Smarthon Documentation

Release 1.0

Smarthon Limited

Jun 15, 2022

CONTENTS

CHAPTER

ONE

TUTORIAL GUIDE

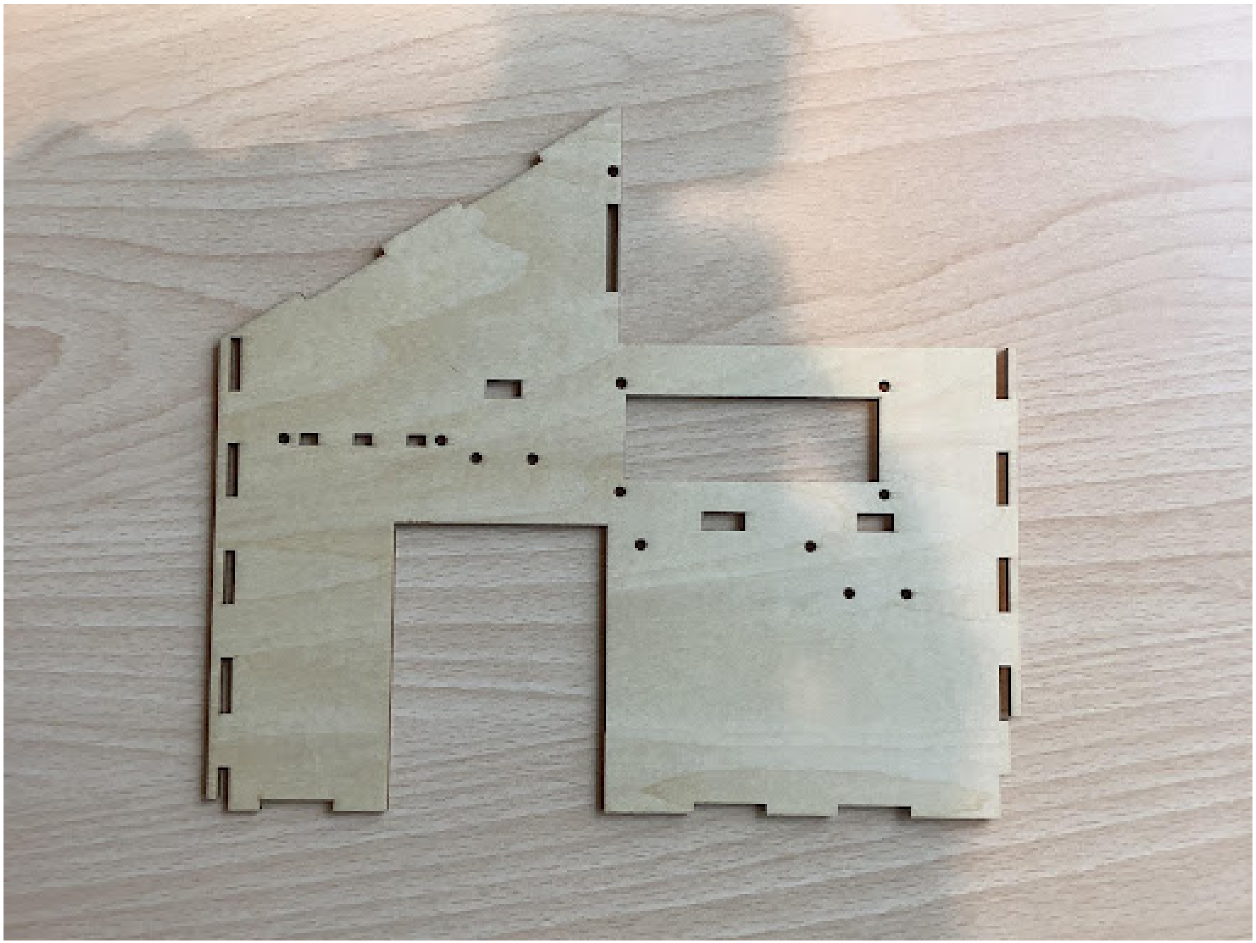
1.1 Smarthon Smart House Kit for micro:bit

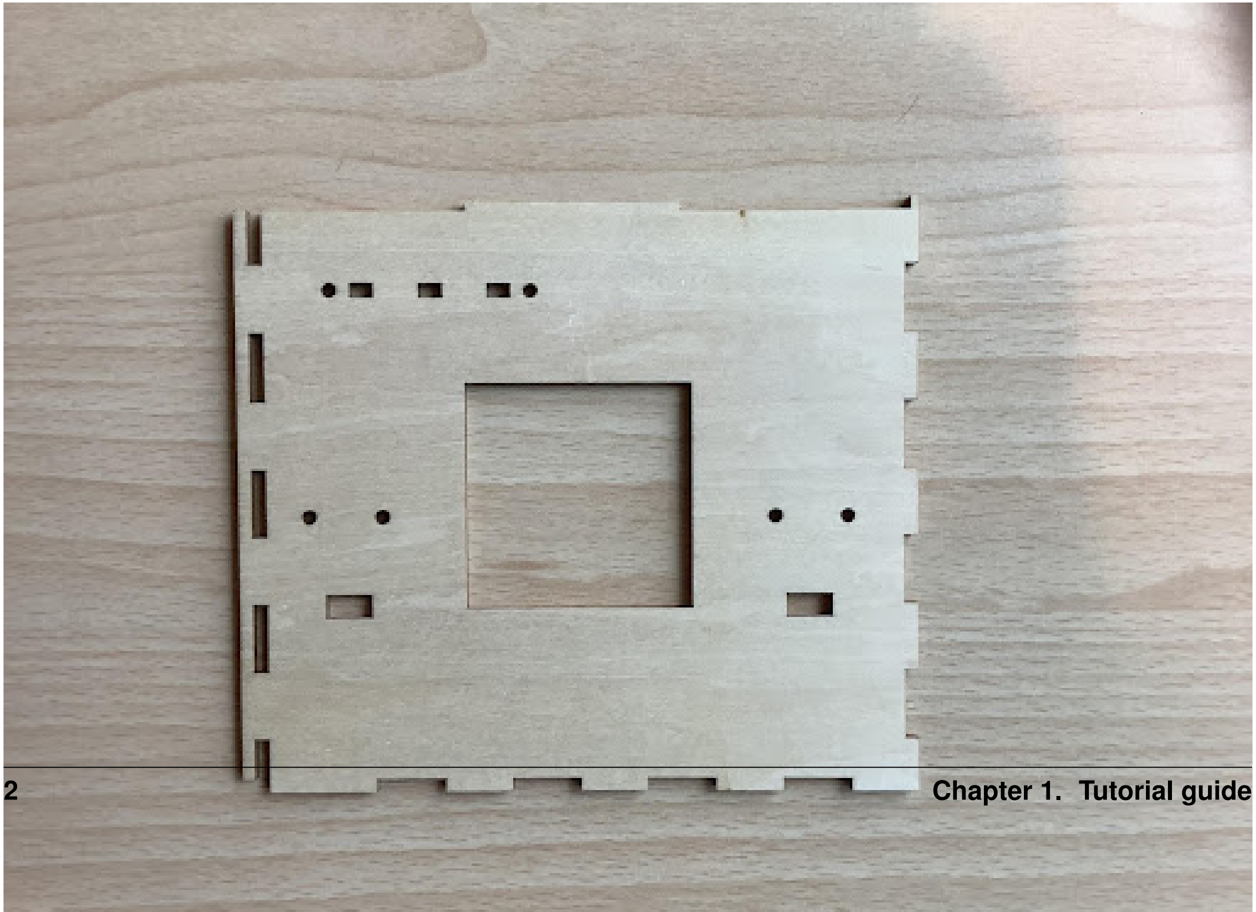
1.1.1 Chapter 1 Know More About Smart Home

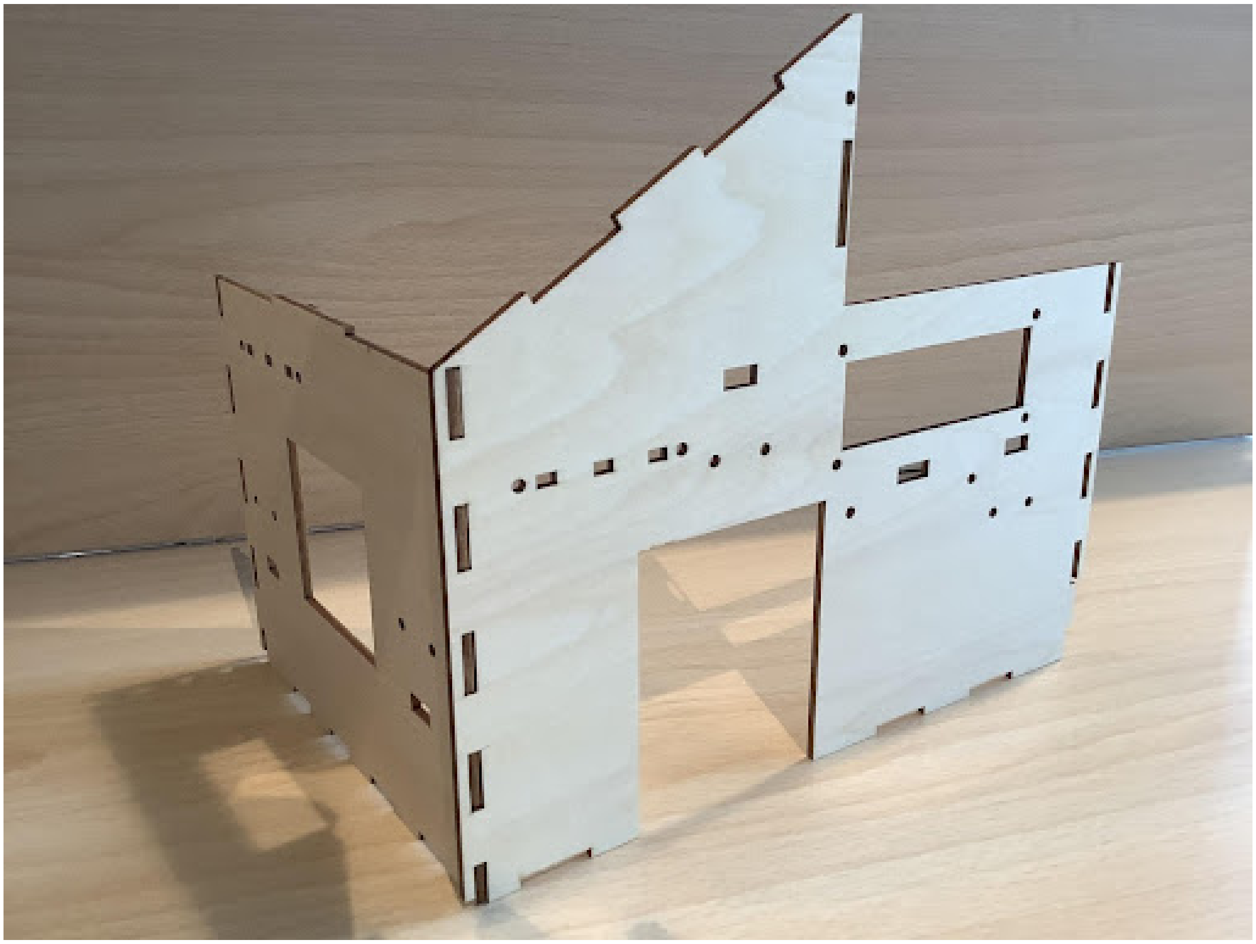
What is Smart Home

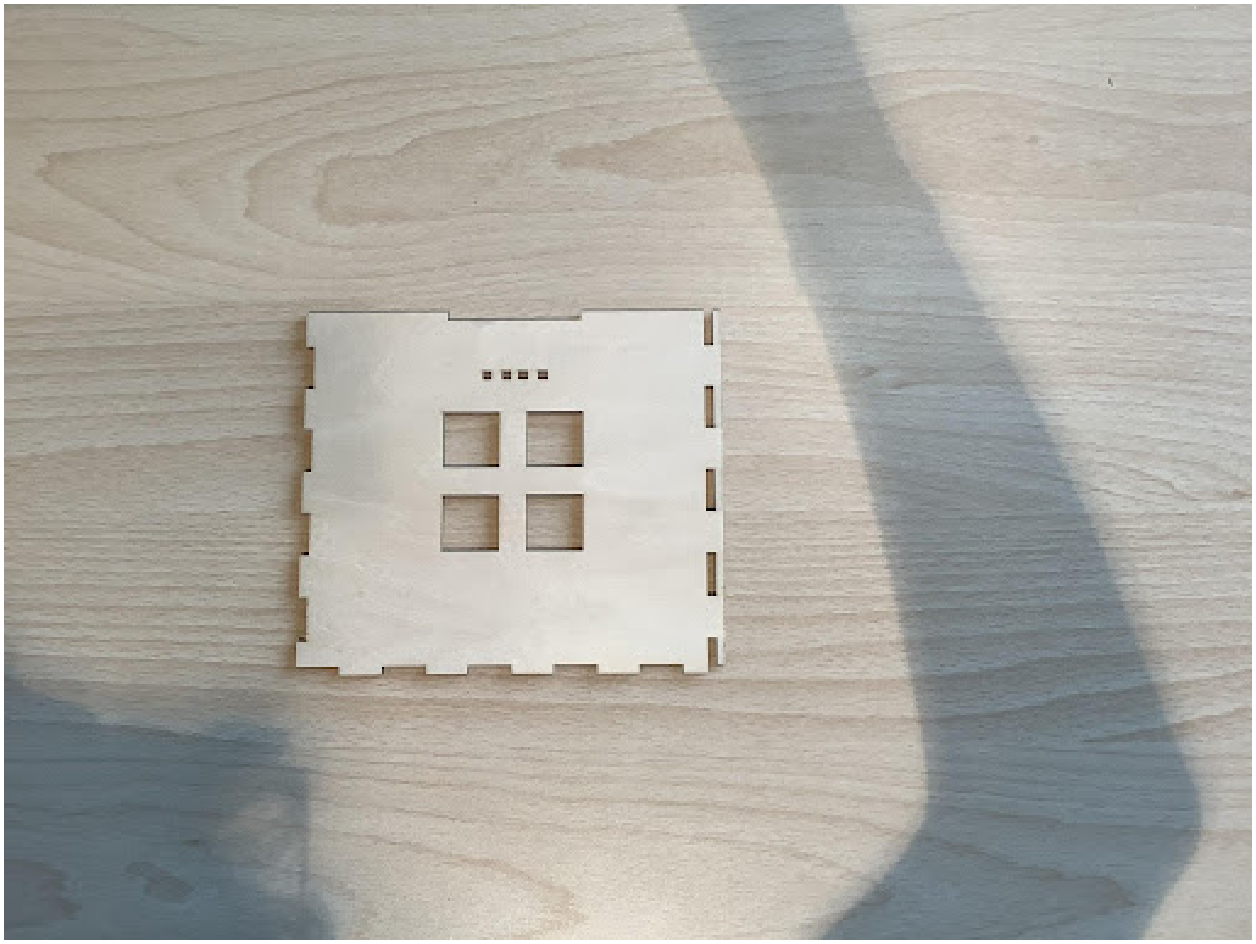
Advantages of Smart Home

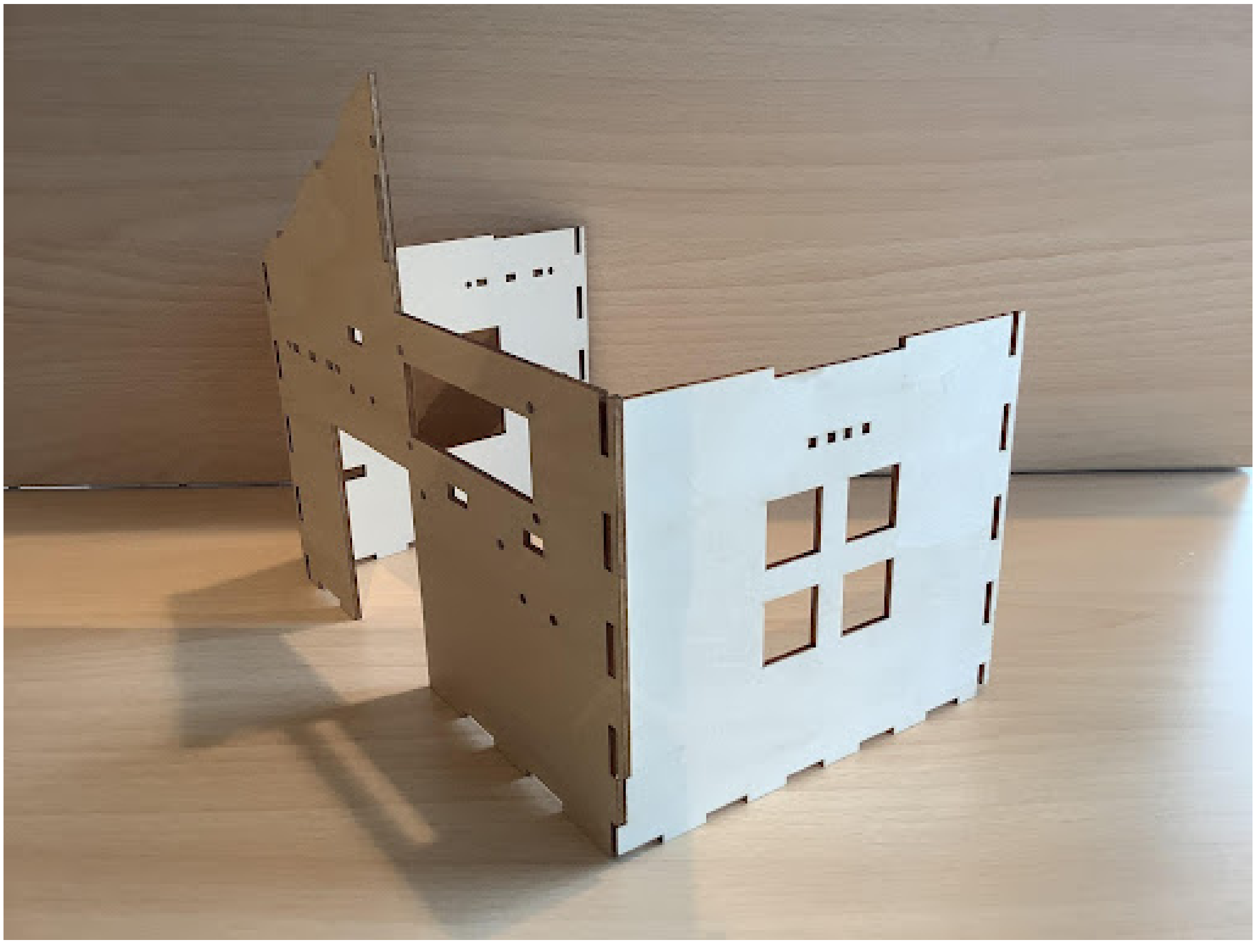
Steps of Building Smart Home Model

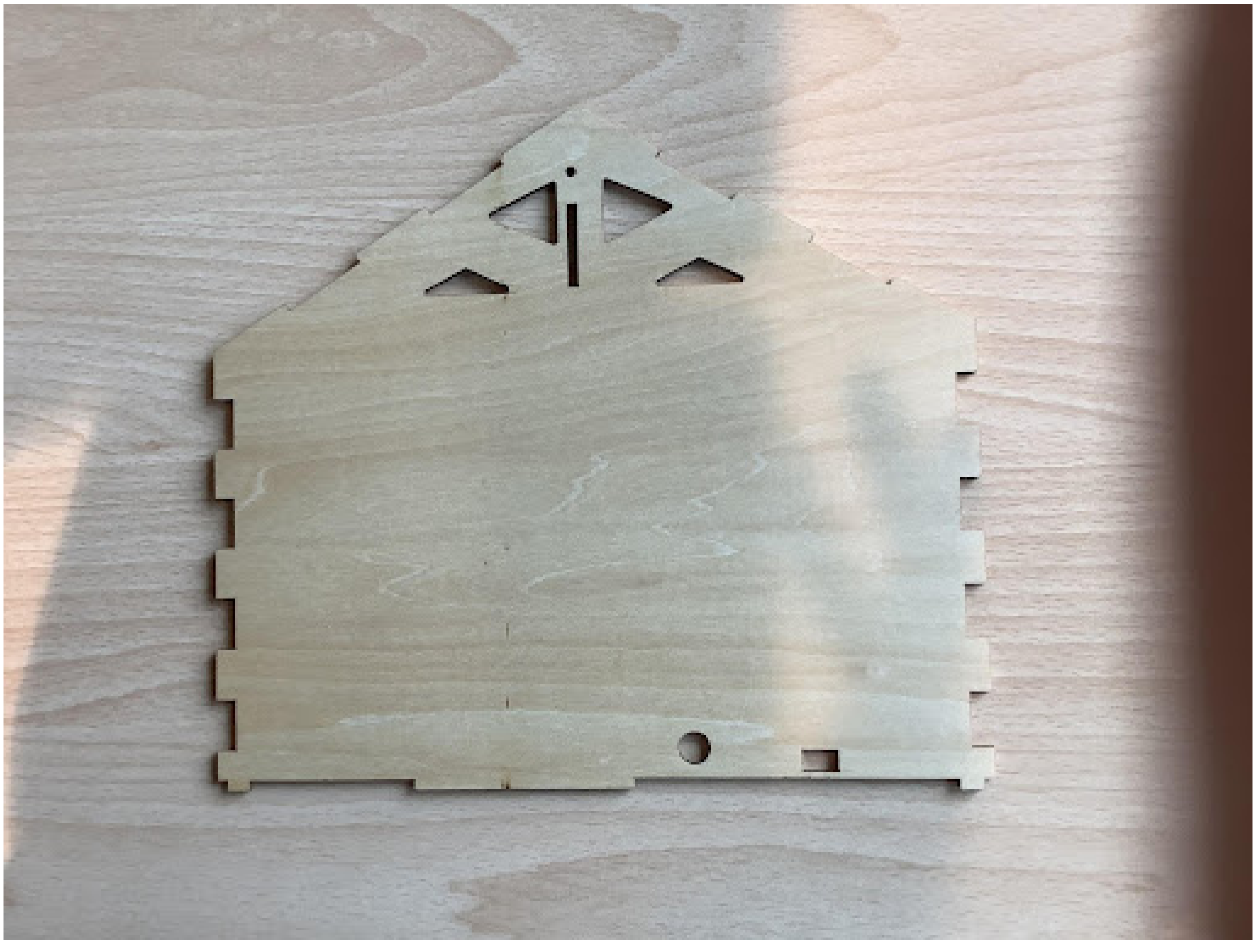


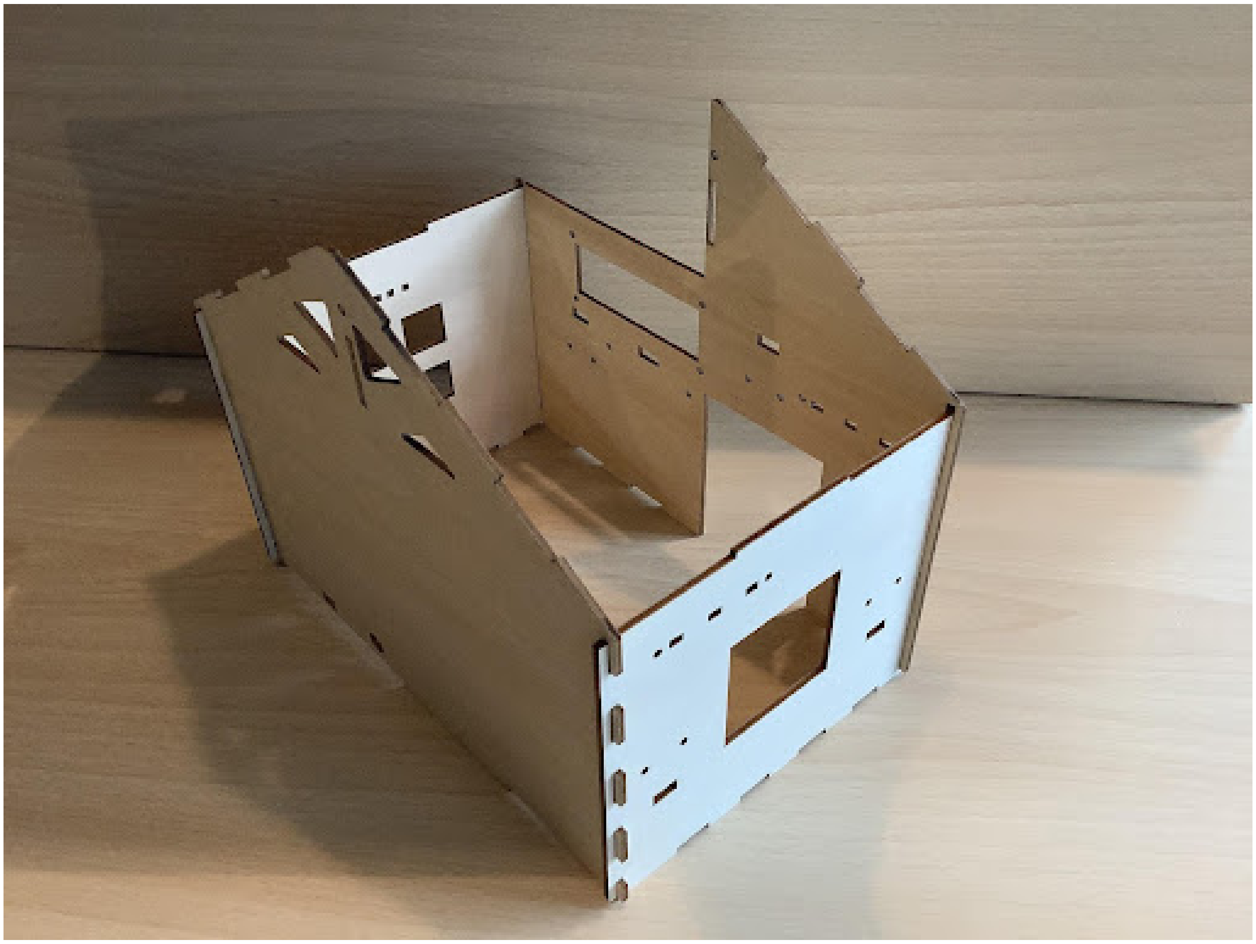


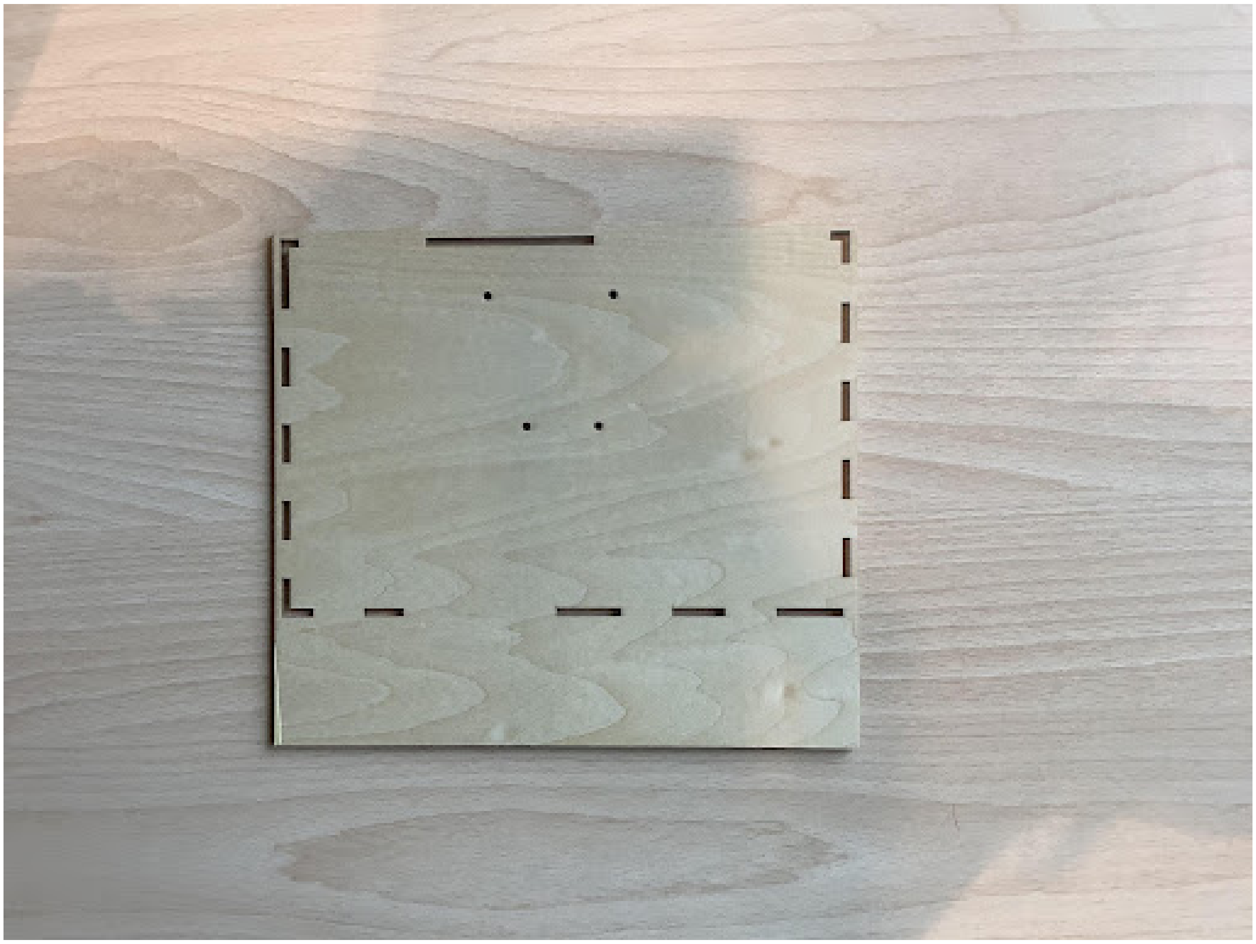


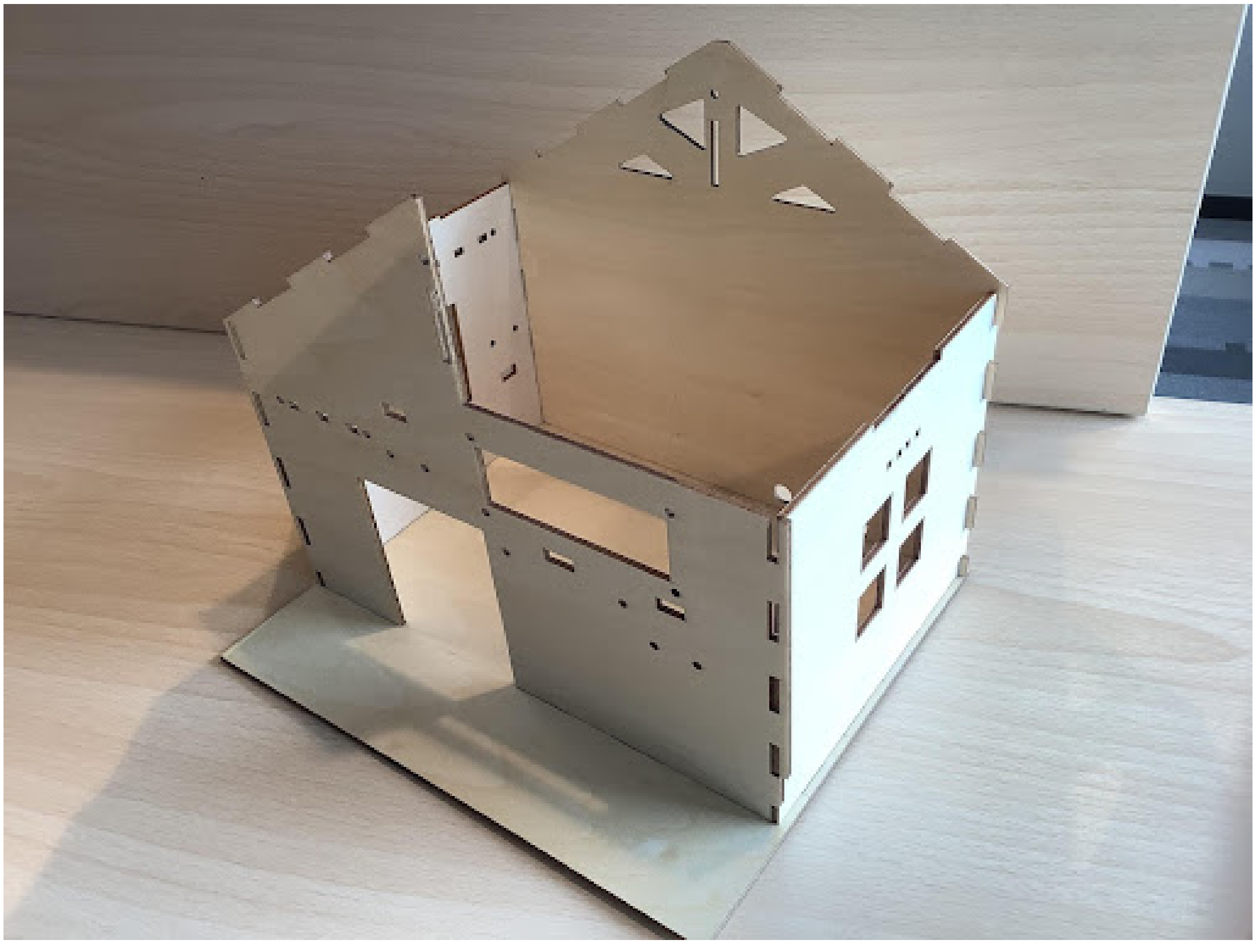


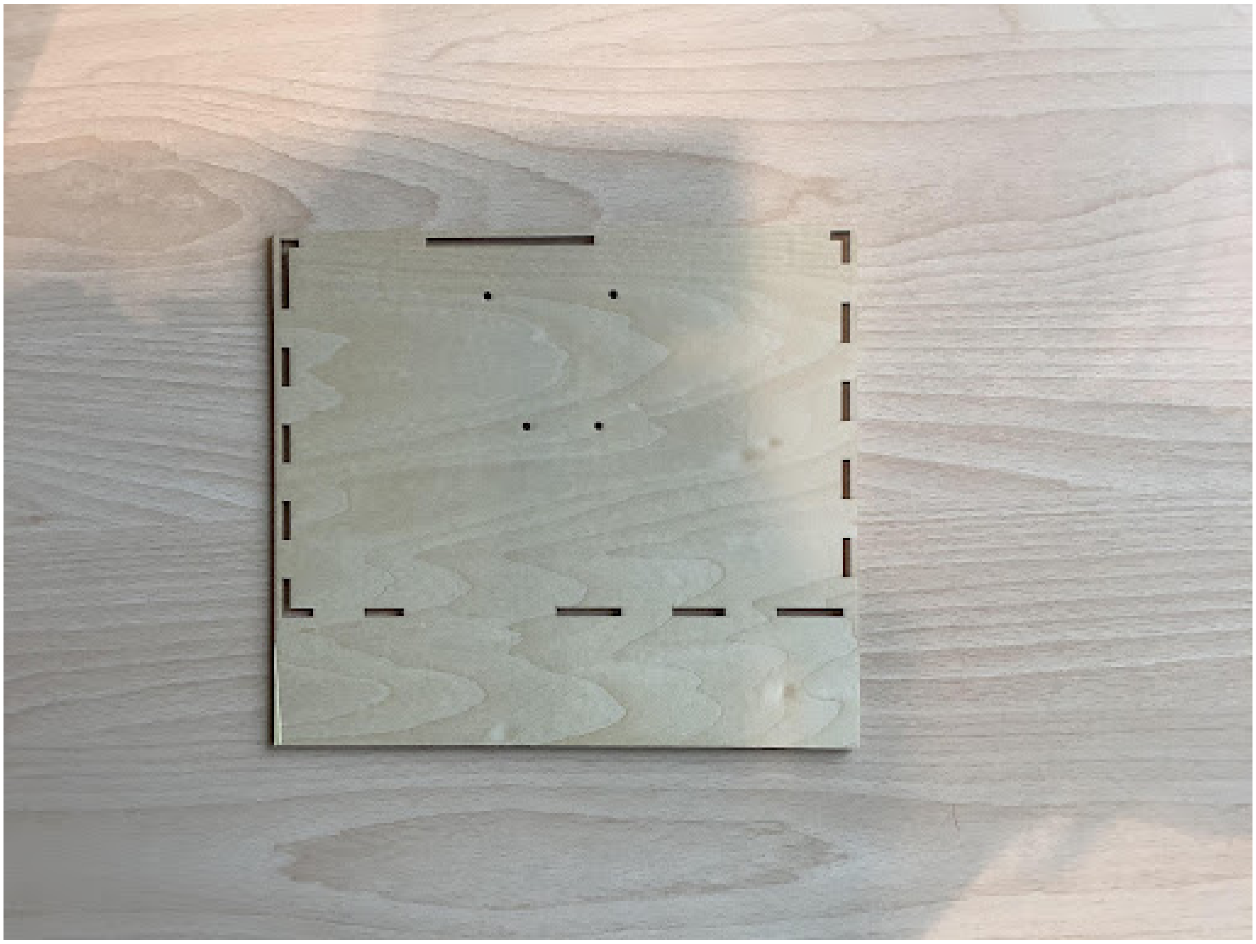


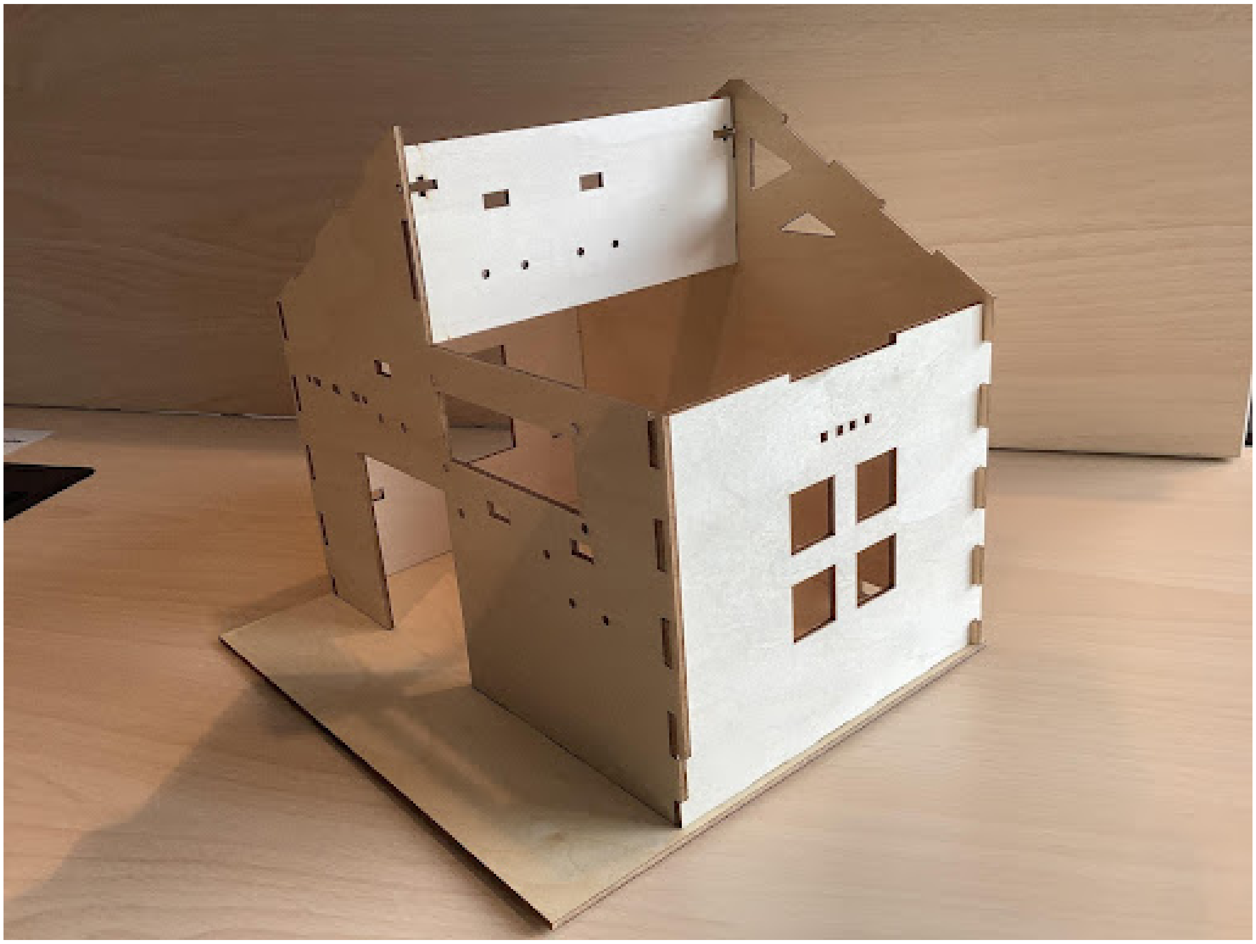


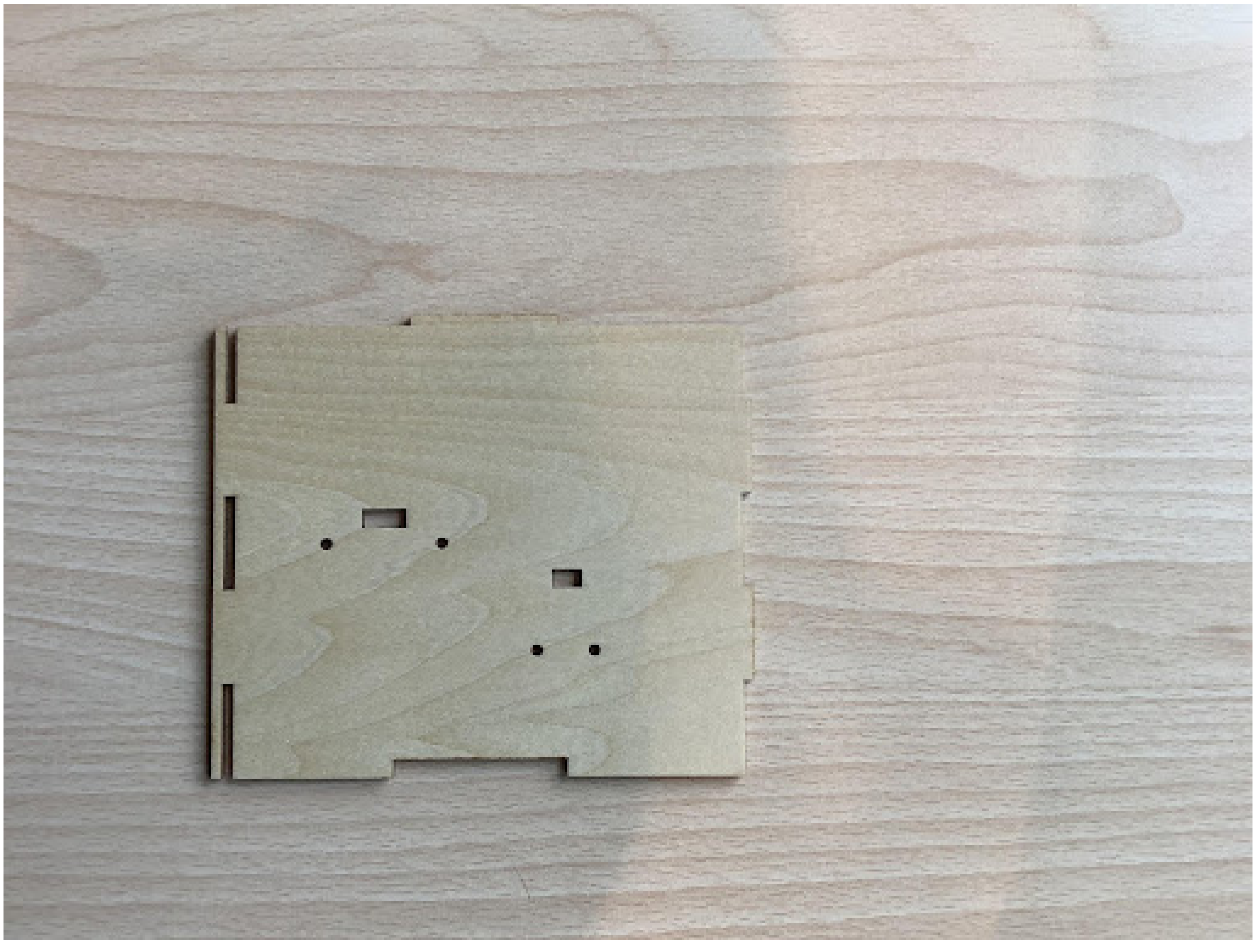


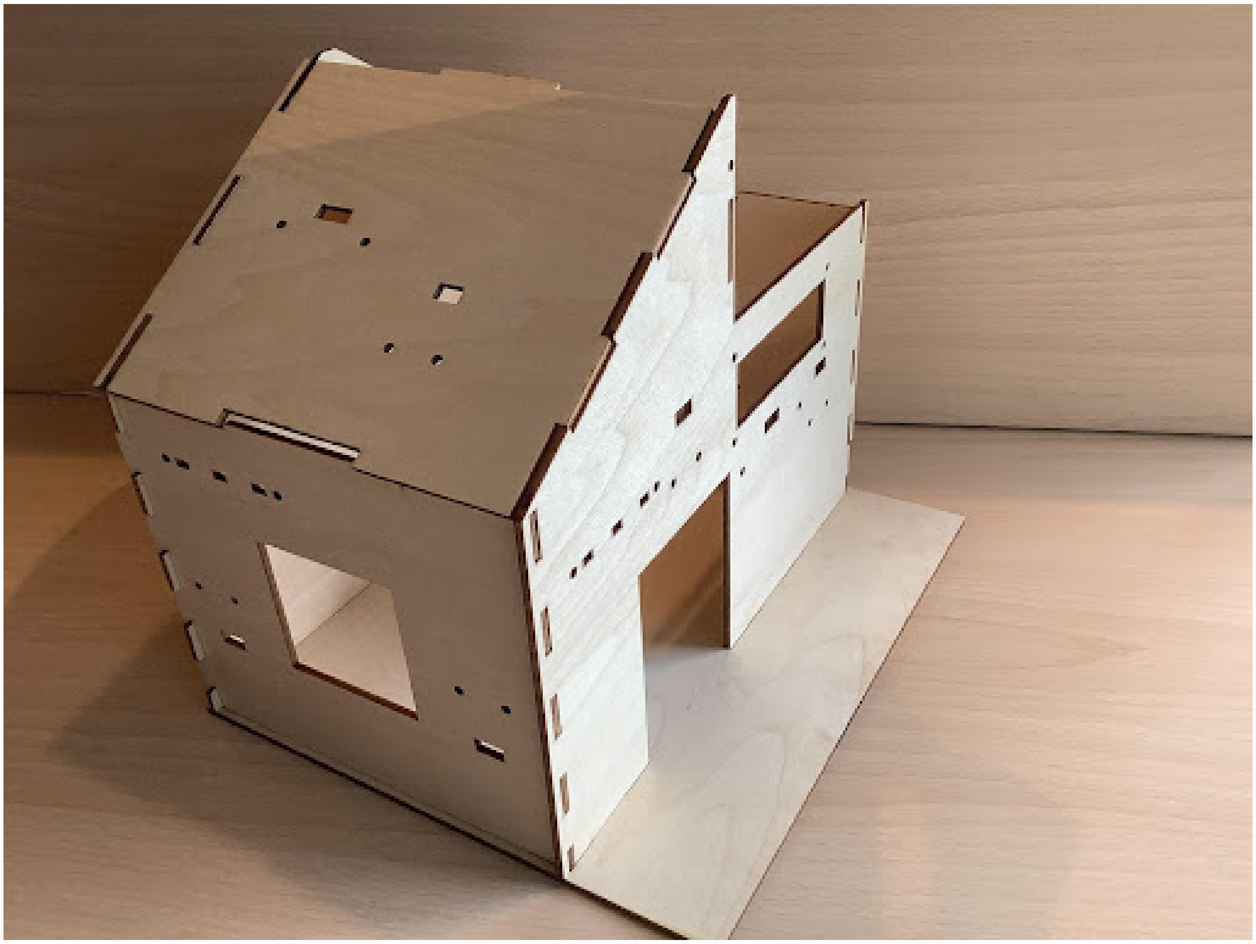




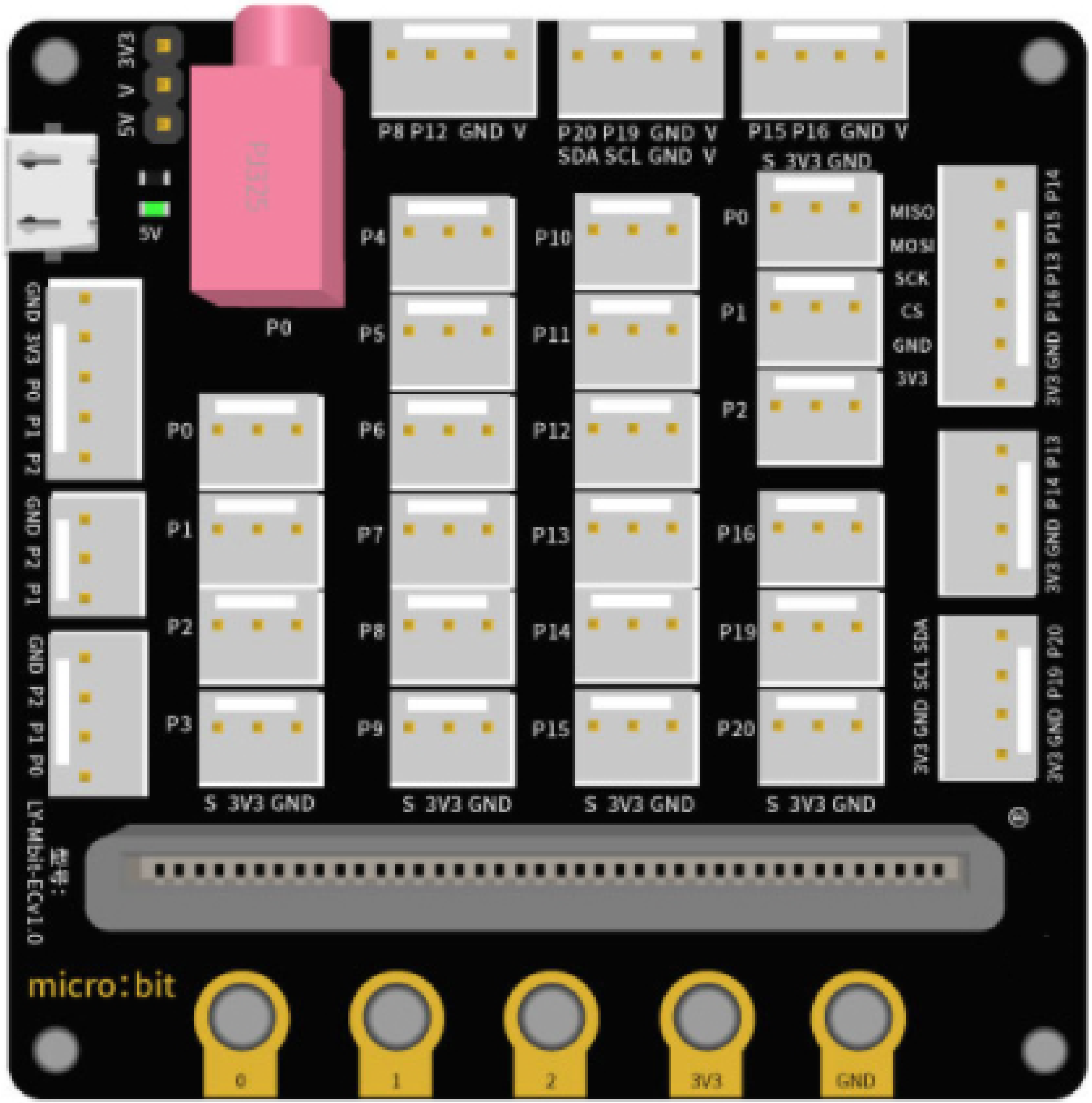




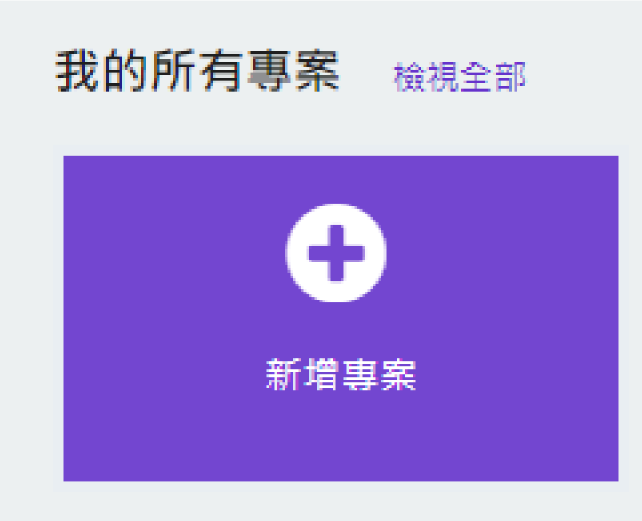


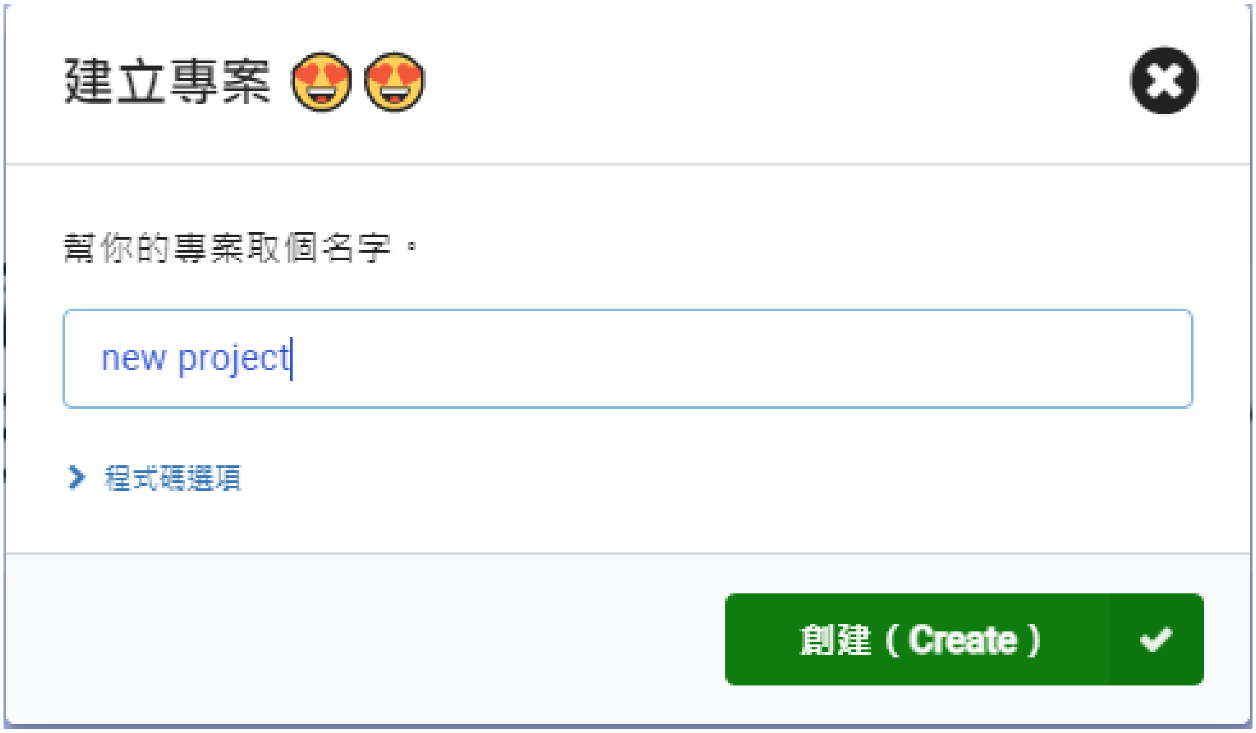


Learn About Micro:bit Expansion Board



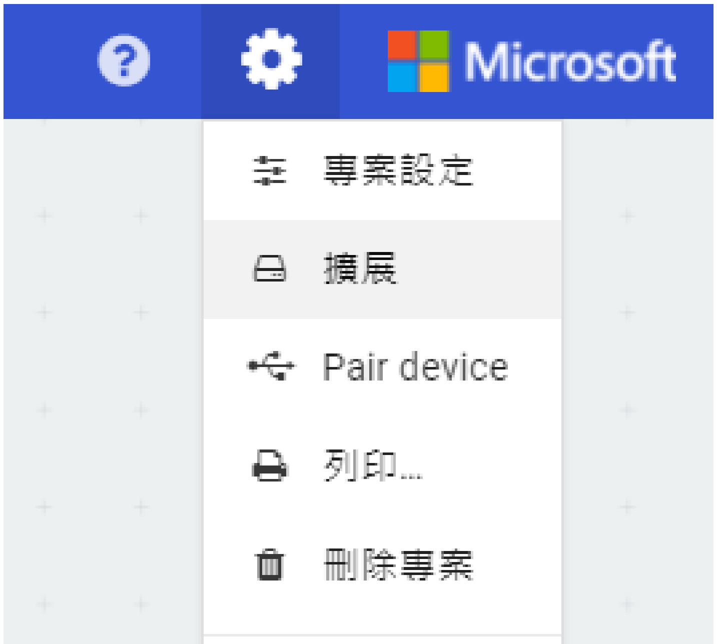
Prepare Micro:bit Programming MakeCode: Add Extensions

