</tr>
<tr>
<td>Connect Board</td>
<td>1</td>
<td>NA</td>
<td>This is a bridge between the micro:bit and sensor modules</td>
</tr>
<tr>
<td>Sensor Module (10)</td>
<td>1</td>
<td>Analog Input</td>
<td>It can be used to detect the intensity of light in the environment</td>
</tr>
<tr>
<td>Light Sensor</td>
<td>1</td>
<td>Analog Input</td>
<td></td>
</tr>
<tr>
<td>Temperature and Humidity Sensor</td>
<td>1</td>
<td>Analog Input</td>
<td>It can get temperature and humidity in the environment</td>
</tr>
<tr>
<td>Sound Sensor</td>
<td>1</td>
<td>Analog Input</td>
<td>It can detect the sound strength of the environment</td>
</tr>
<tr>
<td>Rotary Angle</td>
<td>1</td>
<td>Analog Input</td>
<td></td>
</tr>
<tr>
<td>Mini Fan</td>
<td>1</td>
<td>Analog Output/PWM</td>
<td>A DC motor + orbit fan</td>
</tr>
<tr>
<td>Speaker</td>
<td>1</td>
<td>Analog Output/PWM</td>
<td>Voice output speaker</td>
</tr>
<tr>
<td>PIR Motion Sensor</td>
<td>1</td>
<td>Digital Input</td>
<td>It allows the user to sense motion, usually human movement in its range</td>
</tr>
<tr>
<td>RGB LED</td>
<td>1</td>
<td>Digital Output</td>
<td>A colorful LED. The color and brightness can be programmable</td>
</tr>
<tr>
<td>Switch</td>
<td>1</td>
<td>Digital Input</td>
<td>It can be used to switch ON/OFF</td>
</tr>
<tr>
<td>Relay</td>
<td>1</td>
<td>Digital Output</td>
<td>It is a digital switch can be used to control high-voltage electrical devices. (maximum 250V)</td>
</tr>
<tr>
<td>Cables</td>
<td>8</td>
<td>NA</td>
<td>20cm cable x 2, 10cm cable x 6</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>NA</td>
<td>1 for power input, 1 for micro:bit program upload</td>
</tr>
</tbody>
</table>
How it works

The kit is composed of a connect board and several sensor modules. The connect board is a bridge between the micro:bit main board and the sensor modules. It converts the micro:bit edge connector into several E-brick connector. The sensor modules can be attached to it by cable.

![E-brick Connector](image)

E-brick Connector

The E-brick connectors are compatible with the standard Grove from Seeed. It's a 2.0mm 1x4 connector. With signals include: 1 VCC, 1GND, 2 analog / digital signal.

An E-brick connector have 4 pins:

<table>
<thead>
<tr>
<th>Pin ID</th>
<th>Pin Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin#1</td>
<td>Signal#1</td>
<td>Connect to a micro:bit pin with analog in / digital io / i2c function</td>
</tr>
<tr>
<td>Pin#2</td>
<td>Signal#2</td>
<td>Same to Signal#1. The Signal#2 is often not used, because 1 signal pin is enough for most sensor modules</td>
</tr>
<tr>
<td>Pin#3</td>
<td>VCC</td>
<td>Power pin</td>
</tr>
</tbody>
</table>
Pin#4  GND      Power pin

There are 3 types of the connector:

- 3 analog input connectors
- 3 I2C connectors
- 4 digital IO connectors

**E-brick Connector - Analog In/PWM Type**

<table>
<thead>
<tr>
<th>Pin ID</th>
<th>Pin Name</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin#1</td>
<td>A1</td>
<td>Connect to a micro:bit pin with analog input/PWM function</td>
</tr>
<tr>
<td>Pin#2</td>
<td>A2</td>
<td>Same to A1, but most sensor module may not use the A2 pin</td>
</tr>
</tbody>
</table>