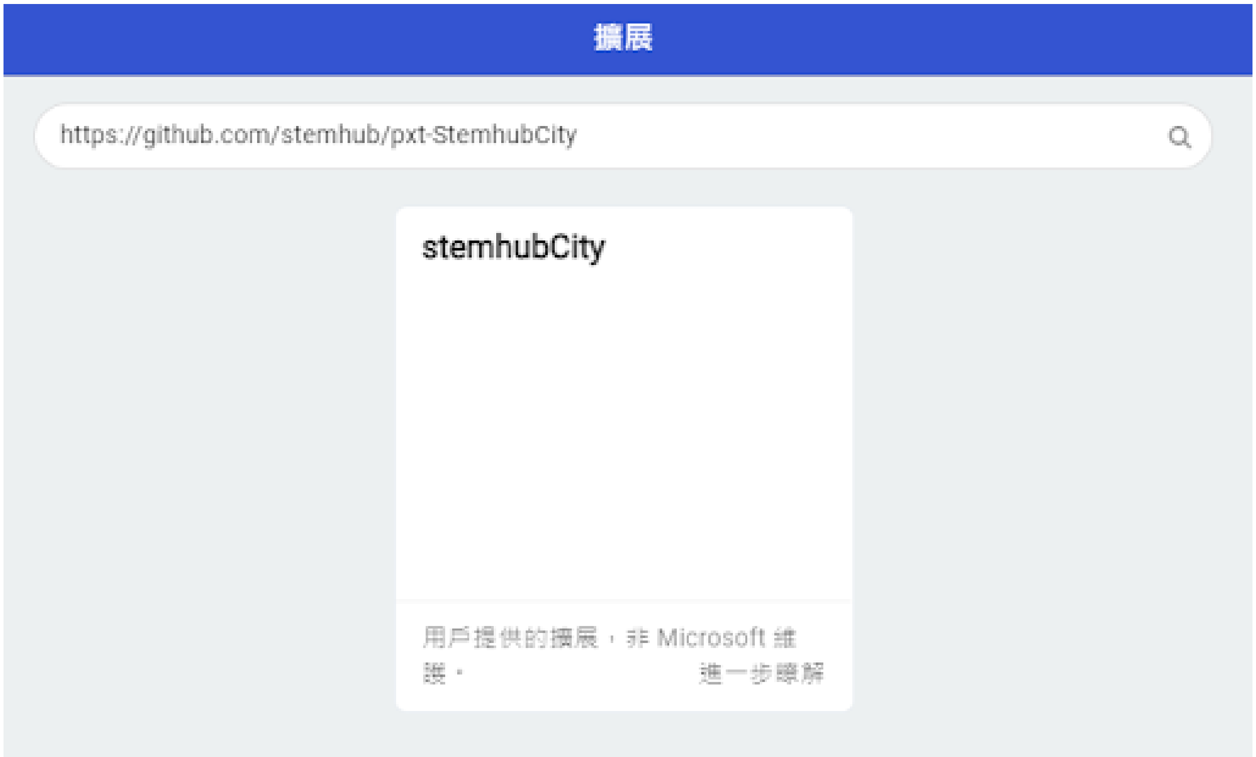




幫你的專案取個名字：

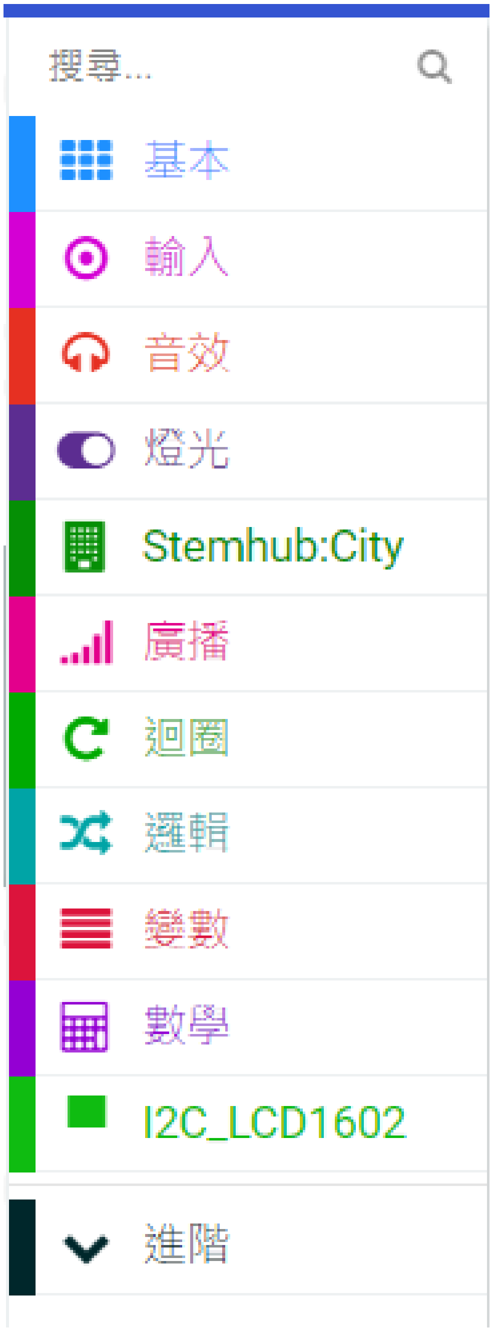
new project





https://github.com/stemhub/pxt-StemhubCity

用戶提供的擴展，非 Microsoft 維護。 進一步瞭解



1.1.2 Chapter 2 Smart Human Body Induction Lamp

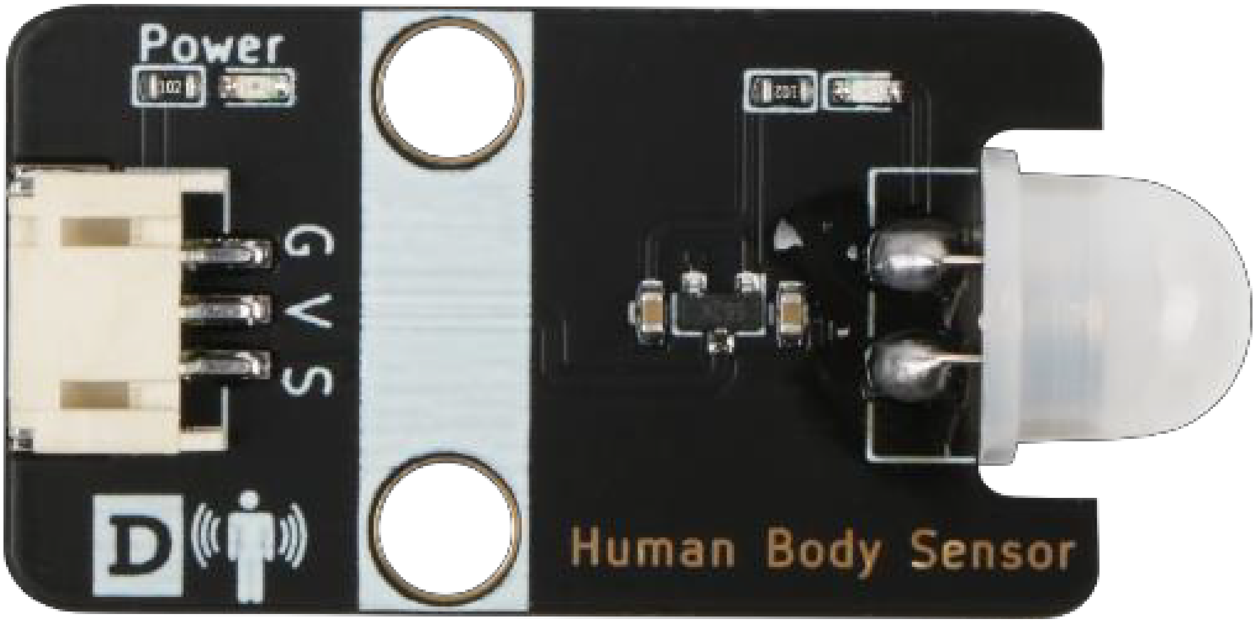
Background

Preparation

Learn About Smart Body Induction Lamp

Learn About Human Sensor Module and the Red and Green Light Module

Human Body Sensor Module

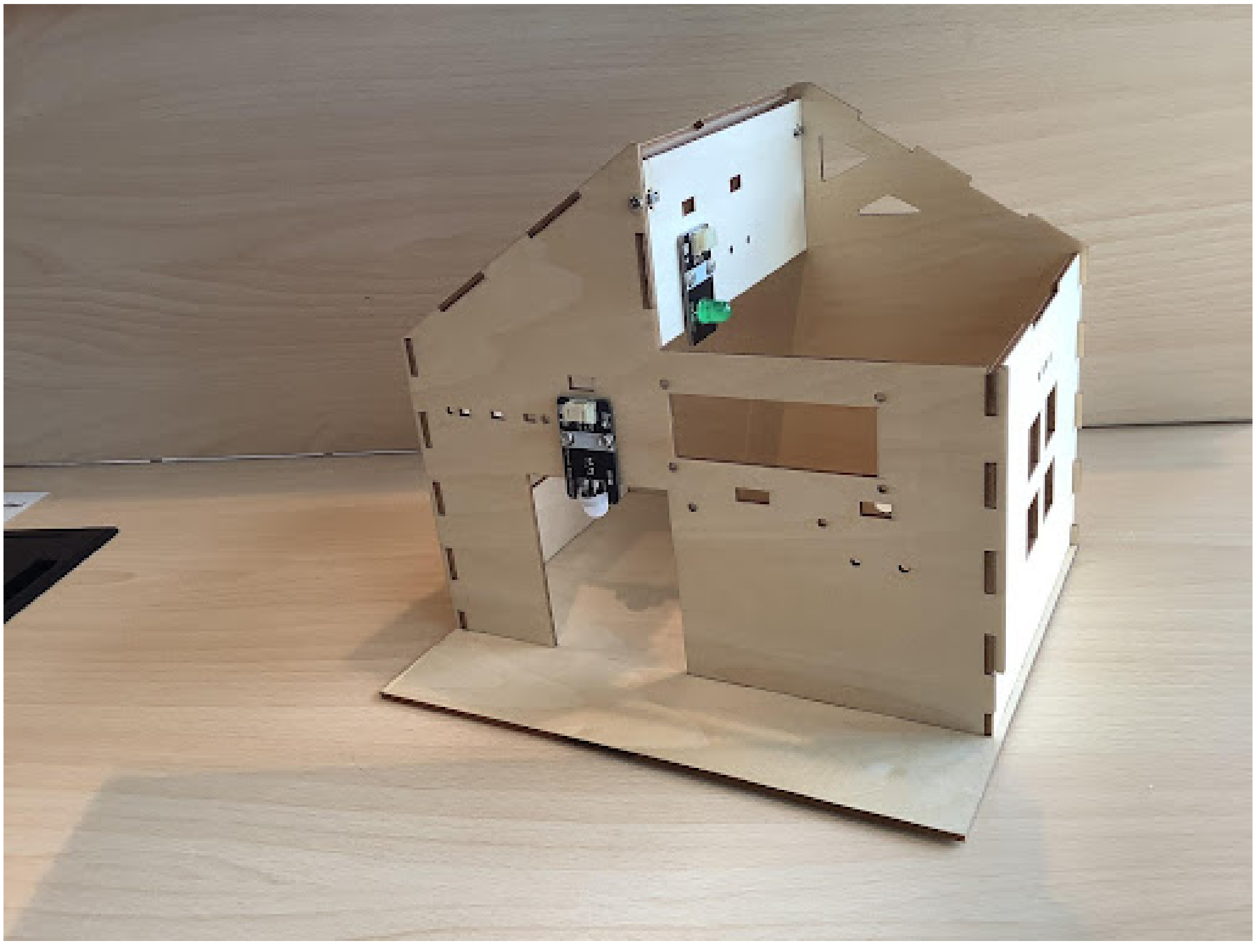


Fully automatic sensing: high level is output when a person enters its sensing range, and low level is output when the person leaves the sensing range with automatic delay to turn off high level.

Repeatable triggering method: After the induction output is in high level, if a person is sensed in the induction area during the delay time period, the output will remain high level until the person leaves and then the high level will be changed to low level (the induction module will automatically delay a delay time period after detecting each human activity, and the time of the last activity will be the starting point of the delay time HC-SR505). The small human body sensor module has three pins, G for GND ground, V for VCC high level or 5v, S is the signal pin.

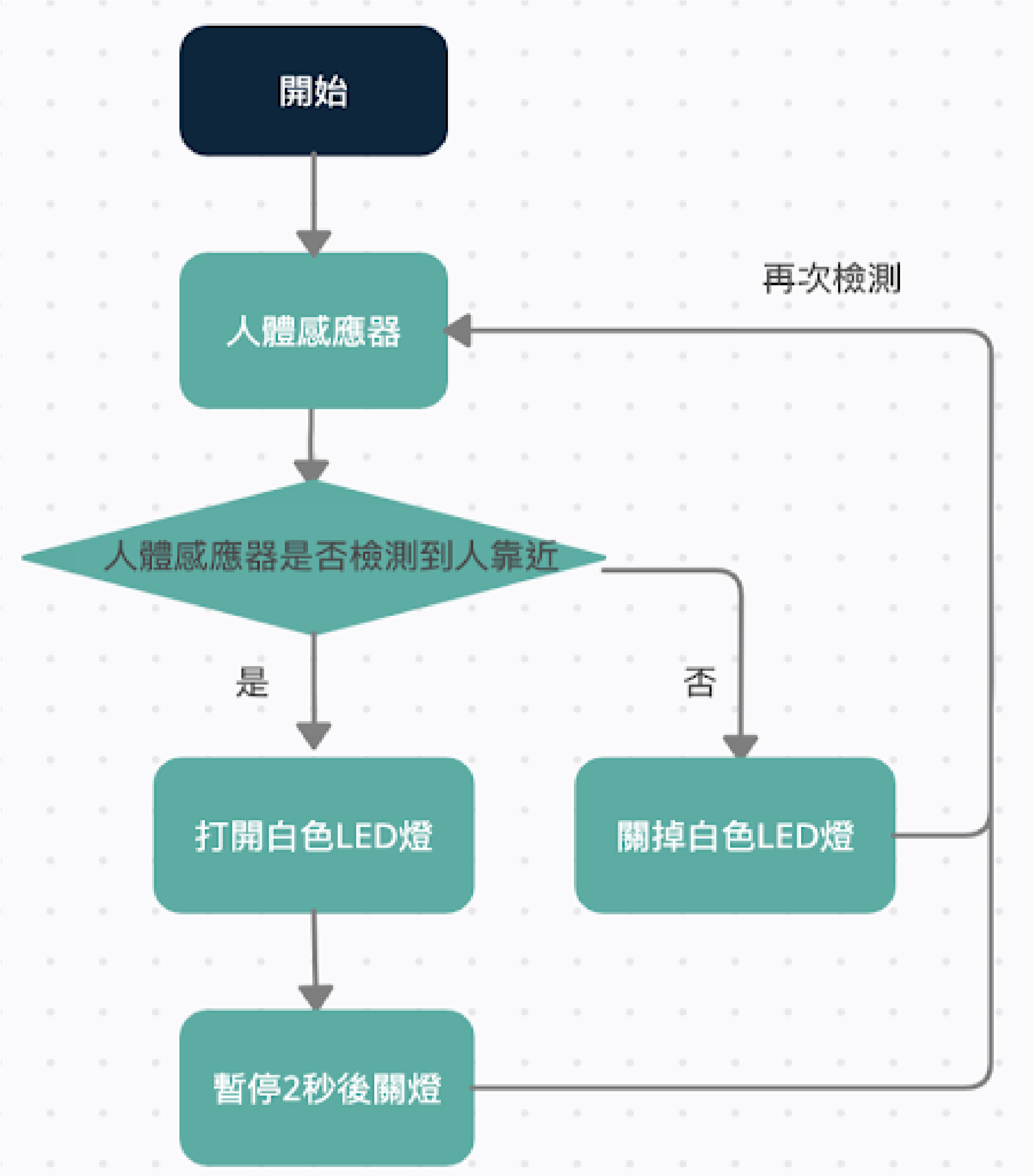
White LED light module

Installation of Human Body Sensor Light

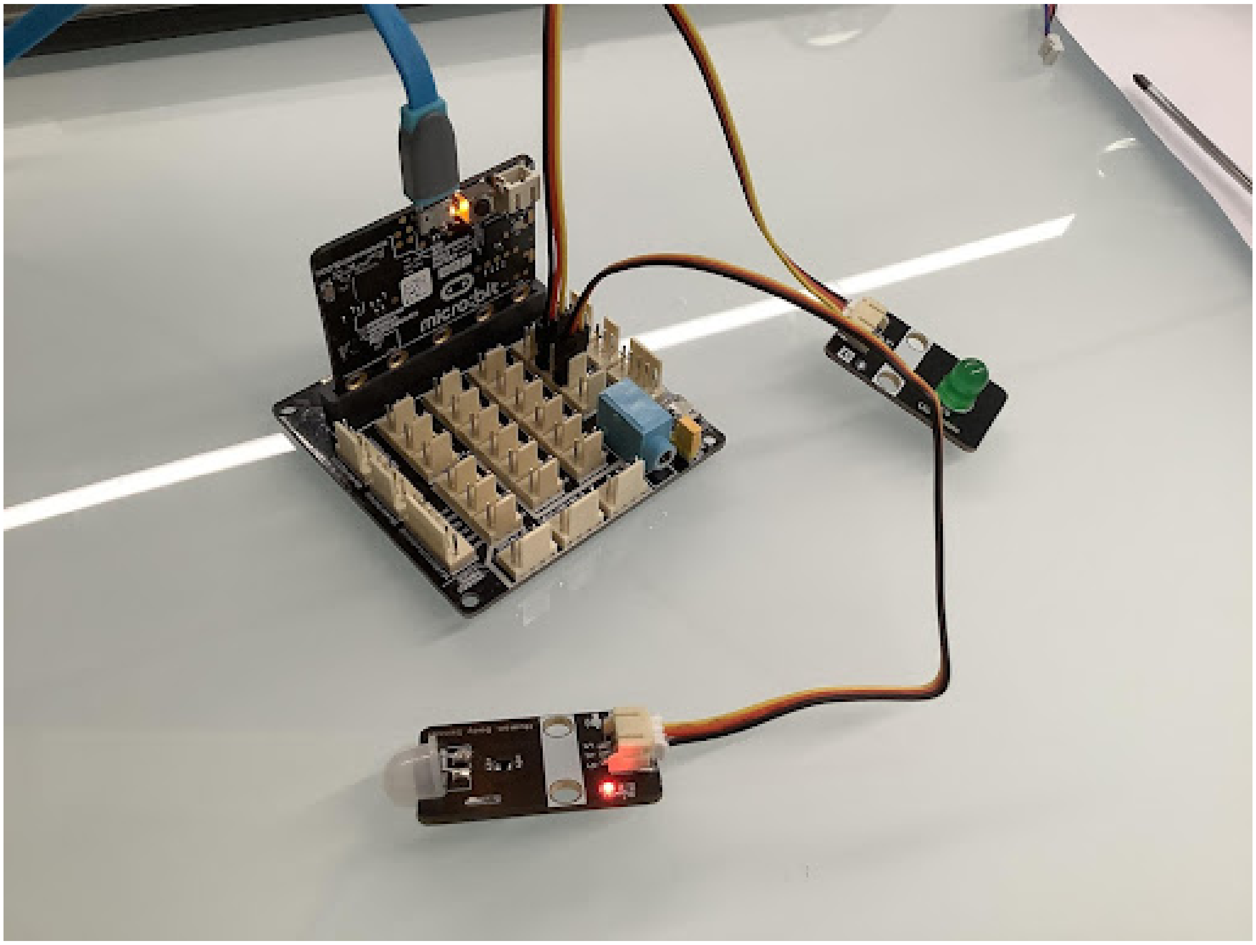


Program Design

Algorithm Design

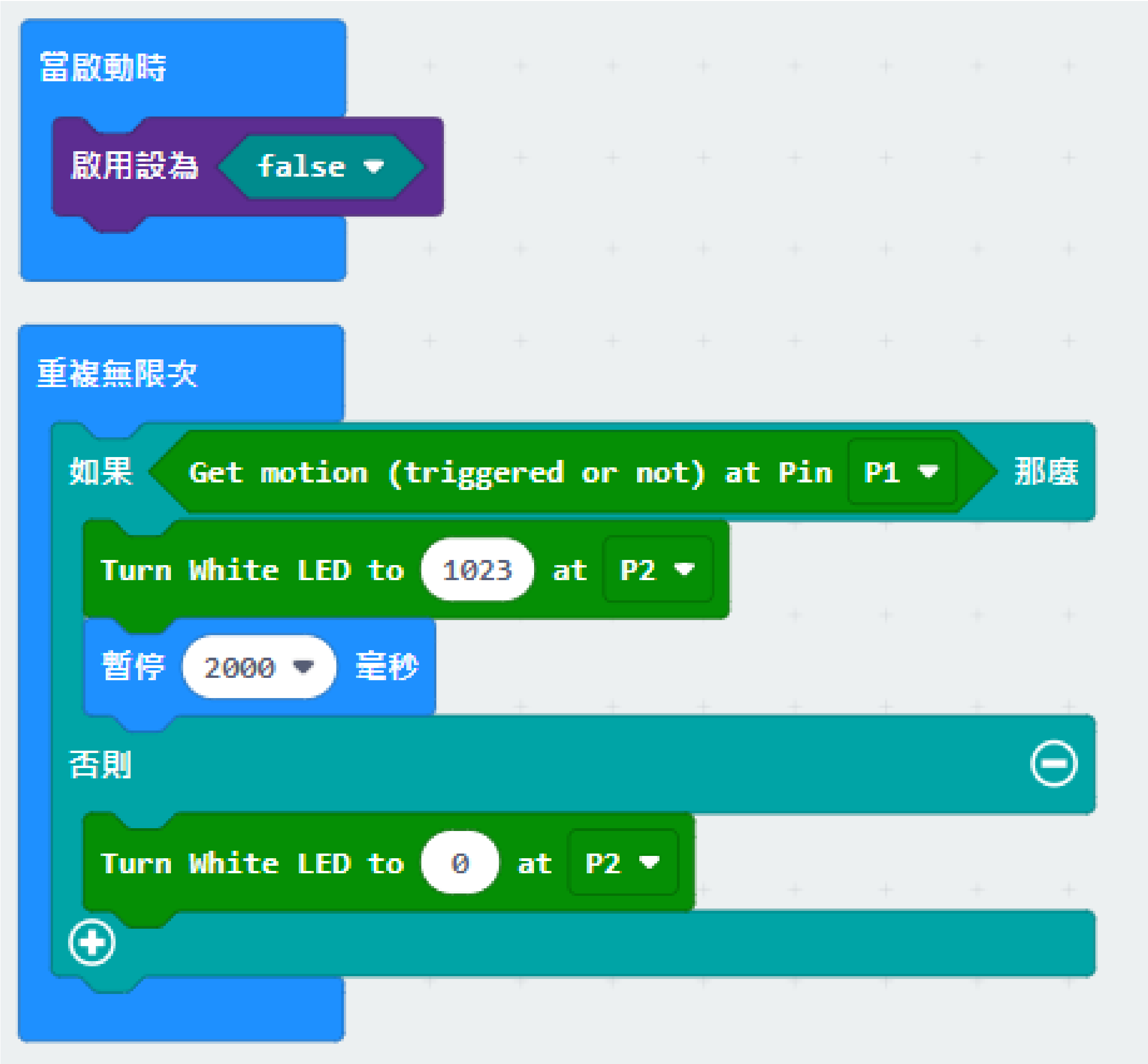


Hardware Connection



Sample Program

Makecode program



```
當啟動時
    啟用設為 false ▾

重複無限次
    如果 Get motion (triggered or not) at Pin P1 ▾ 那麼
        Turn White LED to 1023 at P2 ▾
        暫停 2000 ▾ 毫秒
    否則
        Turn White LED to 0 at P2 ▾
```

Conclusion

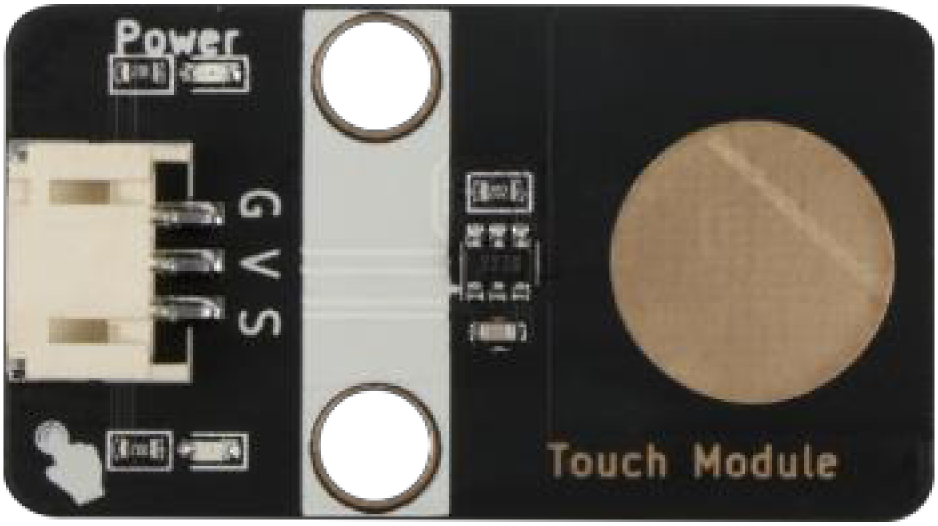
1.1.3 Chapter 3 Music Doorbell

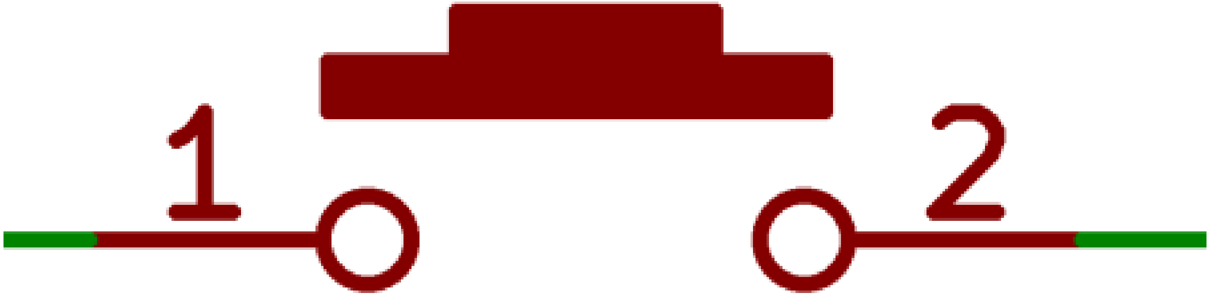
Background

Preparation

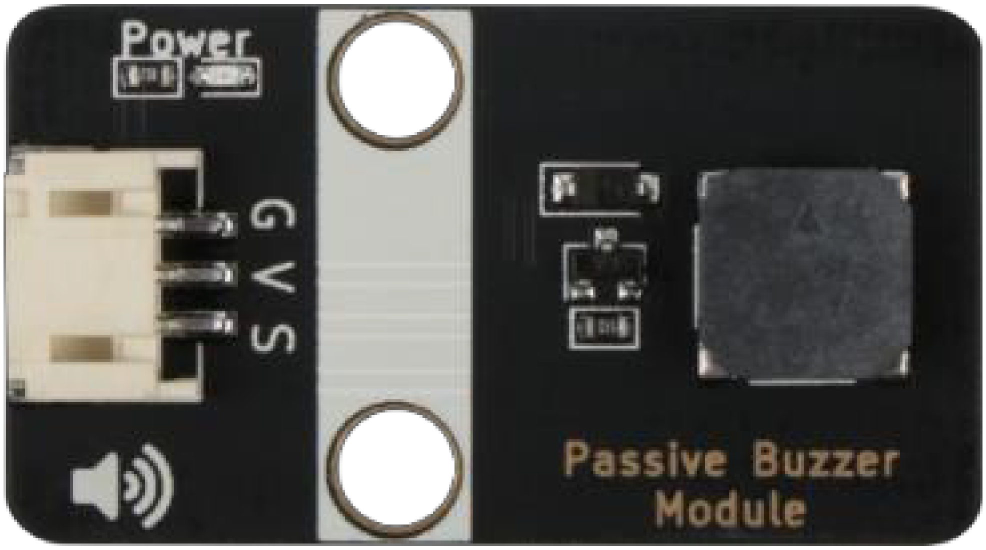
Learn AboutTouch Sensors and Passive Buzzers

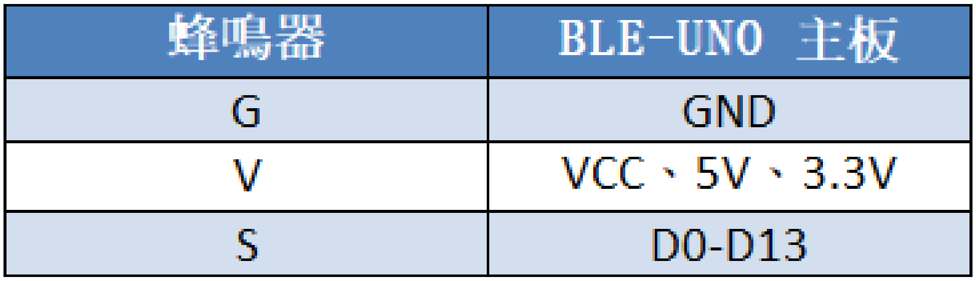
Touch Sensor





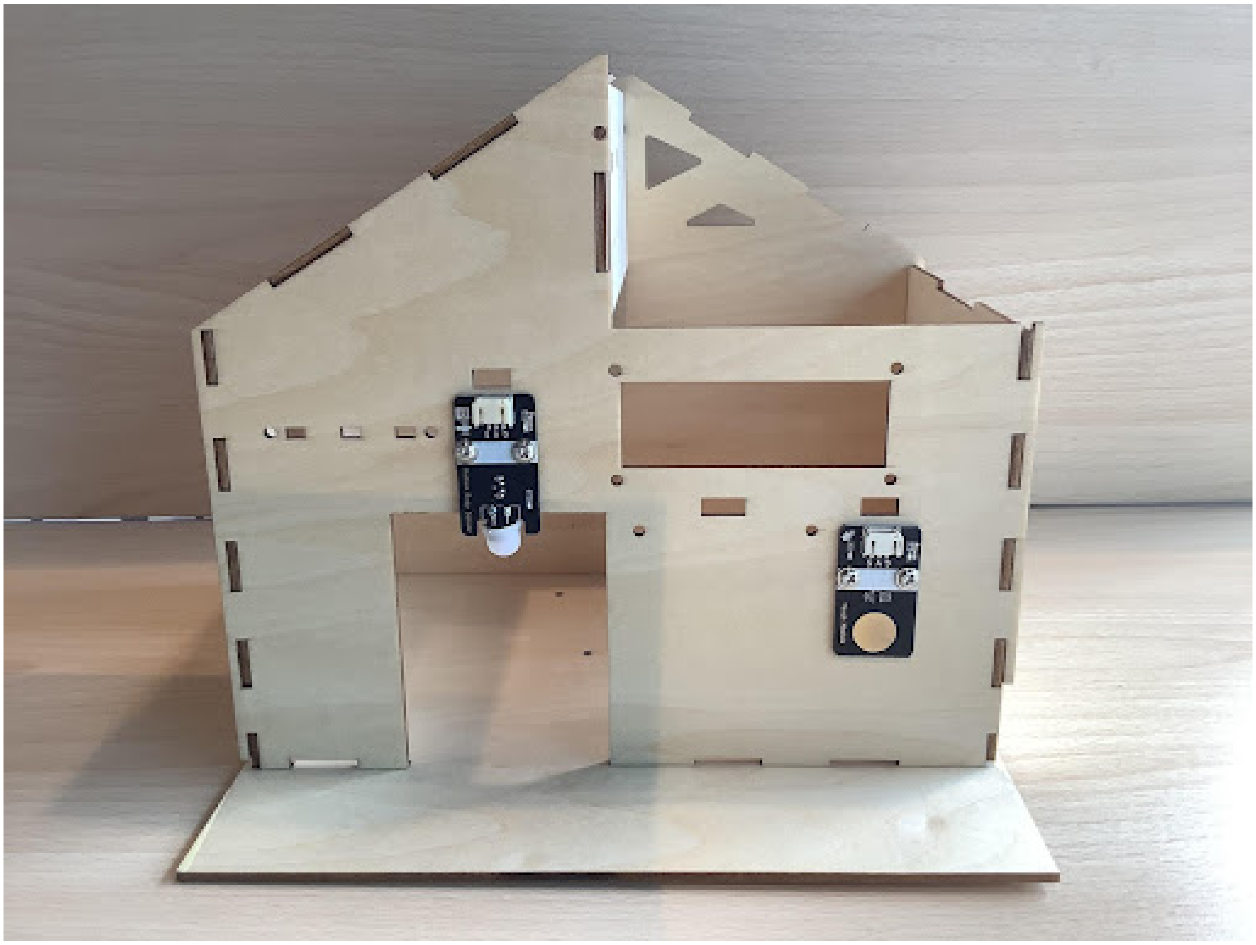
Passive buzzer module

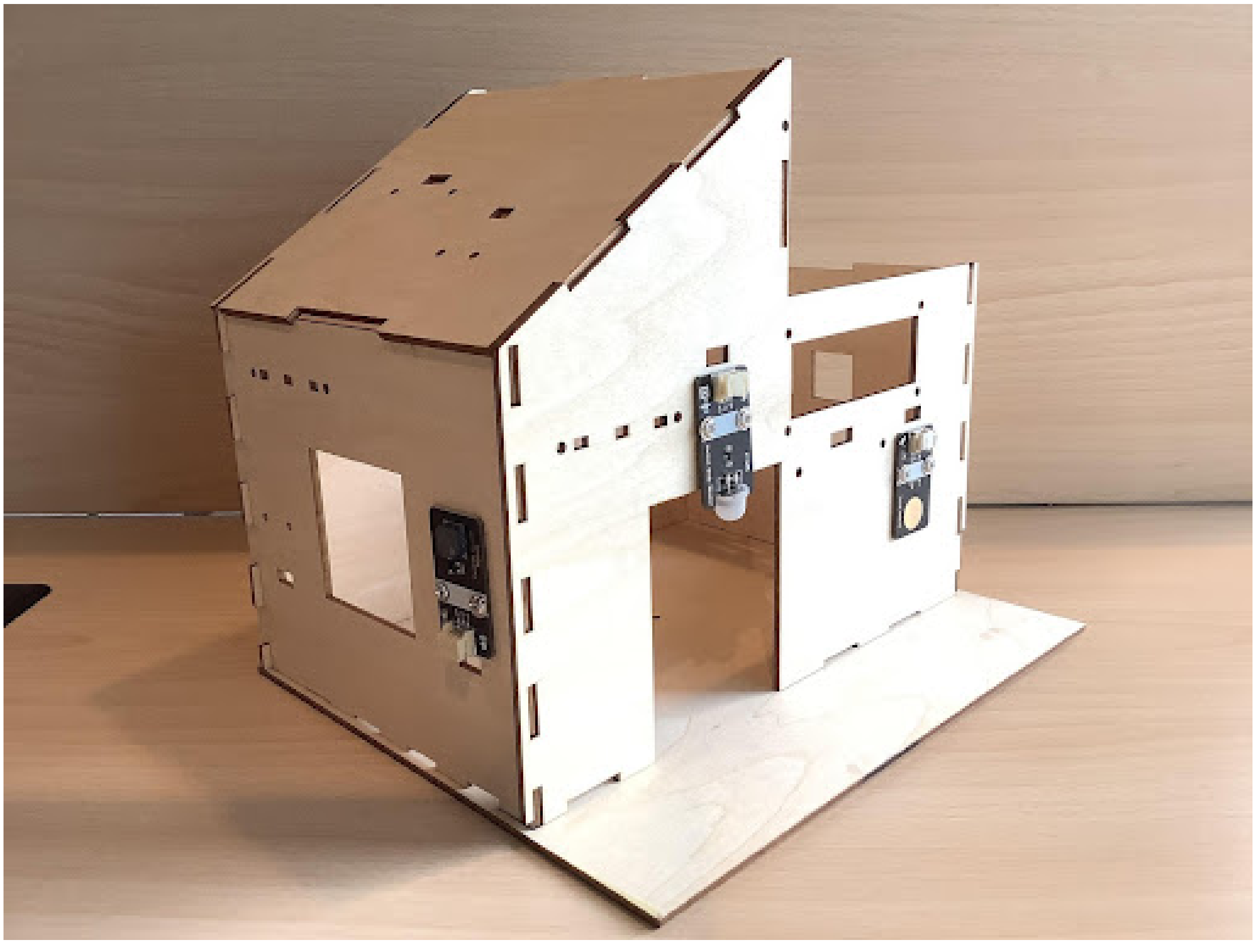




蜂鳴器	BLE-UNO 主板
G	GND
V	VCC、5V、3.3V
S	D0-D13

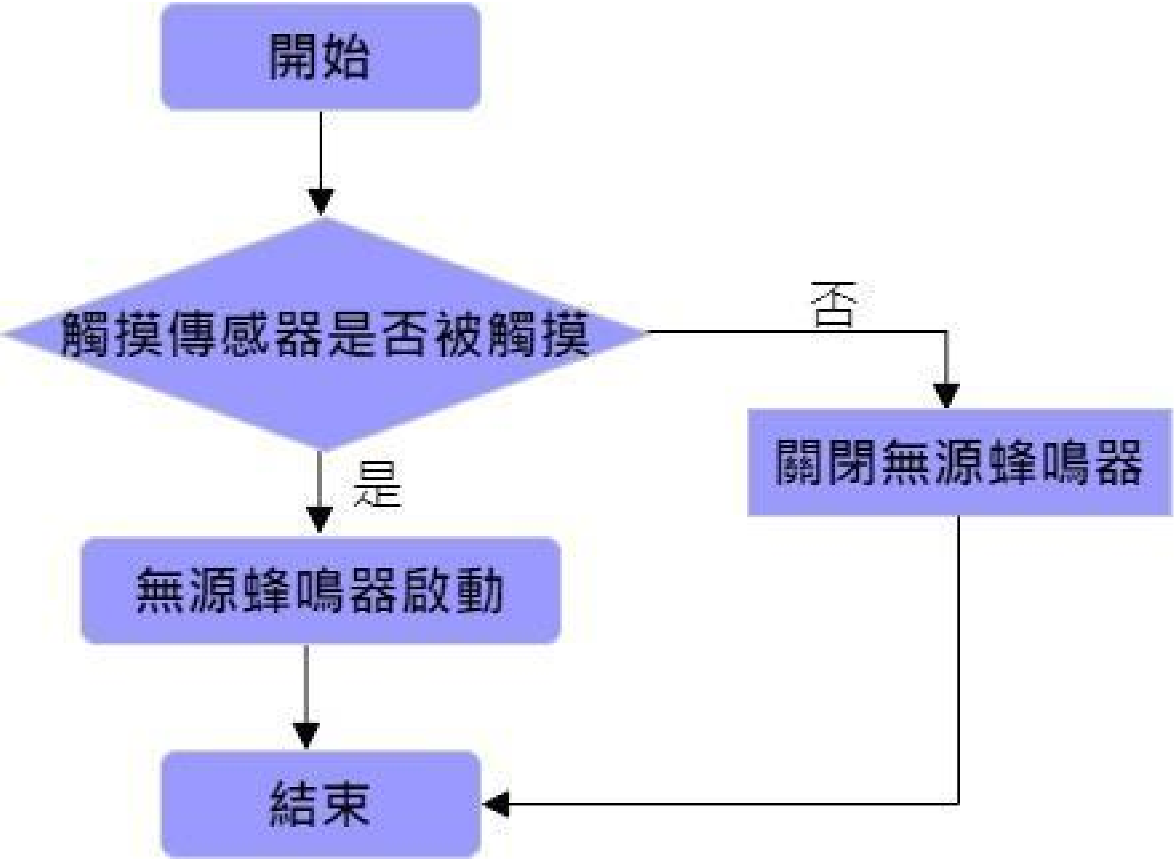
Installation of Doorbell



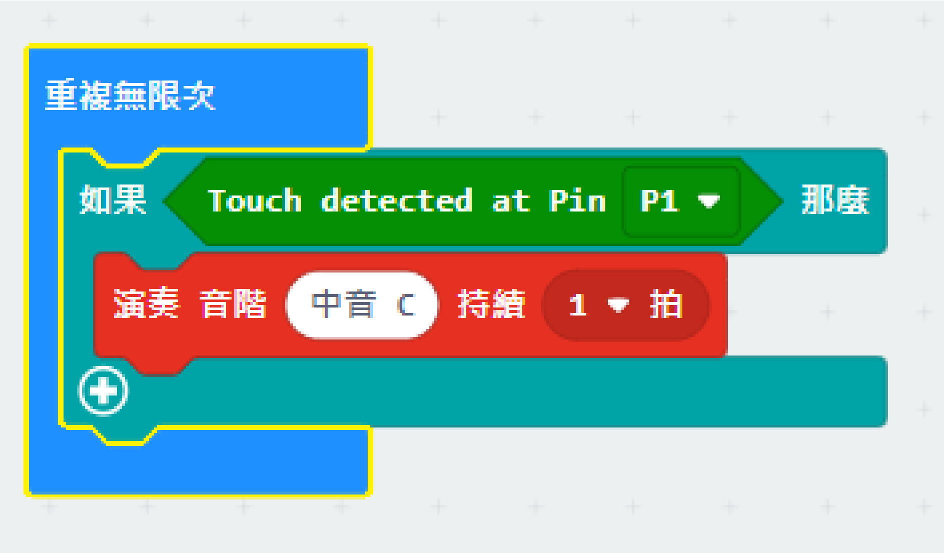


Program Design

Algorithm Design



Hardware Connection



```
重複無限次
  如果 Touch detected at Pin P1 ▾ 那麼
    演奏 音階 中音 C 持續 1 ▾ 拍
  ⊕
```

Conclusion

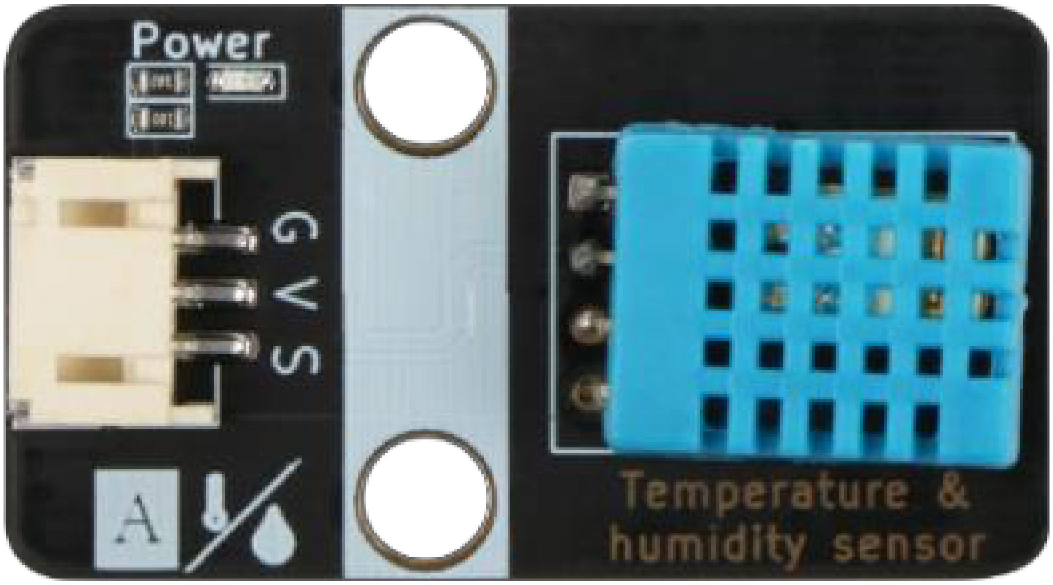
1.1.4 Chapter 4 Smart Temperature Control Fan