**E-brick Connector - Digital IO Type**

<table>
<thead>
<tr>
<th>Pin ID</th>
<th>Pin Name</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin#1</td>
<td>D1</td>
<td>Connect to a micro:bit pin with digital io function</td>
</tr>
<tr>
<td>Pin#2</td>
<td>D2</td>
<td>Same to D1, but most sensor module may not use the D2 pin</td>
</tr>
</tbody>
</table>

**E-brick Connector - IIC Type**

IIC (Inter-Integrated Circuit), pronounced I-squared-C, is a multi-master, multi-slave, single-ended, serial computer bus invented by Philips Semiconductor (now NXP Semiconductors). It is typically used for attaching lower-speed peripheral to microcontrollers in short-distance, intra-board communication.

There are some sensor modules based on the IIC bus. We can attach several modules on the same bus, because they can be identified by different addresses.

<table>
<thead>
<tr>
<th>Pin ID</th>
<th>Pin Name</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin#1</td>
<td>IIC SCL</td>
<td>IIC clock signal. Connect to micro:bit pin19</td>
</tr>
<tr>
<td>Pin#2</td>
<td>IIC SDA</td>
<td>IIC data signal. Connect to micro:bit pin20</td>
</tr>
</tbody>
</table>

**Conversation on Remaping Micro:bit Edge Pin**

Please note the Connector Name. There is a convention in the connector name. A connector name can be indicated the mapped micro:bit signal name.

For example: Connector D12 means: The Signal#1 is remapped to micro:bit pin 12, and Signal#2 is remapped to micro:bit pin 13(12+1)

It's useful for us to position the micro:bit pin quickly when develop code.

<table>
<thead>
<tr>
<th>Connector Type</th>
<th>Connector Name</th>
<th>Micro:bit Pin Name(Signal#1, Signal#2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog Input /</td>
<td>A0</td>
<td>pin0, pin1</td>
</tr>
<tr>
<td>Digital IO</td>
<td>A1</td>
<td>pin1, pin2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>A2</td>
<td>pin2, pin3</td>
<td></td>
</tr>
<tr>
<td>Digital IO</td>
<td>D12</td>
<td>pin12, pin13</td>
</tr>
<tr>
<td></td>
<td>D13</td>
<td>pin13, pin14</td>
</tr>
<tr>
<td></td>
<td>D14</td>
<td>pin14, pin15</td>
</tr>
<tr>
<td></td>
<td>D15</td>
<td>pin15, pin16</td>
</tr>
<tr>
<td>IIC</td>
<td>IIC</td>
<td>pin19, pin20</td>
</tr>
<tr>
<td></td>
<td>IIC</td>
<td>pin19, pin20</td>
</tr>
<tr>
<td></td>
<td>IIC</td>
<td>pin19, pin20</td>
</tr>
</tbody>
</table>

- We usually use A0, A1, A2 as a analog input connector, but it can also be used as a digital io connector.
- D12, D13, D14, D15 can be only used as digital IO connector.
- 3 IIC can be only used as IIC connector.

**How to use it**

Just plug then use it, That’s easy.
**Programming with the kit**

As you know, there are 5 editors we can choose on micro:bit official website.

- Microsoft PXT
- Code Kingdoms JavaScript
- Microsoft Block Editor
- Microsoft Touch Develop
- python

We'll focus more on the **Microsoft PXT**.

**Work with our library**

We'll develop the libraries for the sensor modules. This can reduce your effort on using the modules. You don't have to study the technical details of the modules. For example, the RGB LED is based on the IIC bus, we just need to call the library function to control the color and brightness. We don’t care the IIC bus itself.

With the library, you don't need to care which pin a module connects to. You just need to know what E-brick connector is used.

**NOTE**

The library is based on Microsoft PXT.

**Steps**

- Step 1: Add our library to your code
- Step 2: Refer to the API reference or example code
- Step 3: Drag the library blocks.
Work without our library

You can also use the micro:bit pin library to control the sensor modules directly. In this situation, we can get the micro:bit pin ID by the E-brick connector ID. For example, we connect a sensor module to D12. Then we can get that the corresponding Micro:bit pin is pin12.