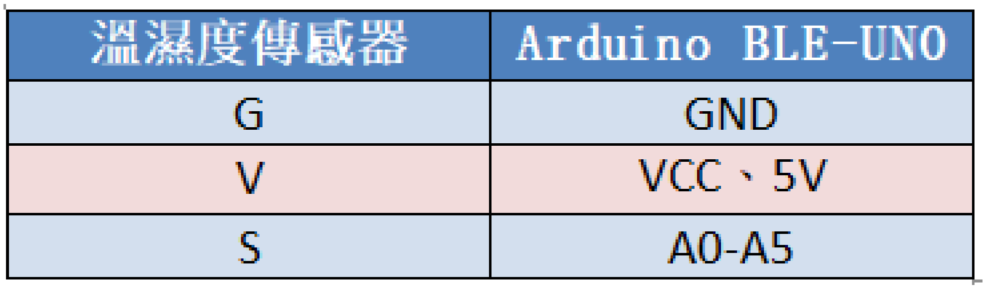
Background

Preparation

Learn About Temperature And Humidity Sensors And DC Motor Fan Modules

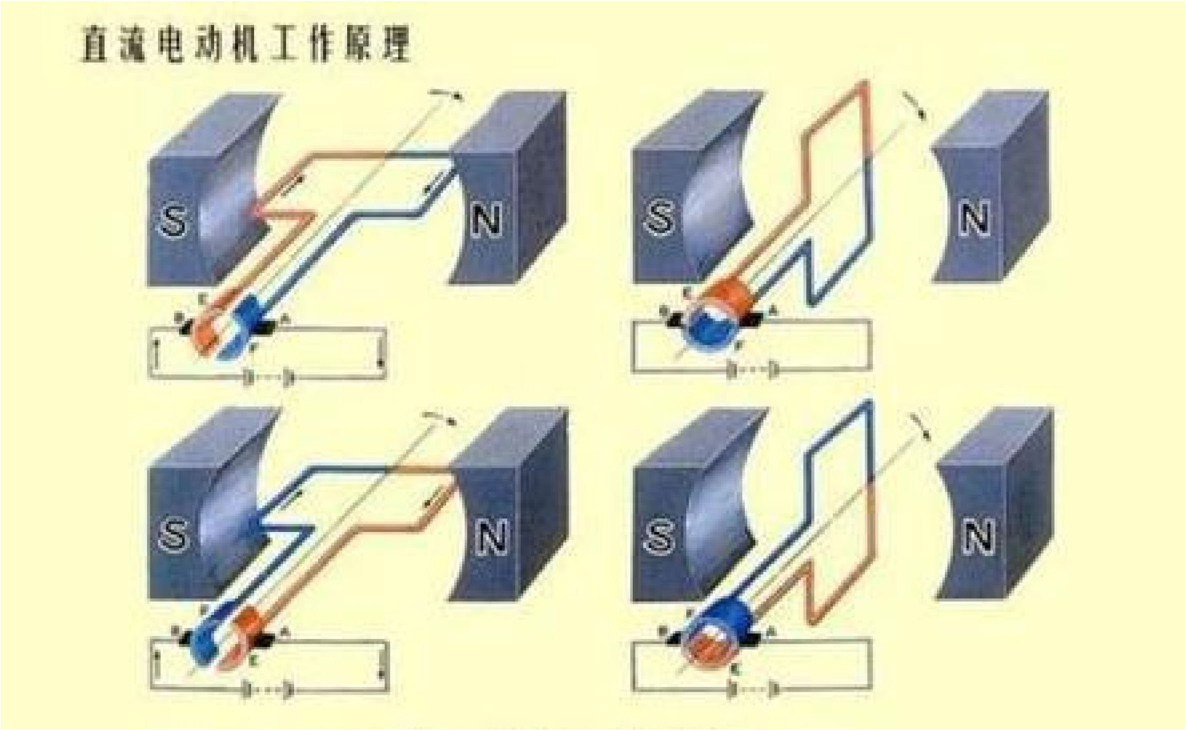
Temperature and Humidity Sensor Module

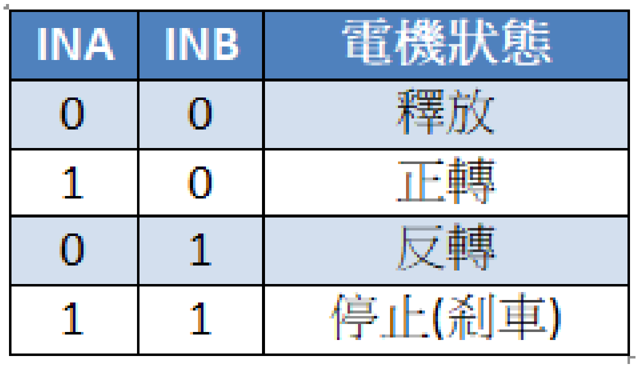




溫濕度傳感器	Arduino BLE-UNO
G	GND
V	VCC、5V
S	A0-A5

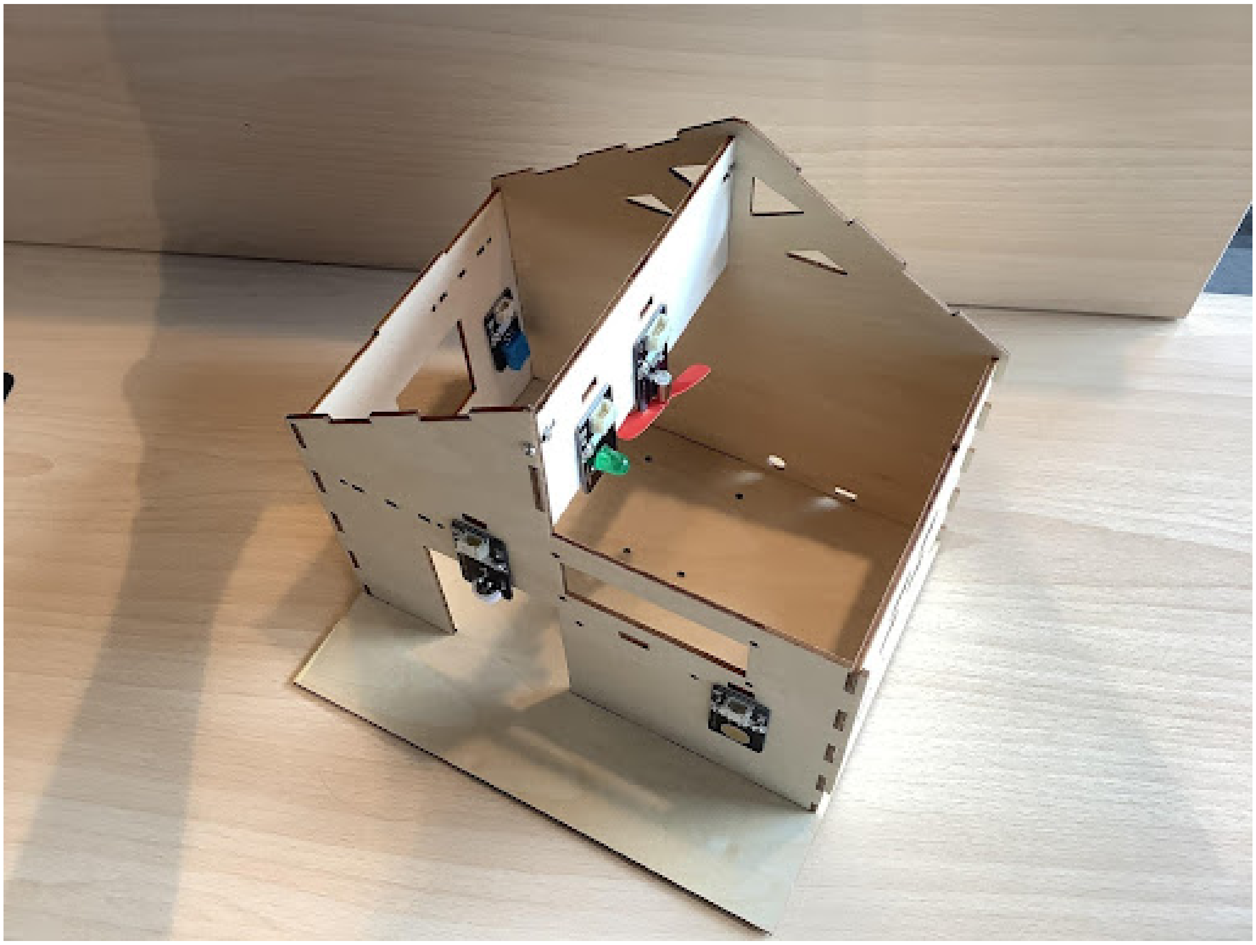
DC Electric Motor Wind Fan Model





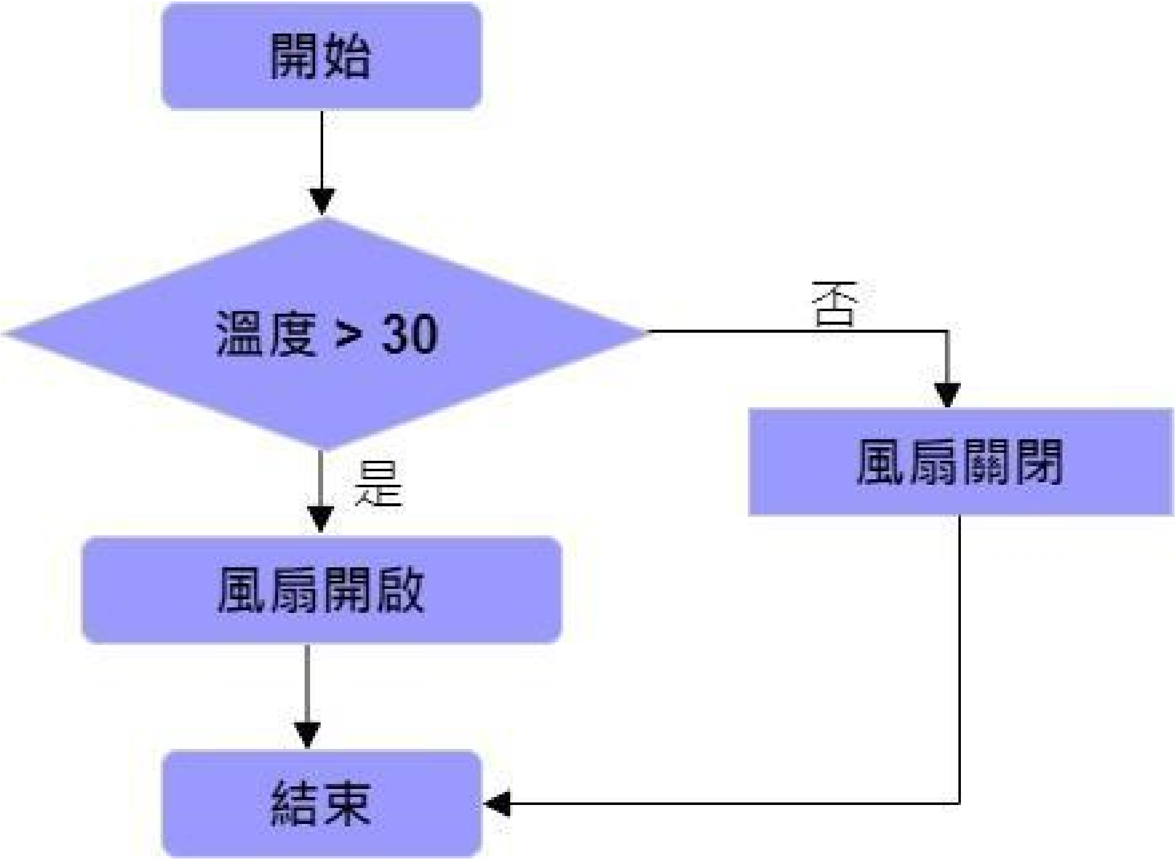
INA	INB	電機狀態
0	0	釋放
1	0	正轉
0	1	反轉
1	1	停止(剎車)

Installation of Temperature-Controlled Fan



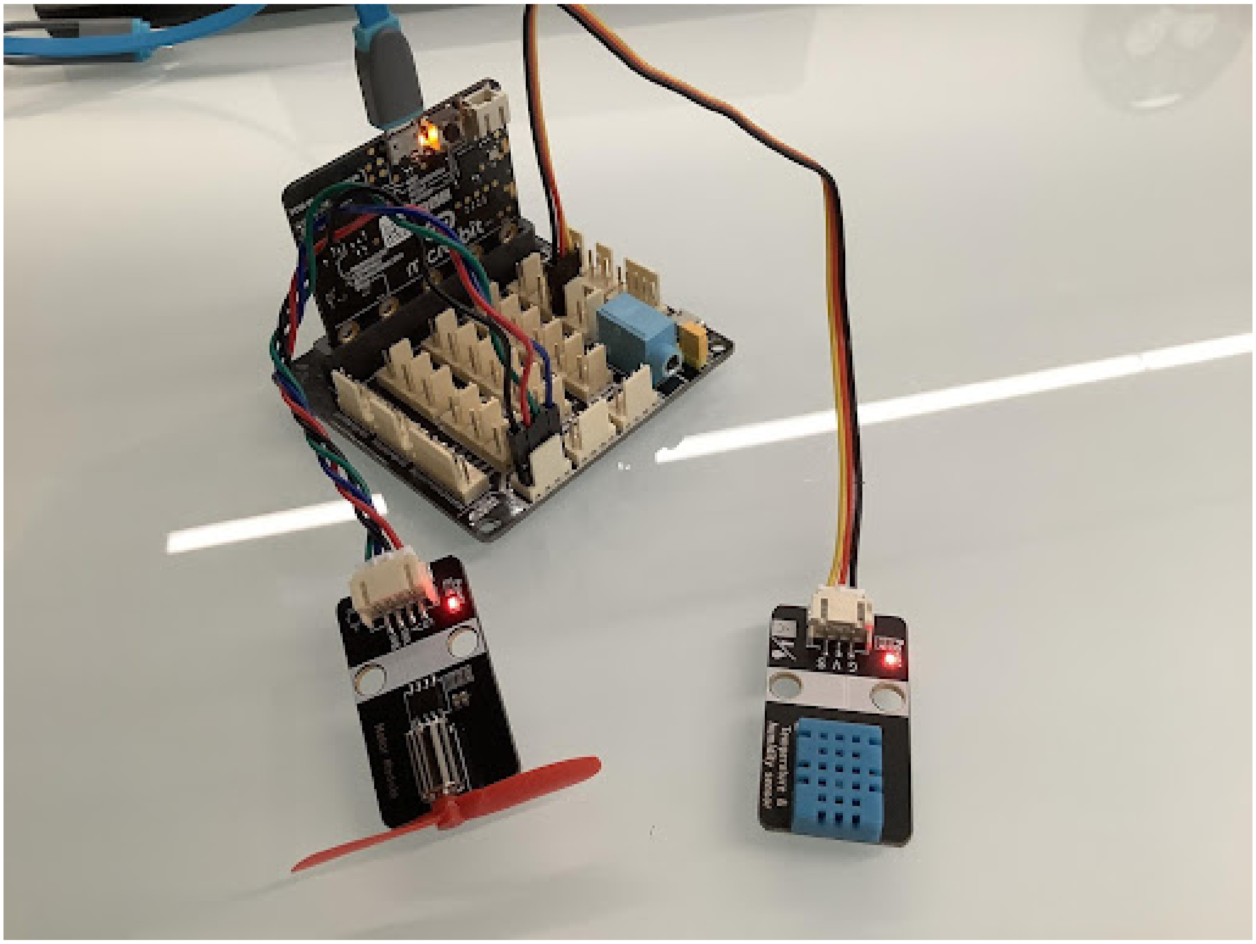
Program Design

Algorithm Design

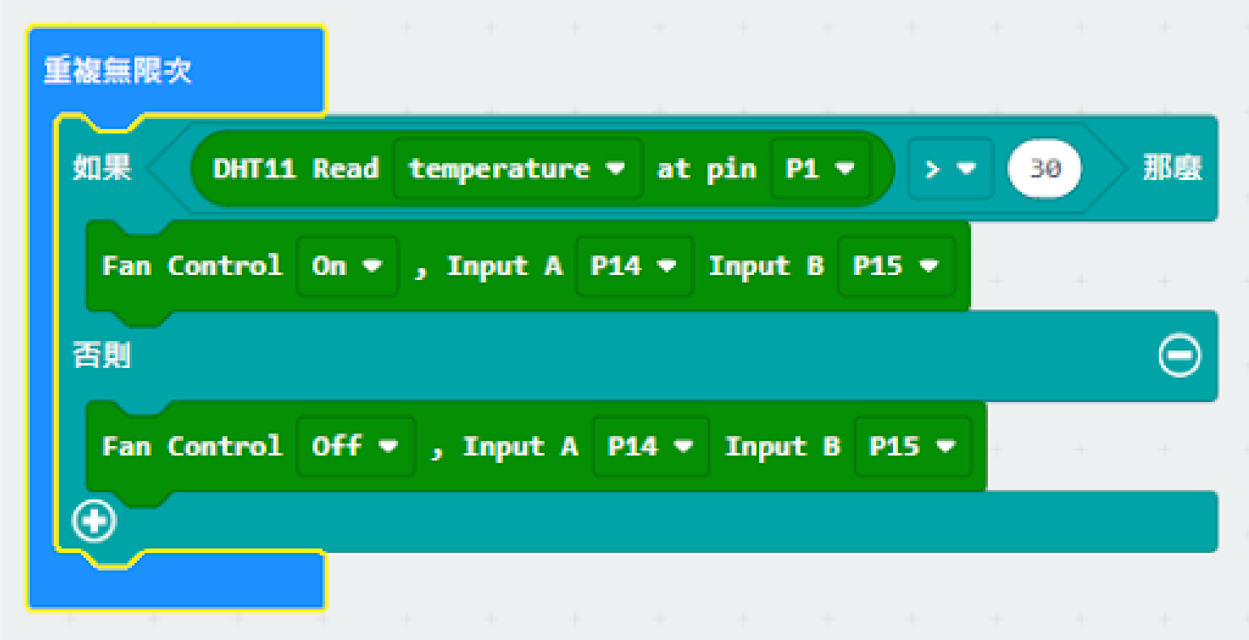


Hardware Connections

Sensors and Actuators | Main Control Board :– | :– Temperature and Humidity Sensor | P1 DC Motor Fan Module | P15P16



Sample Program



```
重複無限次
  如果 DHT11 Read temperature ▾ at pin P1 ▾  > ▾  30  那麼
    Fan Control On ▾ , Input A P14 ▾ Input B P15 ▾
  否則 ⊖
    Fan Control Off ▾ , Input A P14 ▾ Input B P15 ▾
  ⊕
```

Conclusion

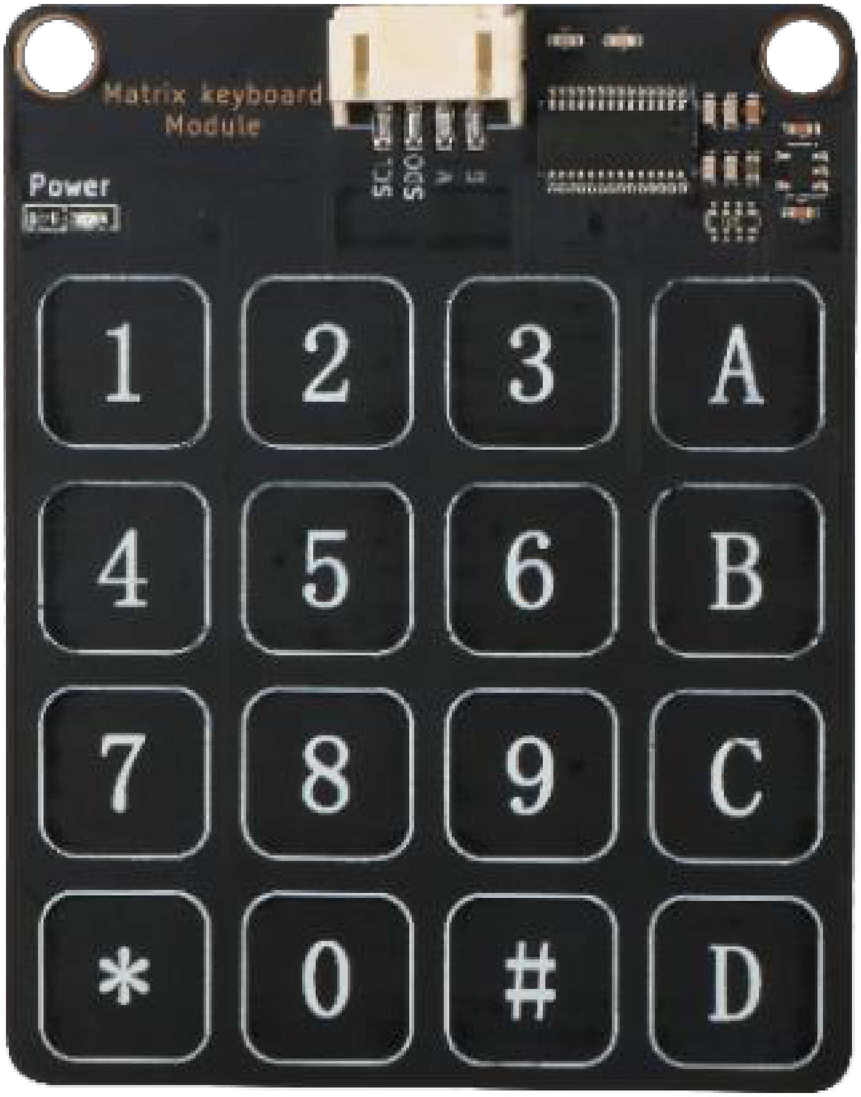
1.1.5 Chapter 5 Smart Access Control

Background

Preparation

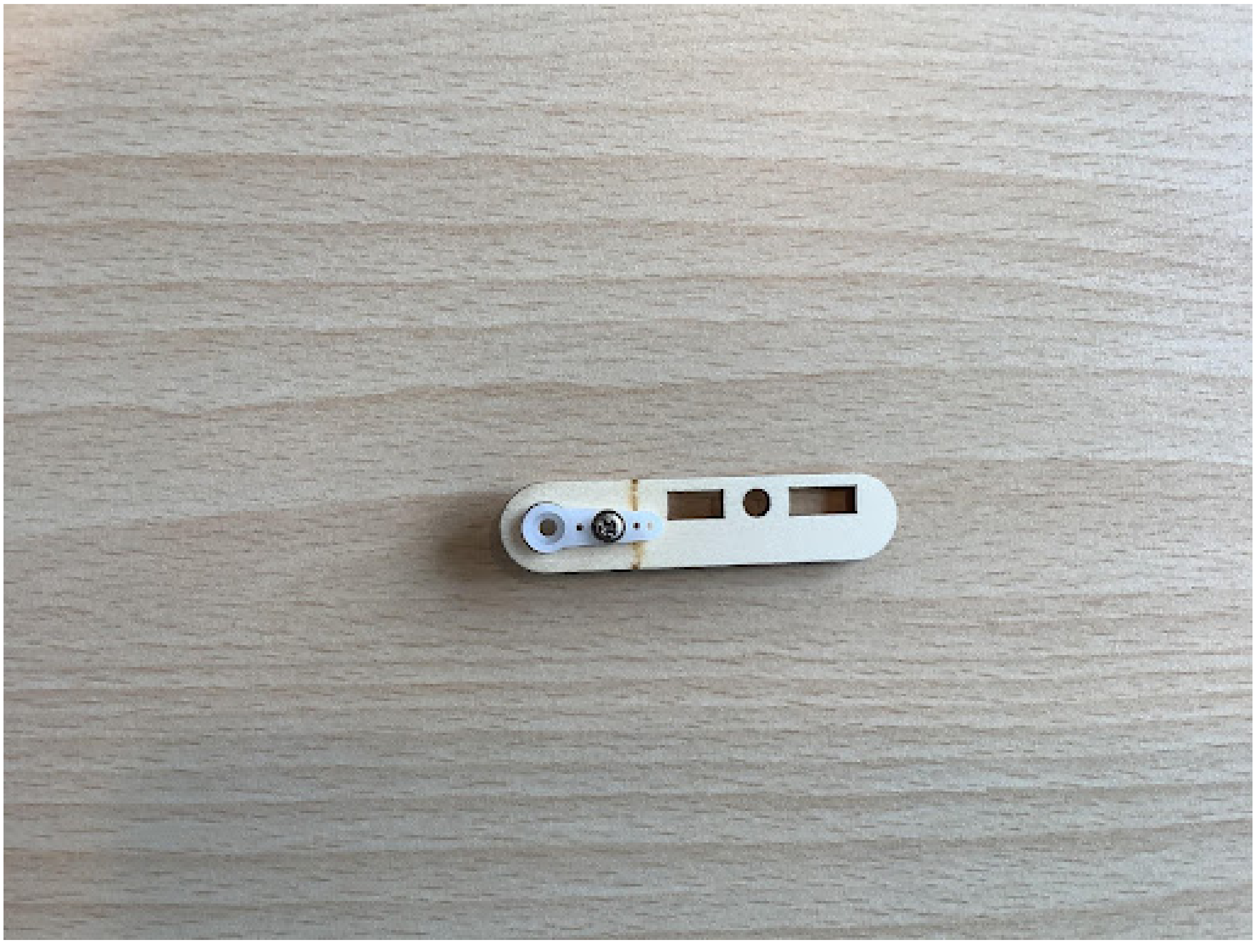
Learn About Smart Locks

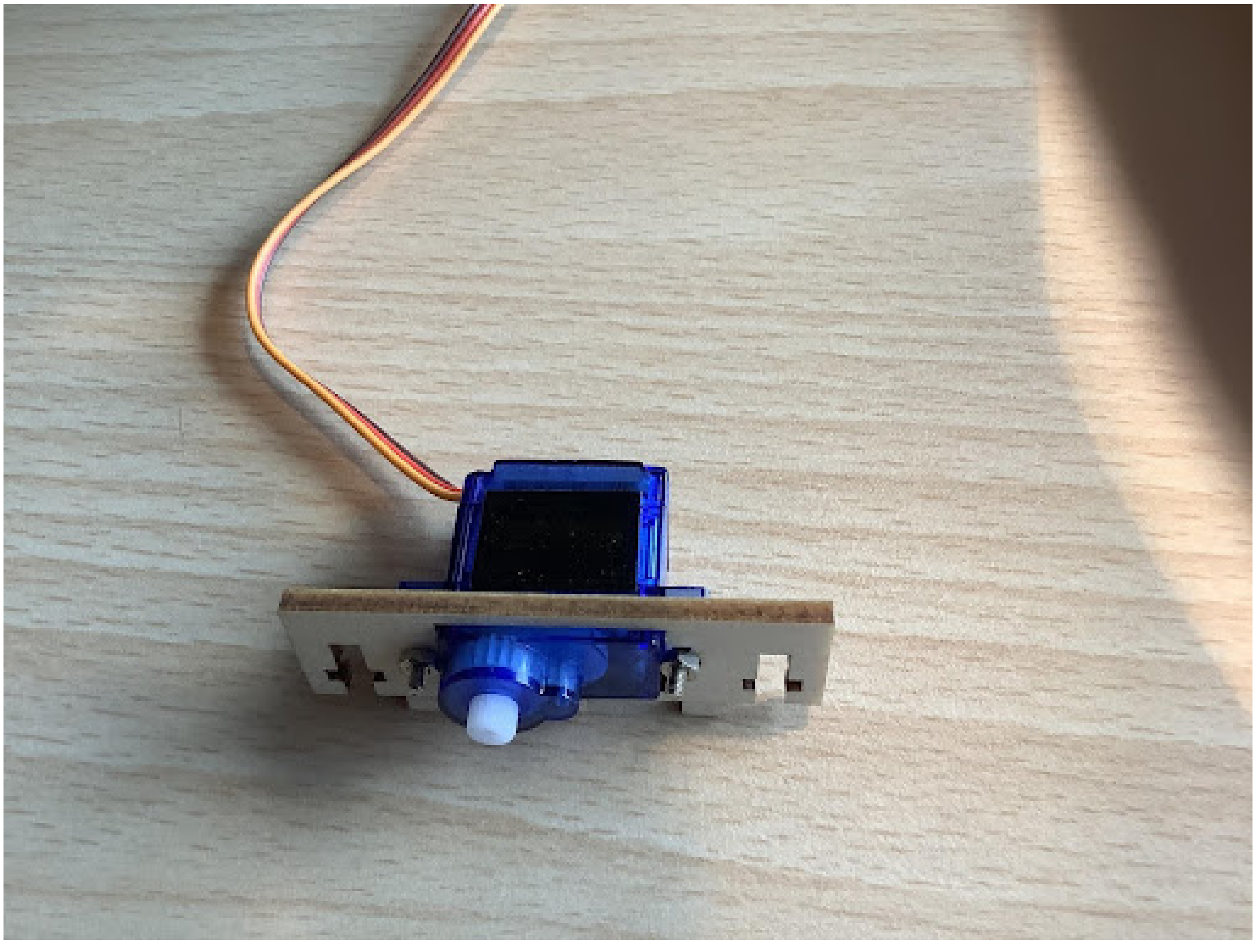
Learn About the Matrix Keyboard Sensor Module

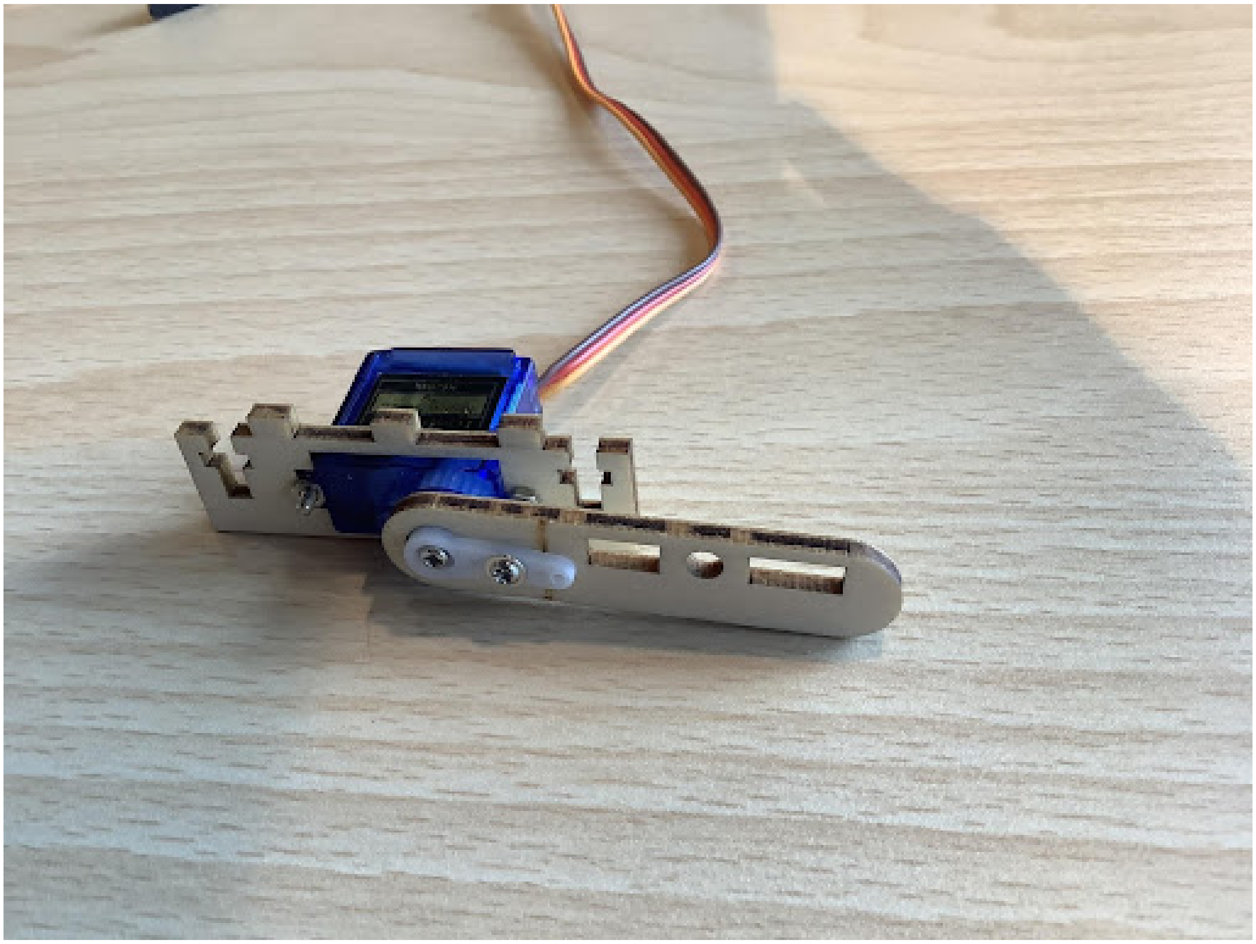


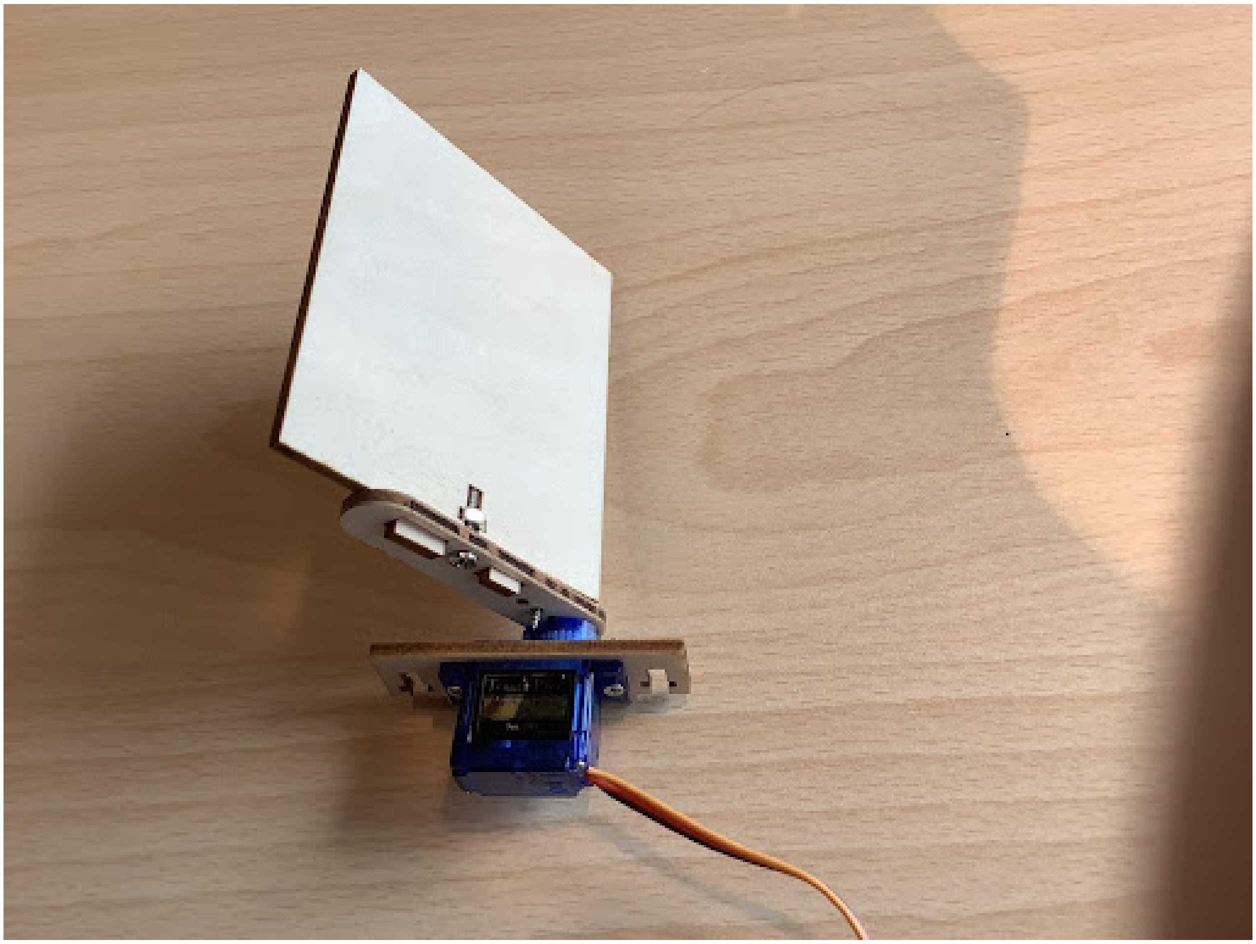
Keyboard Value | Hexadecimal value | Keyboard Value | Hexadecimal value | Keyboard Value | Hexadecimal value :– | :– | :– | :– | :– | :– 1|0xFFFE|7|0xFEFF|D|0x7FFF 2|0xFFFD|8|0xFDFF|C|0xF7FF 3|0xFFFB|9|0xFBFF|B|0xFF7F 4|0xFFEF|*|0xEFFF|A|0xFFF7 5|0xFFDF|0|0xDFFF| 6|0xFFBF|#|0xBFFF|

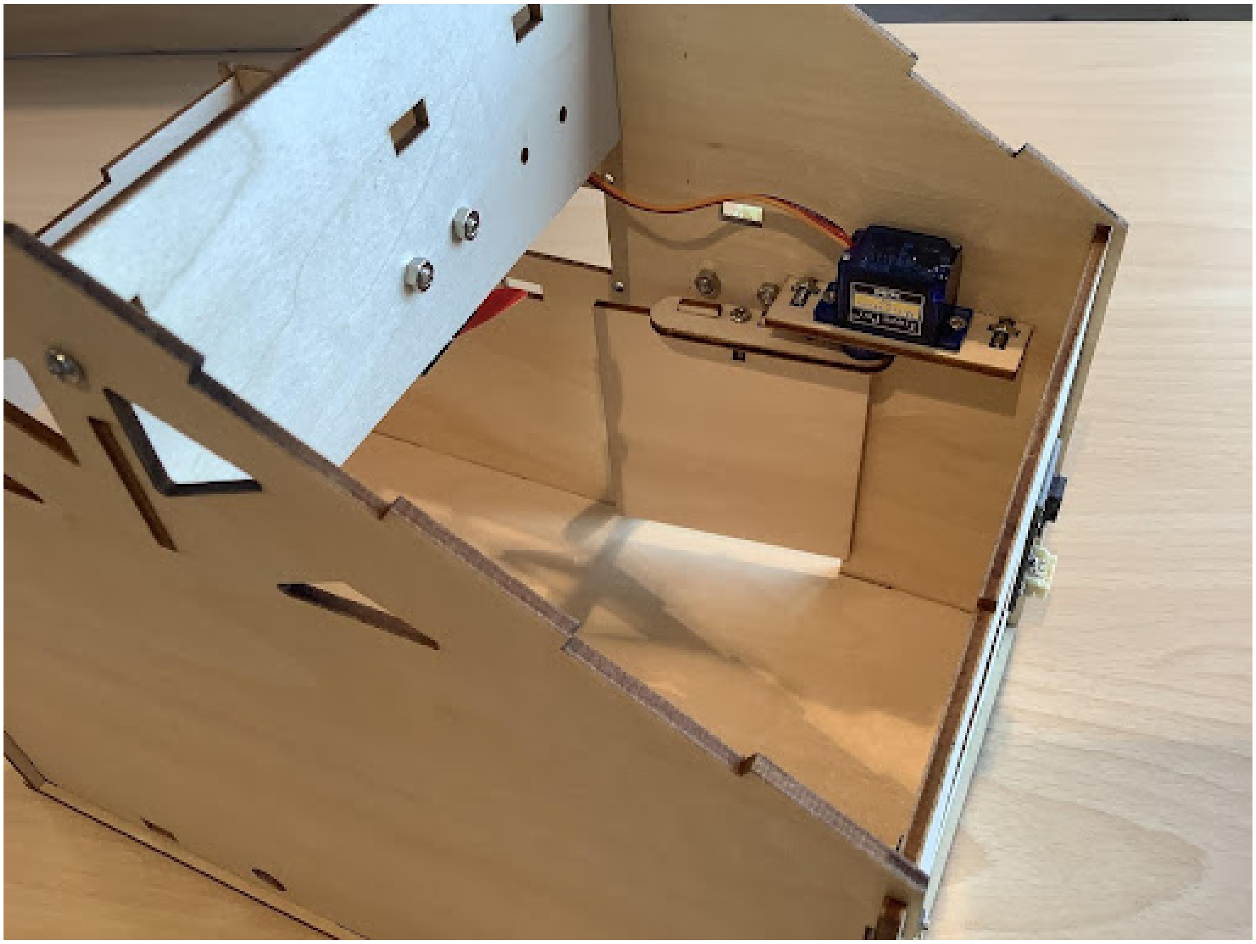
Installation of Smart Access Control

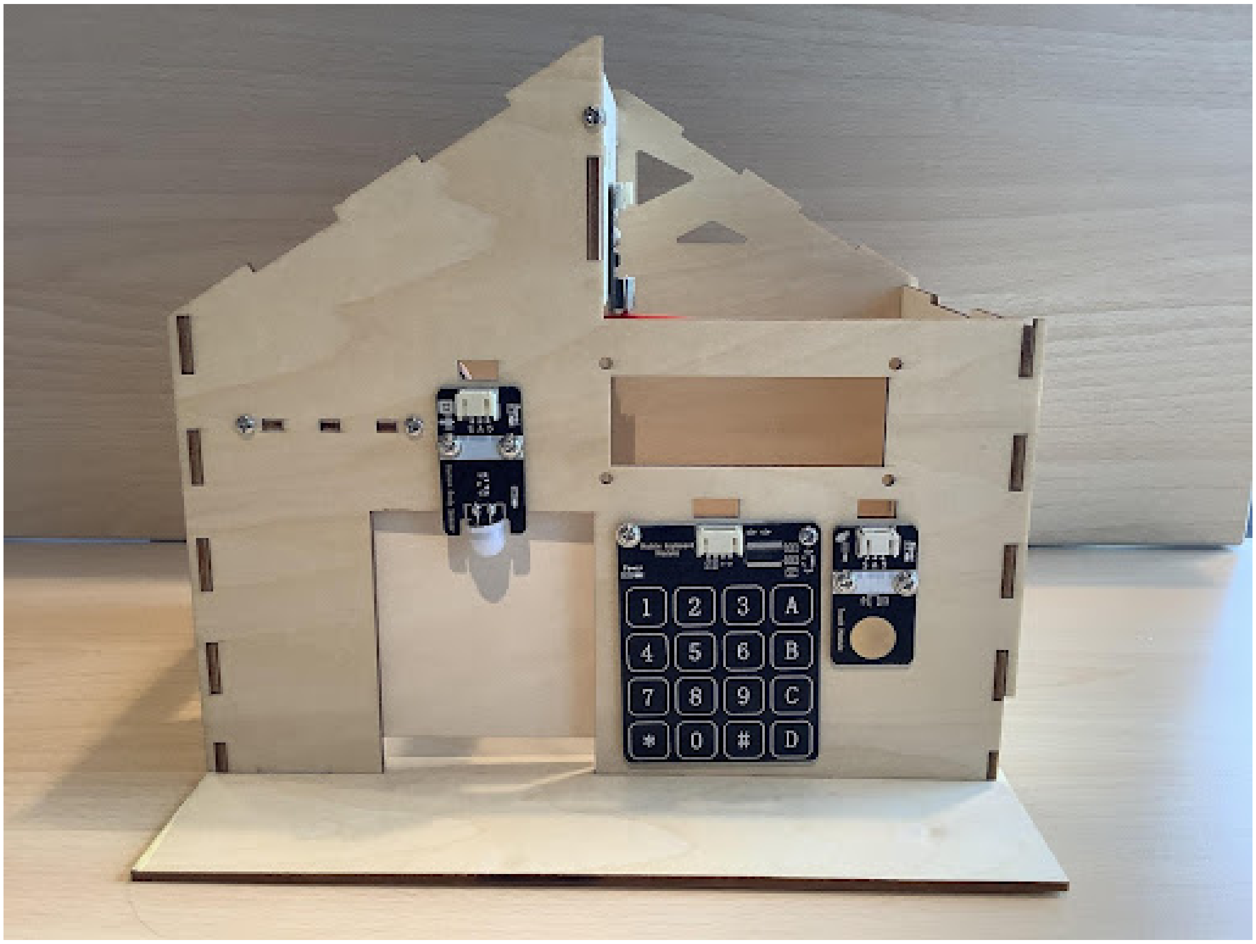










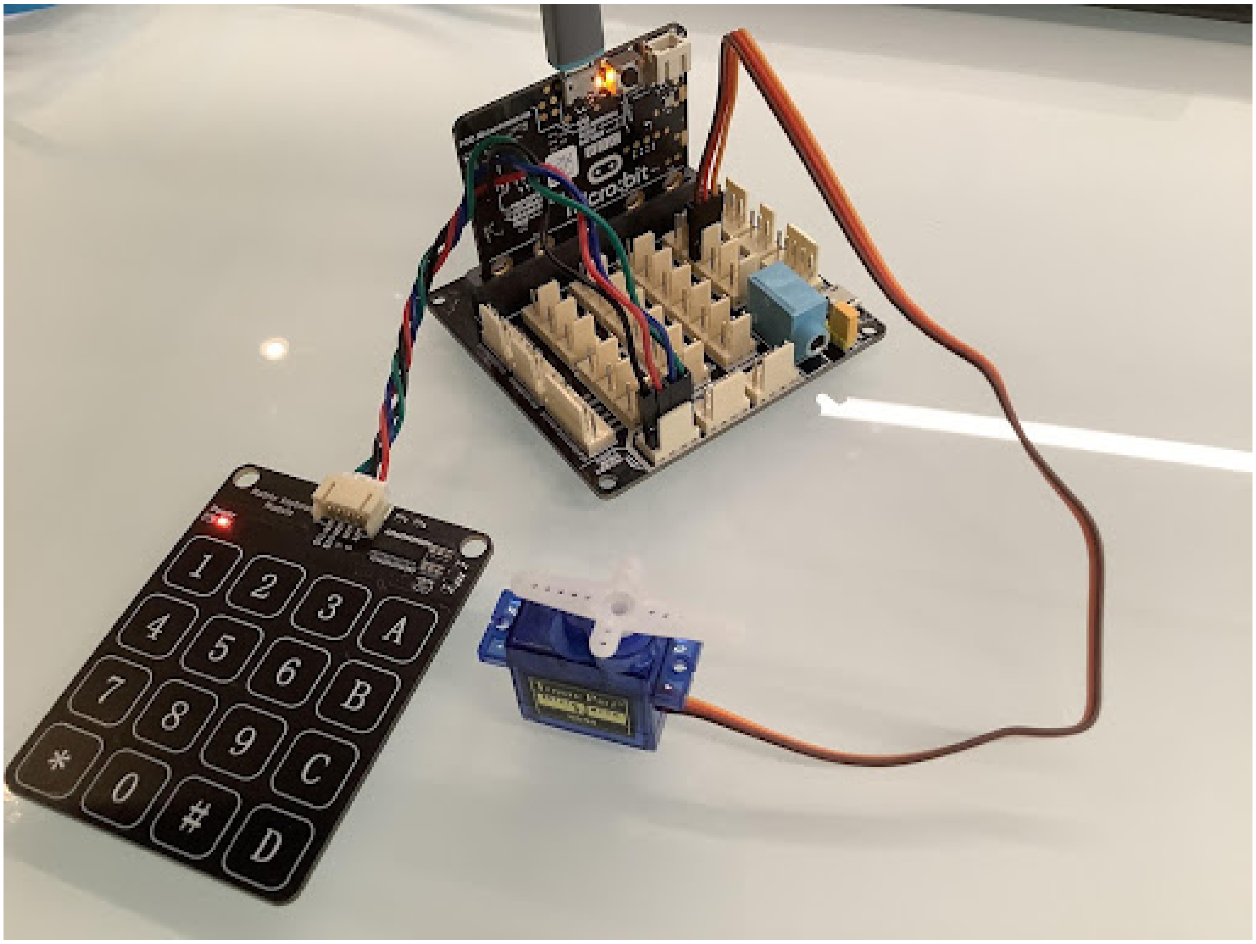


Program Design

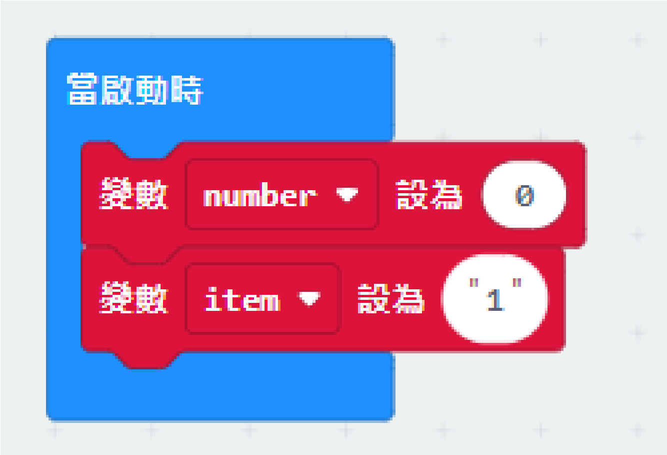
Algorithm Design

Hardware Connections

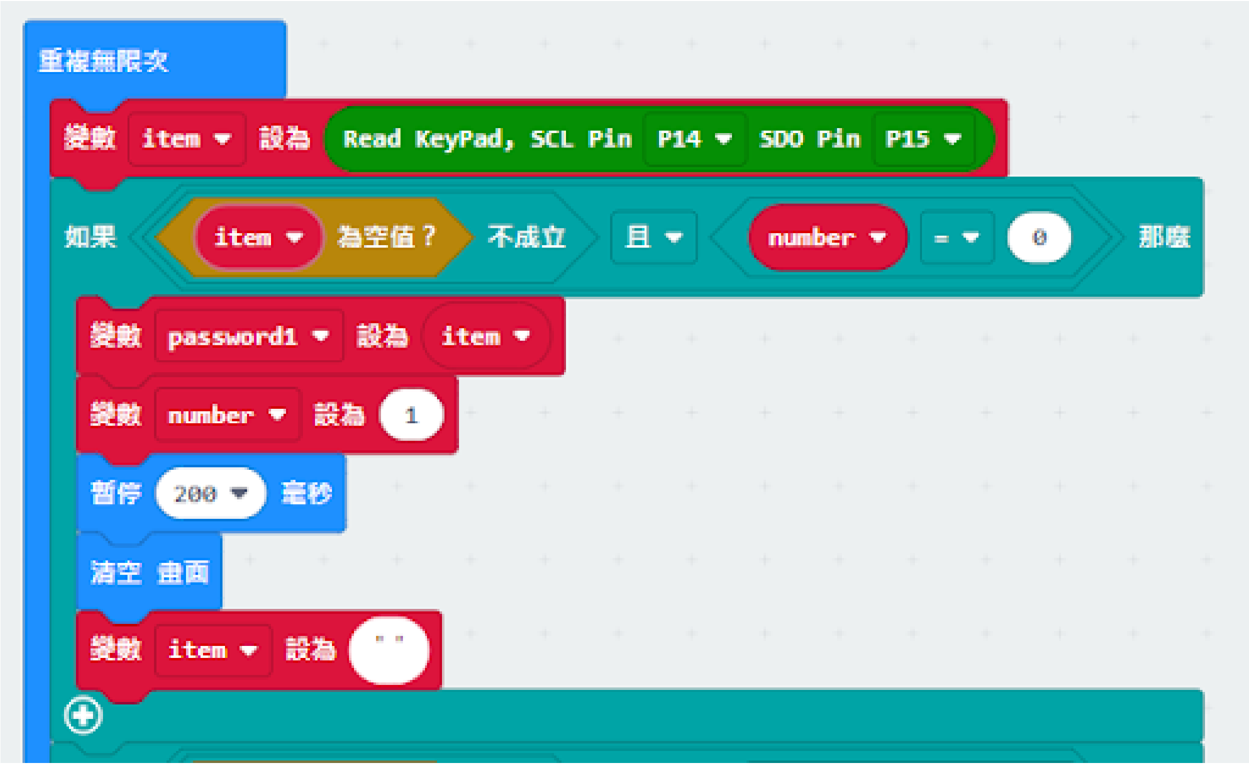
Sensors and Actuators | Main Control Board :– | :– Matrix Keyboard Sensor Module |P15(SCL)P16(SDO) Servo |P2



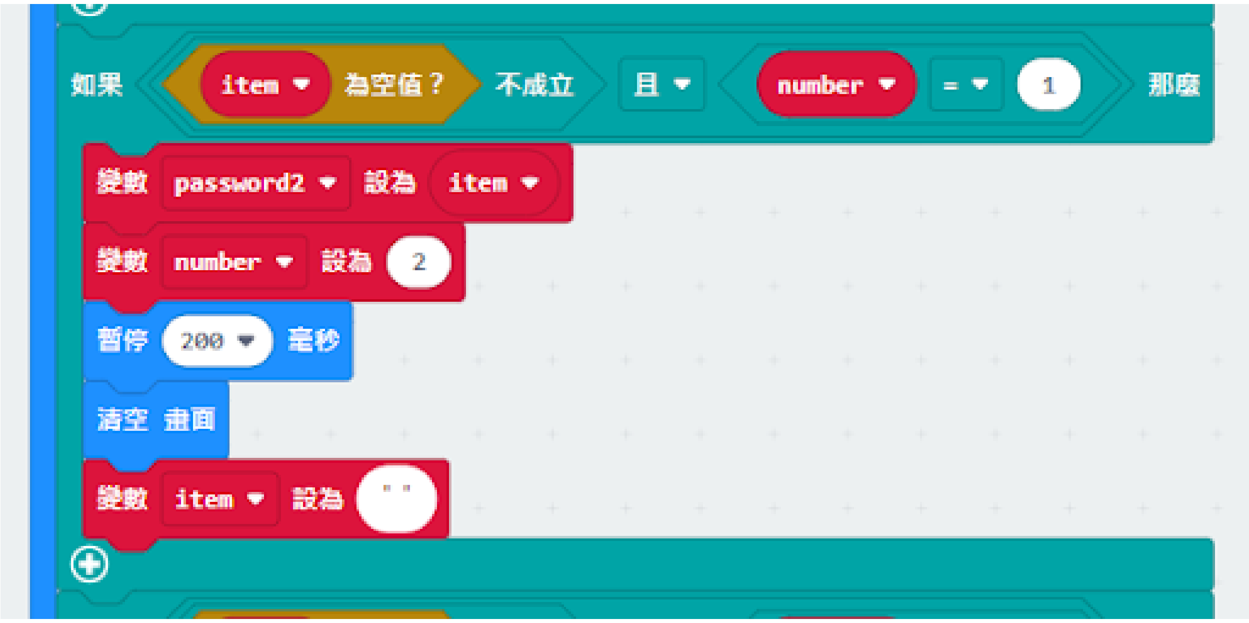
Sample Program



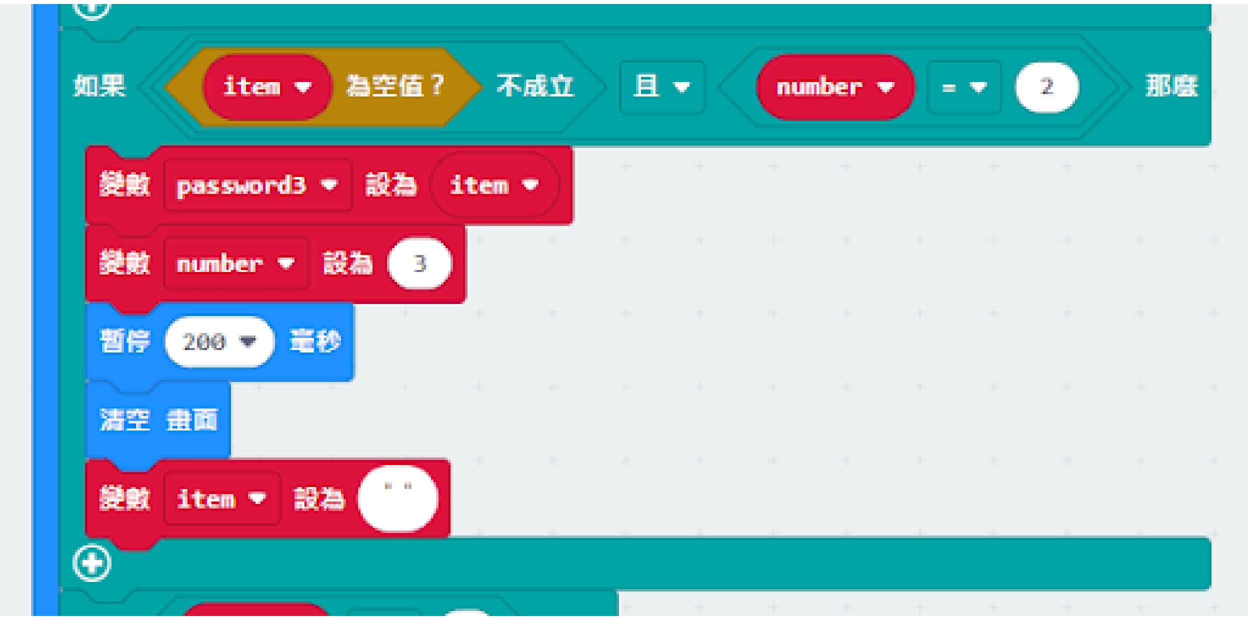
```
當啟動時
    變數 number 設為 0
    變數 item 設為 "1"
```



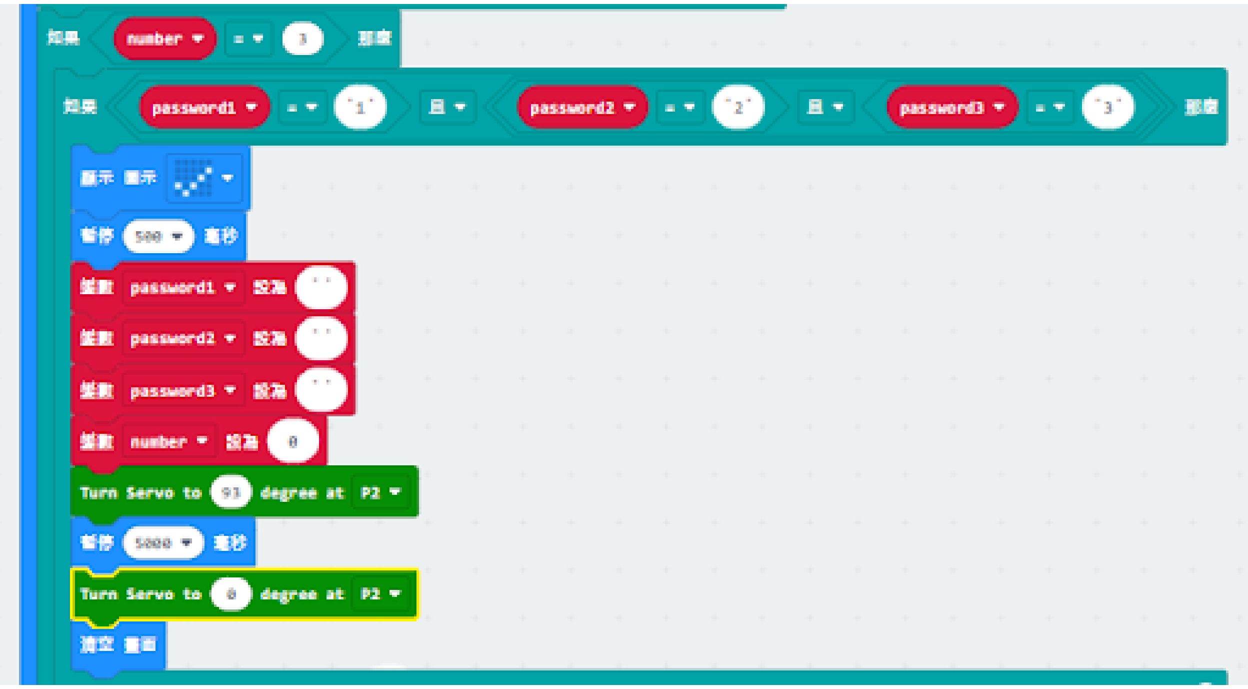
```
重複無限次
    變數 item 設為 Read KeyPad, SCL Pin P14 SDO Pin P15
    如果 item 為空值？ 不成立 且 number = 0 那麼
        變數 password1 設為 item
        變數 number 設為 1
        暫停 200 毫秒
        清空 畫面
        變數 item 設為 ""
```



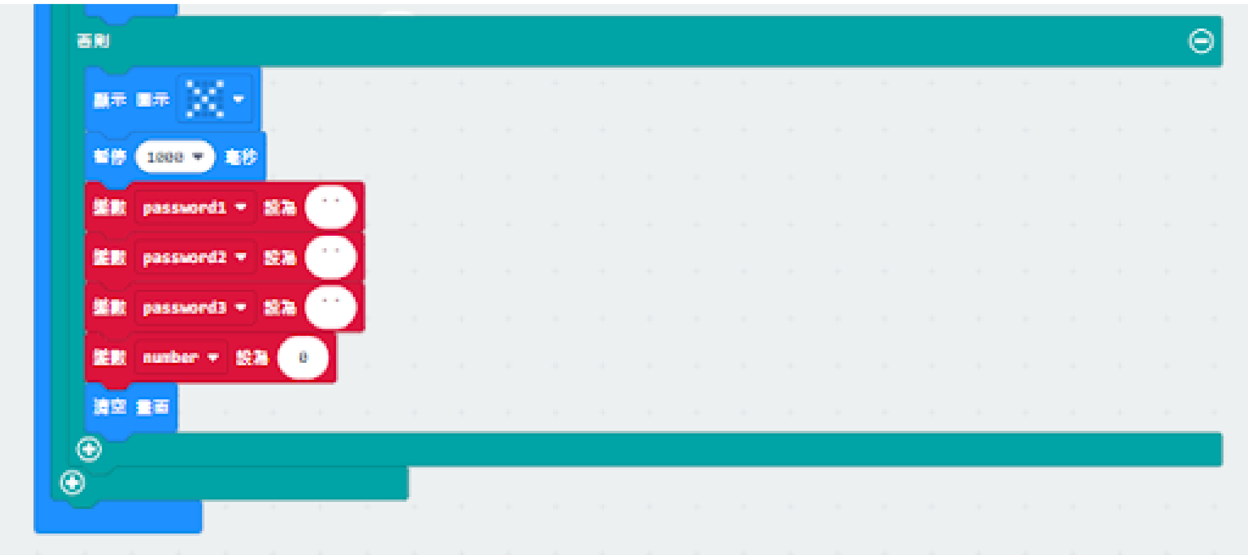
```
    如果 item 為空值？ 不成立 且 number = 1 那麼
        變數 password2 設為 item
        變數 number 設為 2
        暫停 200 毫秒
        清空 畫面
        變數 item 設為 ""
```



```
如果 item 為空值？ 不成立 且 number = 2 那麼
    變數 password3 設為 item
    變數 number 設為 3
    暫停 200 毫秒
    清空 畫面
    變數 item 設為 " "
```



```
如果 number = 3 那麼
    如果 password1 = "1" 且 password2 = "2" 且 password3 = "3" 那麼
        顯示 圖示
        暫停 500 毫秒
        變數 password1 設為 " "
        變數 password2 設為 " "
        變數 password3 設為 " "
        變數 number 設為 0
        Turn Servo to 93 degree at P2
        暫停 5000 毫秒
        Turn Servo to 0 degree at P2
        清空 畫面
```



```
否則
    顯示 圖示
    暫停 1000 毫秒
    變數 password1 設為 " "
    變數 password2 設為 " "
    變數 password3 設為 " "
    變數 number 設為 0
    清空 畫面
```

Conclusion

1.1.6 Chapter 6 Light-controlled Automatic Windows

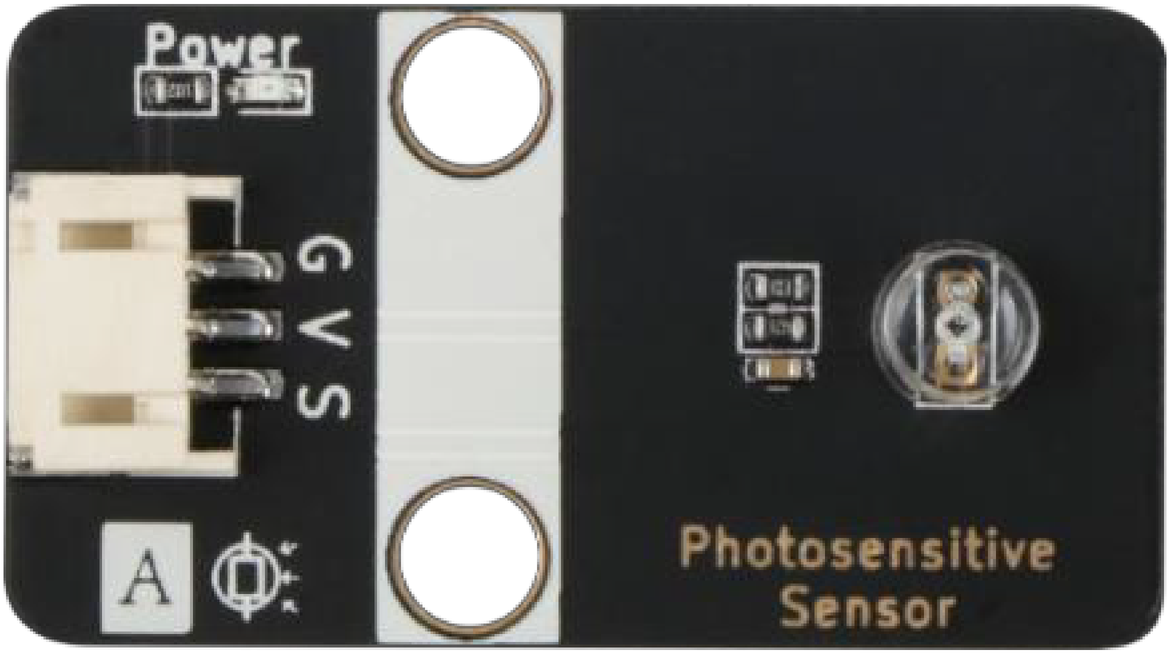
Background

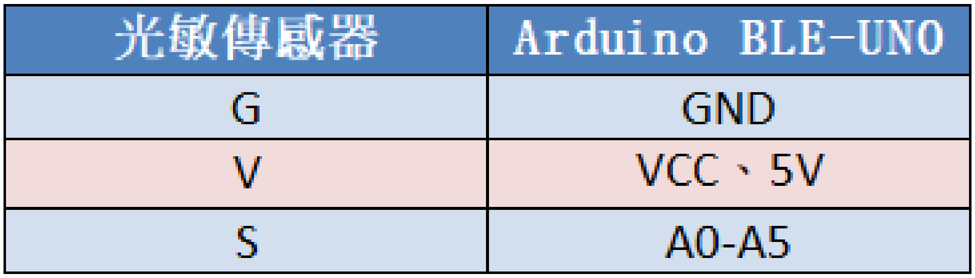
Preparation

Learn About the Principle of Light-controlled Automatic Windows

Learn About Photoreceptors and Digital Tube Displays

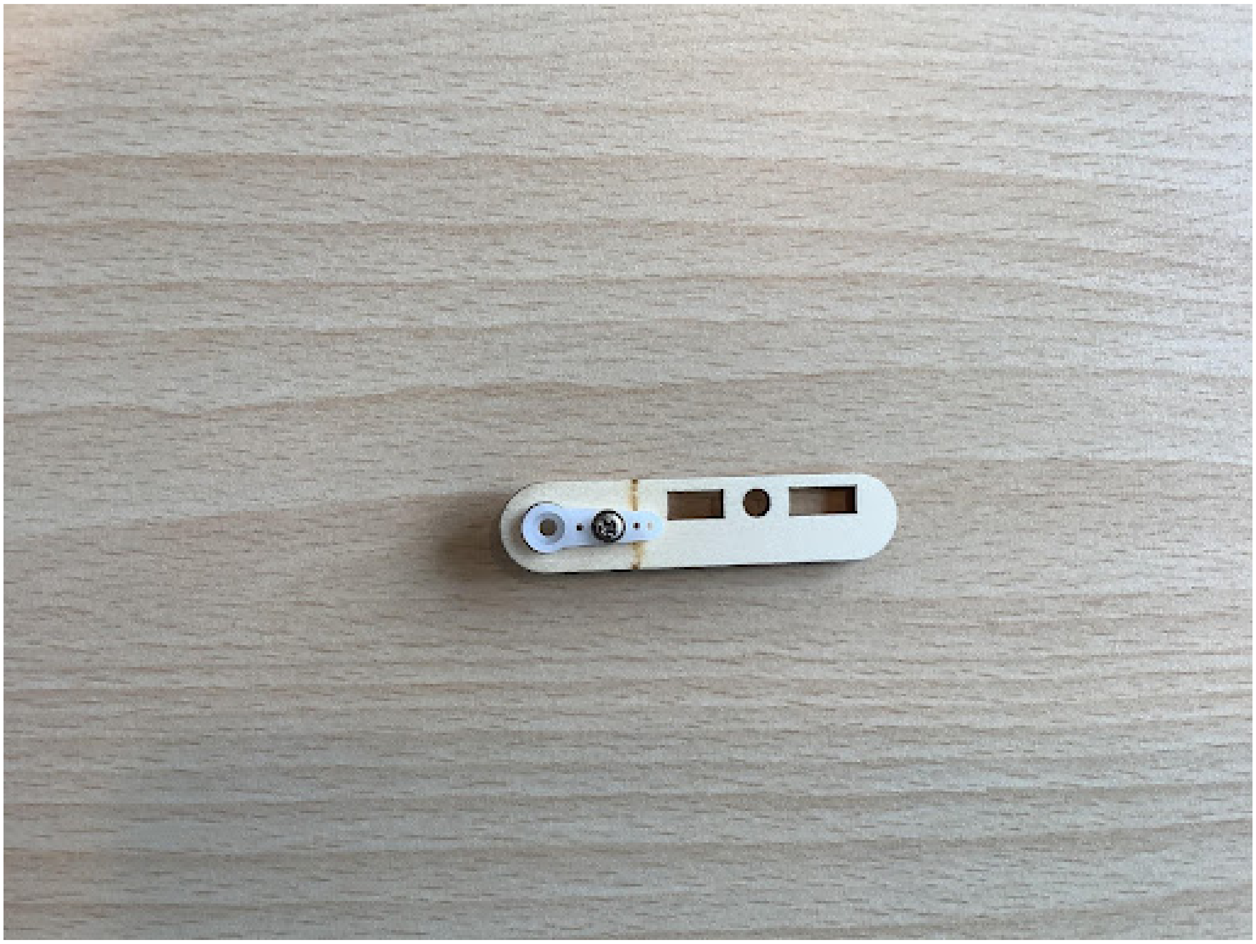
Photosensitive sensor

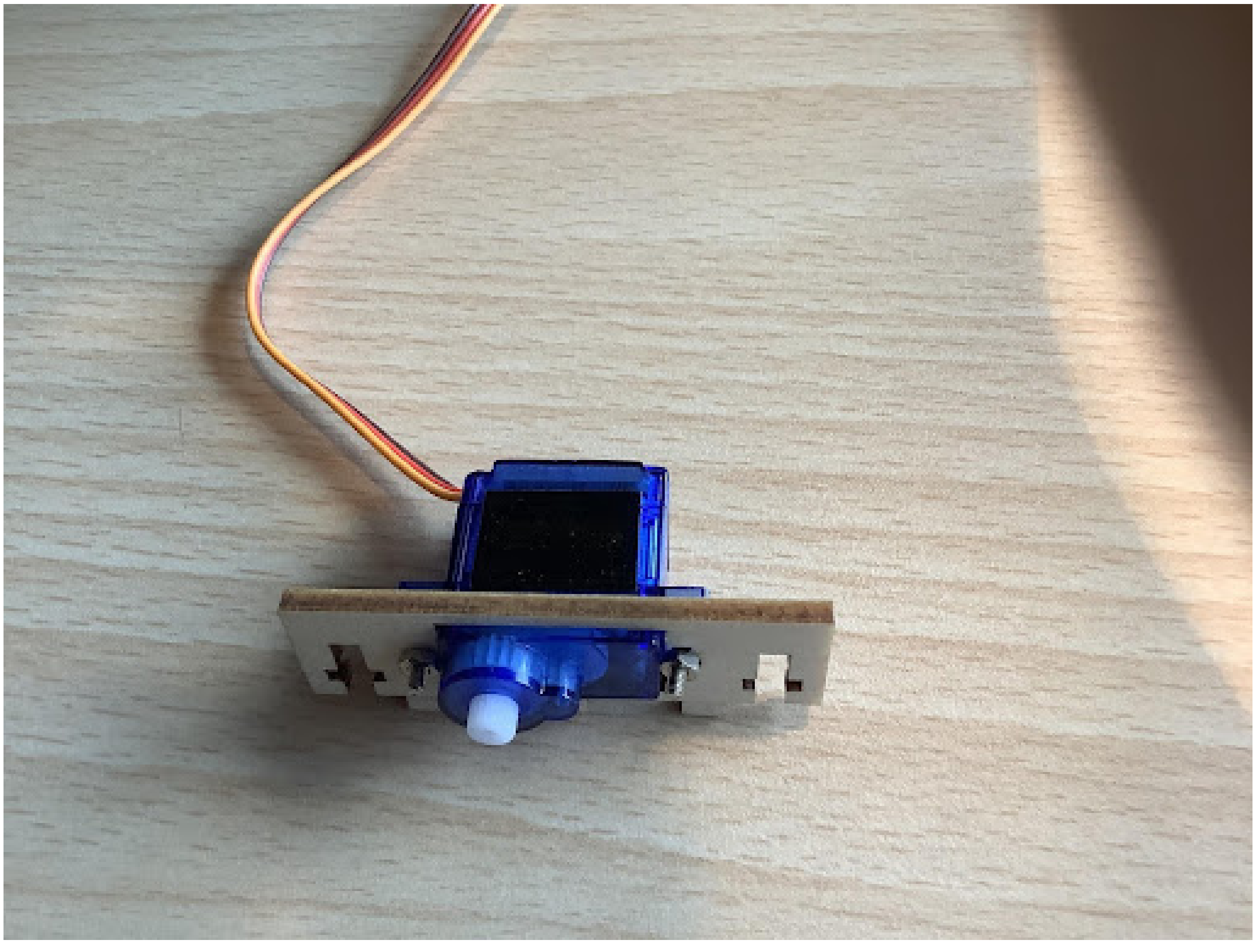


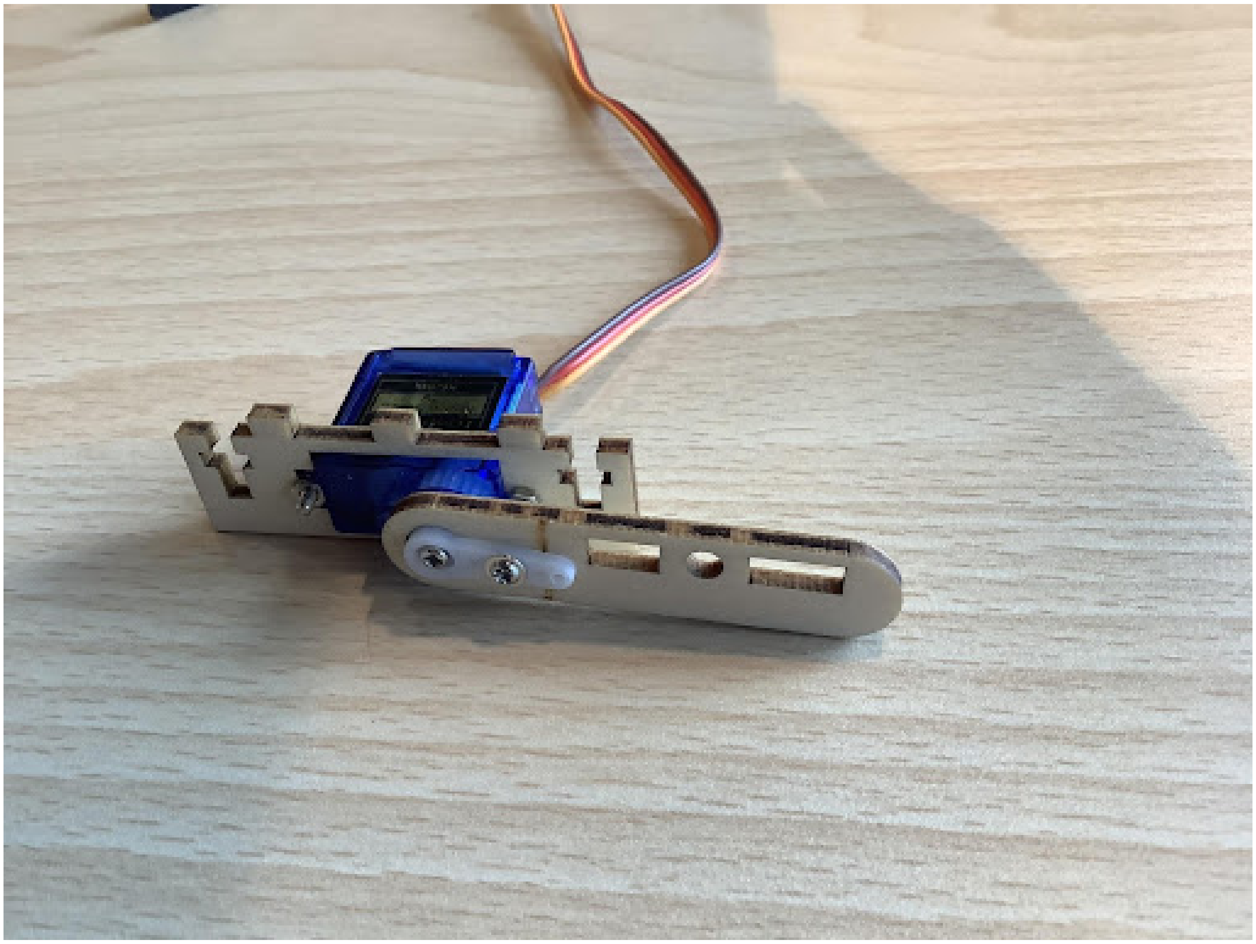


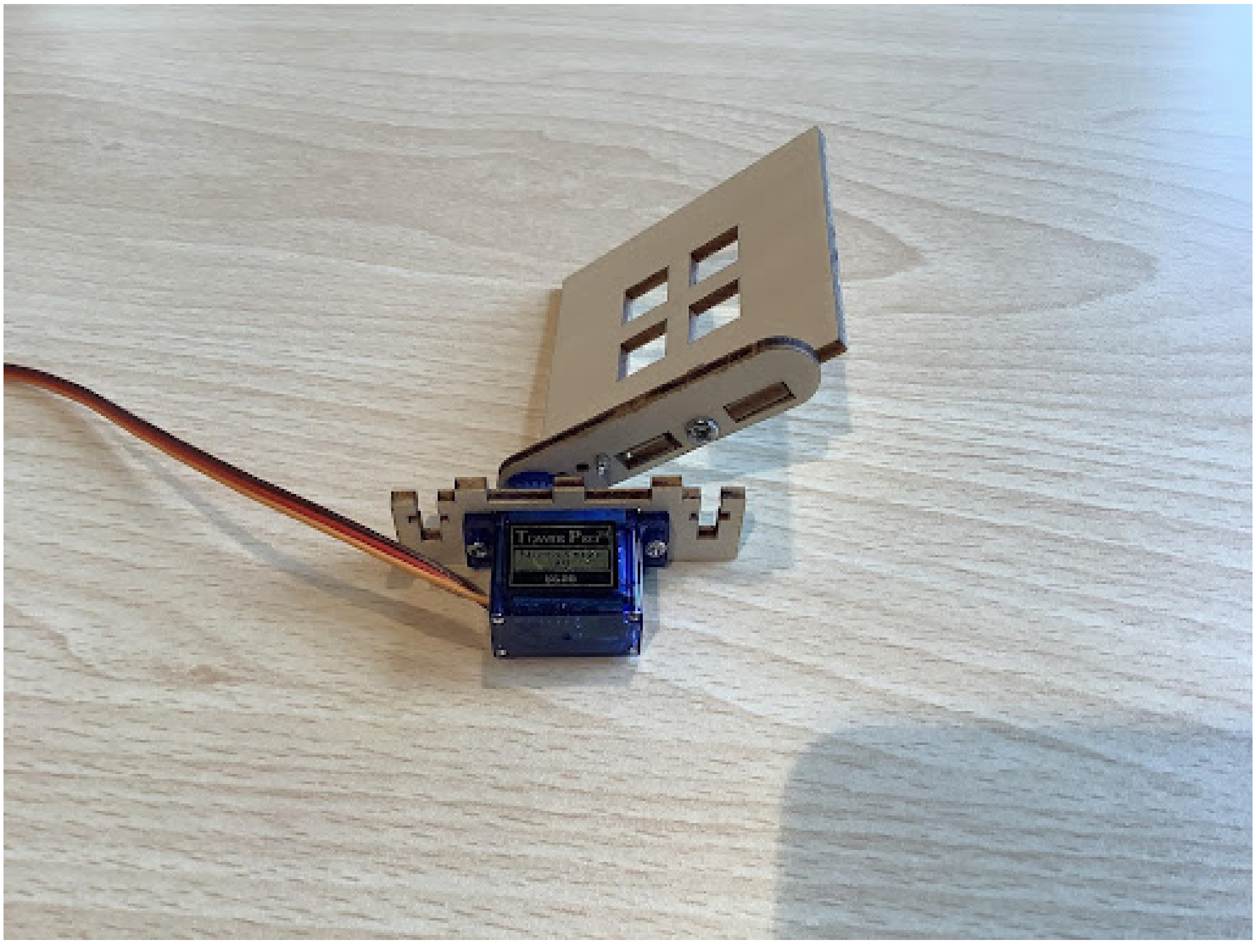
光敏傳感器	Arduino BLE-UNO
G	GND
V	VCC、5V
S	A0-A5

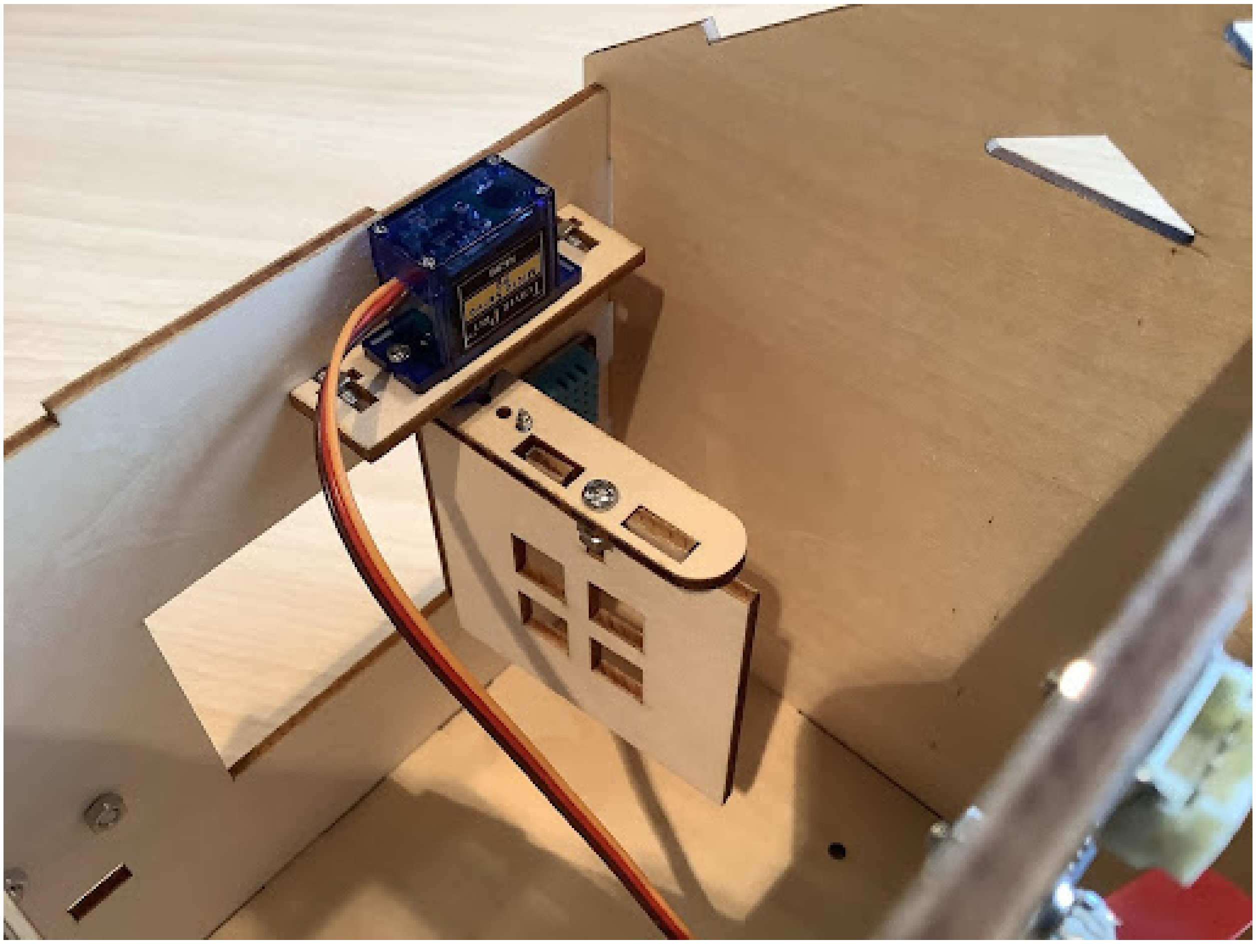
Installation Of Light-controlled Automatic Window

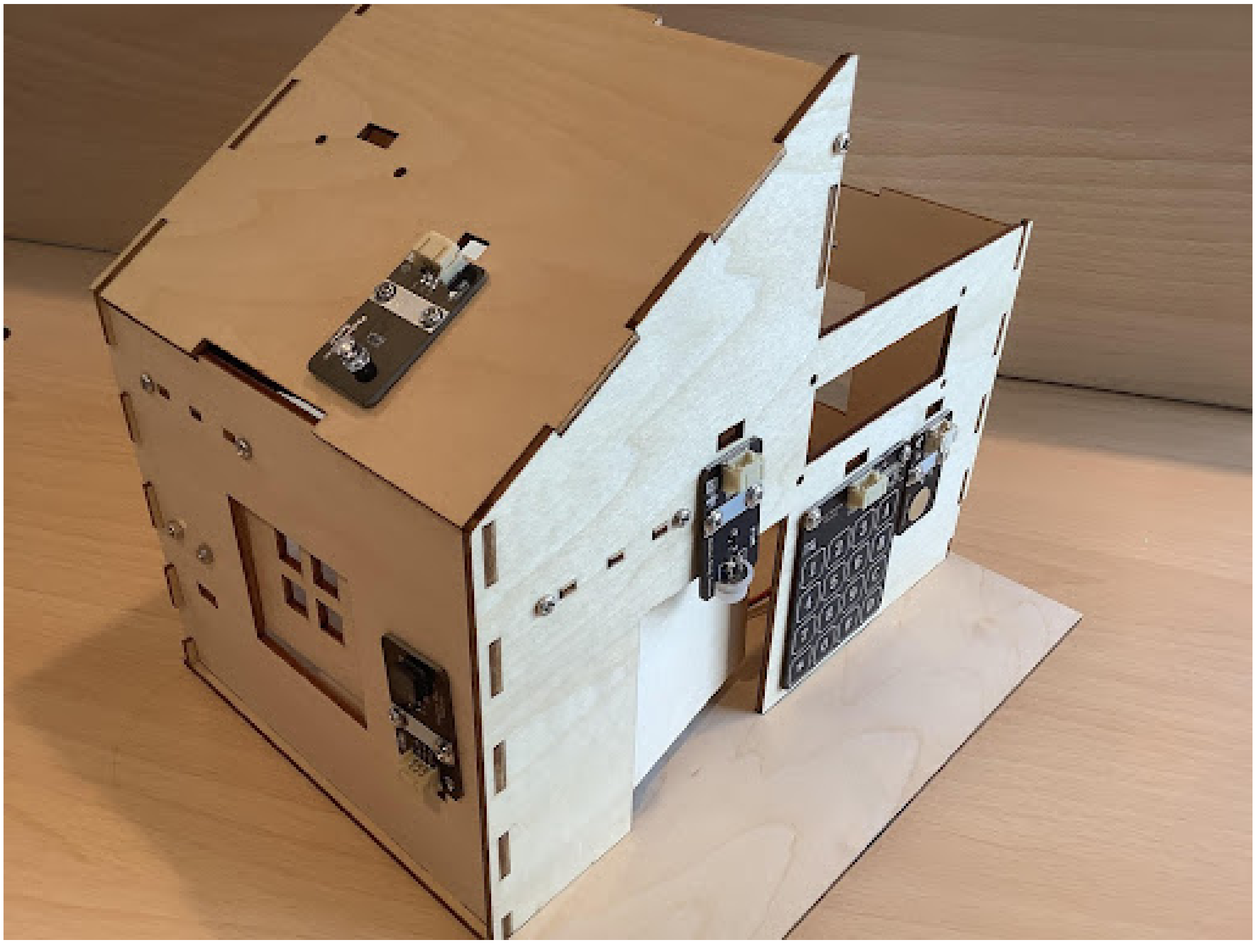






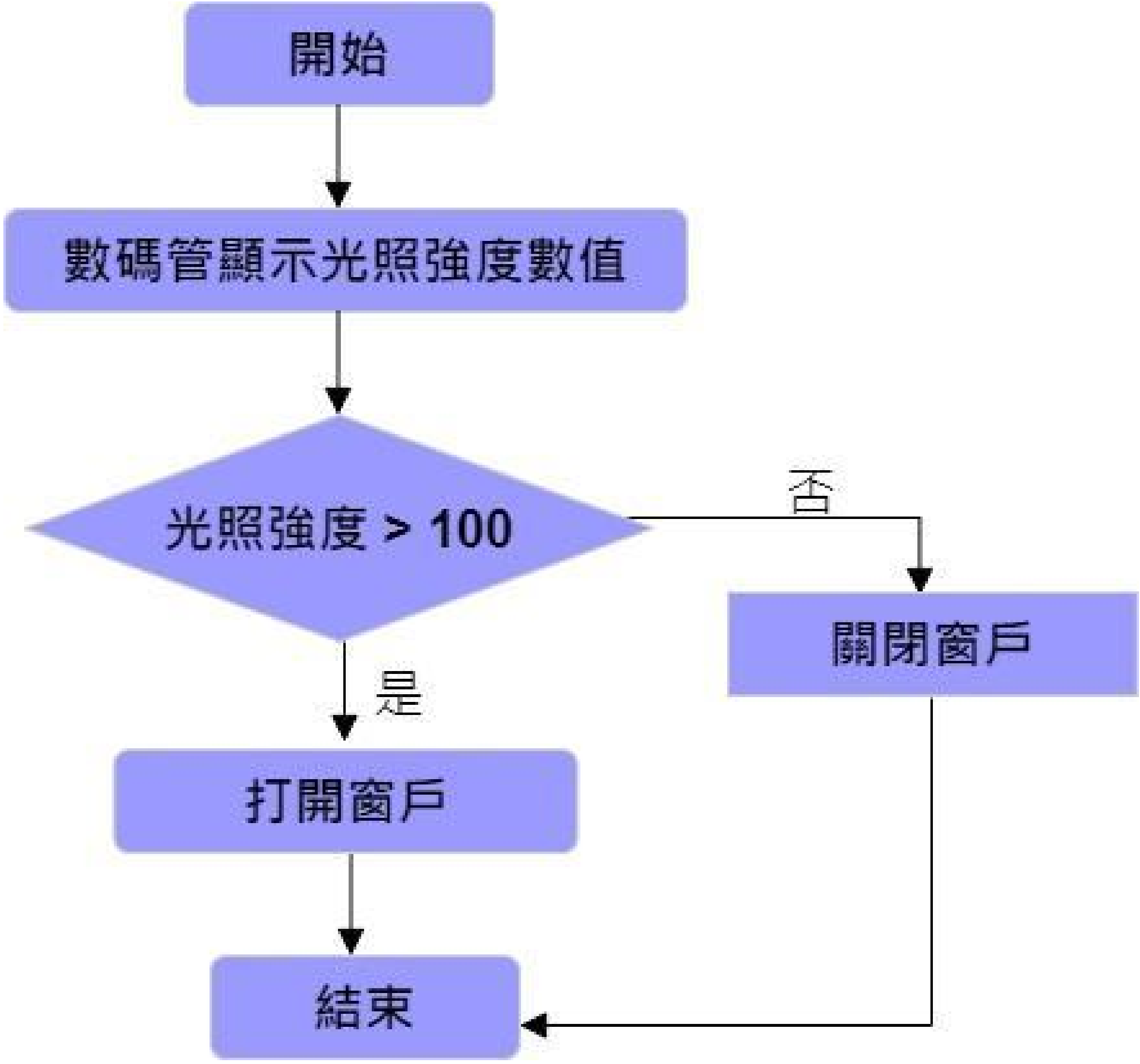






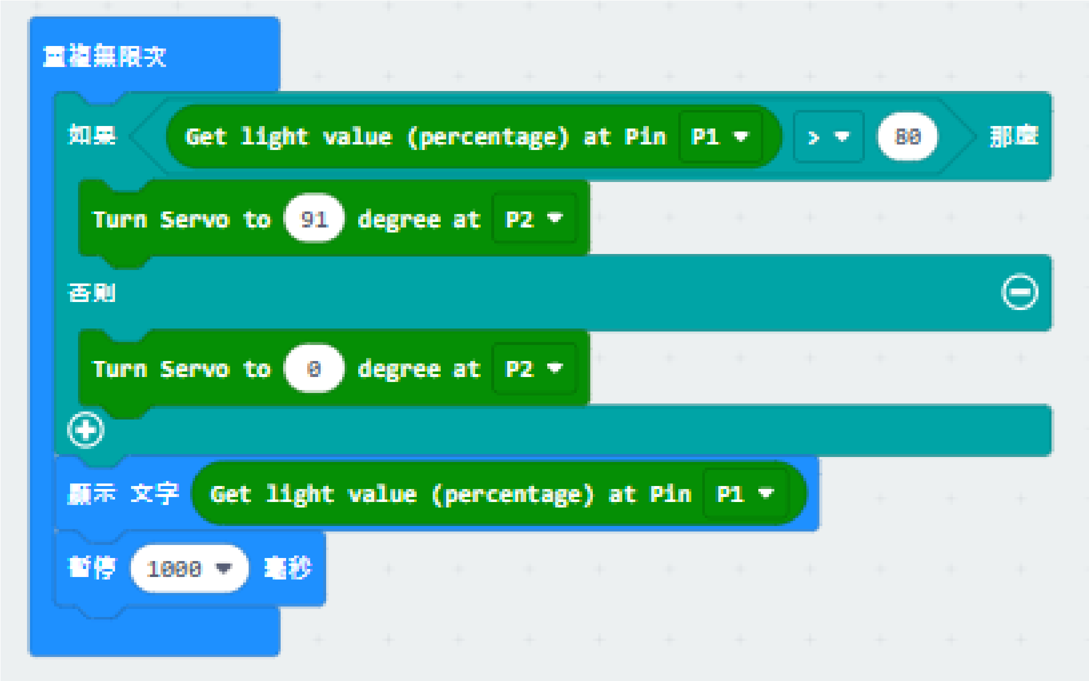
Program Design

Algorithm Design



Hardware Connections

Sample Program



```
重複無限次
  如果 Get light value (percentage) at Pin P1 > 80 那麼
    Turn Servo to 91 degree at P2
  否則
    Turn Servo to 0 degree at P2
  顯示 文字 Get light value (percentage) at Pin P1
  暫停 1000 毫秒
```

Conclusion

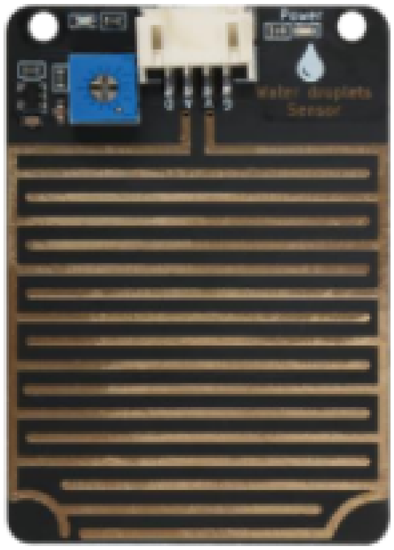
1.1.7 Chapter 7 Smart Rain Control Windows

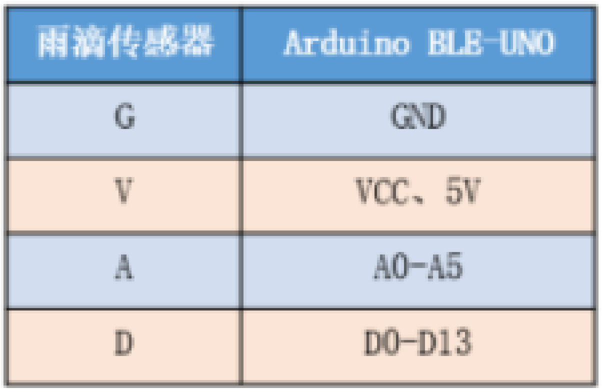
Background

Preparation

Principle of Smart Rain Control Window

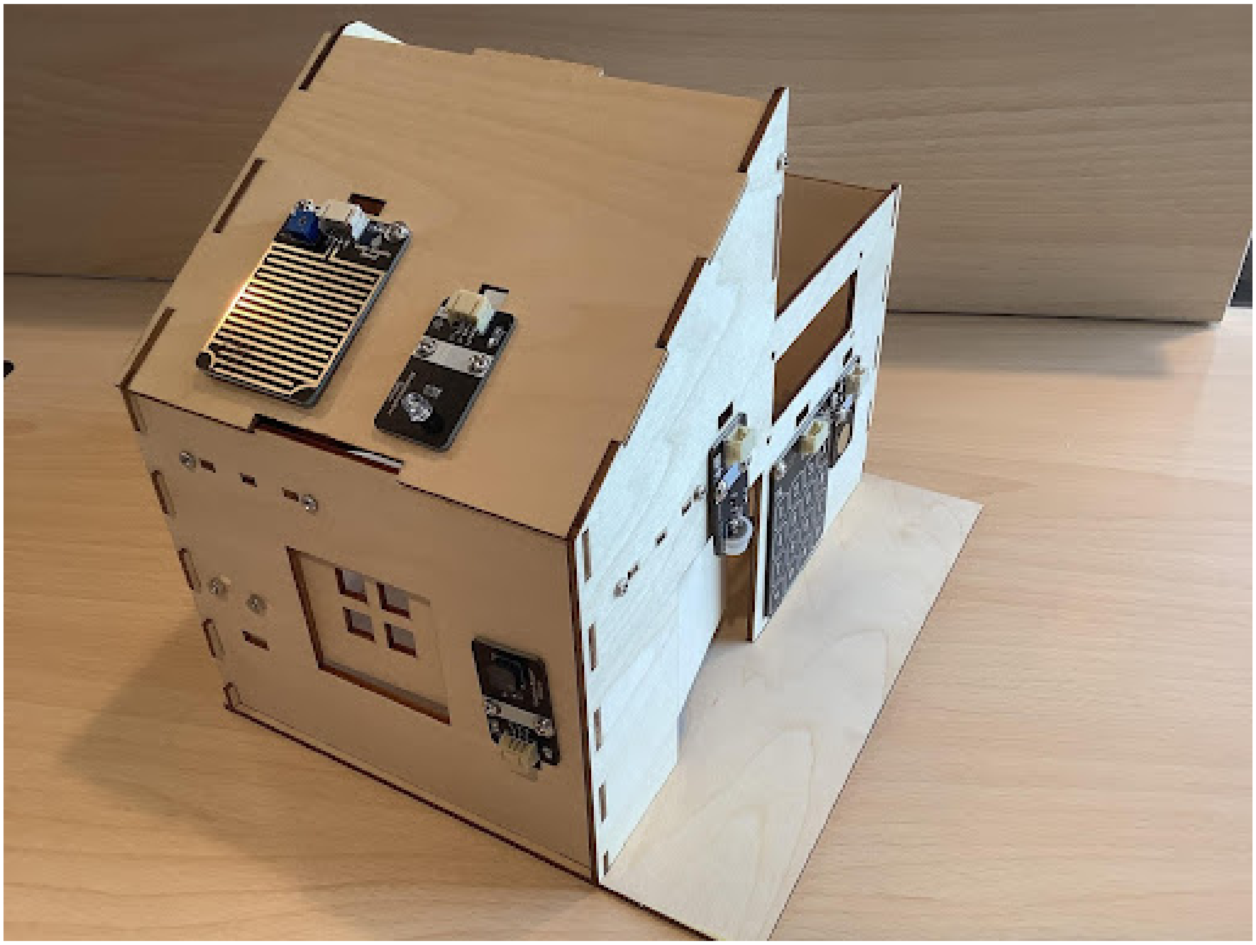
Learn About Raindrop Sensor





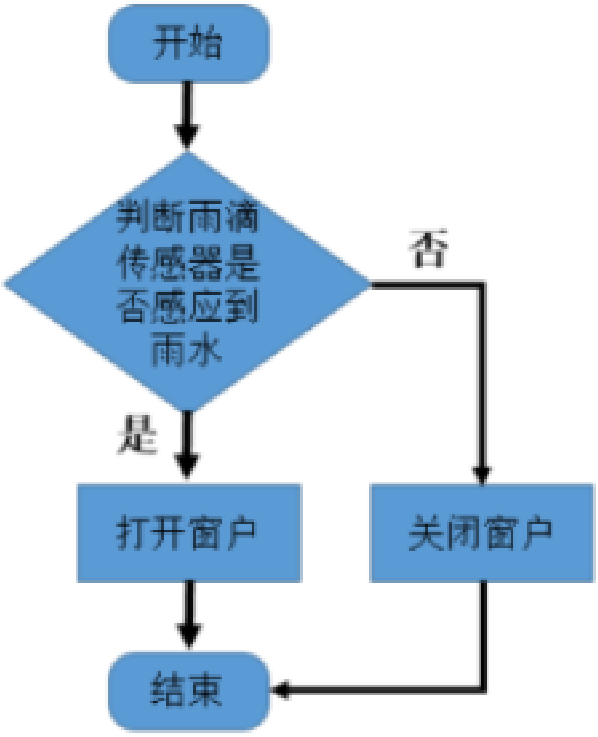
雨滴传感器	Arduino BLE-UNO
G	GND
V	VCC、5V
A	A0-A5
D	D0-D13

Construction of Automatic Smart Window



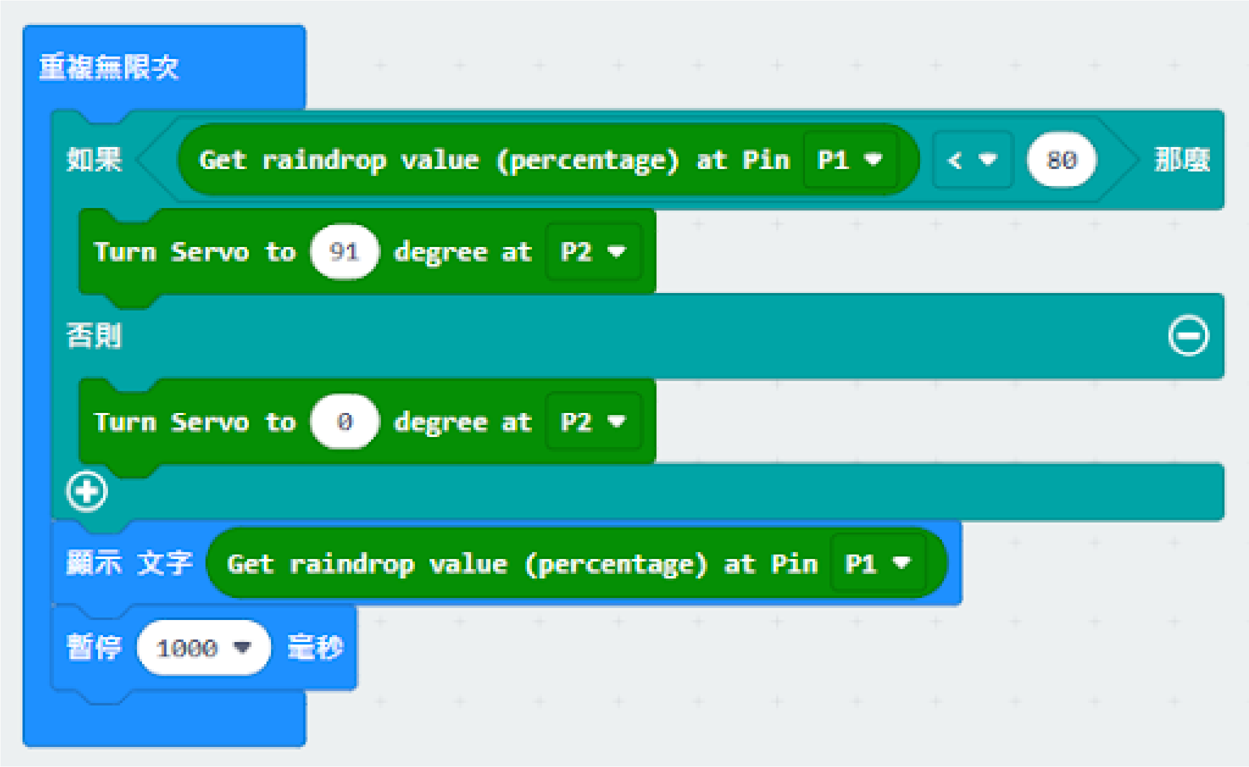
Program Design

Algorithm Design



Hardware Connections

Sample Program



```
重複無限次
  如果 Get raindrop value (percentage) at Pin P1 < 80 那麼
    Turn Servo to 91 degree at P2
  否則
    Turn Servo to 0 degree at P2
  顯示 文字 Get raindrop value (percentage) at Pin P1
  暫停 1000 毫秒
```

Conclusion

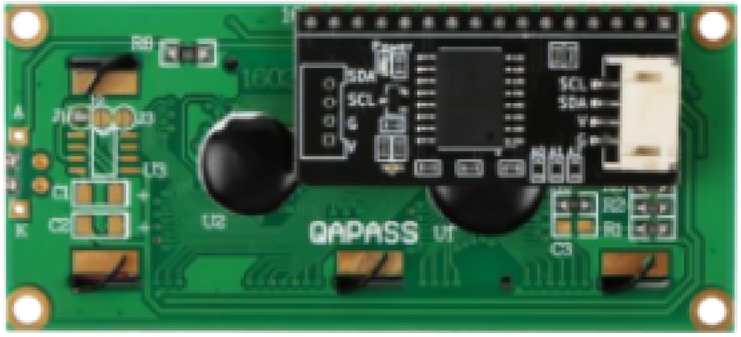
1.1.8 Chapter 8 Environmental Monitoring System