Modules Usage & API Reference

Light Sensor

The Light Sensor module can be used to detect the intensity of light in the environment. We divide the brightness into 5 levels. This module can only be plugged into Analog connector (A0, A1 and A2).
<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Light Sensor</td>
<td>Analog</td>
<td>A0, A1, A2</td>
</tr>
</tbody>
</table>

**Block API**

**Get the light level**

Get current light level, we divide the light intensity into 5 levels, from 1 to 5. Which 1 represents brightest and 5 represents darkness.

```javascript
function LightSensorGetLevel(connName: AnalogConnName): number;
```

**Parameters**

- `connName` is the analog connector's name. This module can only be plugged into analog connector A0, A1, and A2.
Light Sensor event

Configure the MCU check the light level periodically, and then execute the associated code block whenever the light level changes.

```javascript
function onLightSensorEvent(connName: AnalogConnName, body: () => void): void;
```

Parameters

- `connName` is the analog connector's name. This module can only be plugged into analog connector A0, A1 and A2.

Example

Show the light level

This example show you how to get the current light level and show it on the LED screen.

Light level change event

When the light level changes there will show a string on the screen.
DHT11 (Temperature and Humidity Sensor)

This DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. It can get temperature and humidity in the environment.

<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>DHT11</td>
<td>Digital IO</td>
<td>D12, D13, D14, D15</td>
</tr>
</tbody>
</table>

**TODO**

Need hardware engineers to confirm this module's specific parameters.

<table>
<thead>
<tr>
<th>Module</th>
<th>Measurement Range</th>
<th>Humidity Accuracy</th>
<th>Temperature Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DHT11</td>
<td>20-90%RH / 0-50°C</td>
<td>±5%RH</td>
<td>±2°C</td>
</tr>
</tbody>
</table>
Block API

Get the temperature

Get current temperature, you can configure the format of the temperature in Celsius or Fahrenheit.

```javascript
function DHTGetTemperature(connName: ConnName, style: DHTTemStyle): number;
```

**Parameters**
- **connName** is the connector's name. This module can be plugged into both analog connector and digital connector.
- **style** is the format of the temperature. You can choose Celsius or Fahrenheit.

Get humidity

Get current humidity.

```javascript
function DHTGetHumidity(connName: ConnName): number;
```

**Parameters**
- **connName** is the connector's name. This module can be plugged into both analog connector and digital connector.

Temperature change event

Configure the MCU check the temperature periodically, and then execute the associated code block whenever the temperature changes. The smallest unit of changing is 1 degree Celsius.

```javascript
function onDHTEvent(connName: ConnName, body: () => void): void;
```
Parameters

- **connName** is the analog connector's name. This module can be plugged into both analog connector and digital connector.

Example

**Use button to get the temperature and humidity**

This example shows you how to use the button A and B to get the temperature and humidity. When you press the button A, the screen will show the current temperature, if the button is B it will show the humidity.

**Temperature change event**

When the temperature changes the screen will show a smile face!
Sound Sensor

The Sound Sensor Module can be used to detect the sound strength of the environment. We divide the sound into 5 levels, you can use our block to get the current level. This module can only be plugged into analog connector (A0, A1, and A2).

<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sound Sensor</td>
<td>Analog</td>
<td>A0, A1, A2</td>
</tr>
</tbody>
</table>

**Block API**

**Get sound level**

Get current sound level, we divide the sound into 5 levels, range from 1 to 5. Which 1 represent quiet and 5 represent noisy.
function MICGetLevel(connName: AnalogConnName): number;

**Parameters**
- **connName** is the analog connector's name. This module can only be plugged into analog connector A0, A1 and A2.

**Sound Sensor event**

Configure the mcu check the sound level periodically, and then execute the associated code block whenever the sound level changes.

function onMICEvent(connName: AnalogConnName, body: () => void): void;

**Parameters**
- **connName** is the analog connector's name. This module can only be plugged into analog connector A0, A1 and A2.