Background

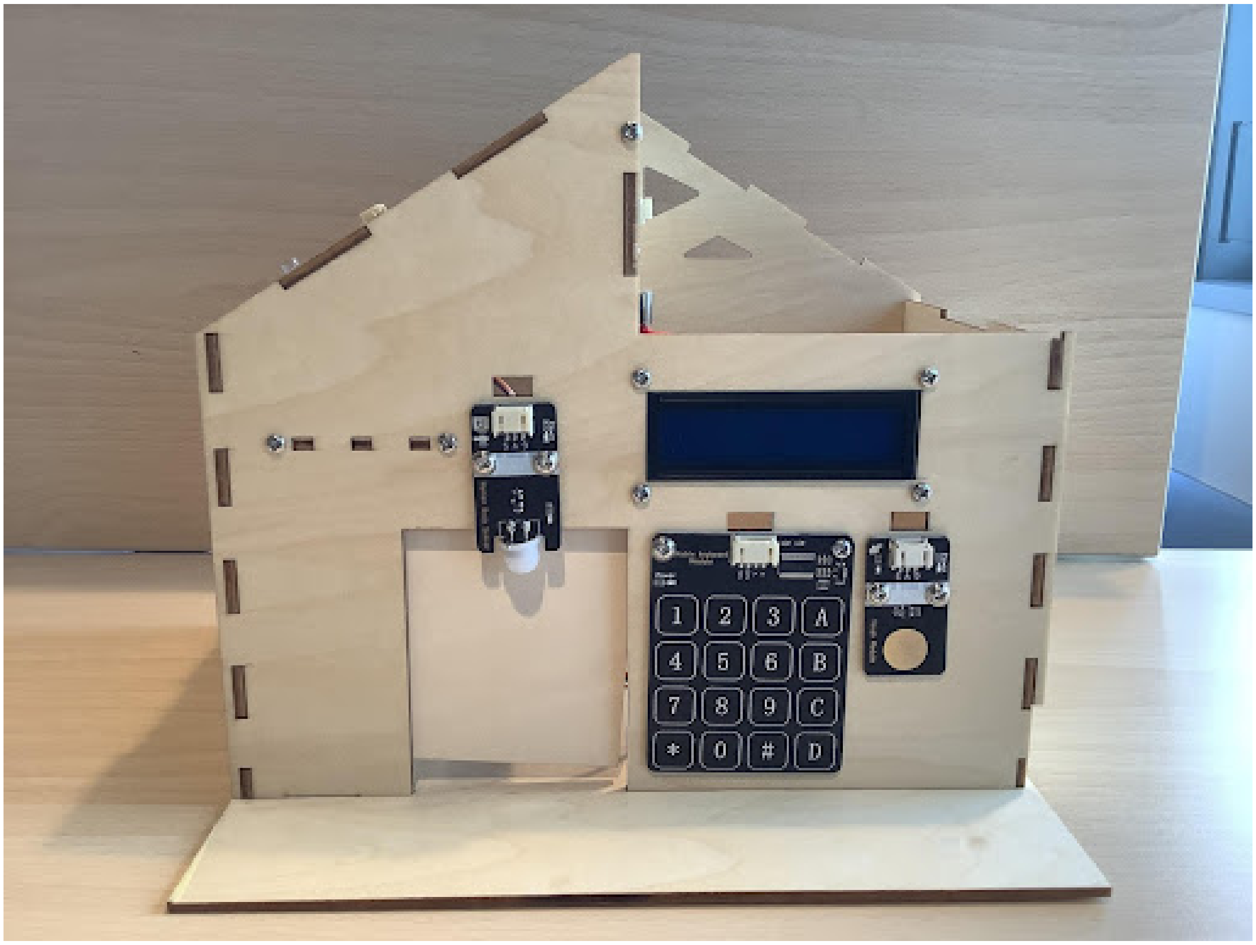
Preparation

Learn About Environmental Monitoring Systems

Learn About LCD Displays

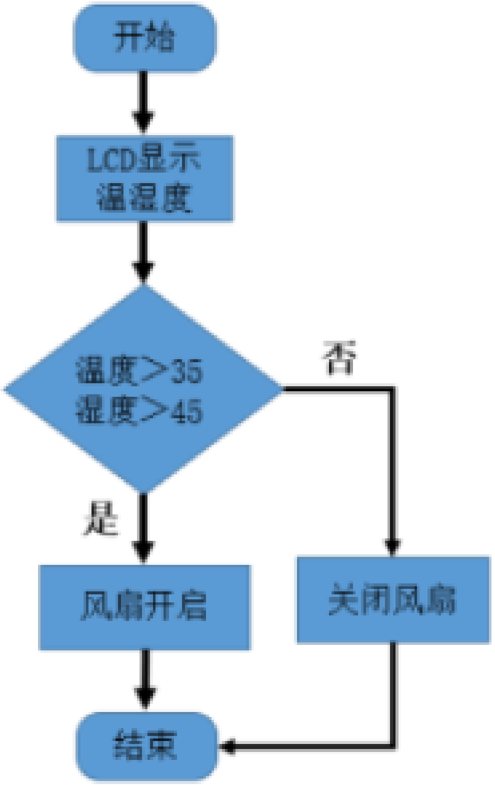


Installation of Environmental Monitoring System



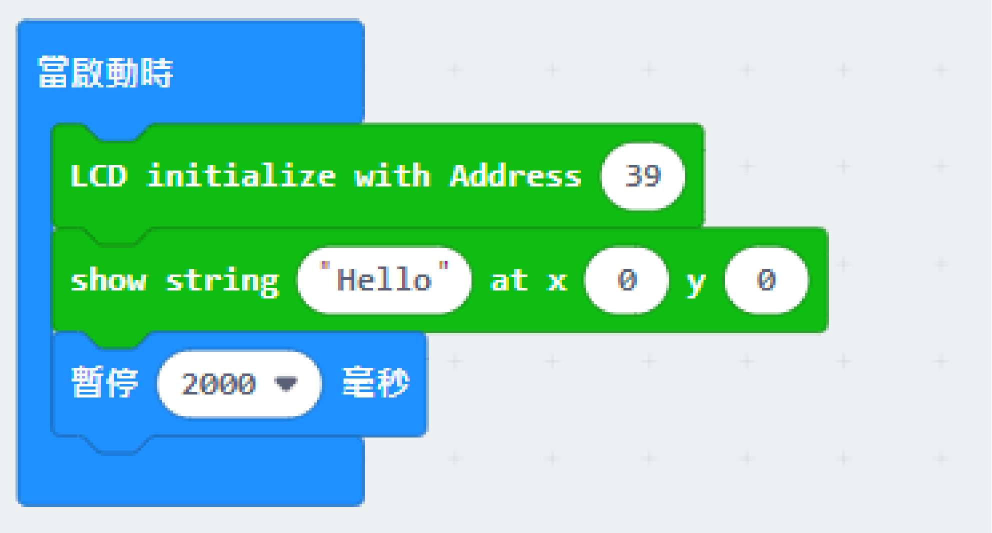
Program Design

Algorithm Design

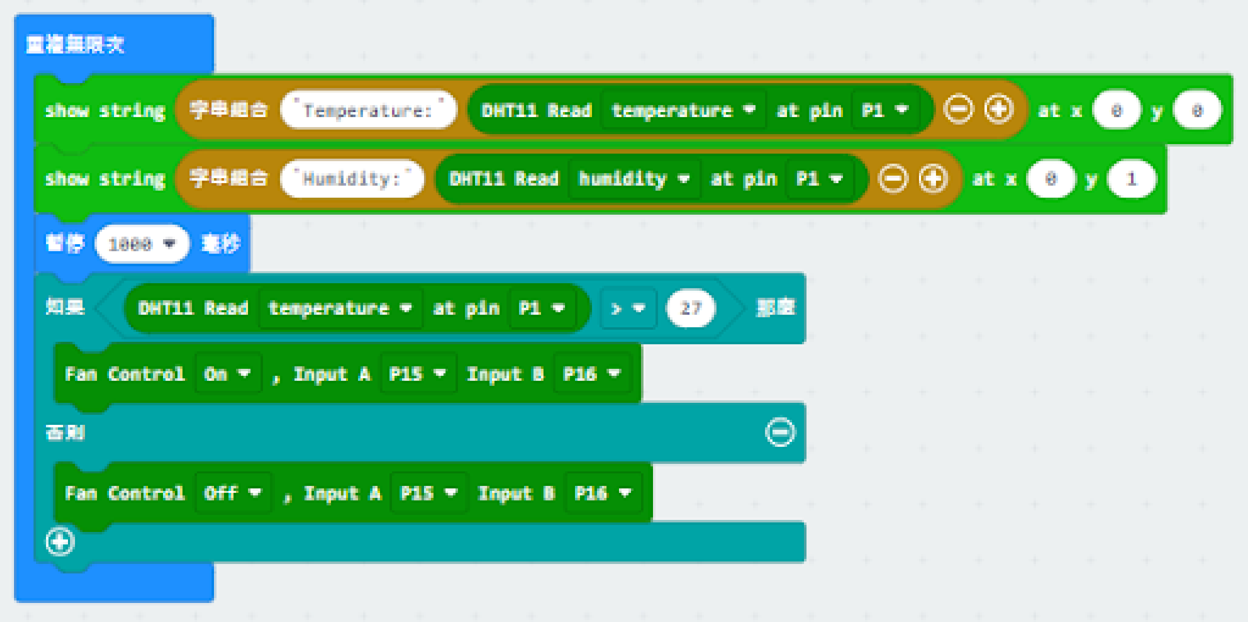


Hardware Connections

Sample Program



```
當啟動時
    LCD initialize with Address 39
    show string "Hello" at x 0 y 0
    暫停 2000 毫秒
```



```
重複無限次
    show string 字串組合 "Temperature:" DHT11 Read temperature at pin P1 at x 0 y 0
    show string 字串組合 "Humidity:" DHT11 Read humidity at pin P1 at x 0 y 1
    暫停 1000 毫秒
    如果 DHT11 Read temperature at pin P1 > 27 那麼
        Fan Control On , Input A P15 Input B P16
    否則
        Fan Control Off , Input A P15 Input B P16
```

Conclusion

1.2 Smart City IoT Starter Kit

1.2.1 Know More About Smart City

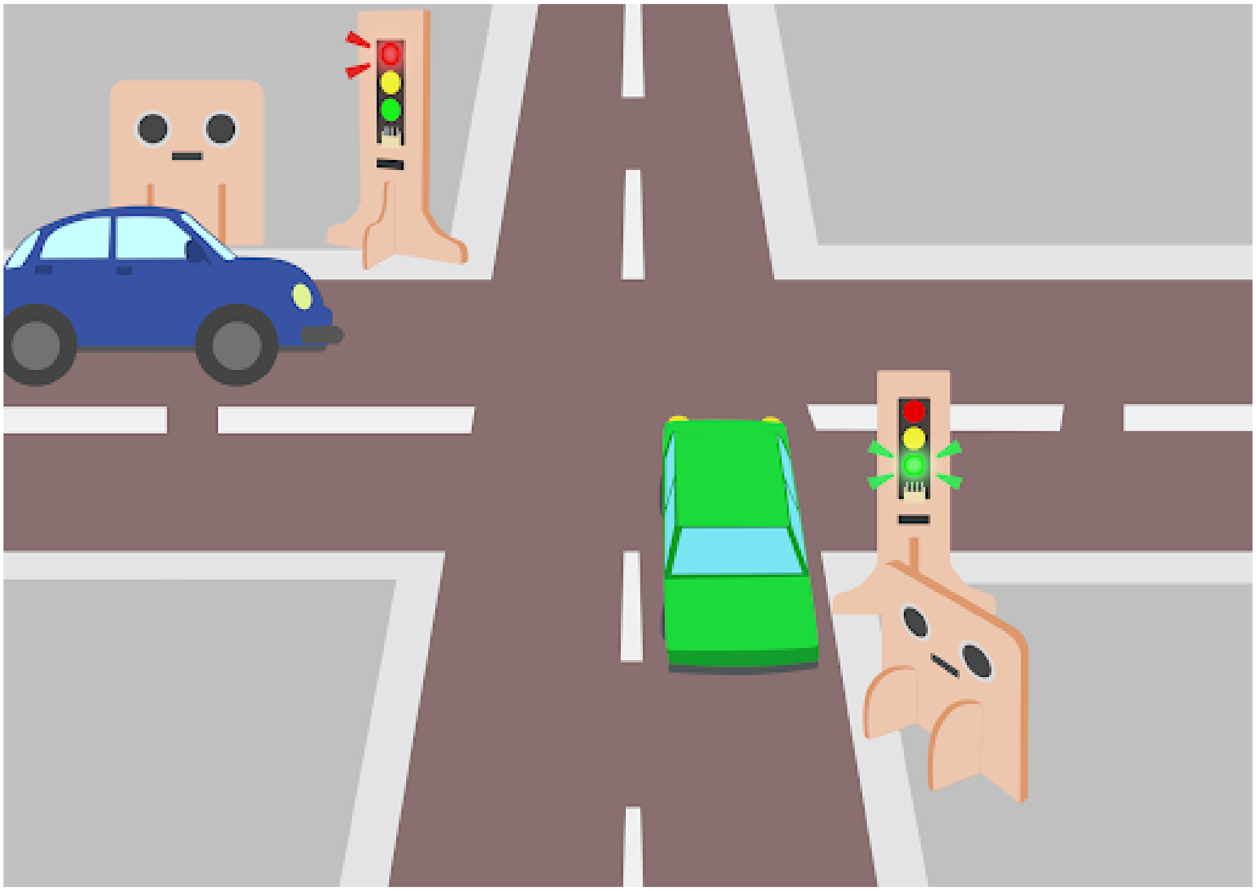
What is a Smart City?

Advantages of Smart City

- More effective, data-driven decision-making

- Enhanced citizen and government engagement
- Safer communities
- Reduced environmental footprint
- Improved transportation

1.2.2 Intersection

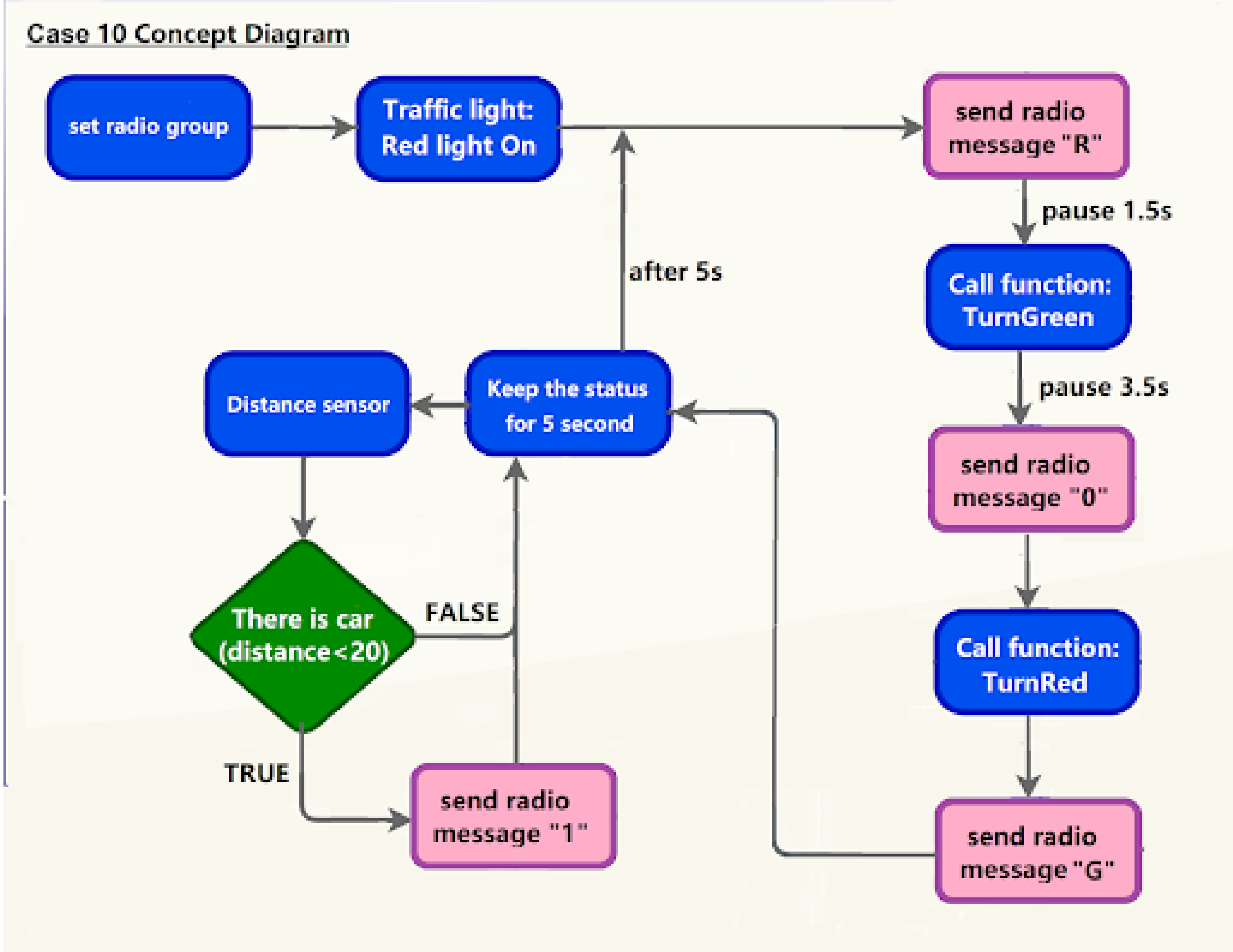


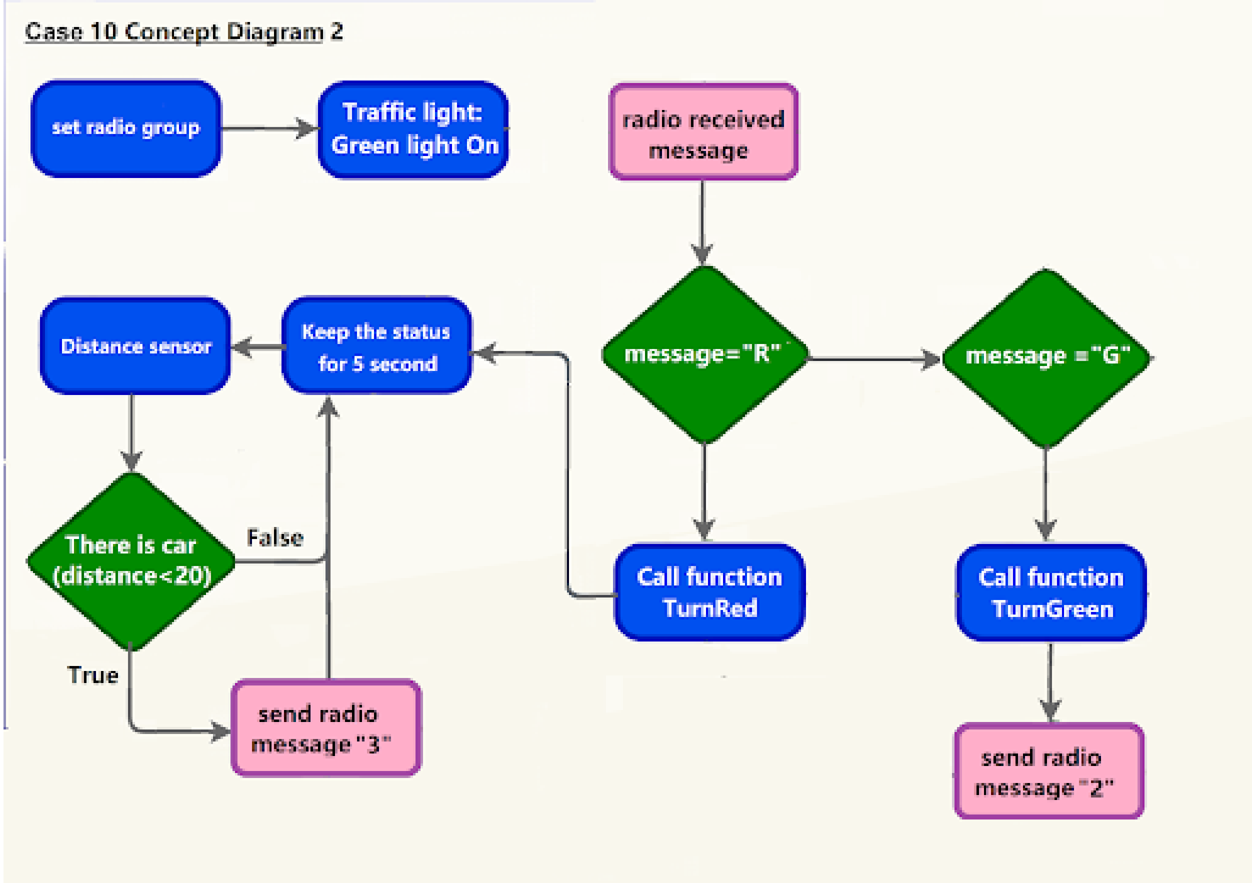
Goal

Background

What is a smart traffic light?

Smart traffic light operation

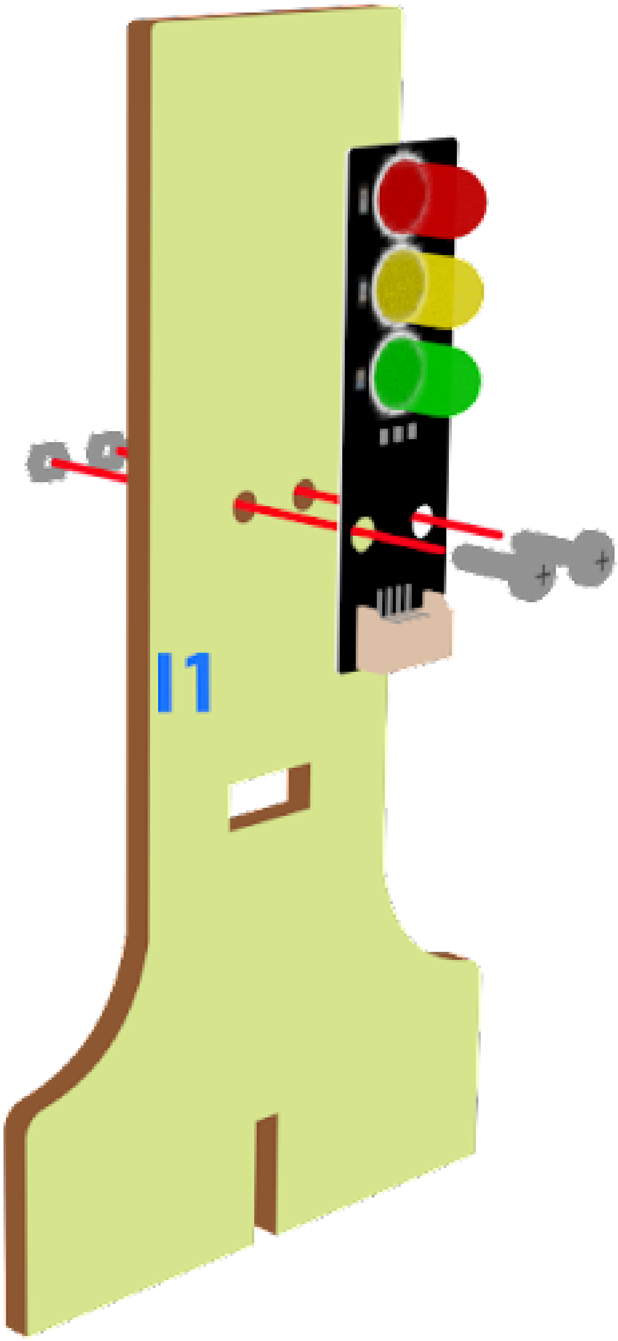




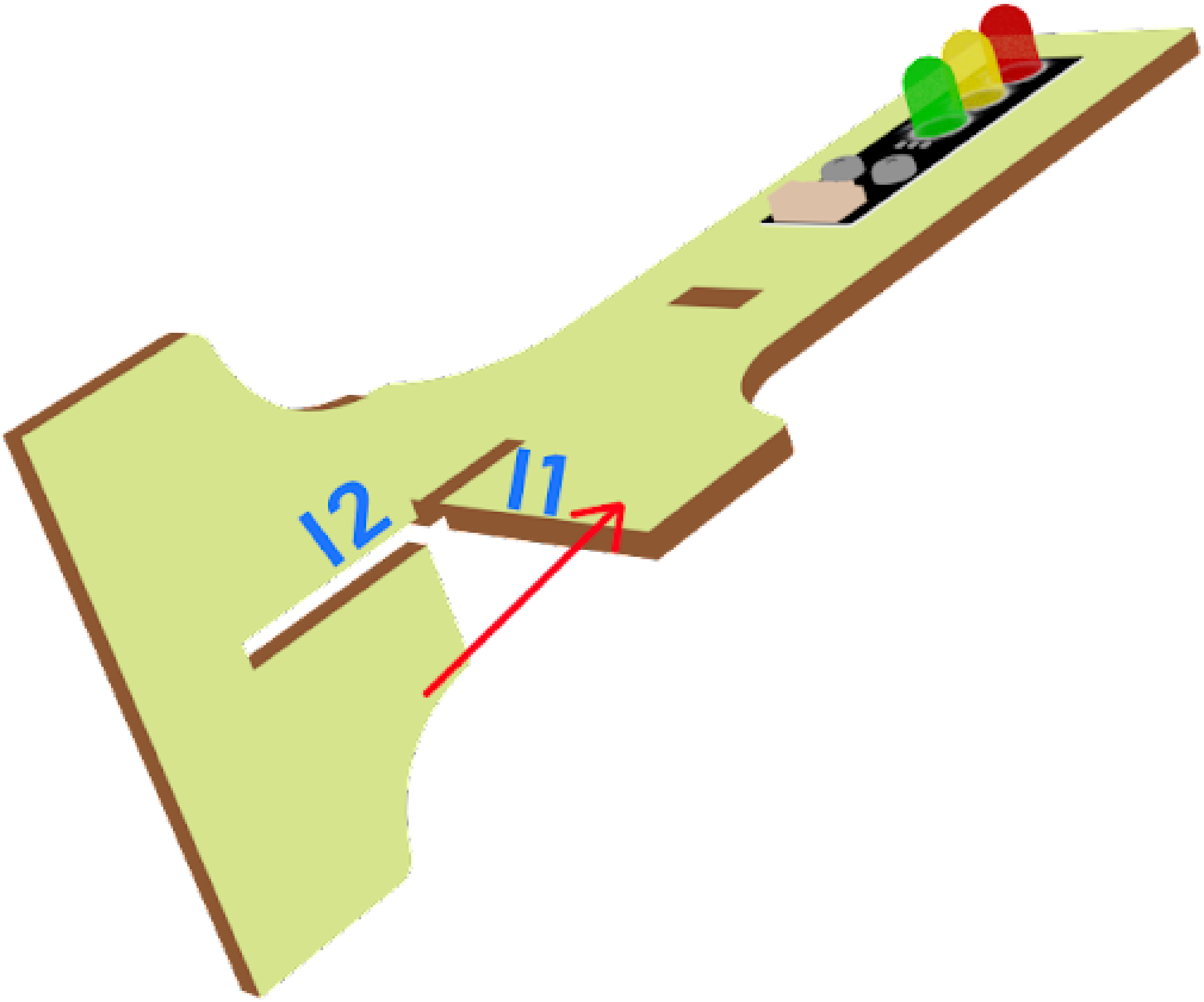
Part List

Assembly step

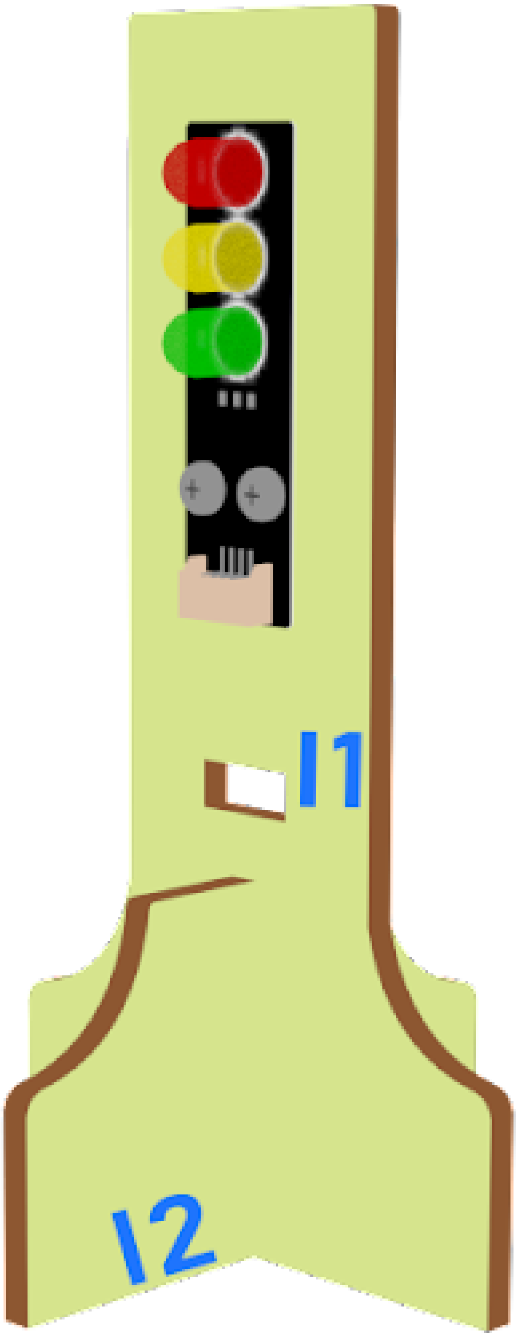
Step 1



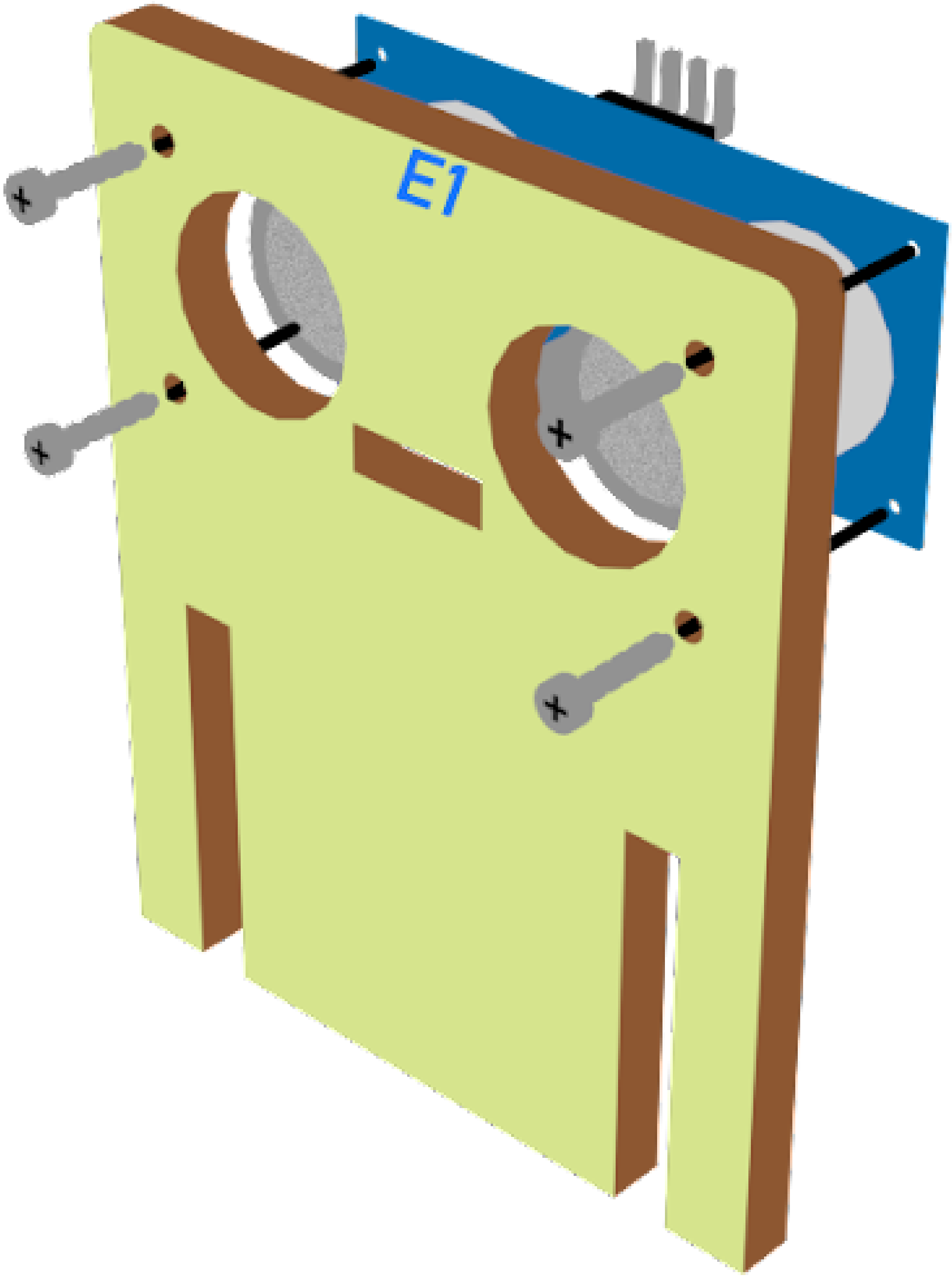
Step 2



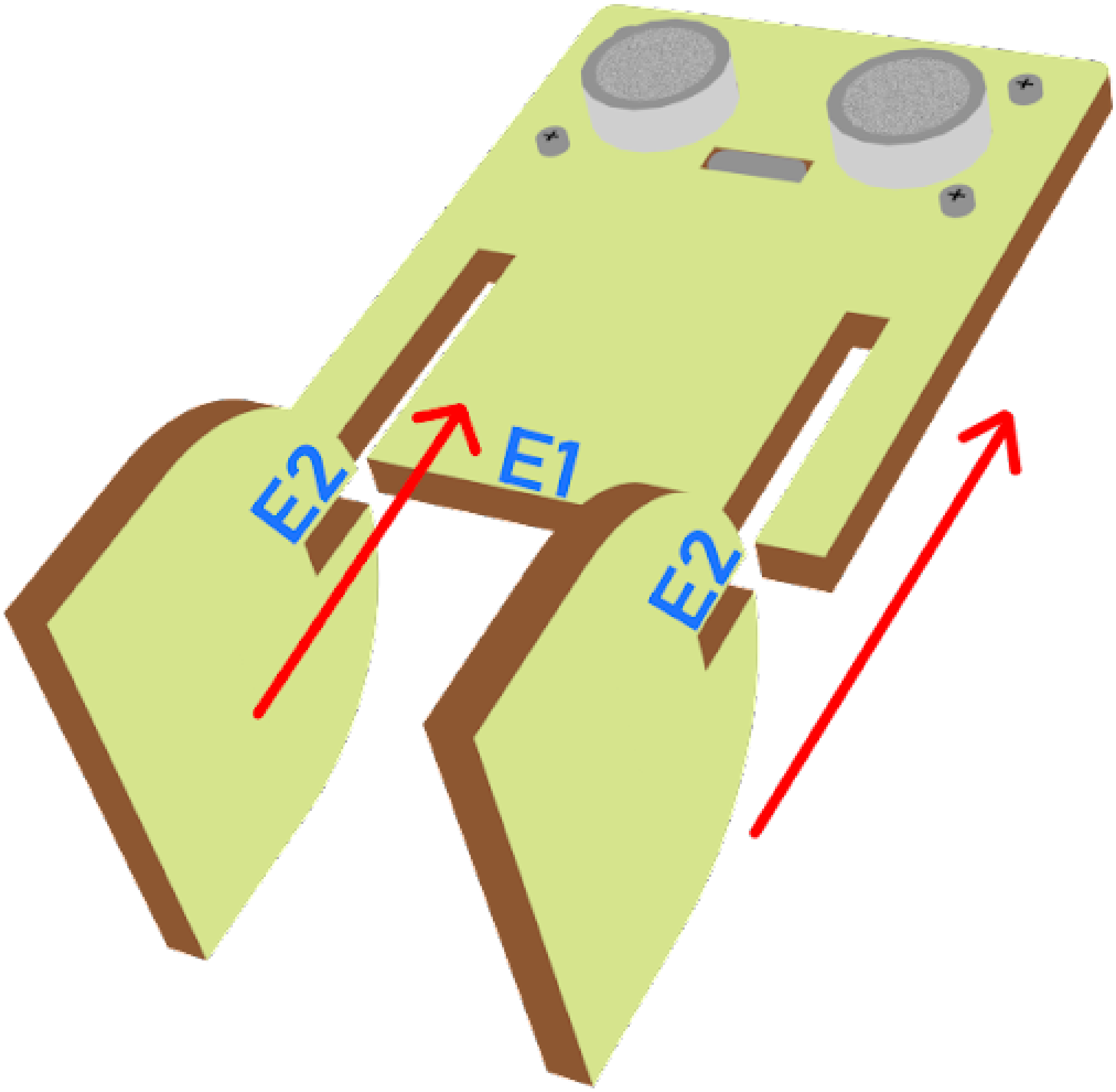
Step 3



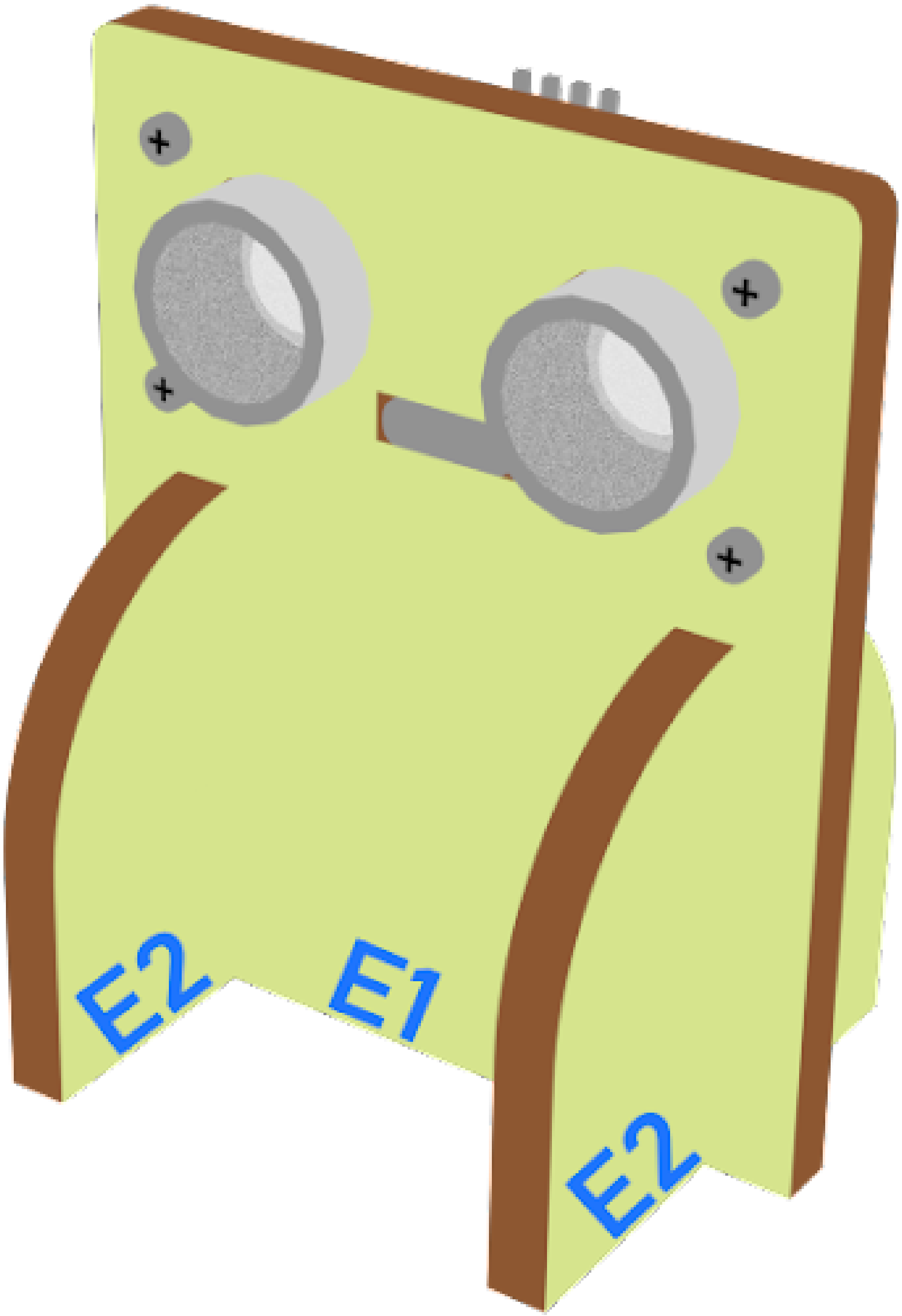
Step 4



Step 5



Step 6



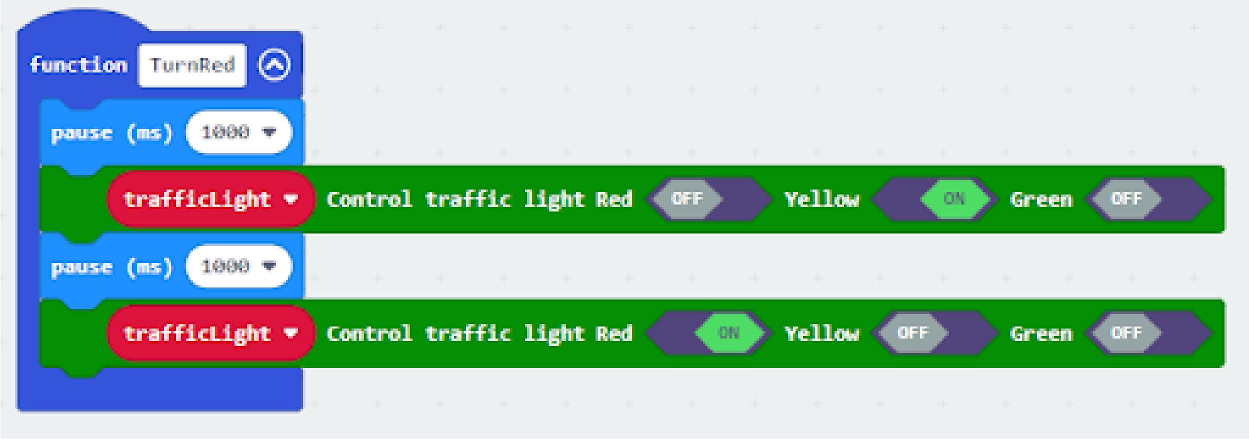
Hardware connect

Programming (MakeCode)

Traffic light 1

Step 1. Set up a new function (TurnRed)

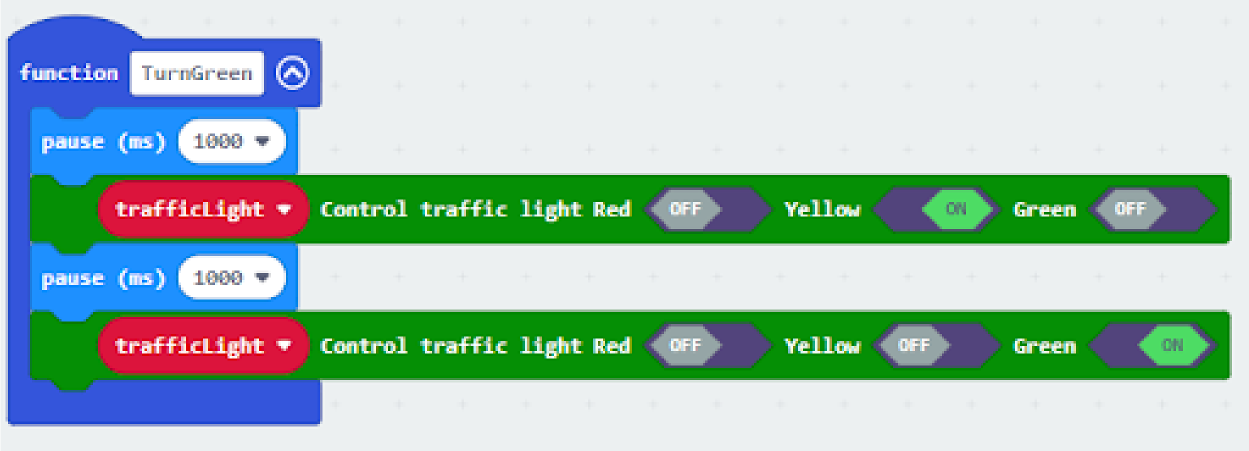
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light red on



```
function TurnRed
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red ON Yellow OFF Green OFF
```

Step 2. Set up a new function (TurnGreen)

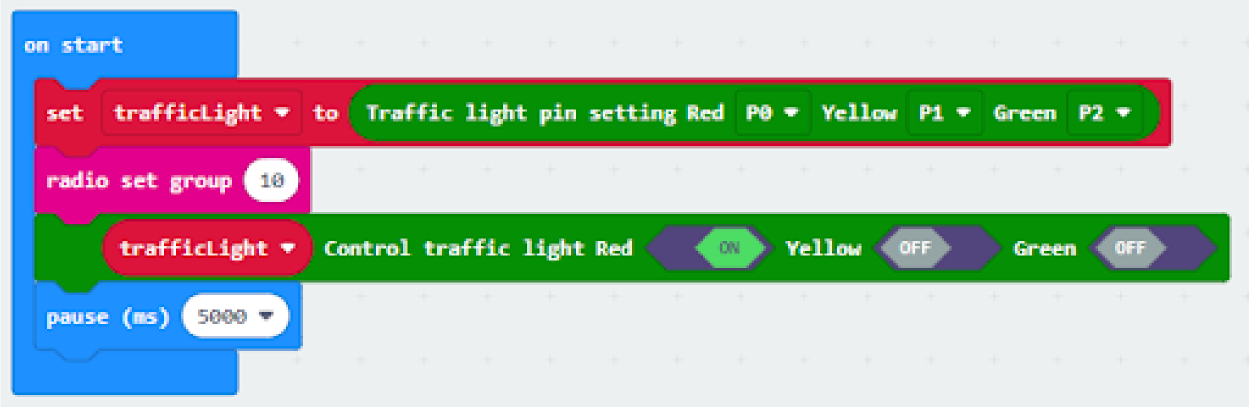
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light green on



```
function TurnGreen
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow OFF Green ON
```

Step 3. Initialize the program

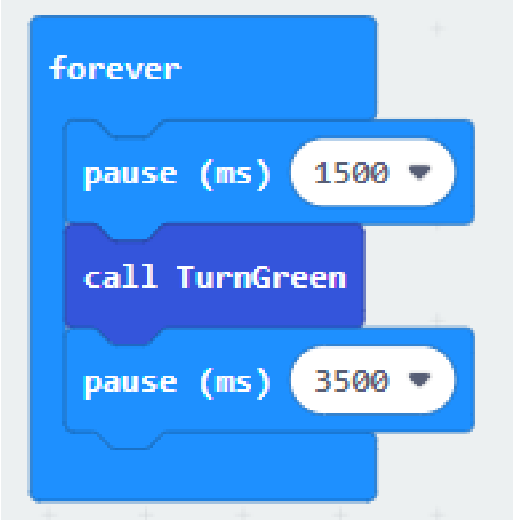
- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Drag radio set group 10 to on start
- Control traffic light red on
- Pause for 5s



```
on start
  set trafficLight ▾ to Traffic light pin setting Red P0 ▾ Yellow P1 ▾ Green P2 ▾
  radio set group 10
  trafficLight ▾ Control traffic light Red ON Yellow OFF Green OFF
  pause (ms) 5000 ▾
```

Step 4. Keep the green light status for 5 second

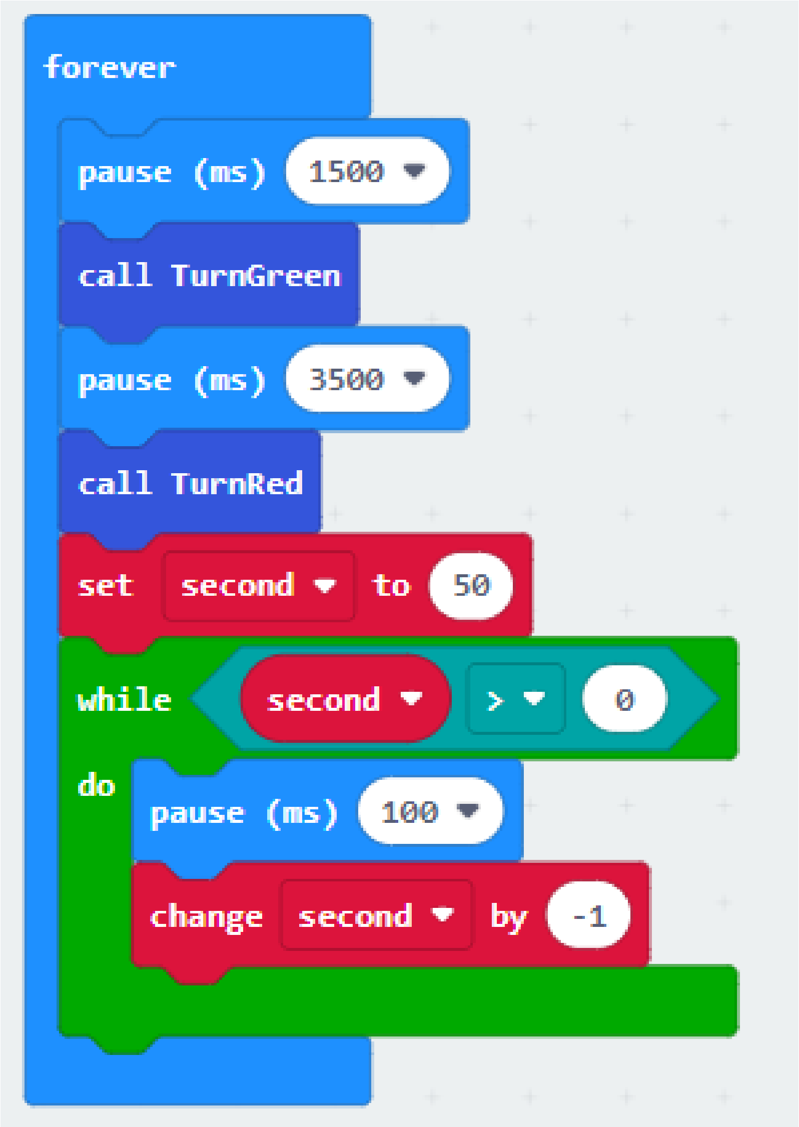
- Call function TurnGreen
- Pause 1.5s before TurnGreen
- Pause 3.5s after TurnGreen



```
forever
  pause (ms) 1500 ▾
  call TurnGreen
  pause (ms) 3500 ▾
```

Step 5. Keep the red light status for 5 second

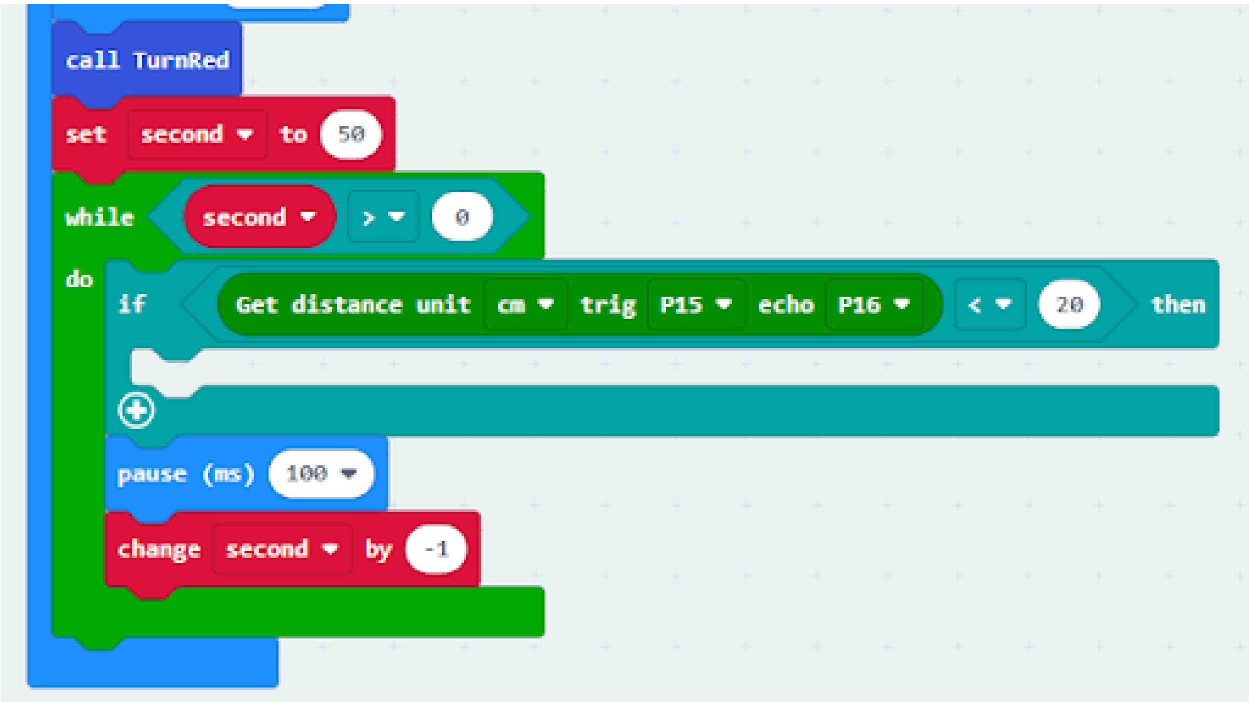
- Call function TurnRed
- set variable second to 50
- While second > 0, snap pause to 0.1 second and change second by -1.



```
forever
    pause (ms) 1500
    call TurnGreen
    pause (ms) 3500
    call TurnRed
    set second to 50
    while second > 0
    do
        pause (ms) 100
        change second by -1
```

Step 6. Get distance value

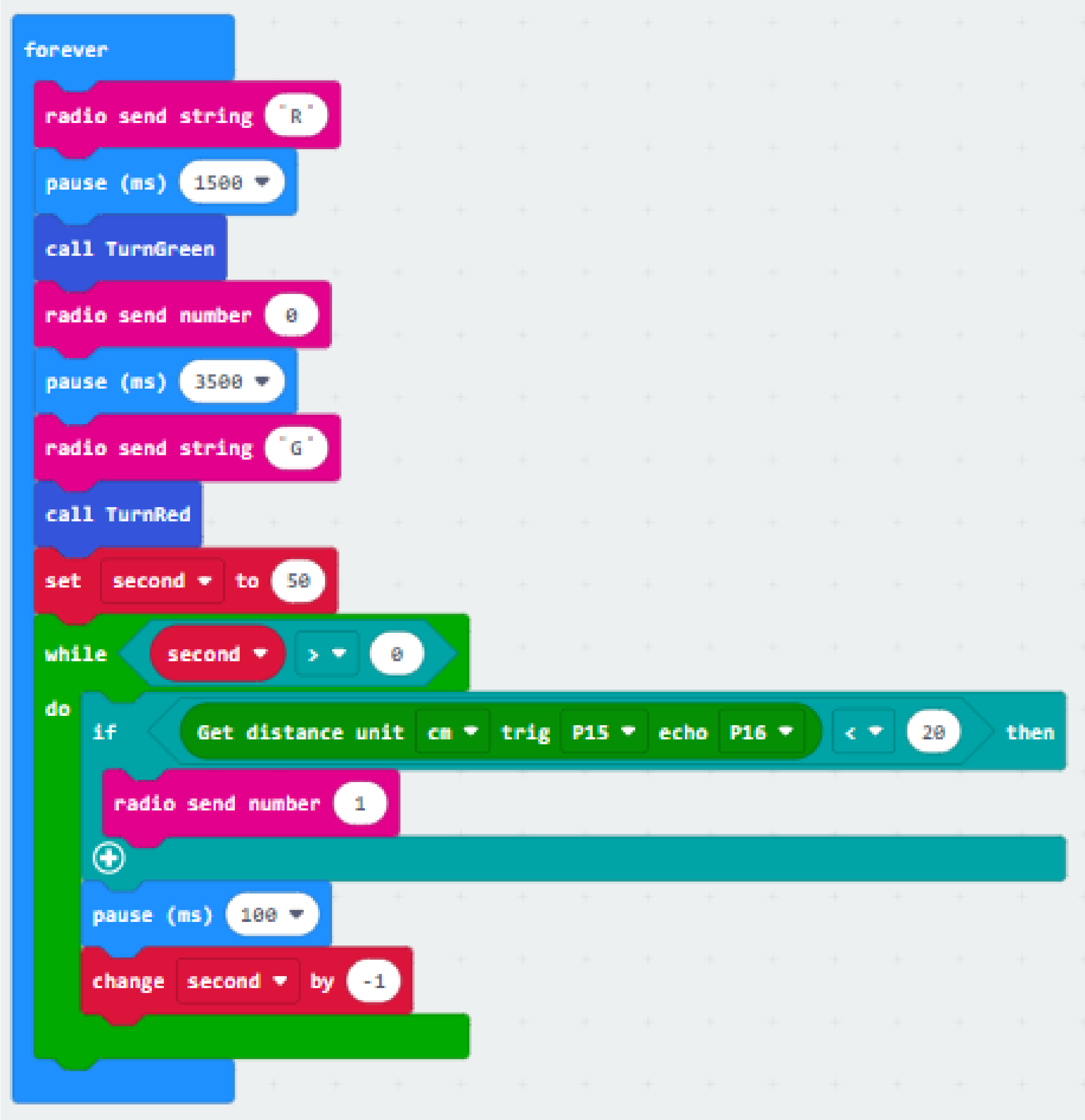
- Snap if statement into while loop, set get distance unit cm trig P15 echo P16 < 20



```
call TurnRed
set second ▾ to 50
while second ▾ > ▾ 0
do
    if Get distance unit cm ▾ trig P15 ▾ echo P16 ▾ < ▾ 20 then
    ⊕
    pause (ms) 100 ▾
    change second ▾ by -1
```

Step 7. Control traffic light 2 and car by sending radio number

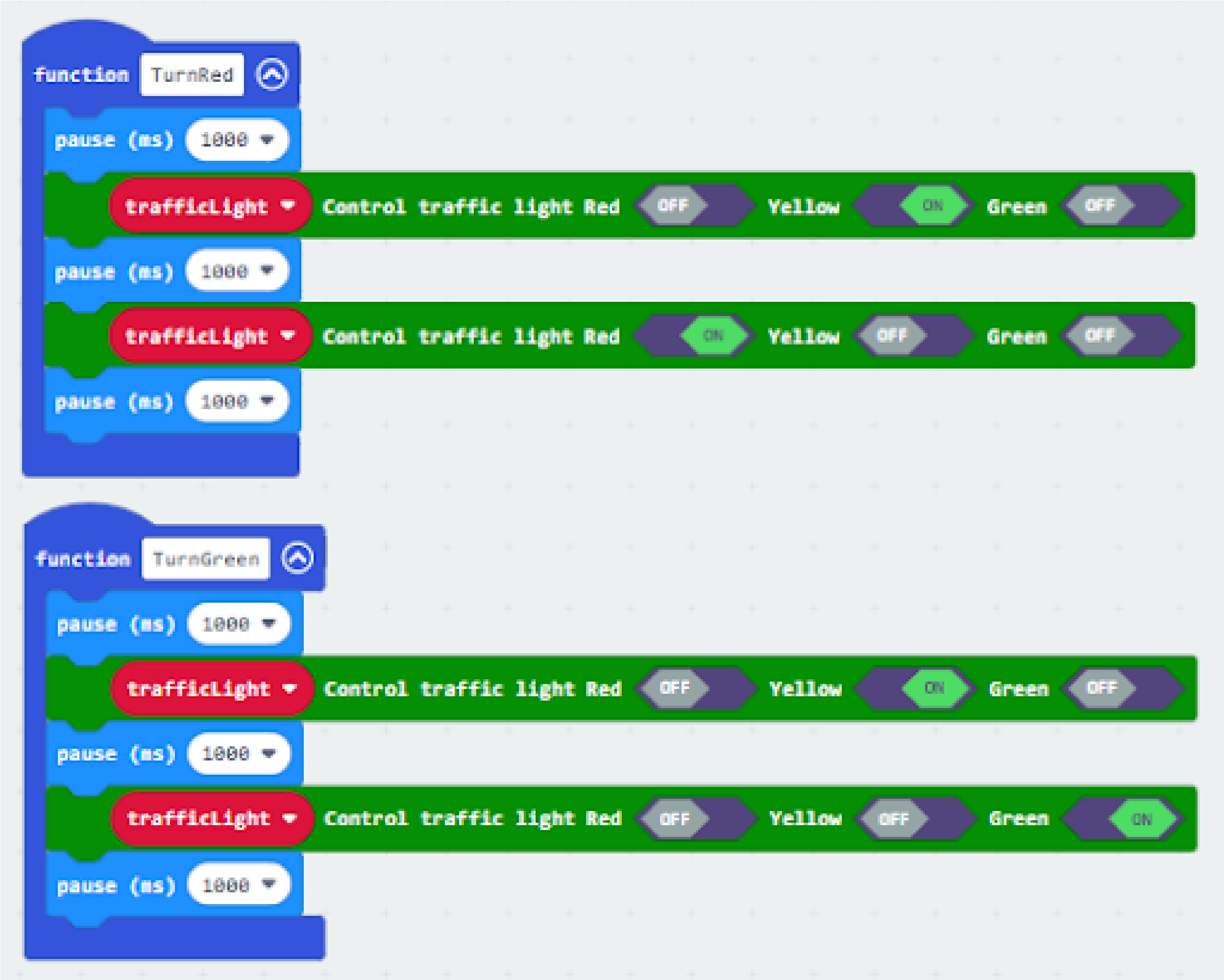
- Drag radio send number 1 into if
- Drag radio send number 0 after TurnGreen
- Drag radio send string "R" before TurnGreen
- Drag radio send string "G" before TurnRed



```
forever
    radio send string "R"
    pause (ms) 1500
    call TurnGreen
    radio send number 0
    pause (ms) 3500
    radio send string "G"
    call TurnRed
    set second to 50
    while second > 0
    do
        if Get distance unit cm trig P15 echo P16 < 20 then
            radio send number 1
        ⊕
        pause (ms) 100
        change second by -1
```

Traffic light 2

Step 1. Set up new functions

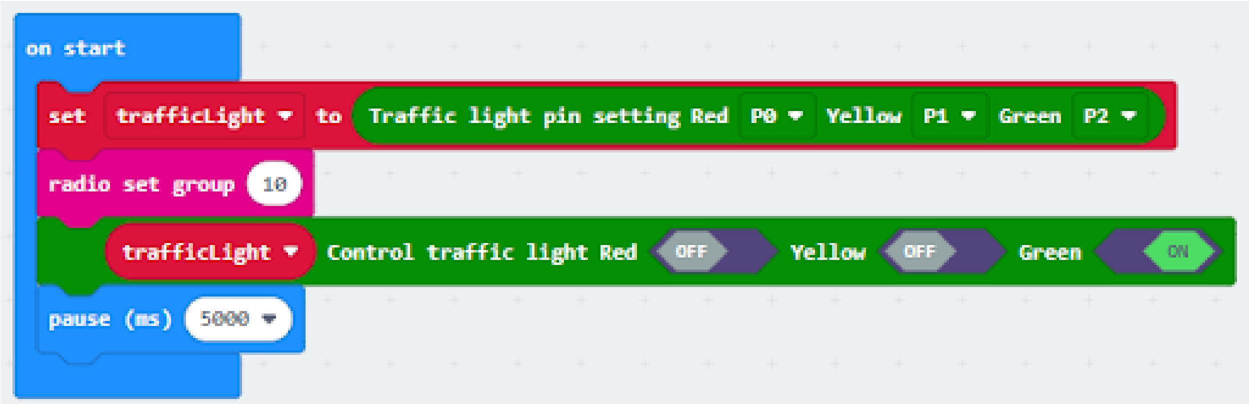


```
function TurnRed
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red ON Yellow OFF Green OFF
    pause (ms) 1000

function TurnGreen
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow OFF Green ON
    pause (ms) 1000
```

Step 2. Initialize the program

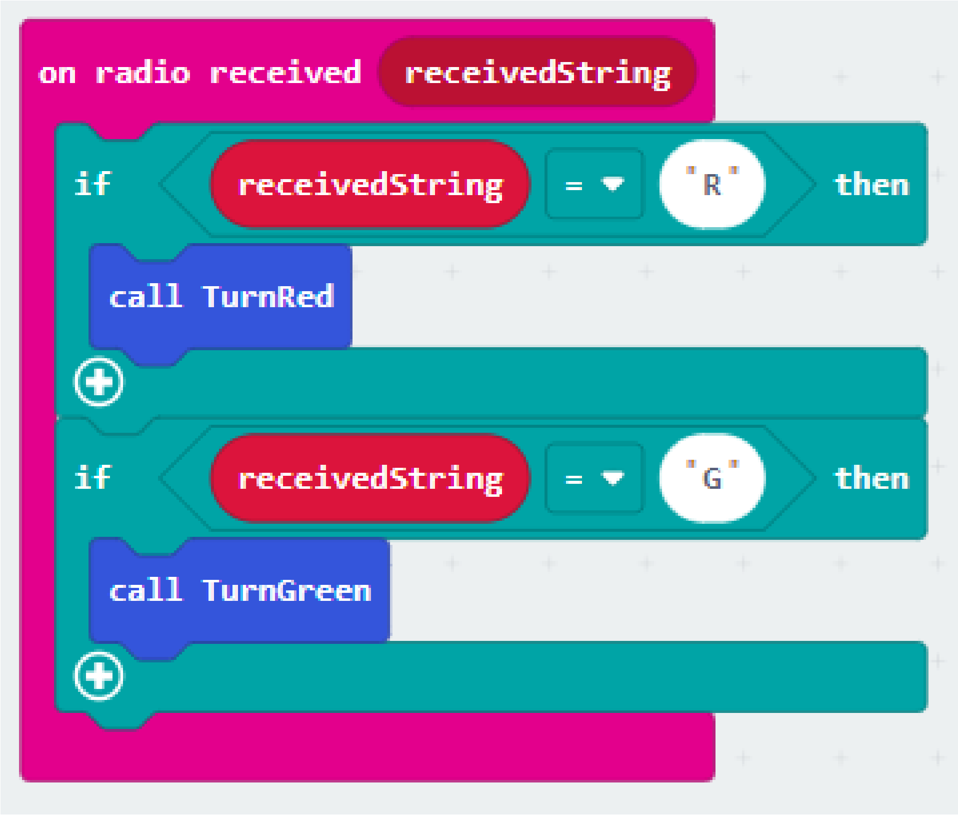
- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Drag radio set group 10 to on start
- Control traffic light green on
- Pause for 5s



```
on start
  set trafficLight ▾ to Traffic light pin setting Red P0 ▾ Yellow P1 ▾ Green P2 ▾
  radio set group 10
  trafficLight ▾ Control traffic light Red OFF Yellow OFF Green ON
  pause (ms) 5000 ▾
```

Step 3. Control traffic light and car by receiving different number

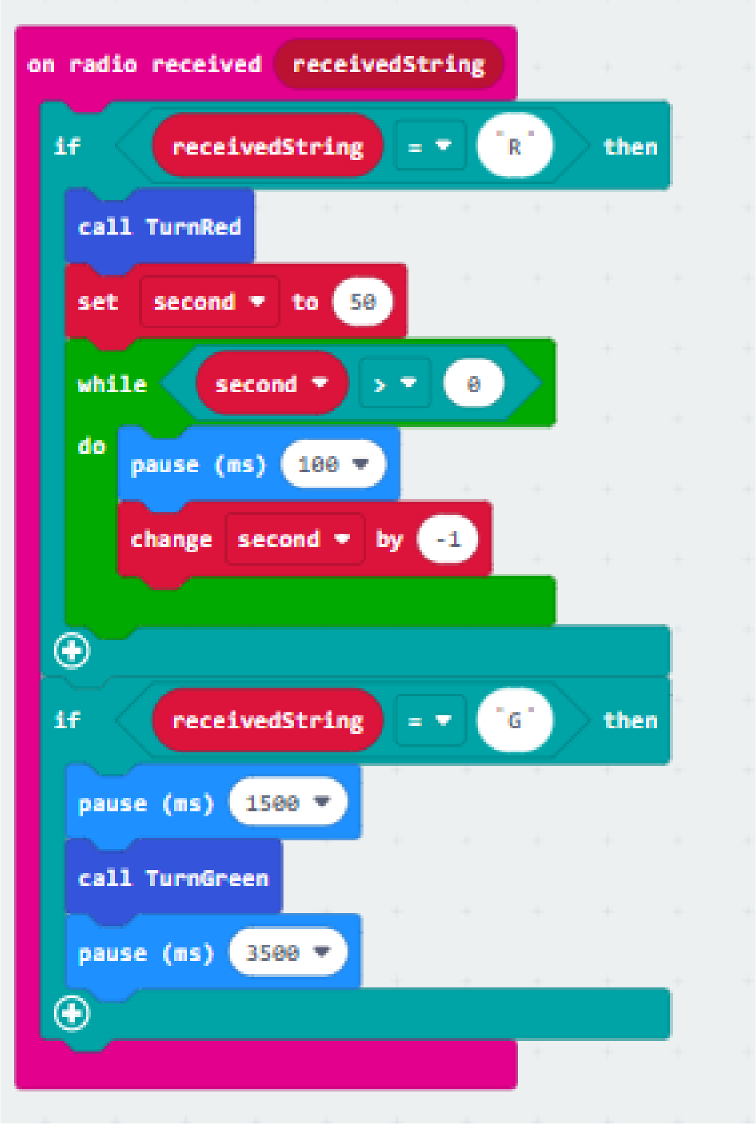
- Snap if statement into on radio received receivedString
- Set receivedString =“R” and call TurnRed
- Set receivedString =“G” and call TurnGreen



```
on radio received receivedString
  if receivedString = ▾ "R" then
    call TurnRed
  ⊕
  if receivedString = ▾ "G" then
    call TurnGreen
  ⊕
```

Step 4. Keep the red light status for 5 second

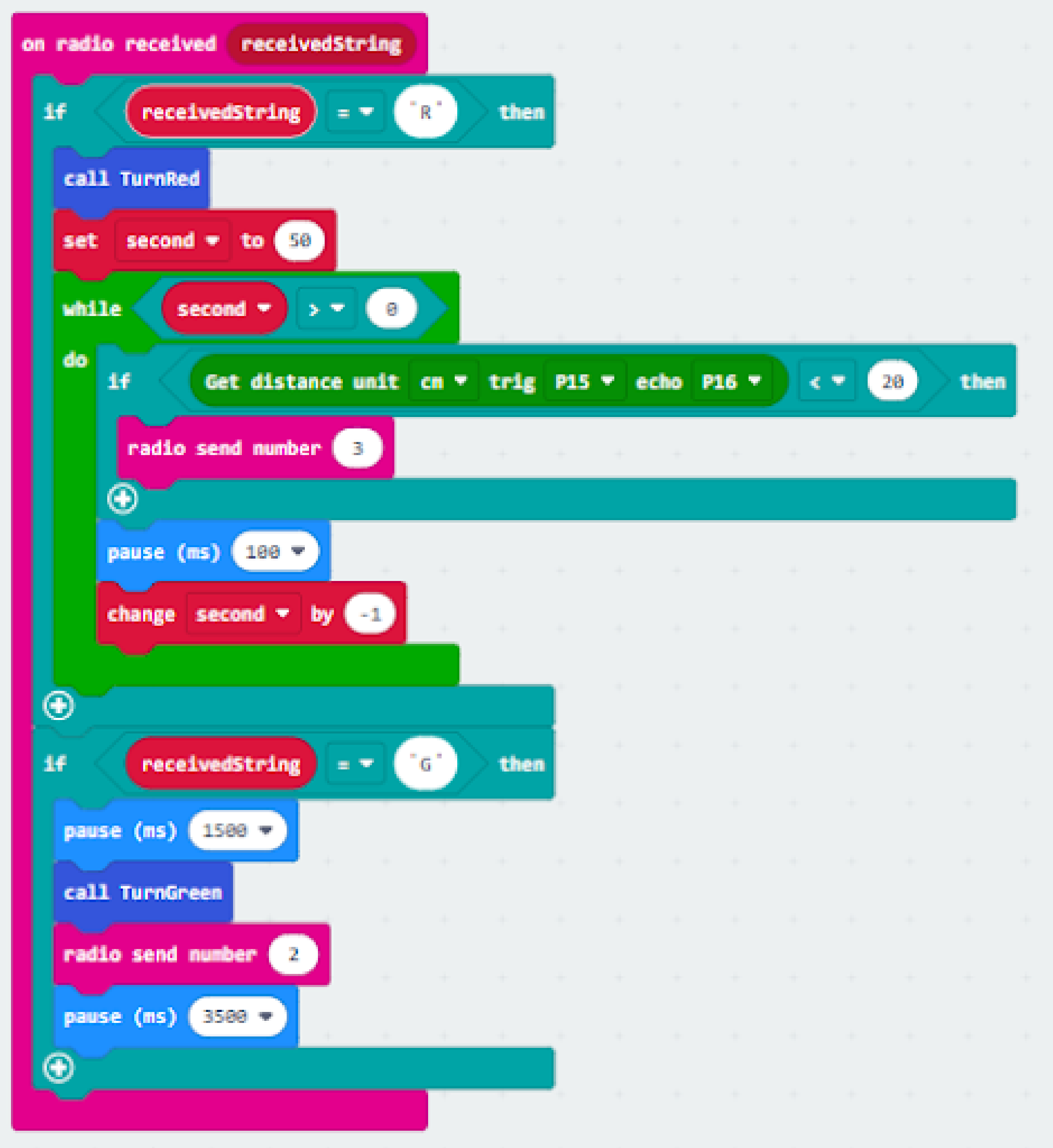
- After TurnRed, set variable second to 50
- While second > 0, snap pause to 0.1 second and change second by -1.
- Before TurnGreen, pause 1.5s
- After TurnGreen, pause 3.5s



```
on radio received receivedString
    if receivedString = "R" then
        call TurnRed
        set second to 50
        while second > 0
        do
            pause (ms) 100
            change second by -1
    ⊕
    if receivedString = "G" then
        pause (ms) 1500
        call TurnGreen
        pause (ms) 3500
    ⊕
```

Step 5. Get distance value and control the car

- Snap if statement into while loop, set get distance unit cm trig P15 echo P16 < 20
- Drag radio send number 3 into if 1 if

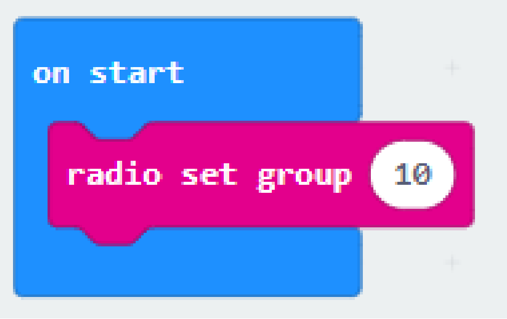


```
on radio received receivedString
  if receivedString = "R" then
    call TurnRed
    set second to 50
    while second > 0
    do
      if Get distance unit cm trig P15 echo P16 < 20 then
        radio send number 3
      ⊕
      pause (ms) 100
      change second by -1
  ⊕
  if receivedString = "G" then
    pause (ms) 1500
    call TurnGreen
    radio send number 2
    pause (ms) 3500
  ⊕
```

Car 1:

Step 1. Set radio set group at start position

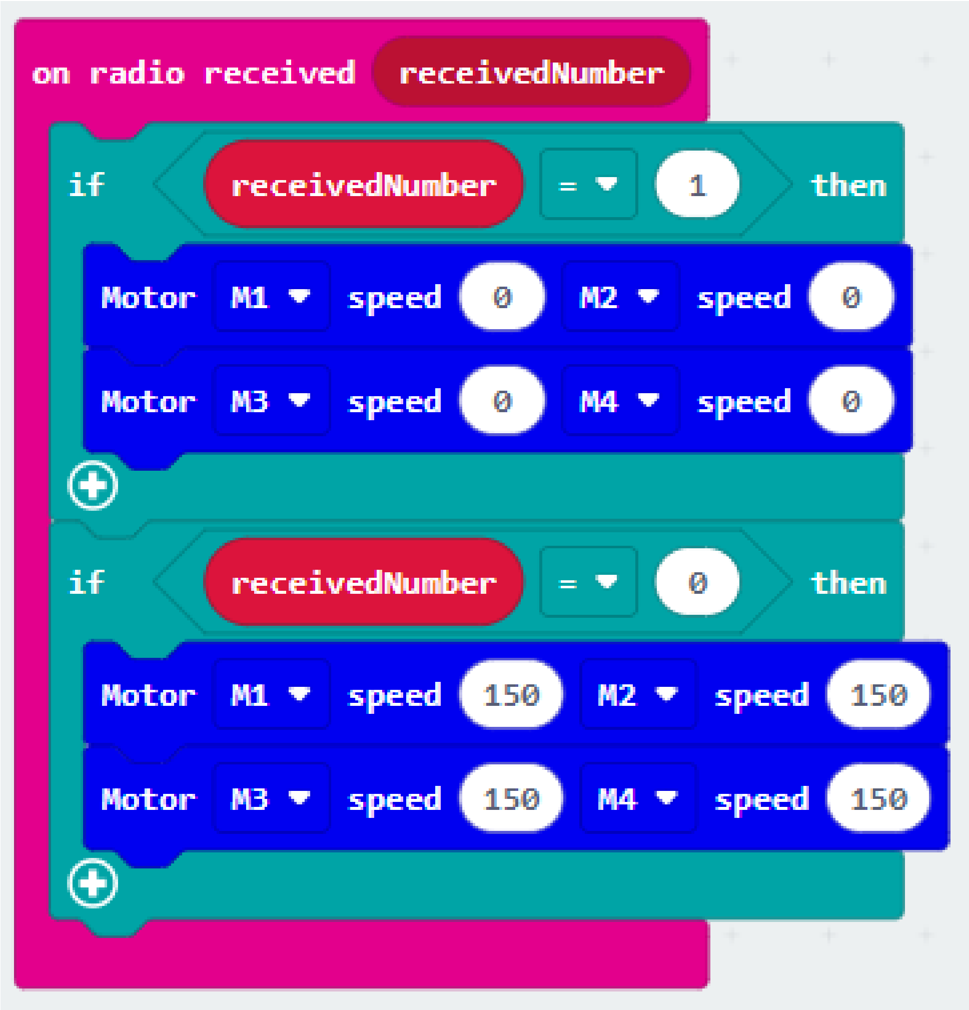
- Drag radio set group 10 to on start



```
on start
    radio set group 10
```

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward

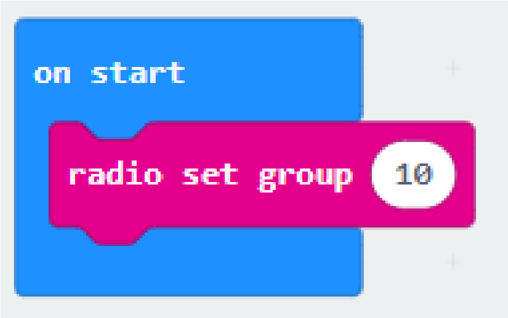


```
on radio received receivedNumber
    if receivedNumber = 1 then
        Motor M1 speed 0 M2 speed 0
        Motor M3 speed 0 M4 speed 0
    if receivedNumber = 0 then
        Motor M1 speed 150 M2 speed 150
        Motor M3 speed 150 M4 speed 150
```

Car 2:

Step 1. Set radio set group at start position

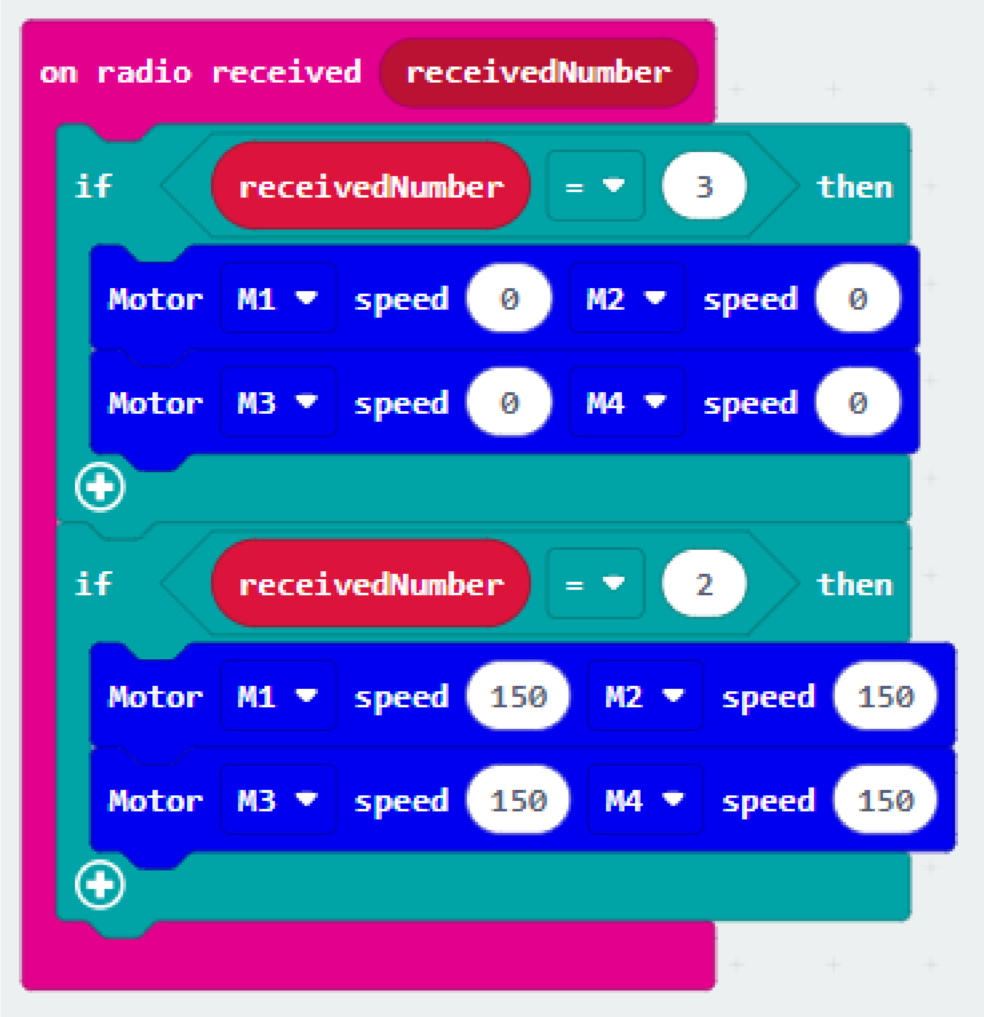
- Drag radio set group 10 to on start



```
on start
    radio set group 10
```

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =3 and make the car stop
- Set receivedNumber=2 and make the car move forward

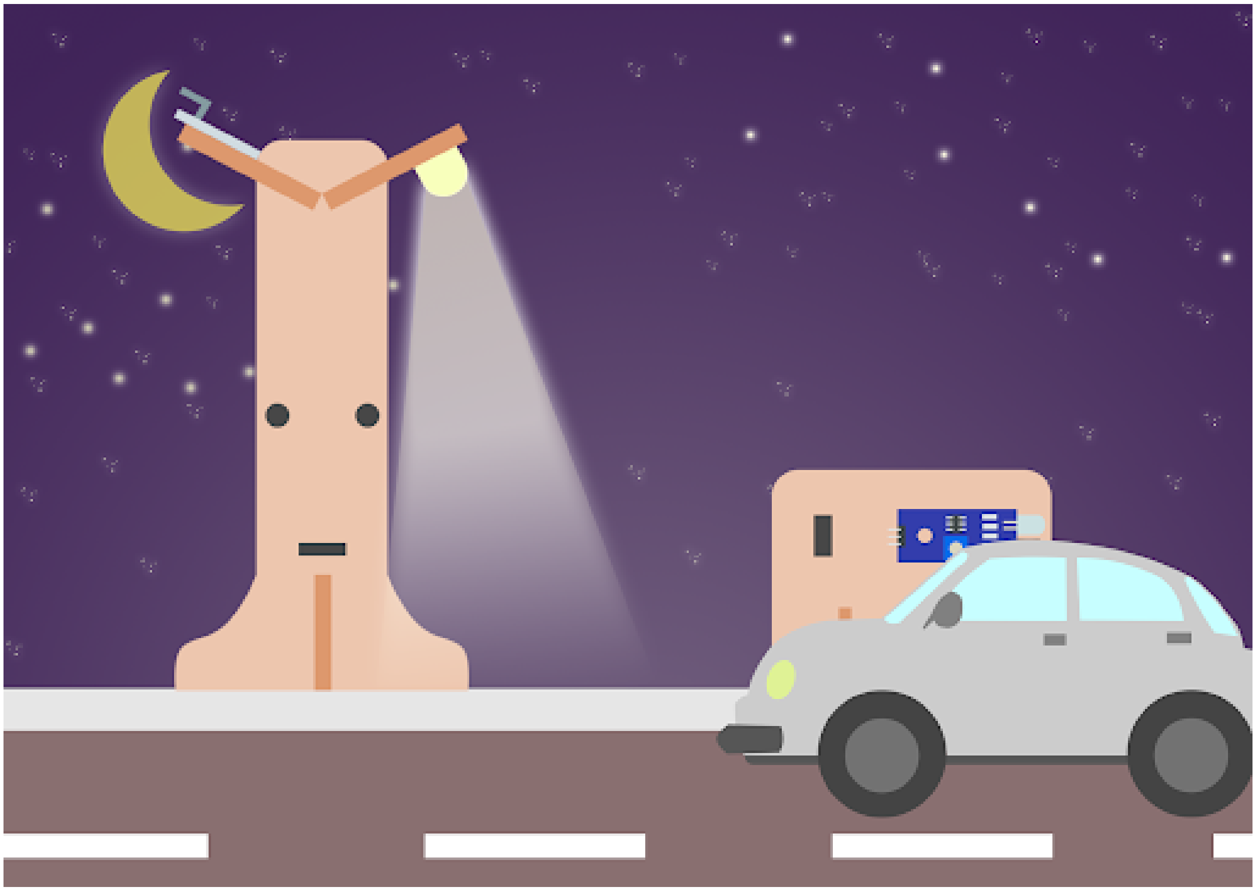


```
on radio received receivedNumber
  if receivedNumber = 3 then
    Motor M1 speed 0 M2 speed 0
    Motor M3 speed 0 M4 speed 0
  if receivedNumber = 2 then
    Motor M1 speed 150 M2 speed 150
    Motor M3 speed 150 M4 speed 150
```

Result

Think

1.2.3 Automated Smart Street Lamp

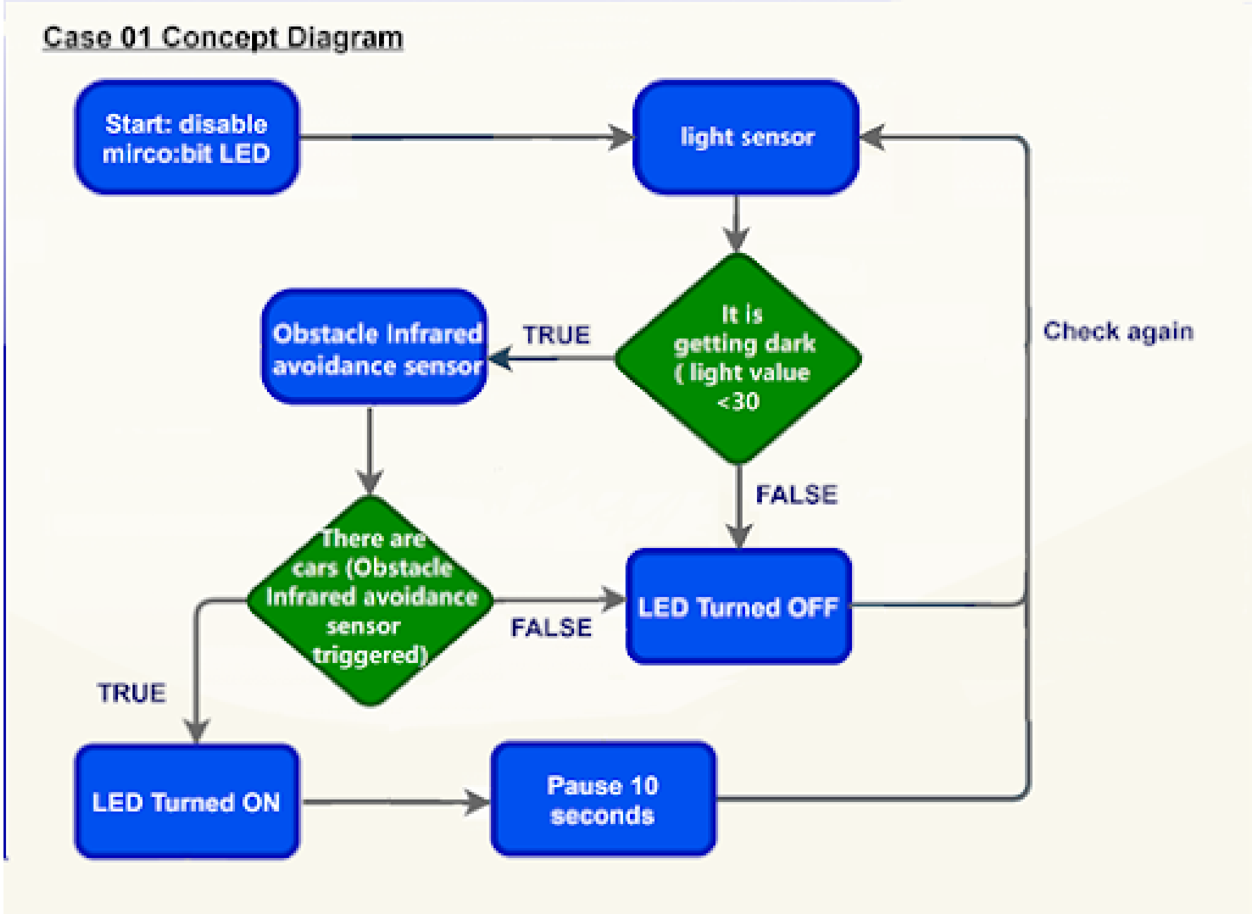


Goal

Background

What is a smart street lamp?

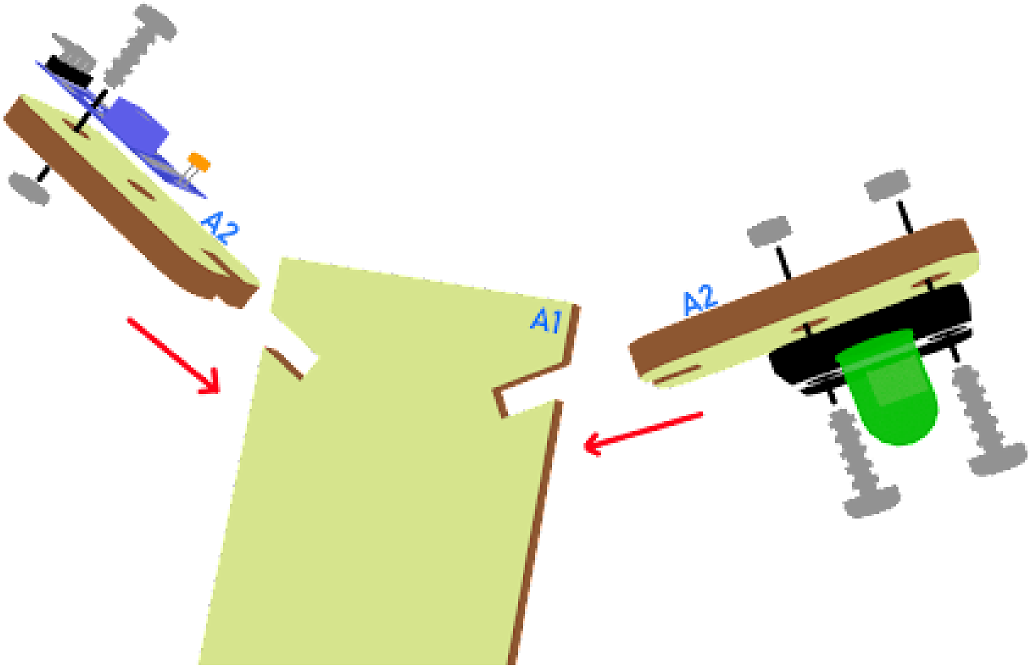
Smart street lamp operation


Case 01 Concept Diagram
Start: disable mirco:bit LED
light sensor
It is getting dark (light value <30
TRUE
Obstacle Infrared avoidance sensor
FALSE
Check again
There are cars (Obstacle Infrared avoidance sensor triggered)
FALSE
LED Turned OFF
TRUE
LED Turned ON
Pause 10 seconds