**Example**

**Show the sound level**

This example show you how to get the current sound level, and show the level number on the LED screen.

**Sound level change event**

When the sound level changes a string will show on the screen.
Rotary Module

The Rotary can produce analog output between 0 and Vcc by adjusting the angular range from 0 ~ 300. This module can only be plugged into analog connectors (A0, A1, A2).

<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotary Sensor</td>
<td>Analog</td>
<td>A0, A1, A2</td>
</tr>
</tbody>
</table>

Block API

Get the percentage

Get the current rotary percentage. This value means how much you have rotated the rotary module.
function RotaryGetPercentage(connName: AnalogConnName): number;

Parameters
- **connName** is the analog connector's name. This module can only be plugged into analog connector A0, A1, and A2.

**Light Sensor event**

Configure the mcu to check the rotary AD value periodically, and then execute the associated code block whenever the AD value changes.

function onRotaryEvent(connName: AnalogConnName, body: () => void): void;

Parameters
- **connName** is the analog connector's name. This module can only be plugged into analog connector A0, A1, and A2.

**Example**

**Show the percentage**

This example shows you how to get the current percentage and show it on the LED screen.
Rotary control the LED show

When the percentage is smaller than 50 the screen will show nothing, otherwise there will be a smile face on the screen.

Rotary change event

When you rotating the potentiometer's knob, the LED screen will show a smile face.
Mini Fan Module

The mini fan module contains a DC motor and orbit fan. We can control the speed of the motor according to different situations.

<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mini Fan Module</td>
<td>Analog</td>
<td>A0, A1, A2</td>
</tr>
</tbody>
</table>

Block API

Control the motor speed

We use this block to control the motor's speed. You can change the speed by adjusting the second parameter, range from 0 to 100.

```javascript
function FanControl_1(connName: AnalogConnName, speed: number): void
```
Parameters

- **connName** is the analog connector's name. This module can only be plugged into analog connector and digital connector.

- **speed** is the speed of the motor. The adjustment range is from 0 to 100. Which 0 means the motor is shutting down and 100 means the fastest speed.

Example

**Use button to control the motor's speed.**

This example show you how to use the button A and B control the motor speed. Button A control the motor to speed up, button B control the motor to deceleration.
**Speaker Module**

The Speaker can be used to make a sound by using the Music blocks.

**NOTE**

As the default music is speaking through pin P0, so we need to connect our speaker module to connector A0.
<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speaker Module</td>
<td>Analog</td>
<td>A0</td>
</tr>
</tbody>
</table>

**Example**

*Let your speaker make a sound*

The speaker will circle the sound of the three beat.
**PIR Module**

The PIR Motion Sensors allows user to sense motion, usually human movement in its range. When this module detects that there is an object moving, the PIR signal line will jump from the low level to the high level, and keep the high level for 3 seconds.
<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIR Module</td>
<td>Digital IO</td>
<td>D12, D13, D14, D15</td>
</tr>
</tbody>
</table>

**Block API**

**Check the PIR status**

Check whether the switch is triggered or not. when the pir module is triggered the status signal which was detected will be in high level.

```
pir A0 ▼ is triggered
```

function PIRIsTriggered(connName: ConnName): boolean;

**Parameters**

- `connName` is the connector's name. this module can be plugged into both analog connector and digital connector.
**PIR event**

Configure the specified pin for digital input, and then execute the associated code block whenever the pin in rising edge which means the PIR has detected movement.

Function: `function onPIREvent(connName: ConnName, body: () => void): void;`

**Parameters**
- `connName` is the connector's name. This module can be plugged into both analog connector and digital connector.

**Example**

**Detecting the movement**

When the PIR detected the moving objects, the screen will show a smile face. When there are no moving things in its range, the screen will show nothing.
**RGB LED**

A colorful LED. The color and brightness can be programable. And the color and brightness is controled by the gray value of red, green and blue.