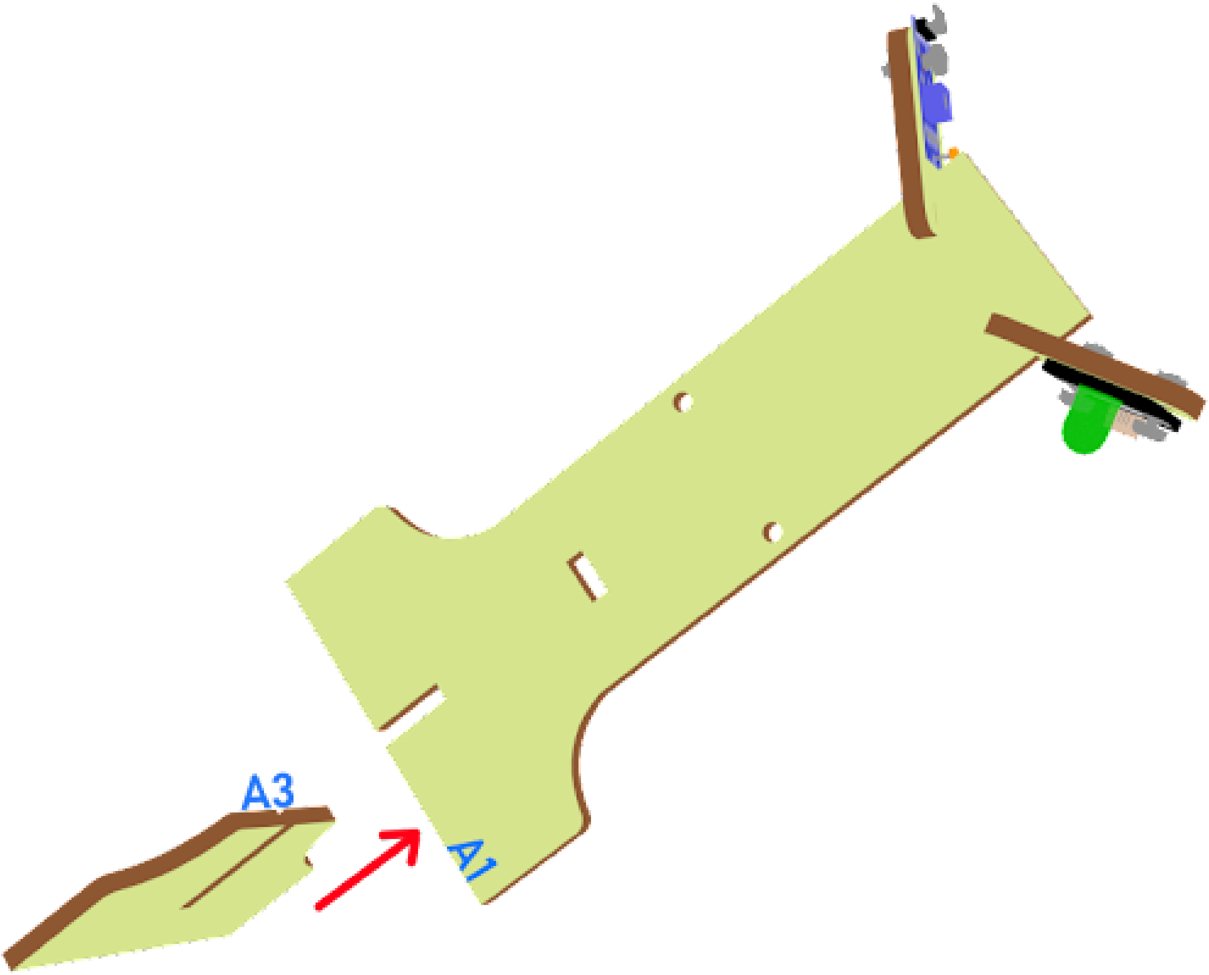
Part List

Assembly step

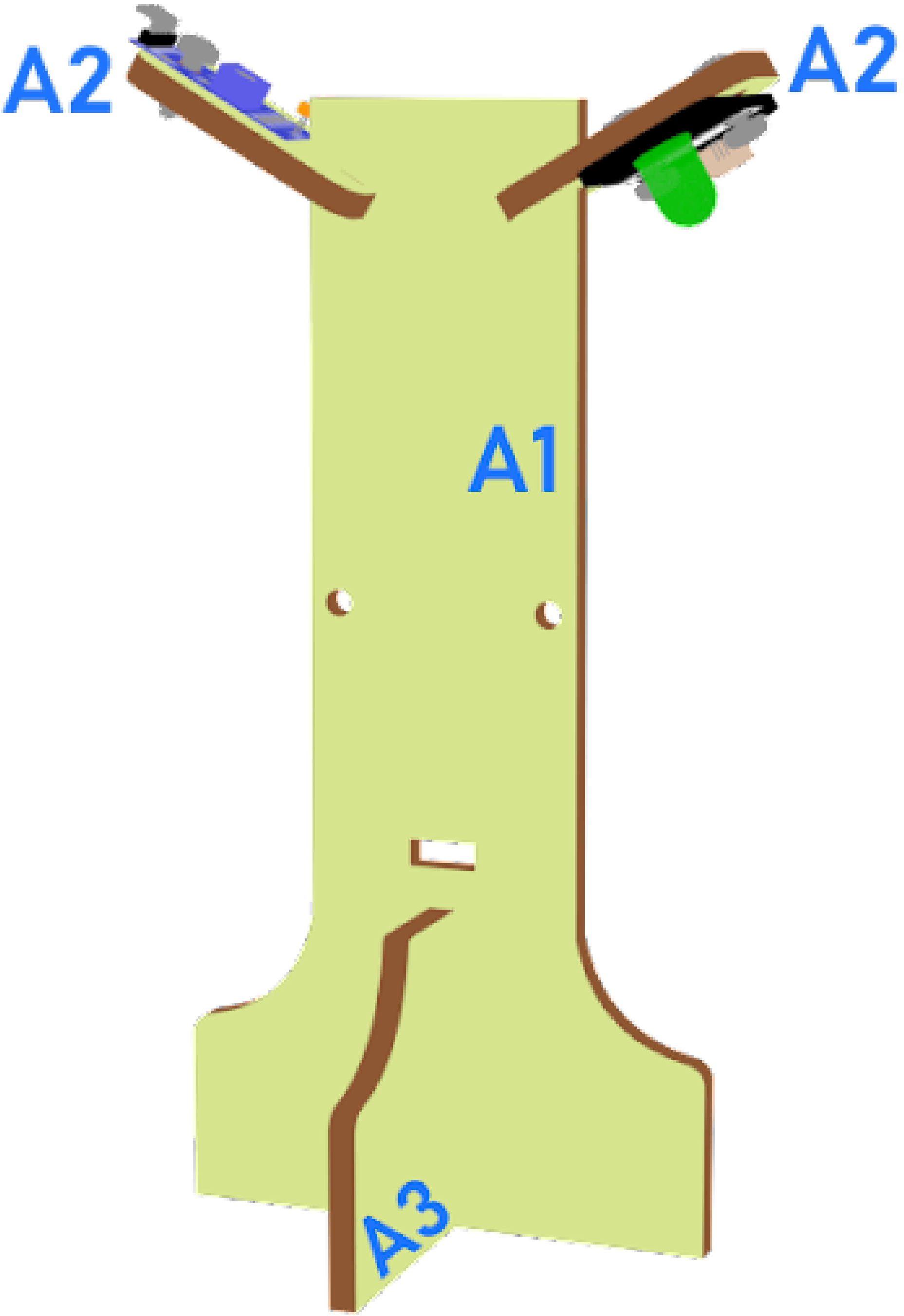
Step 1



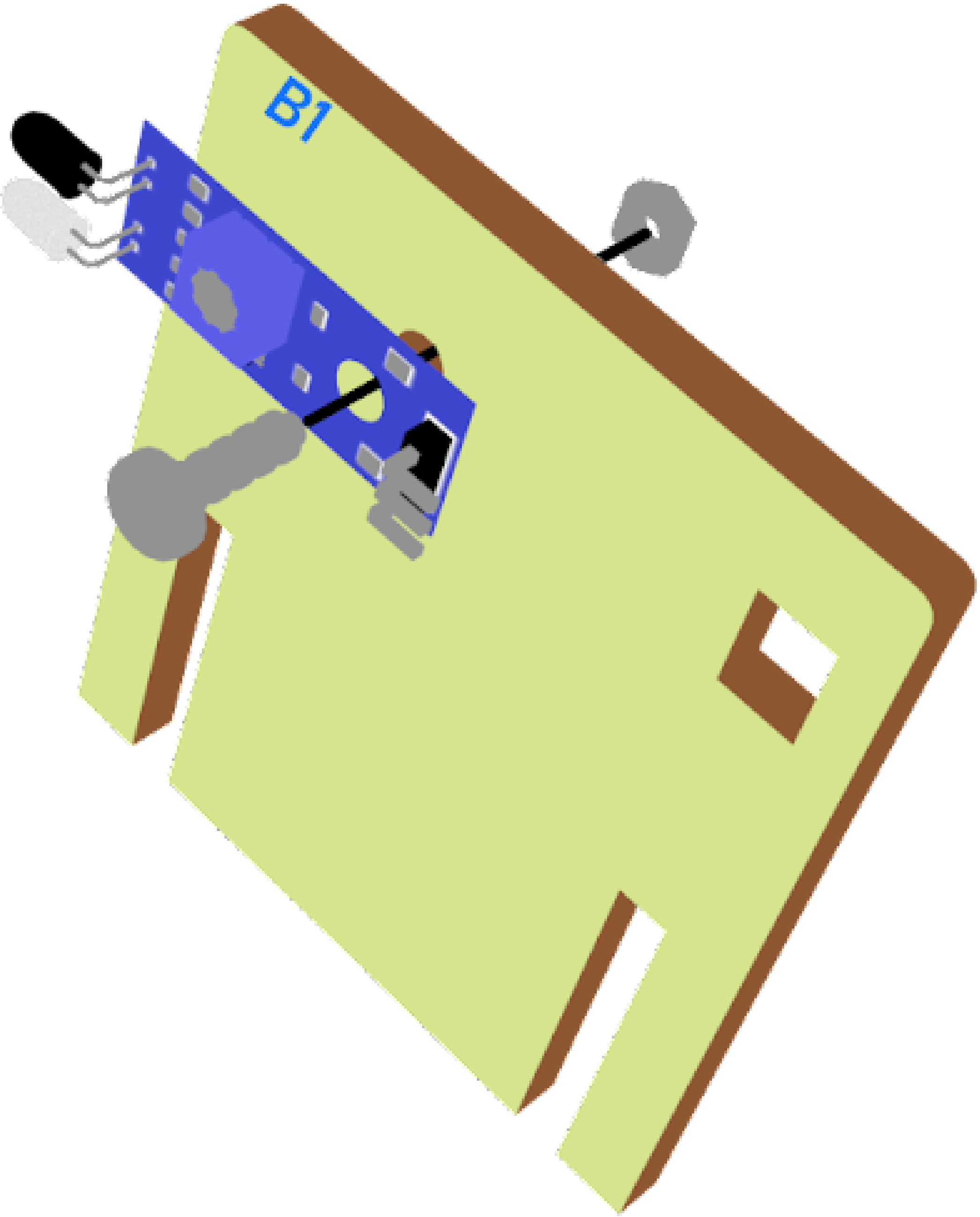
Step 2



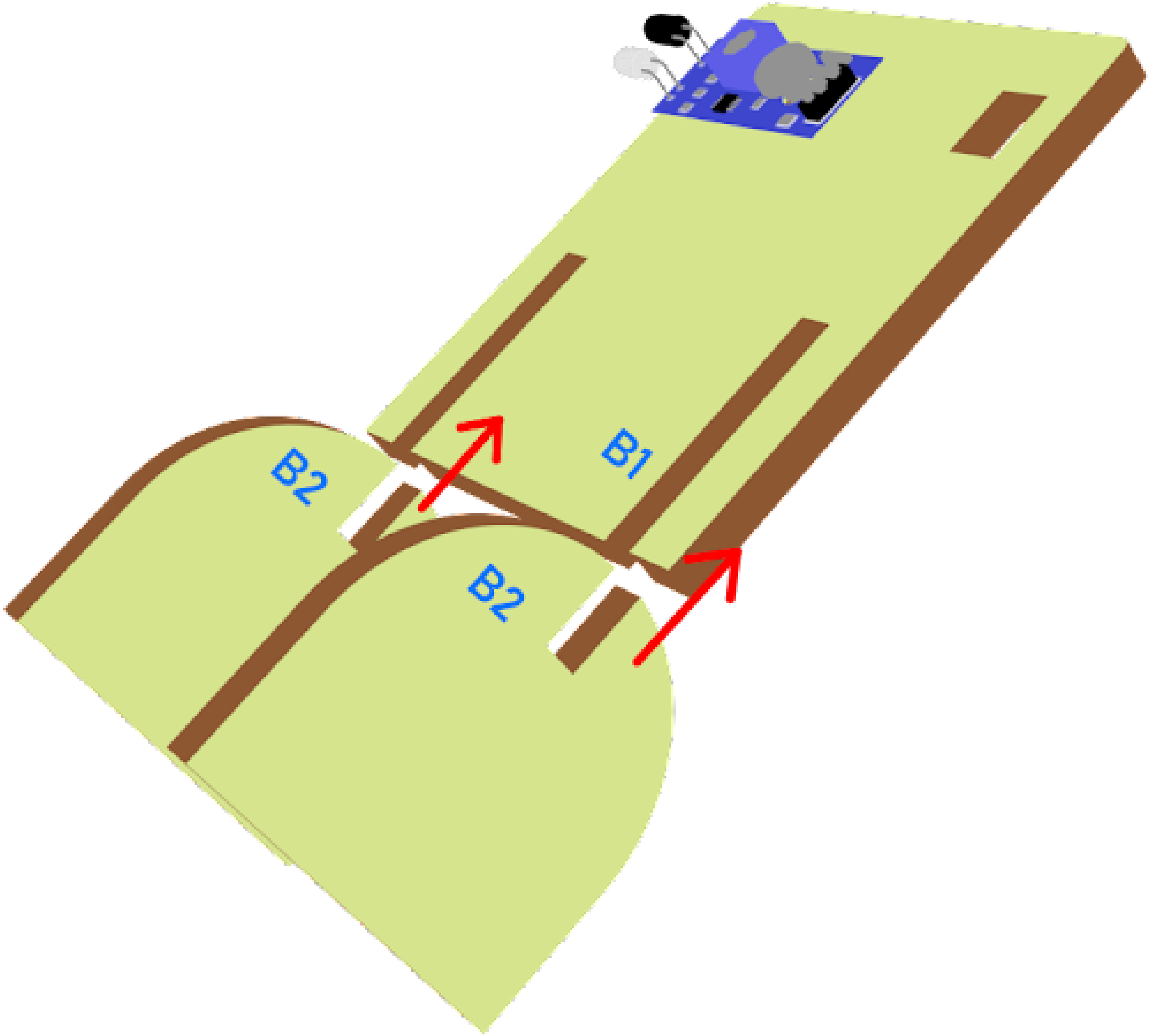
Step 3



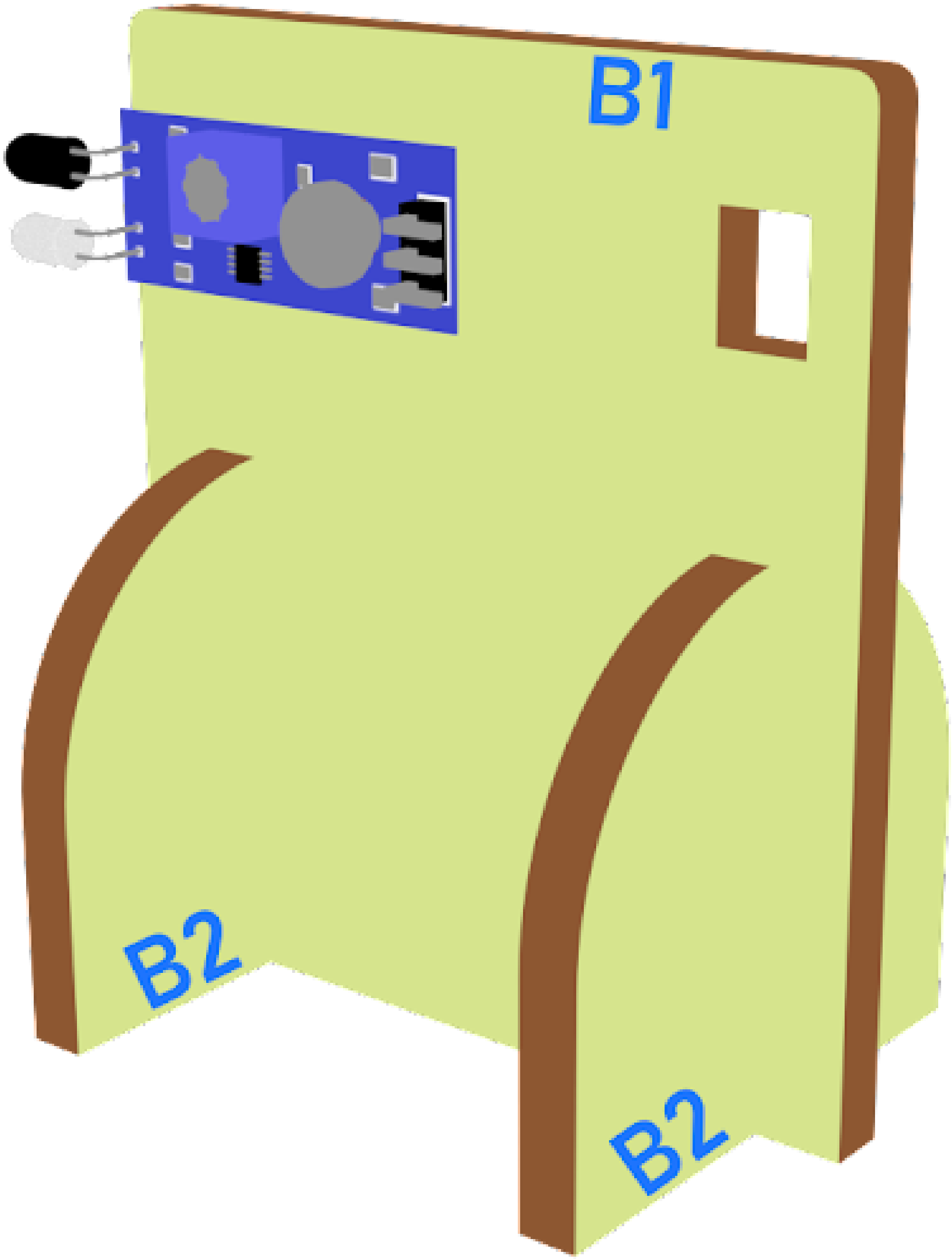
Step 4



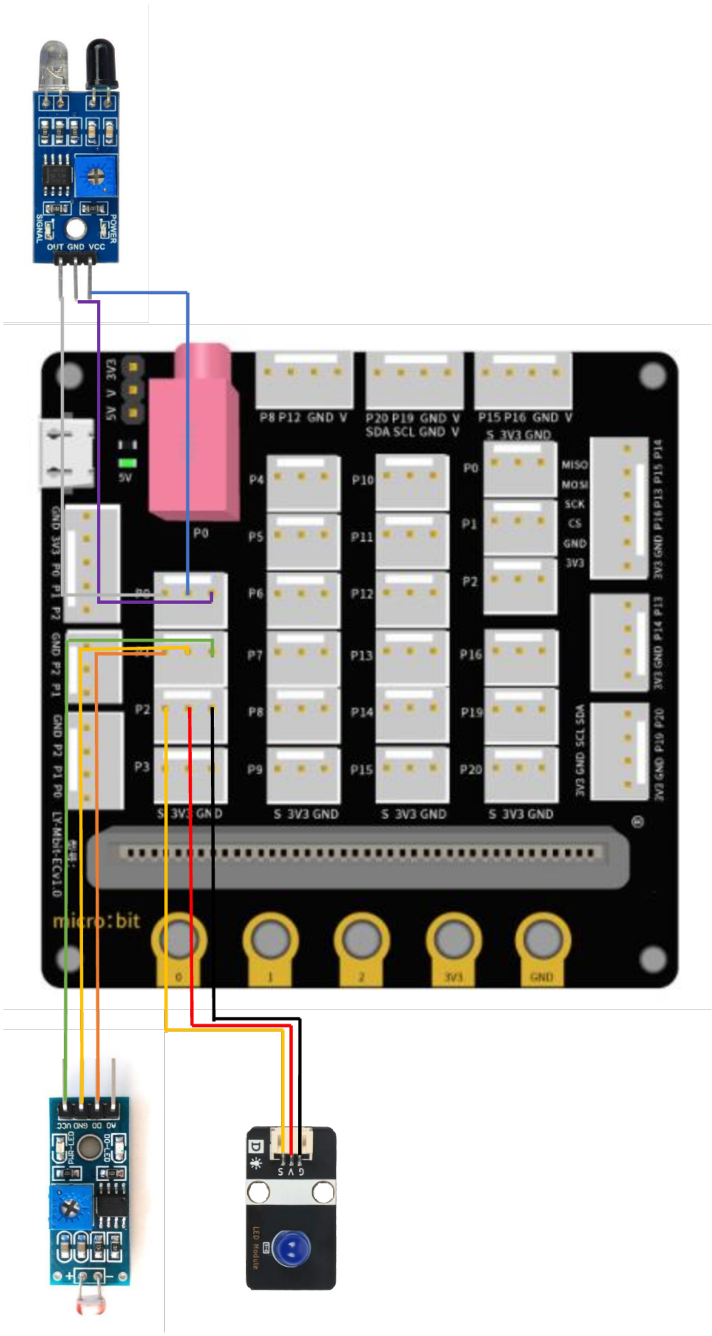
Step 5

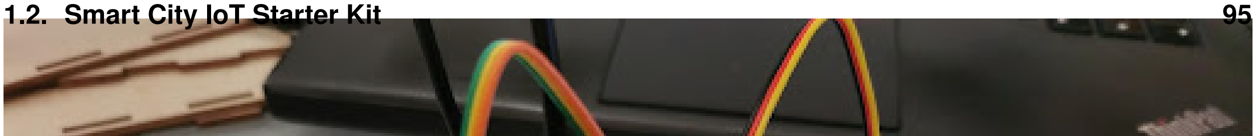


Step 6



Hardware connect

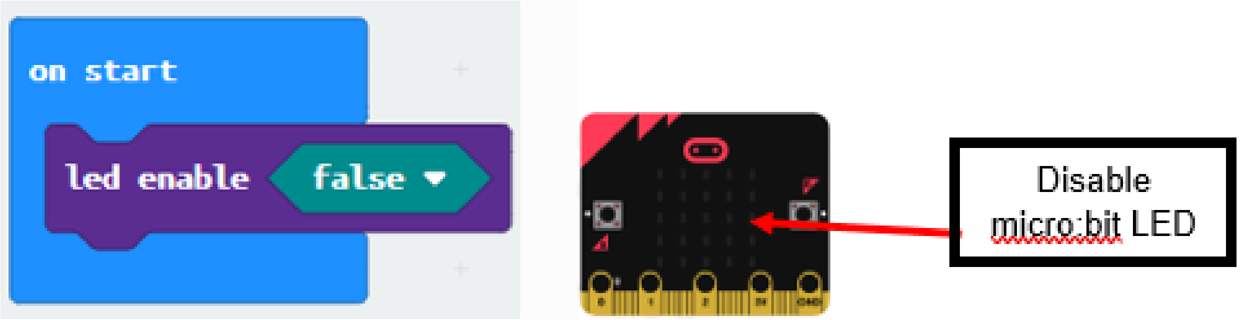




Programming (MakeCode)

Step 1. Disable micro:bit LED.

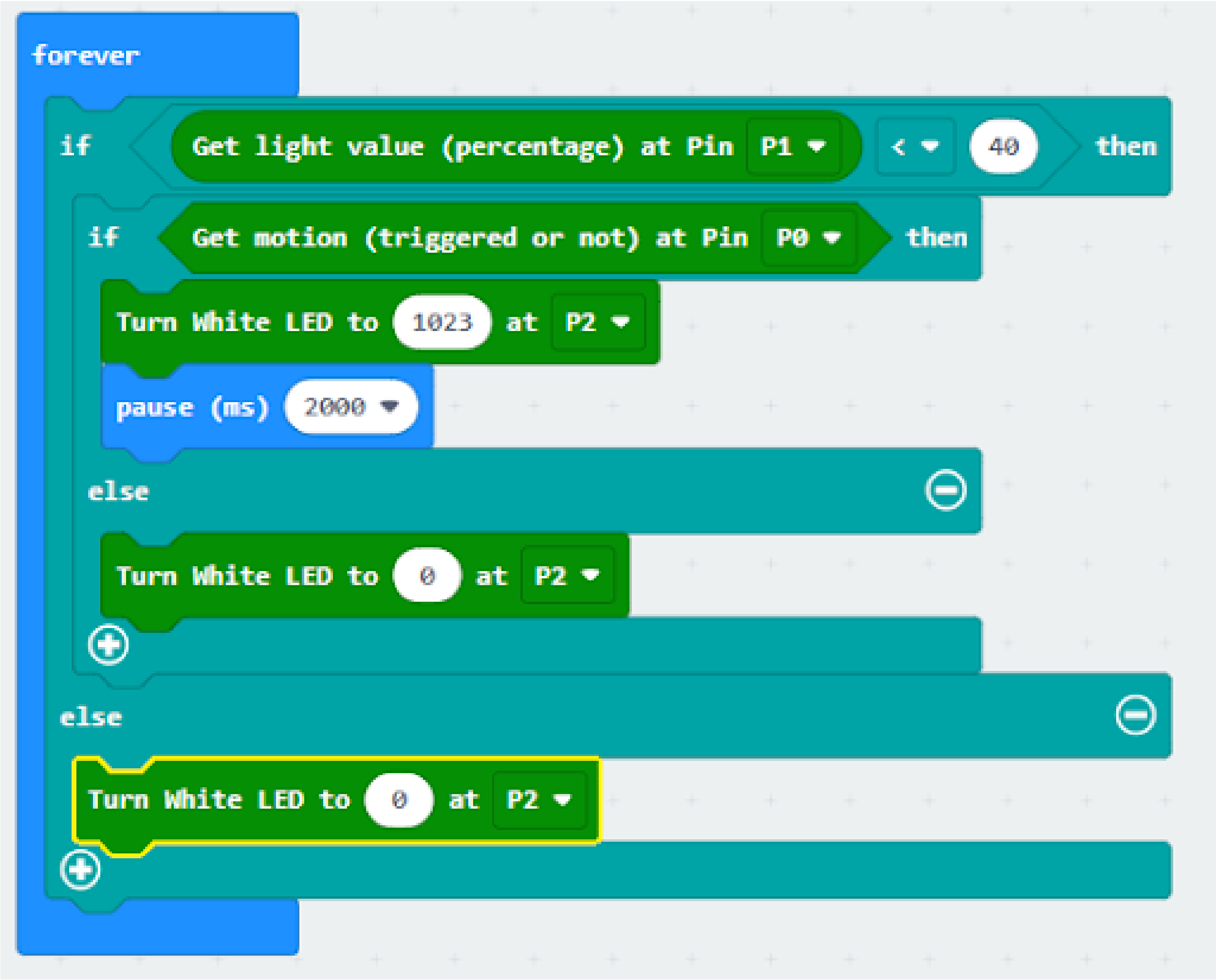
- Snap led enable false to on start
- Note that P3 is used as LED in default setting, LED need to be disable



```
on start
    led enable  false ▾
```

Step 2. Turn on LED by light sensor and obstacle Infrared avoidance sensor

- Drag forever from Basic
- Snap if statement into forever
- Set get light value (percentage) at P1 <40 and get motion (triggered or not) at P0 = true, into if statement that says motion is triggered, someone passes by.
- Then, turn white LED to 1023 at P2 as turning on white LED and pause 10 seconds.
- Else, turn white LED at P2 to 0 as turning off.

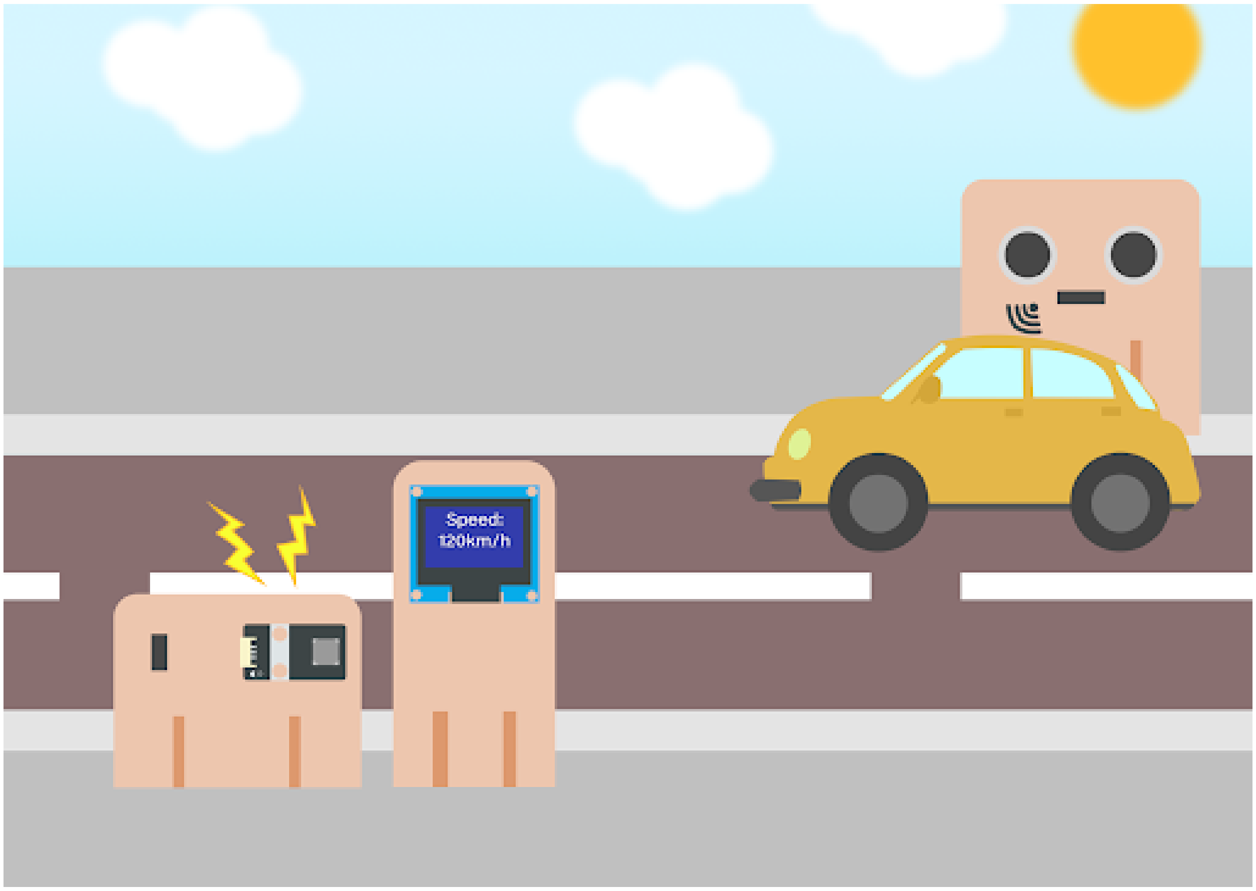


```
forever
  if  Get light value (percentage) at Pin P1 ▾  < ▾  40  then
    if  Get motion (triggered or not) at Pin P0 ▾  then
      Turn White LED to 1023 at P2 ▾
      pause (ms) 2000 ▾
    else
      Turn White LED to 0 at P2 ▾
  else
    Turn White LED to 0 at P2 ▾
```

Result

Think

1.2.4 Car Speed Monitoring

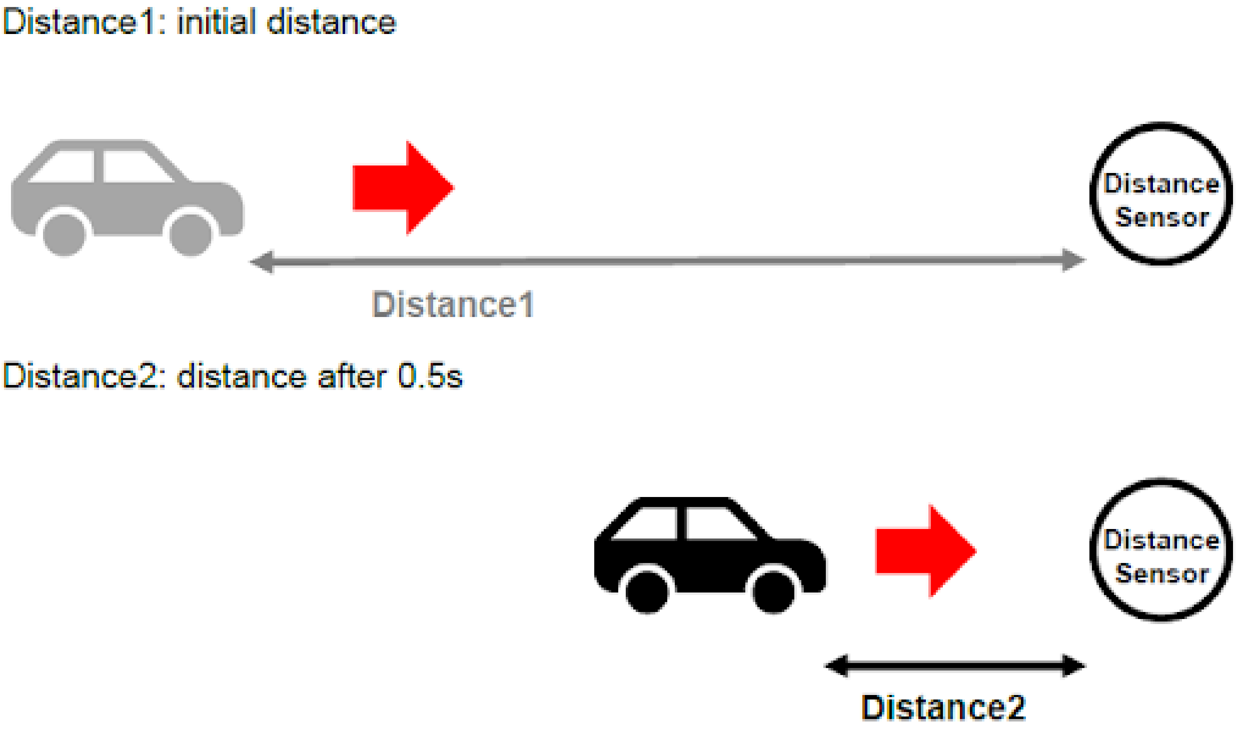


Goal

background

What is car speed monitoring?

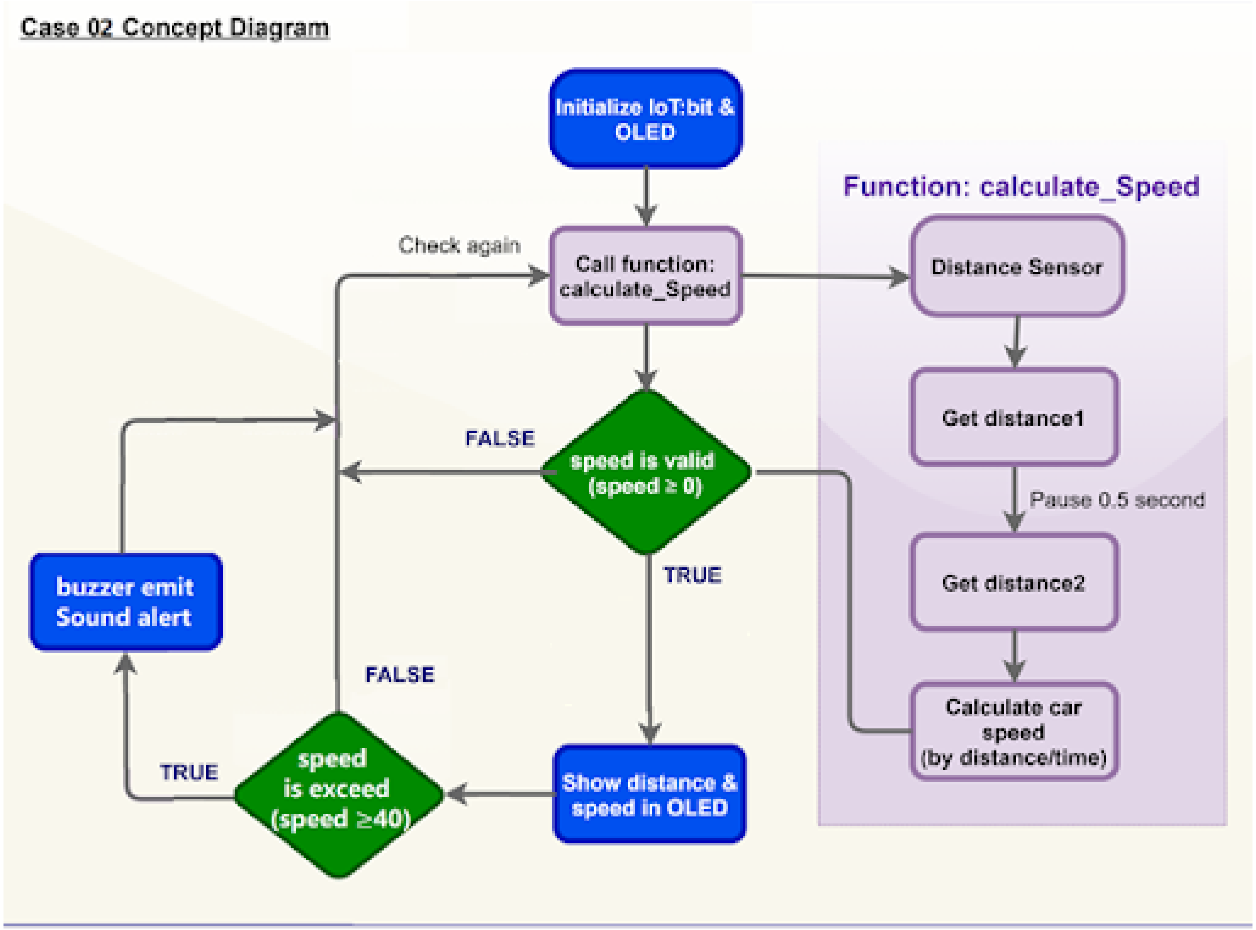
Car speed monitor operation



Distance1: initial distance

Distance2: distance after 0.5s

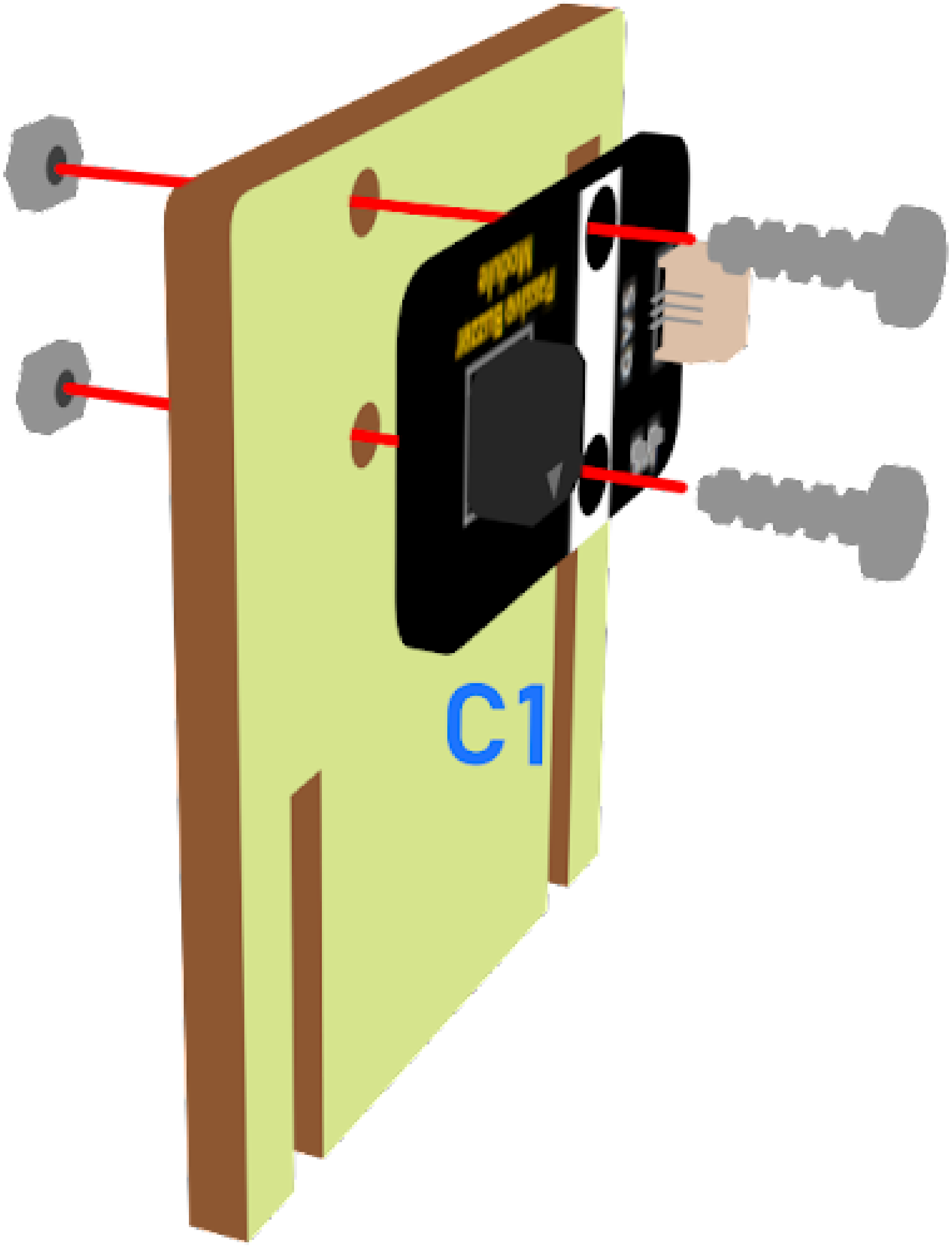
By equation, $\text{Speed} = \frac{\text{distance}}{\text{time}}$,we found that $\text{Speed} = \frac{\text{distance1} - \text{distance2}}{\text{time}}$


Case 02 Concept Diagram
Initialize IoT:bit & OLED
Call function: calculate_Speed
Check again
Function: calculate_Speed
Distance Sensor
Get distance1
Pause 0.5 second
Get distance2
Calculate car speed (by distance/time)
FALSE
speed is valid (speed ≥ 0)
TRUE
buzzer emit Sound alert
FALSE
TRUE
speed is exceed (speed ≥40)
Show distance & speed in OLED