<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB LED</td>
<td>Digital IO</td>
<td>D12, D13, D14, D15</td>
</tr>
</tbody>
</table>

**TODO** [HW Team] Need hardware engineers to add this module's specific parameters.

**Block API**

**Choose a RGB color**

You can choose a given color to show.

```function RGBChooseColor(connName: ConnName, color: MiNodeColor): void;```

**Parameters**

- **connName** is the analog connector's name. This module can be plugged into both analog connector and digital connector.

- **color** is set of colors. That is red, green, blue, yellow, pink, cyan and white. You can choose one color from the given color to light up your RGB module.

**Set a specific RGB color by setting rgb gray value**

You can change the color of the RGB by setting the three gray values.
function RGBSetColor(connName: ConnName, red: number, green: number, blue: number): void;

Parameters

- **connName** is the analog connector's name. This module can be plugged into both analog connector and digital connector.
- **red** is the gray value of red, range from 0 to 255.
- **green** is the gray value of green, range from 0 to 255.
- **blue** is the gray value of blue, range from 0 to 255.

Example

Use button to show different color.

When you press button A the RGB module will show red, if you press button B the color will be green.

Set specific color

This example shows you how to set a specific color. There are two different colors controlled by button A and B. Press the two buttons, you will get the corresponding color.
Switch Module

The switch module can be used to switch ON/OFF.
<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch Module</td>
<td>Digital IO</td>
<td>D12, D13, D14, D15</td>
</tr>
</tbody>
</table>

**TODO** [HW Team] Need hardware engineers to add this module's specific parameters.

**Block API**

**Check the switch's status**

Check whether the switch is opened or not.

```plaintext
switch A0 is opened
```

Function: `switchIsOpened(connName: ConnName): boolean;`

**Parameters**

- `connName` is the connector's name. This module can be plugged into both analog connector and digital connector.

**Switch event**

Configure the specified pin for digital input, and then execute the associated code block whenever the switch is opened or closed.

```plaintext
switch A0 on open
```

Function: `onSwitchEvent(connName: ConnName, event: SwitchEvent, body: () => void): void;`

**Parameters**

- `connName` is the analog connector's name. This module can be plugged into both analog connector and digital connector.
- `event` represents two statuses of switch, open and close.

**Example**

**Switch control the screen show**

If the switch is opened, the screen will show a smile face. Otherwise, it will show nothing.
Show the switch module's status

In this example we use the switch event to show the switch's status. When the switch is opened, the led screen will show a string 'Open', otherwise the led screen will show 'Close'.
Relay Module

The relay is an electrically operated switch. It is a digital switch can be used to control high-voltage electrical devices, such as some home appliances. (maximum 250V)

Danger

It's dangerous for kids to attach the relay module to a AC(110V/220V) device. Our purpose for this module is just showing how to control home appliances. You can hear some sound when you switch the relay ON/OFF. You do not have to connect to a real appliance.
<table>
<thead>
<tr>
<th>Module</th>
<th>Connect Type</th>
<th>Available Connectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relay Module</td>
<td>Digital IO</td>
<td>D12, D13, D14, D15</td>
</tr>
</tbody>
</table>

**TODO [HW Team]** Need hardware engineers to add this module's specific parameters.

**Block API**

**Set relay status(open/close)**

Set the relay module's status open/close.

```
relay A0 set open
```

```typescript
function RelayControl(connName: ConnName, status: FanStatus): void
```

**Parameters**

- **connName** is the connector's name. This module can be plugged into both analog connector and digital connector.

- **status** is the status of the relay open or close.
Example

Control the relay by button

Button A and B control the open and close of the relay module.

Appendix

Microsoft PXT

- Home: https://pxt.microbit.org/
- Getting Started: https://pxt.microbit.org/getting-started

Support

- Email: support@embest-tech.com
- Website: http://minode.embest-tech.com
- Github repo: http://github.com/minodekit