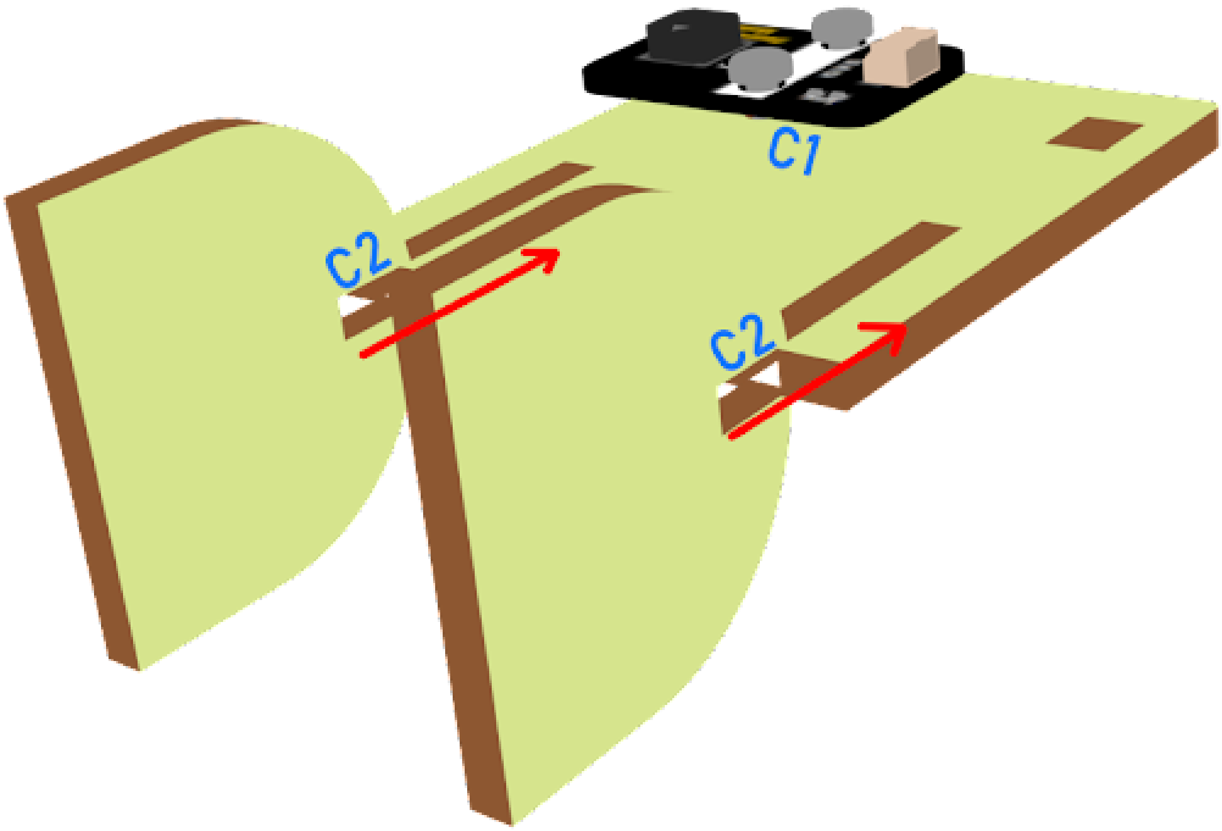
Part List

Assembly step

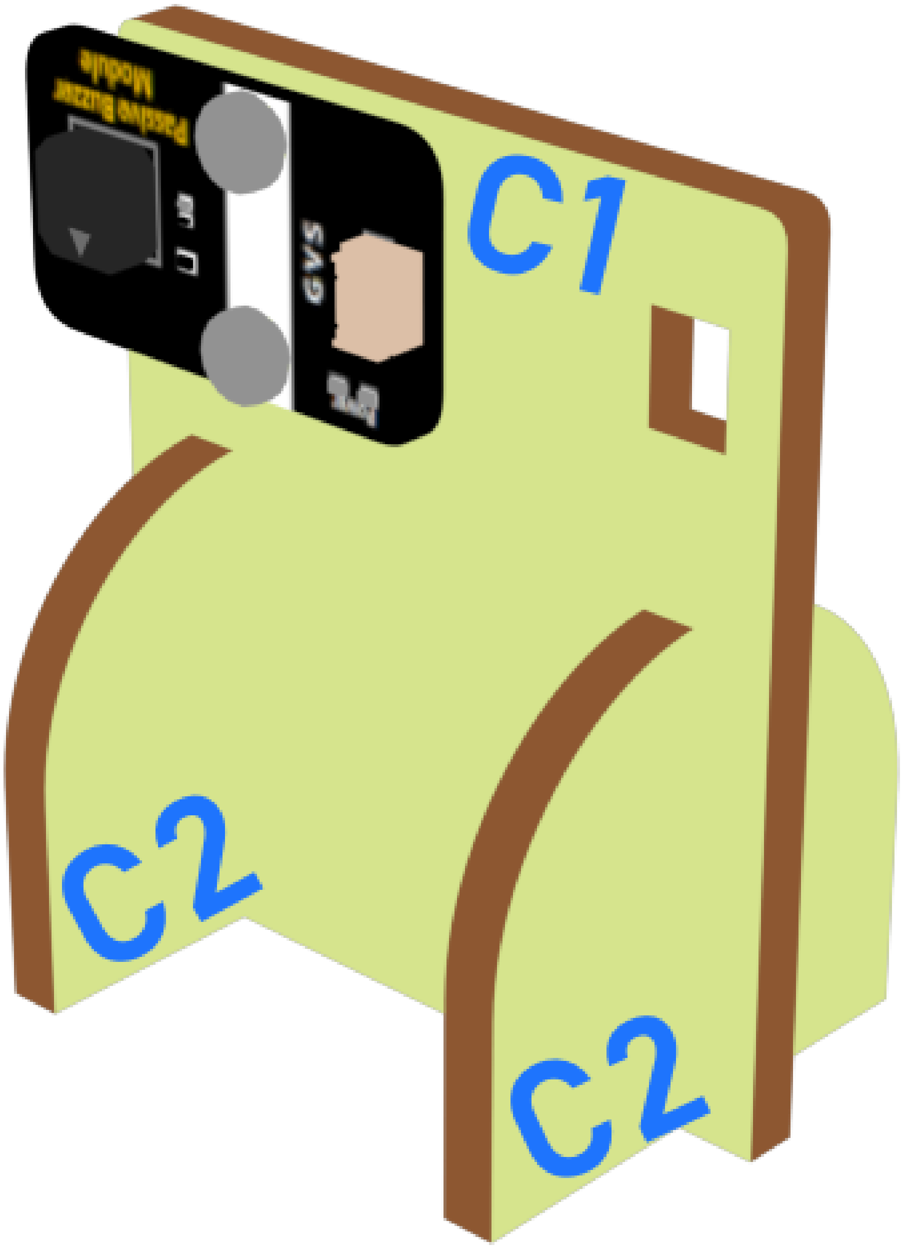
Step 1



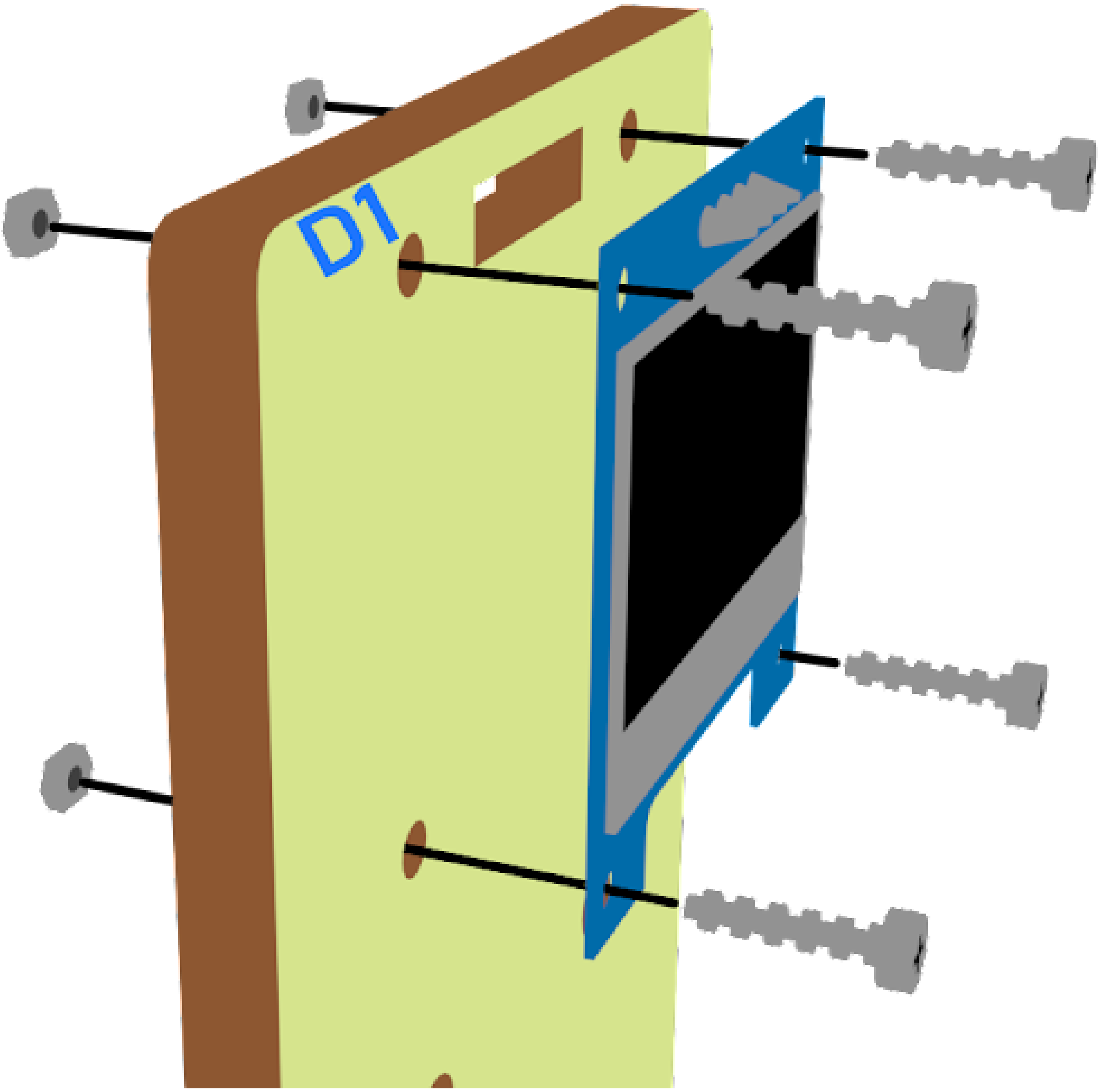
Step 2



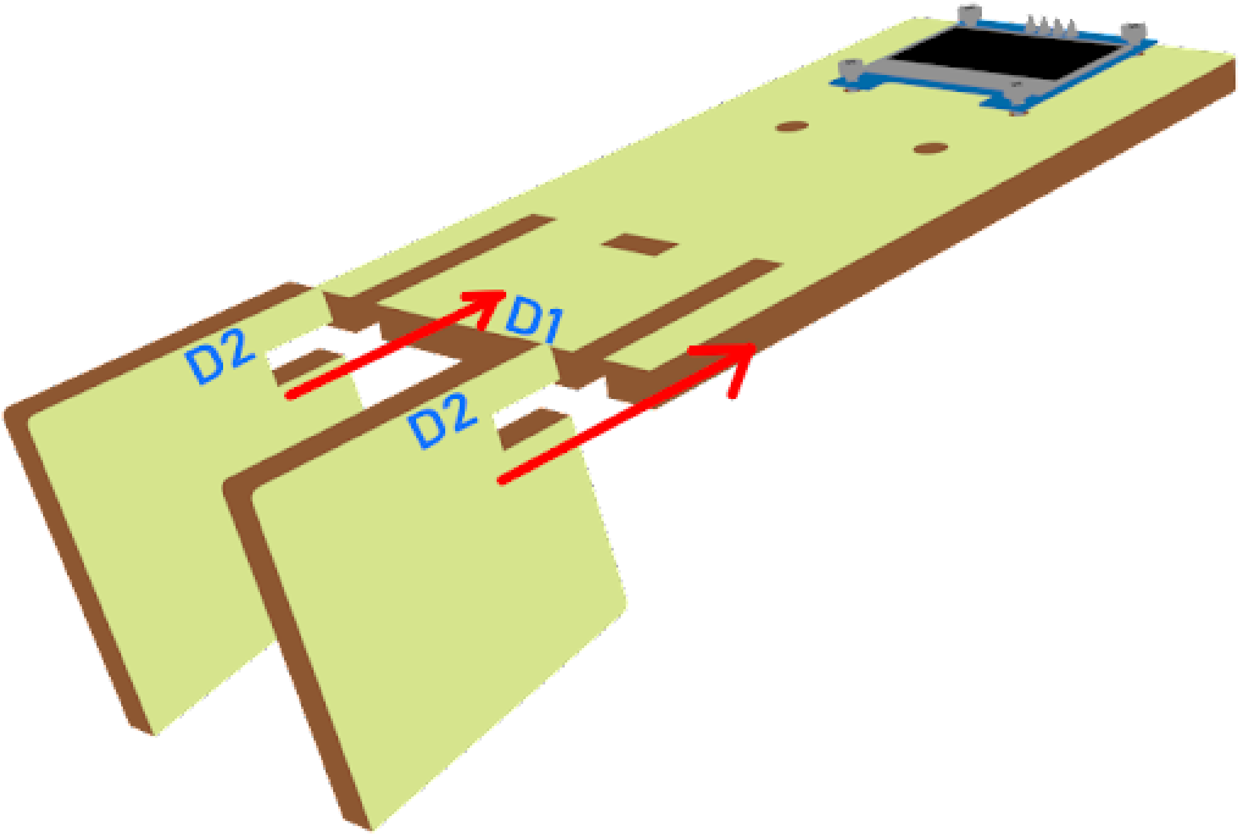
Step 3



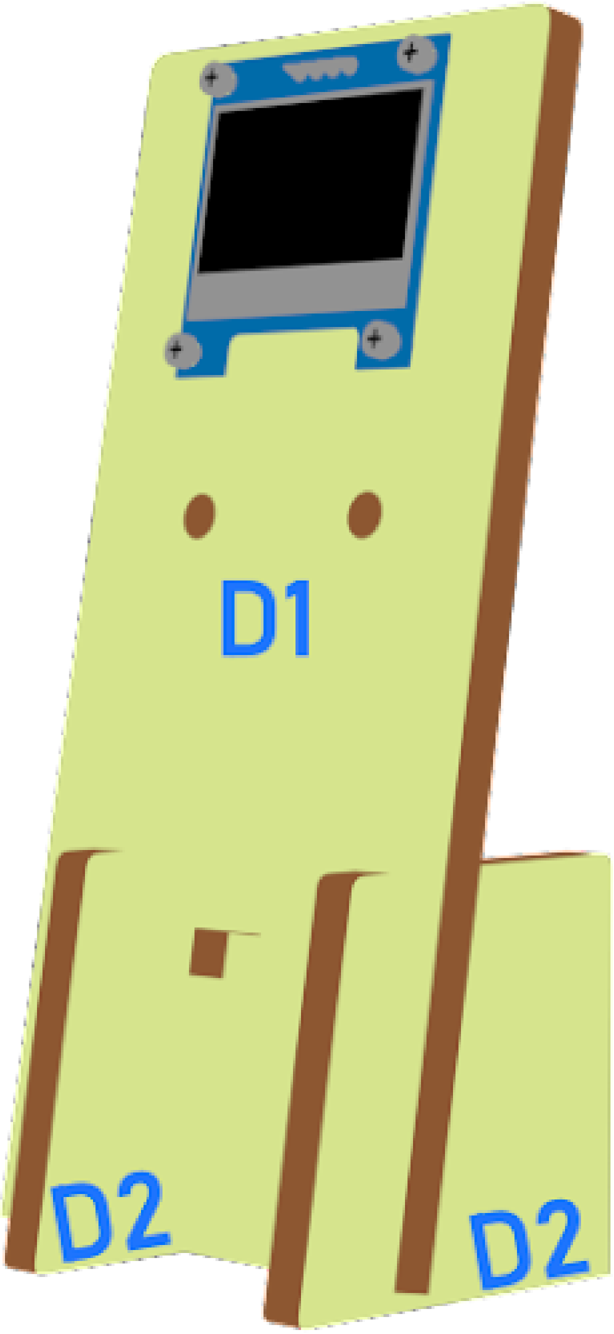
Step 4



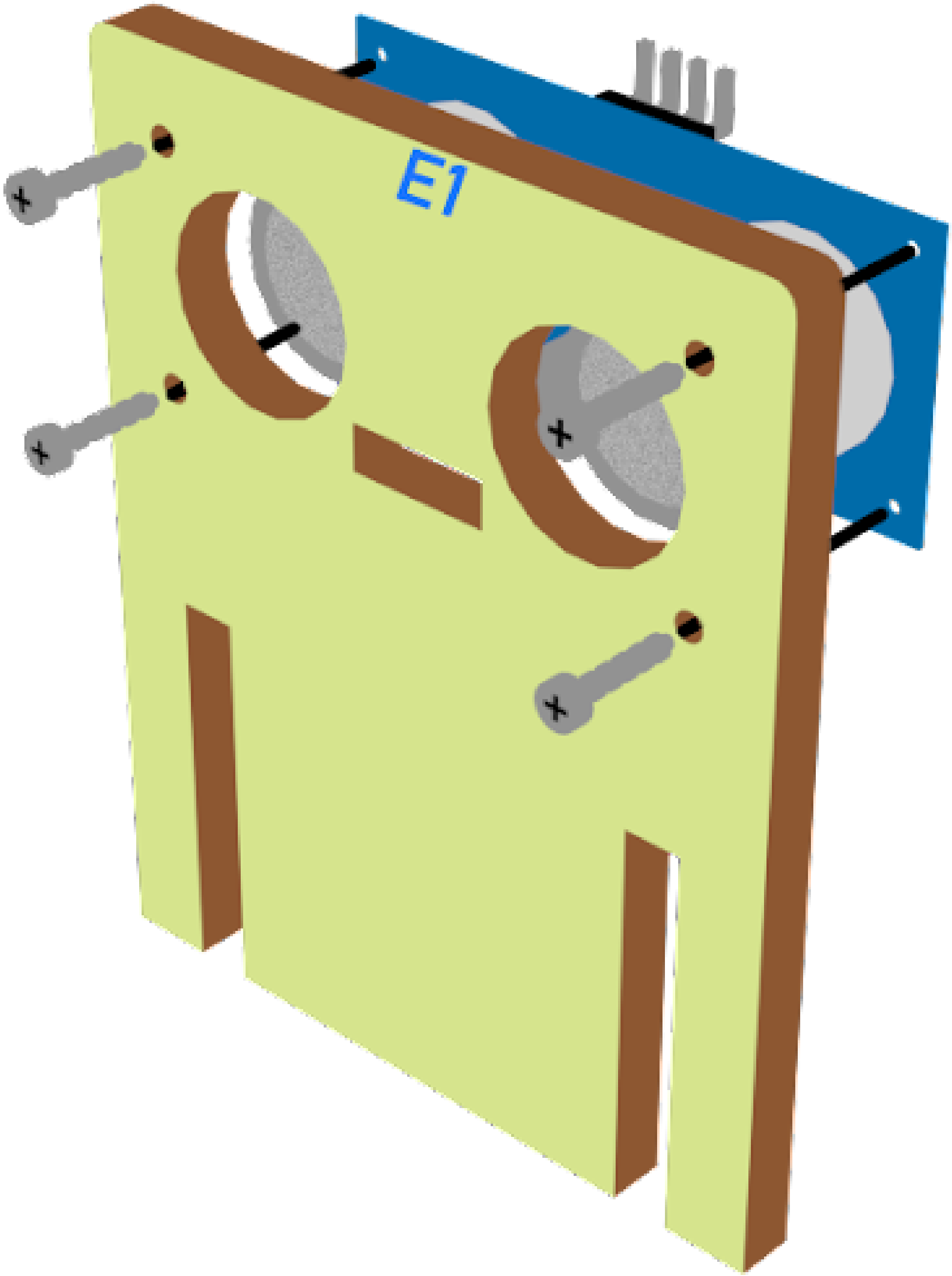
Step 5



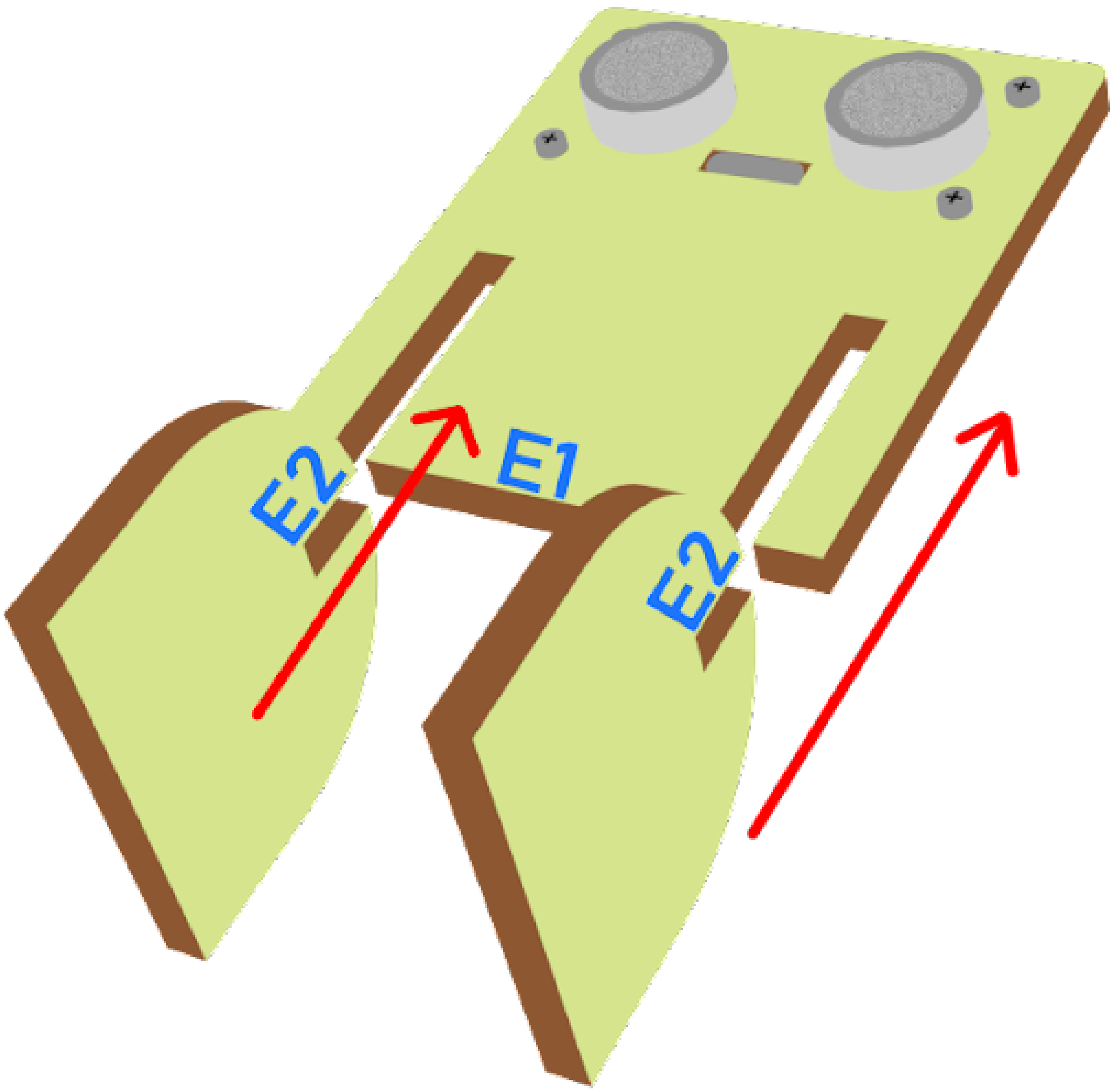
Step 6



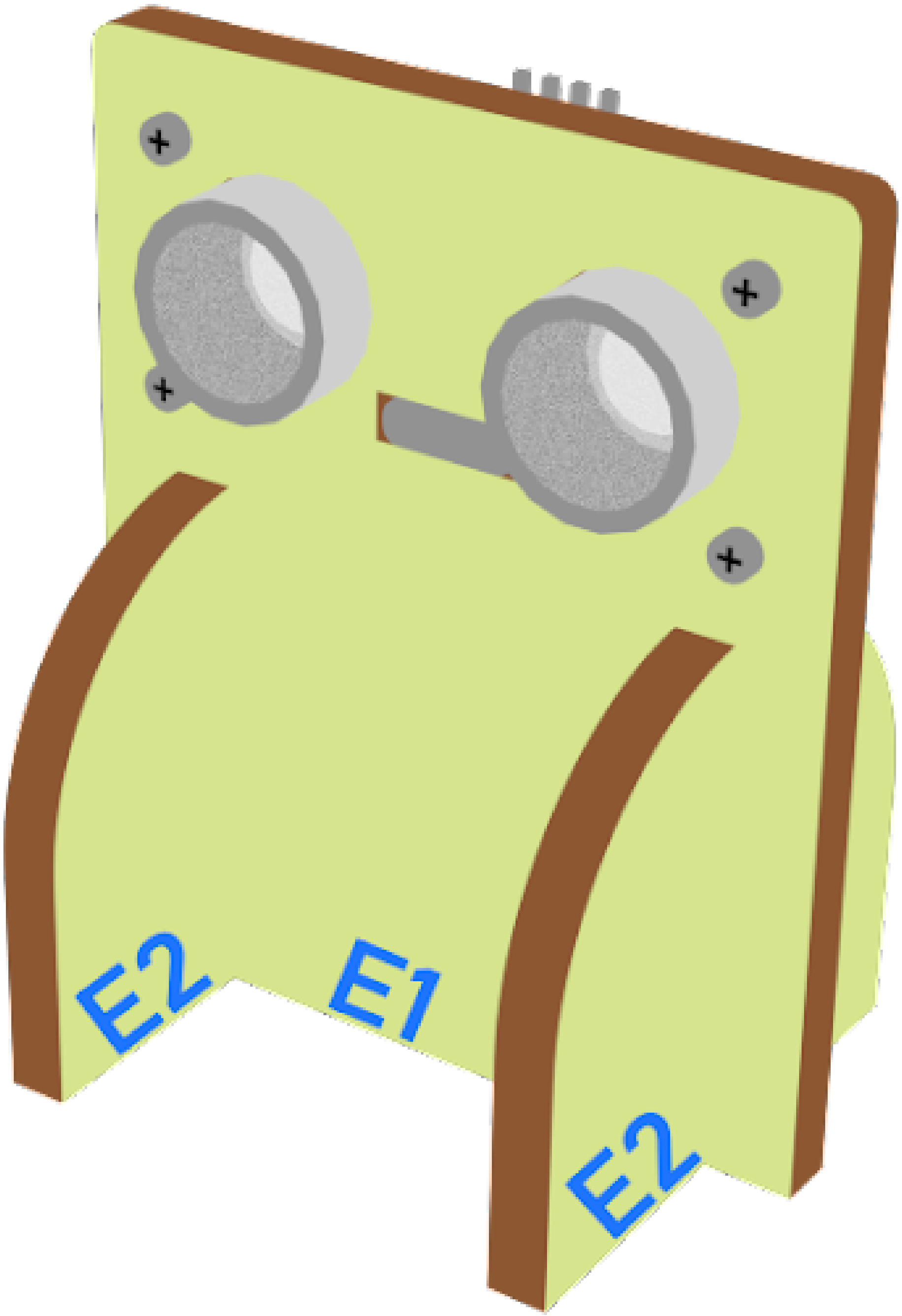
Step 7



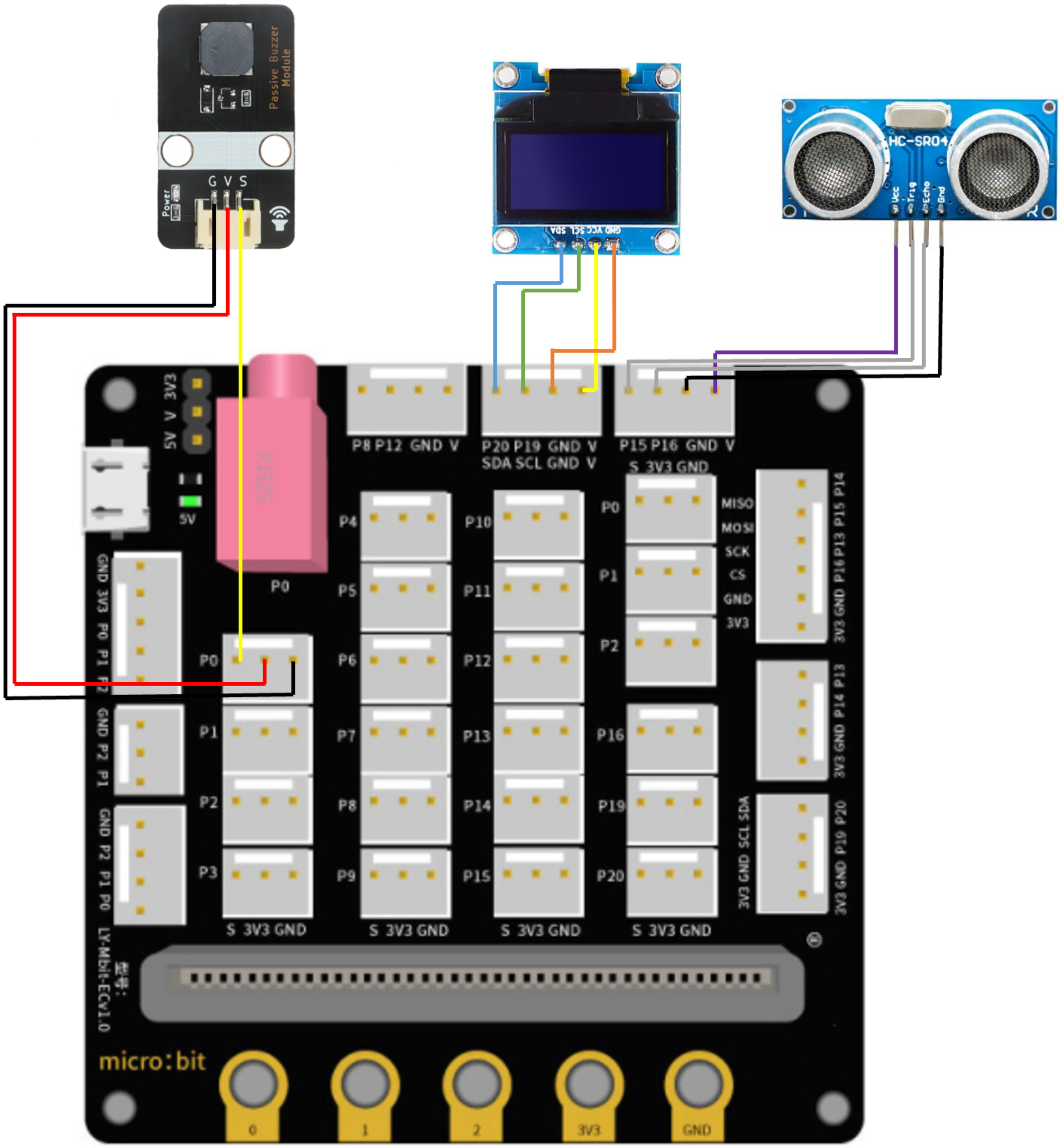
Step 8

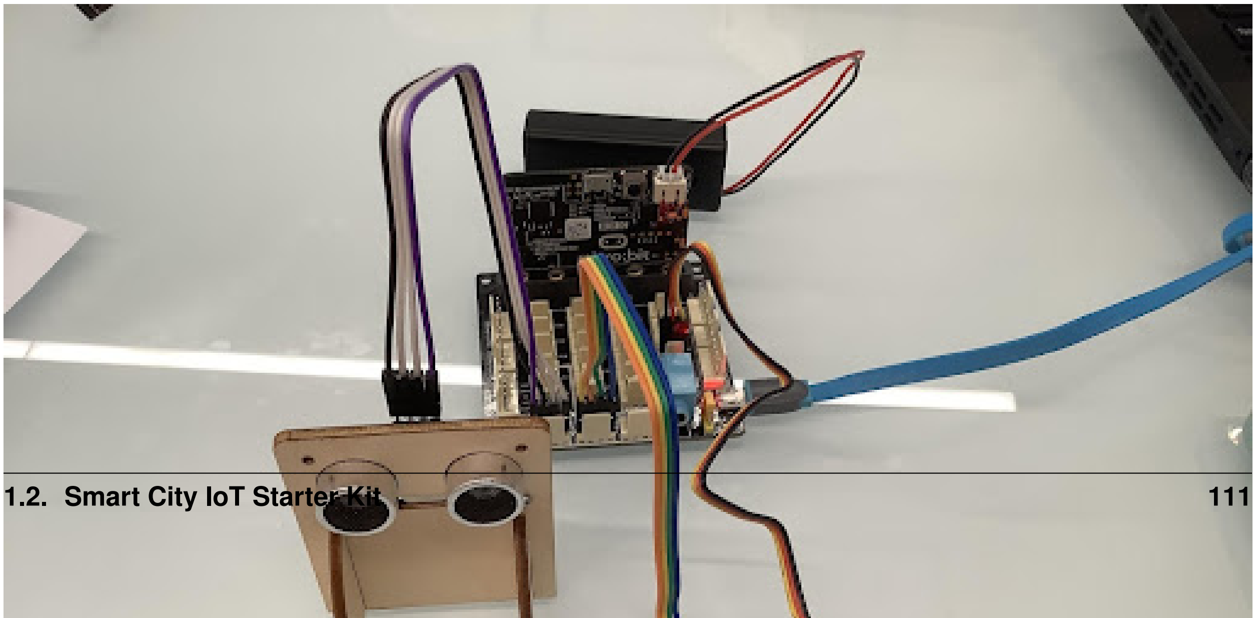


Step 9



Hardware connect

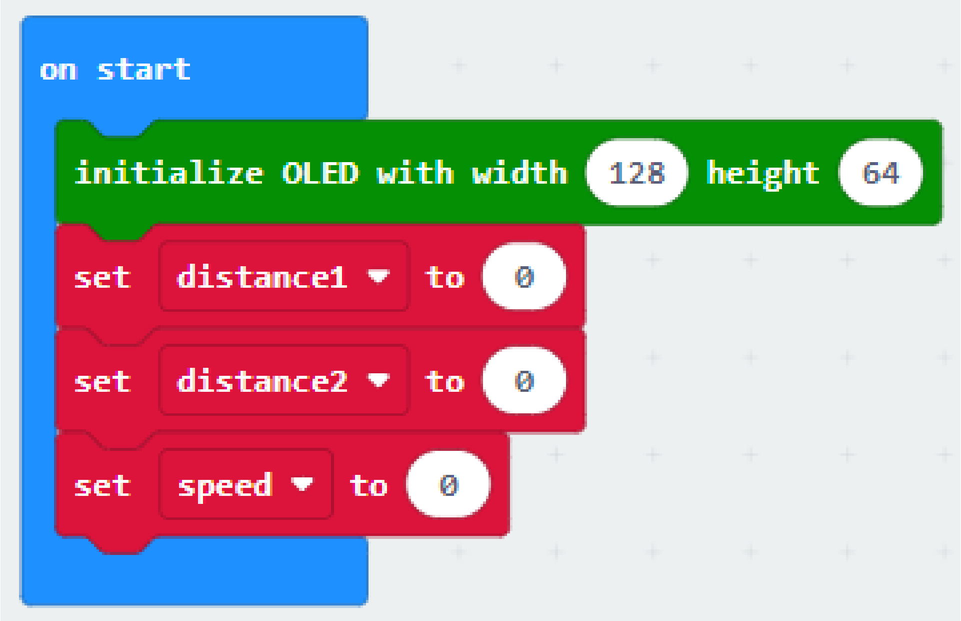




Programming (MakeCode)

Step 1. Initialize OLED screen

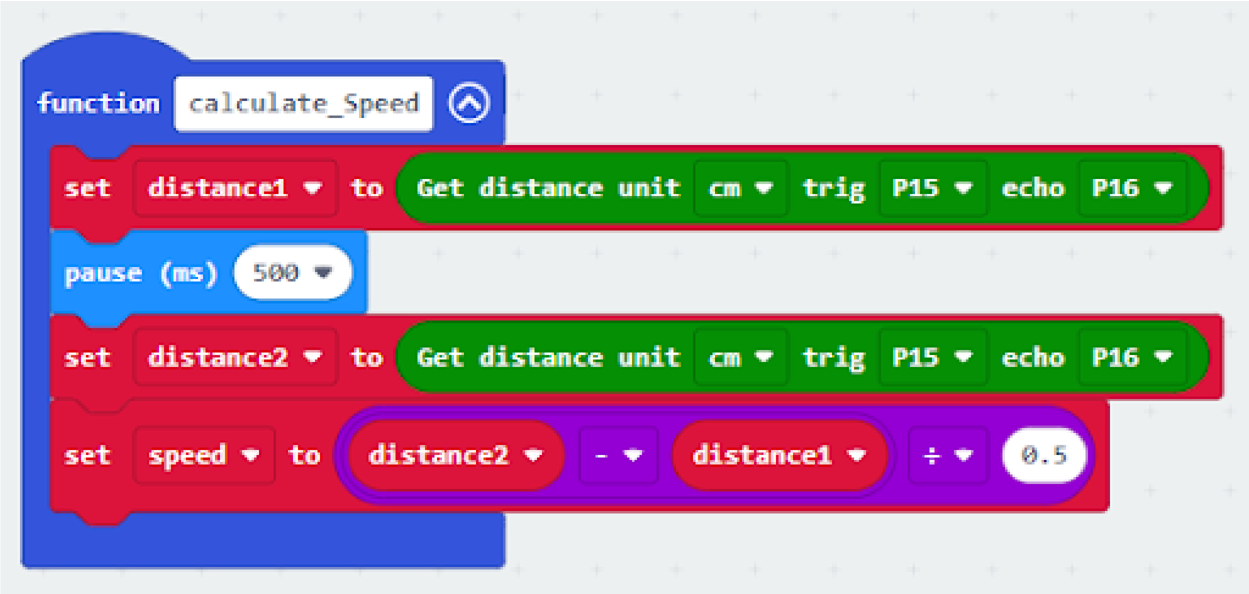
- Drag Initialize OLED with width:128, height: 64 to on start
- Set distance1, distance2 and speed to 0 from variables



```
on start
    initialize OLED with width 128 height 64
    set distance1 to 0
    set distance2 to 0
    set speed to 0
```

Step 2. Set up function (calculate_Speed)

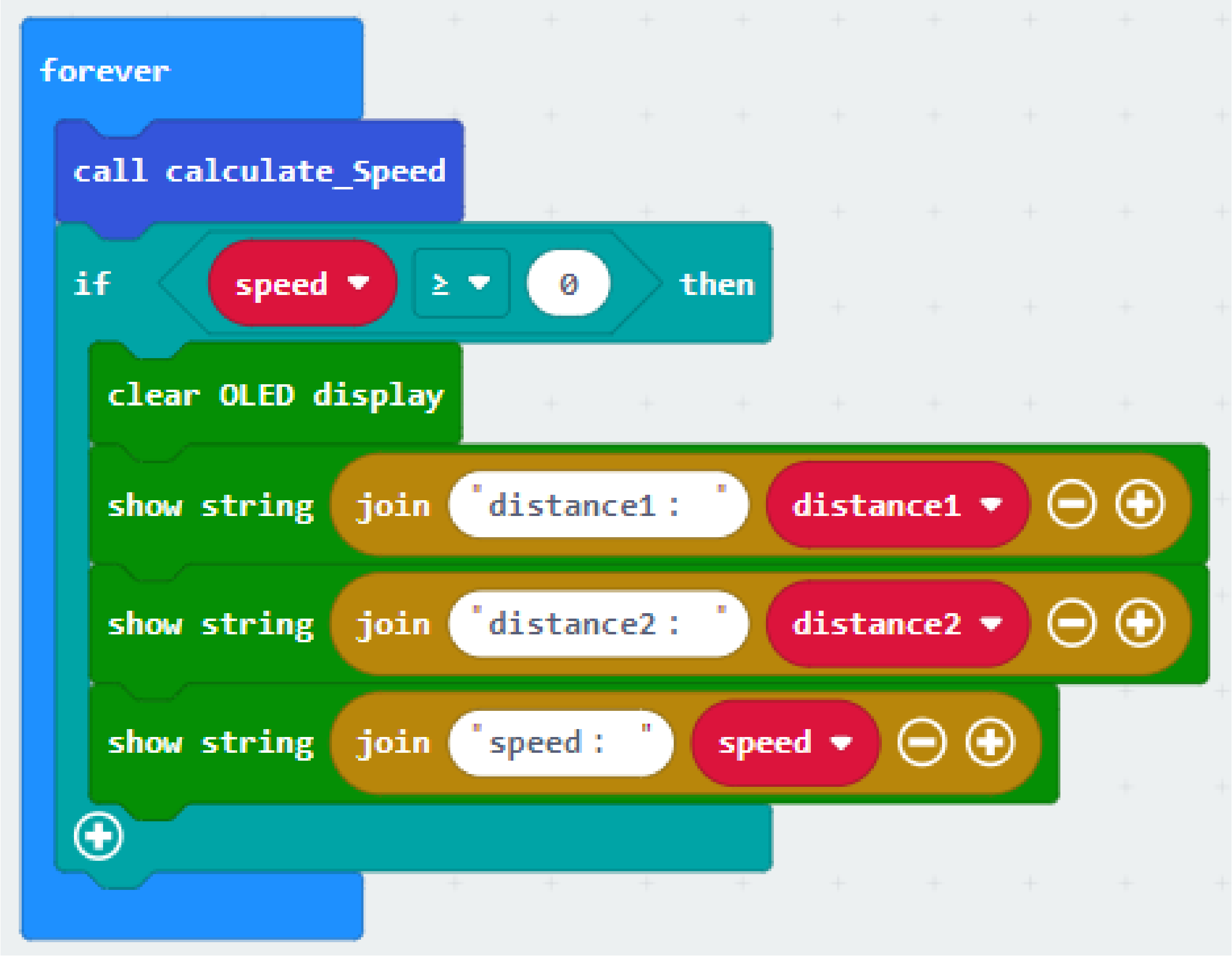
- Set up a new function calculate_Speed from Advanced > Functions.
- Set distance1 to get distance unit cm trig P14 echo P15 (distance from the car to the distance sensor before 0.5 second) Drag Pause to wait 500ms and set distance2 to get distance unit cm trig P14 echo P15 (distance from the car to the distance sensor after 0.5 second)
- By the equation of speed = distance / time. We get the speed of the moving car to (distance1-distance2)/0.5 (unit: cm/s)



```
function calculate_Speed
    set distance1 to Get distance unit cm trig P15 echo P16
    pause (ms) 500
    set distance2 to Get distance unit cm trig P15 echo P16
    set speed to distance2 - distance1 ÷ 0.5
```

Step 3. Calculate car speed

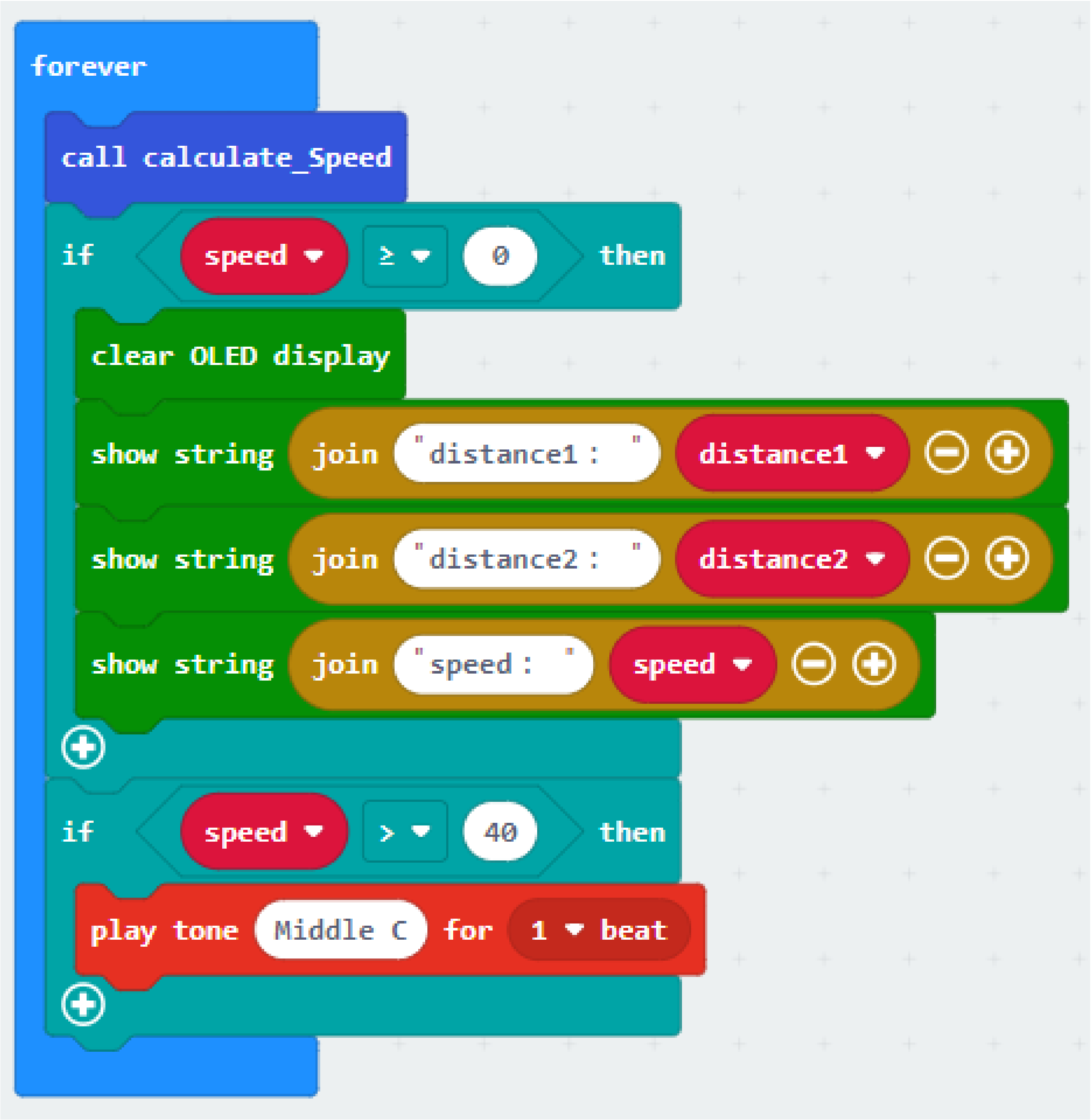
- In block forever, call function calculate_Speed from Advanced > Functions to get the speed of the moving car
- Snap If statement into the loop
- Snap clear OLED display from OLED to avoid overlap
- Snap show string and show value of variables distance1, distance2 and speed



```
forever
    call calculate_Speed
    if speed ≥ 0 then
        clear OLED display
        show string join "distance1 : " distance1
        show string join "distance2 : " distance2
        show string join "speed : " speed
```

Step 4. buzzer

- Snap If statement into the loop
- If speed 40, then snap play tone Middle C for 1 beat from music

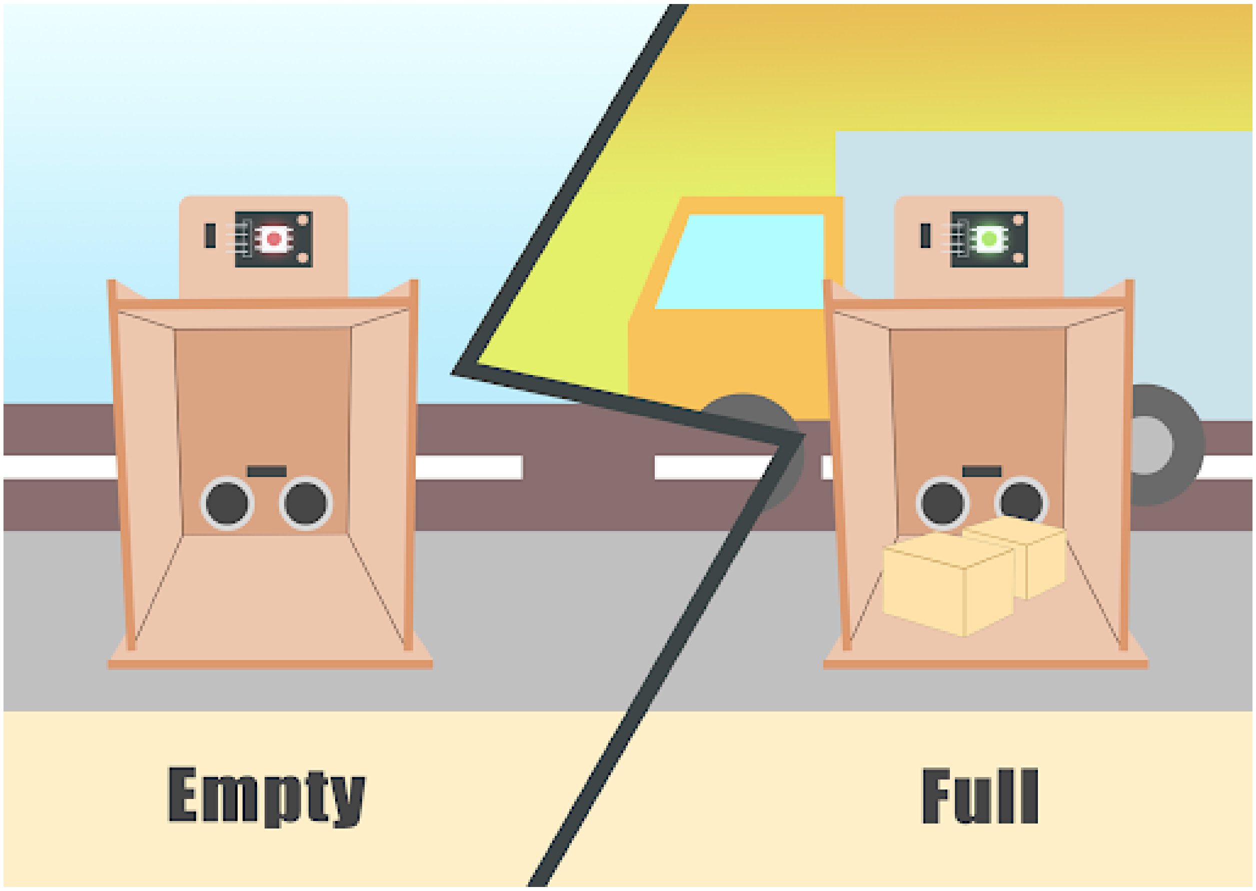


```
forever
    call calculate_Speed
    if speed ≥ 0 then
        clear OLED display
        show string join "distance1 : " distance1
        show string join "distance2 : " distance2
        show string join "speed : " speed
    if speed > 40 then
        play tone Middle C for 1 beat
```

Result

Think

1.2.5 Unloading Alert System

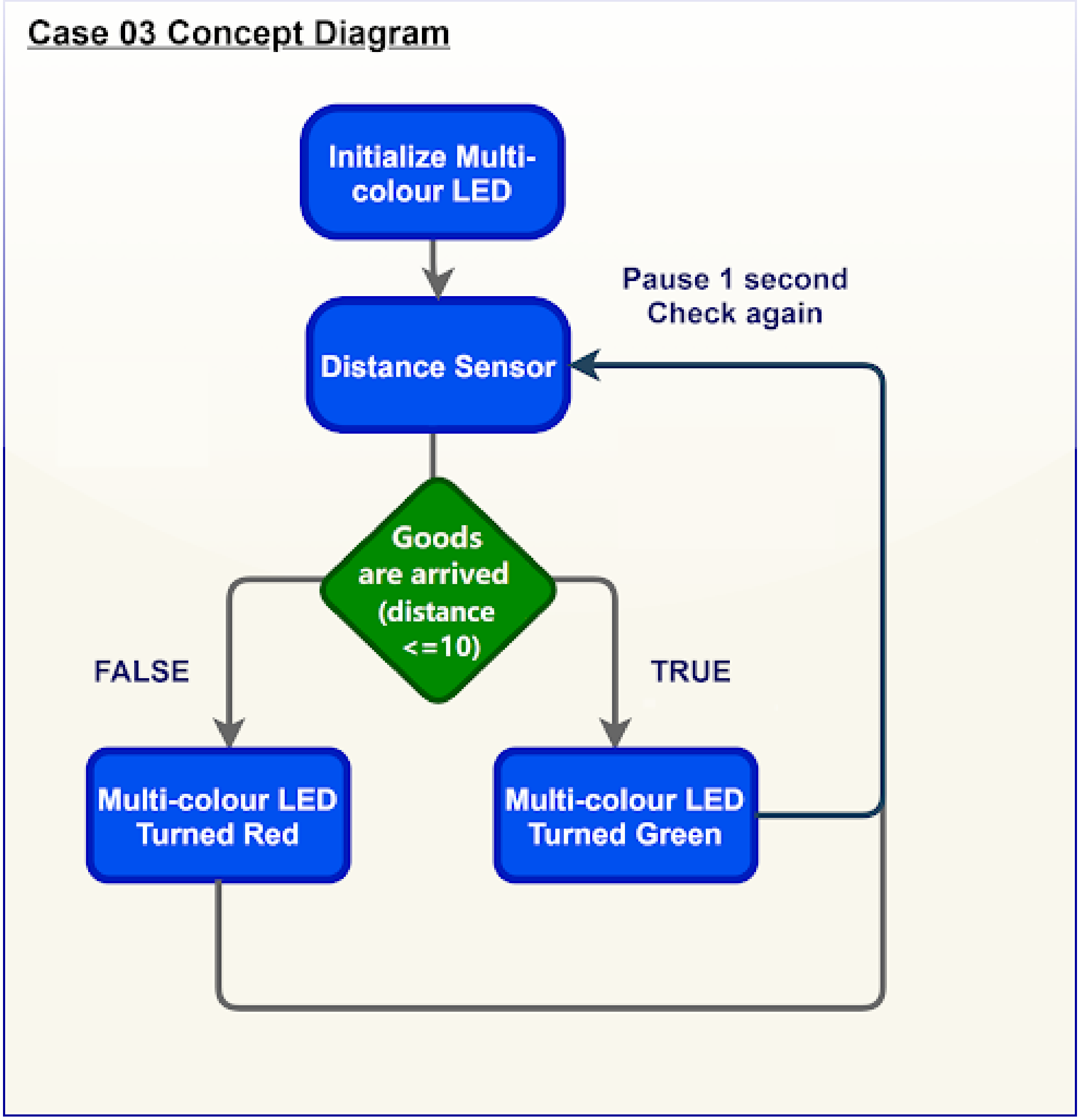


Goal

Background

What is an Unloading alert system?

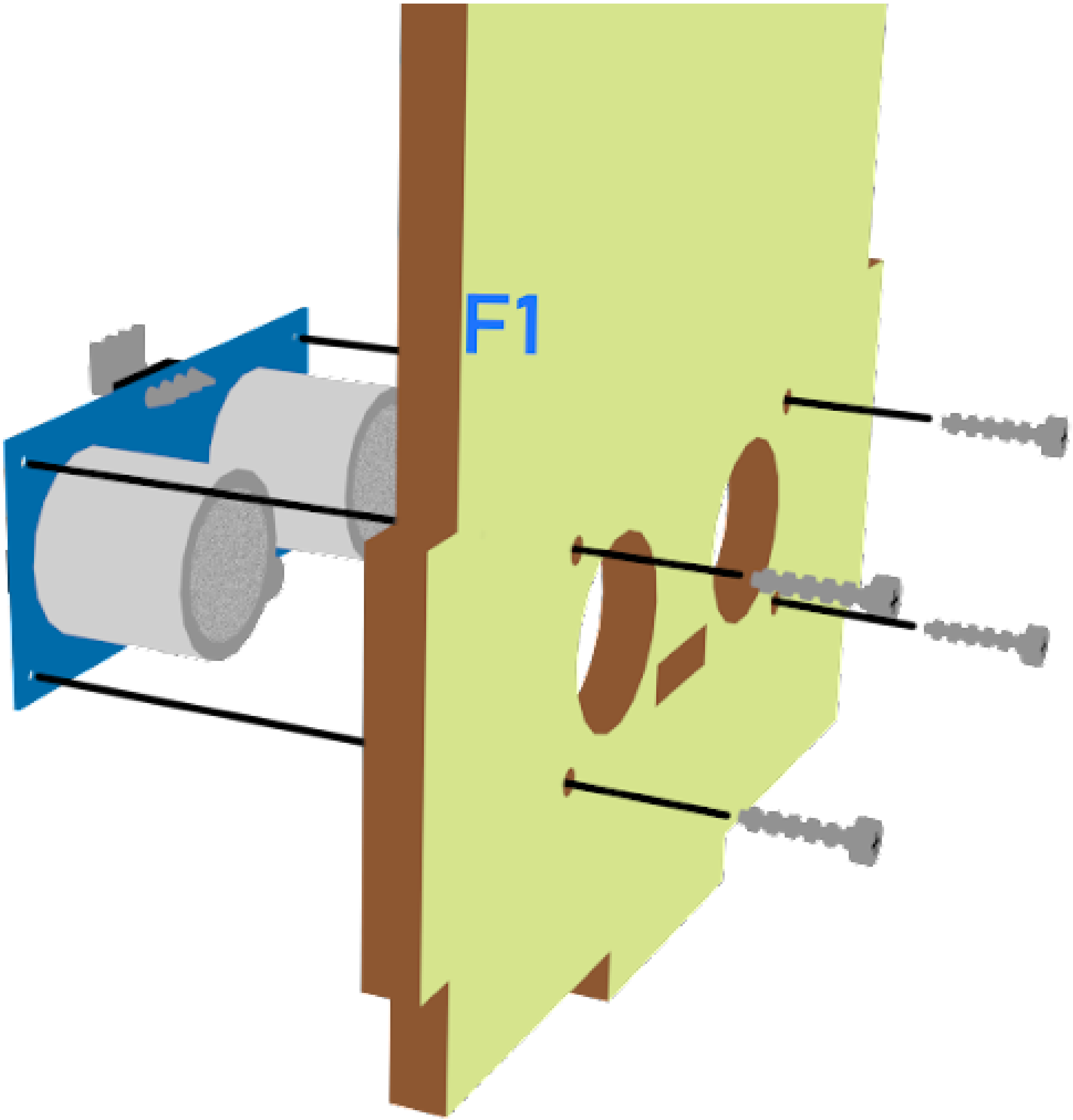
Unloading alert system operation


Case 03 Concept Diagram
Initialize Multi-colour LED
Distance Sensor
Pause 1 second
Check again
Goods are arrived (distance <=10)
FALSE
TRUE
Multi-colour LED Turned Red
Multi-colour LED Turned Green

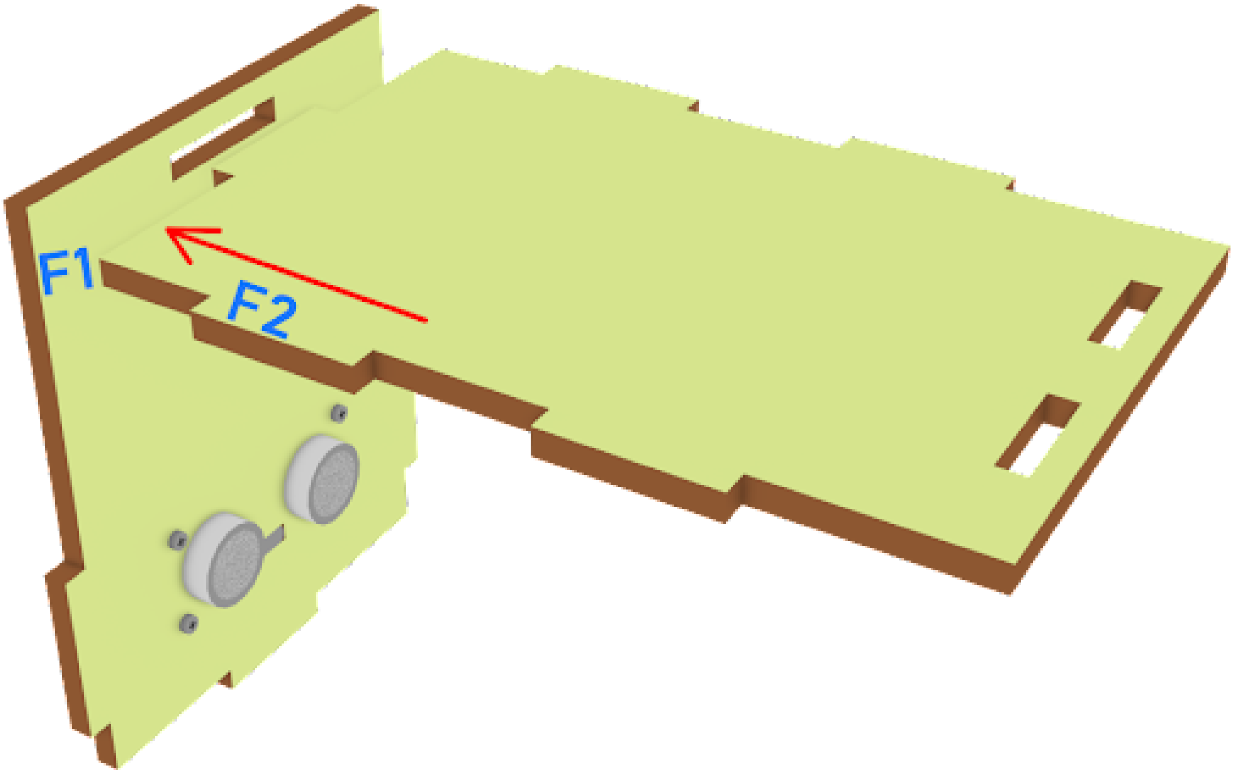
Part List

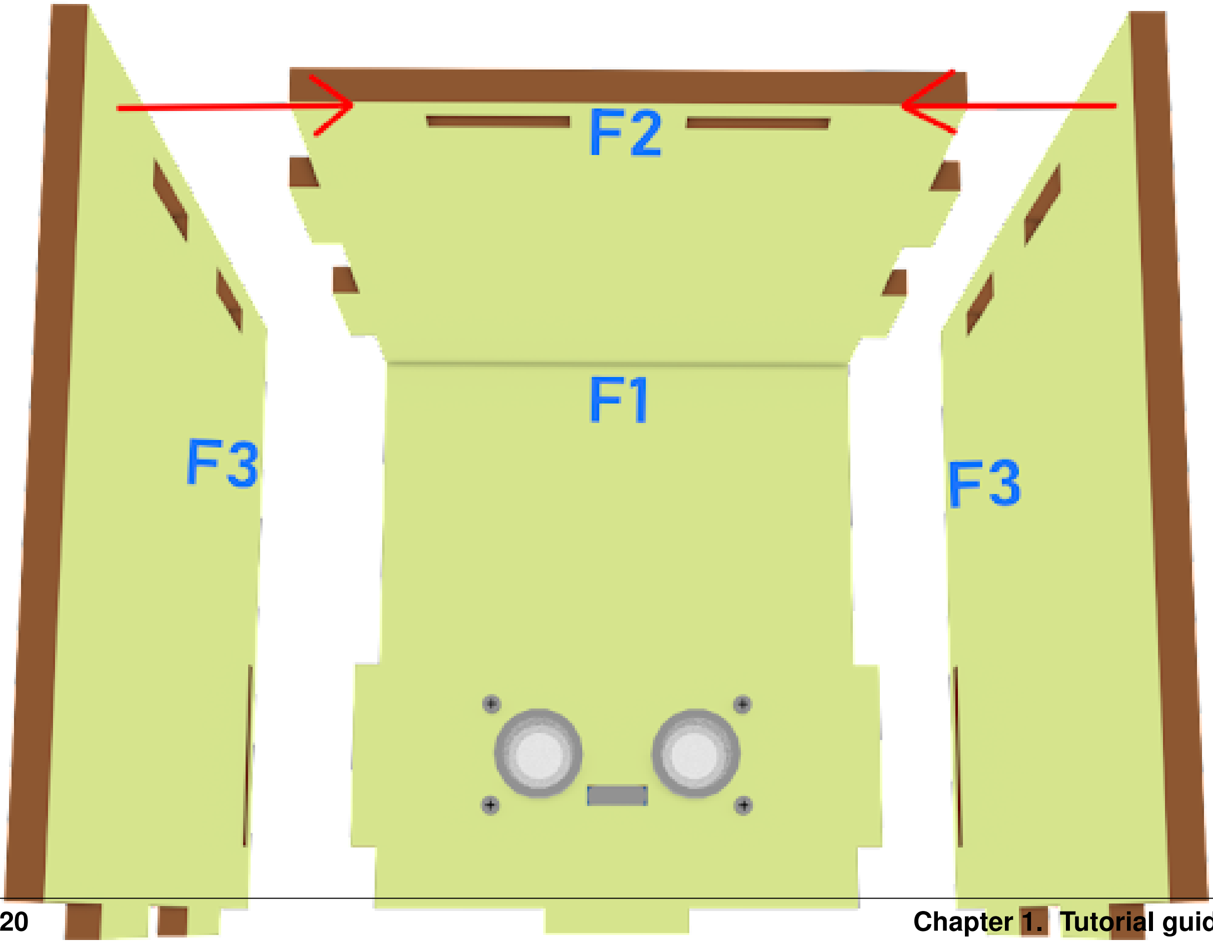
Assembly step

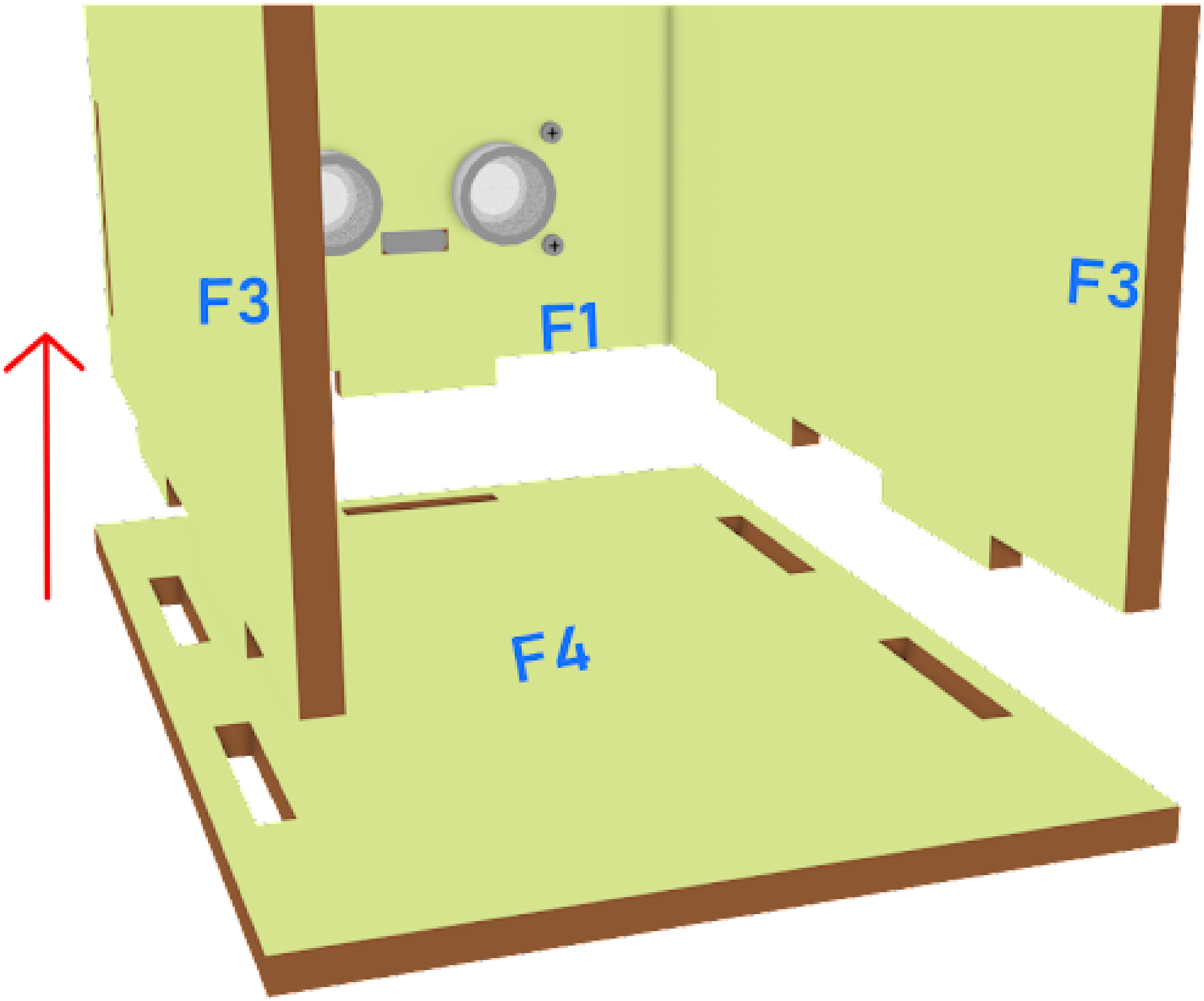
Step 1



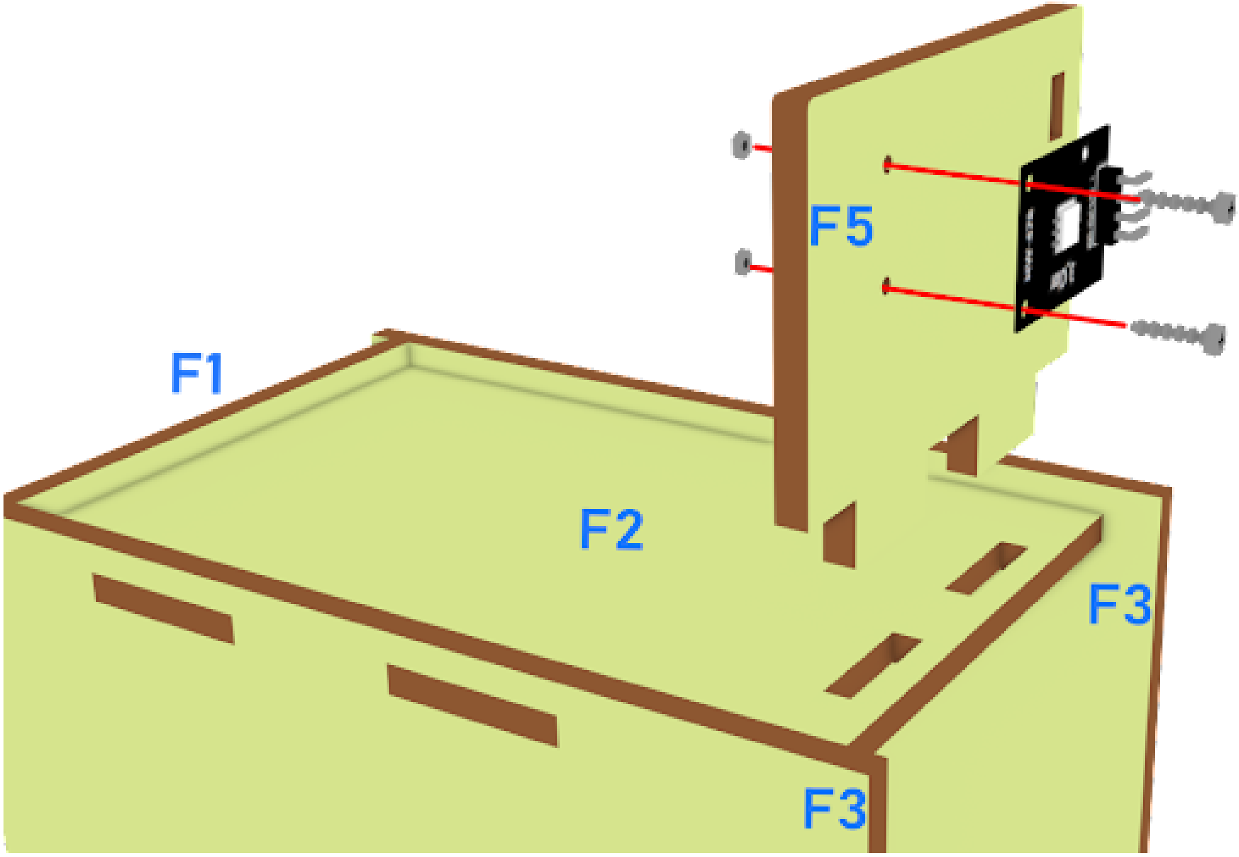
Step 2



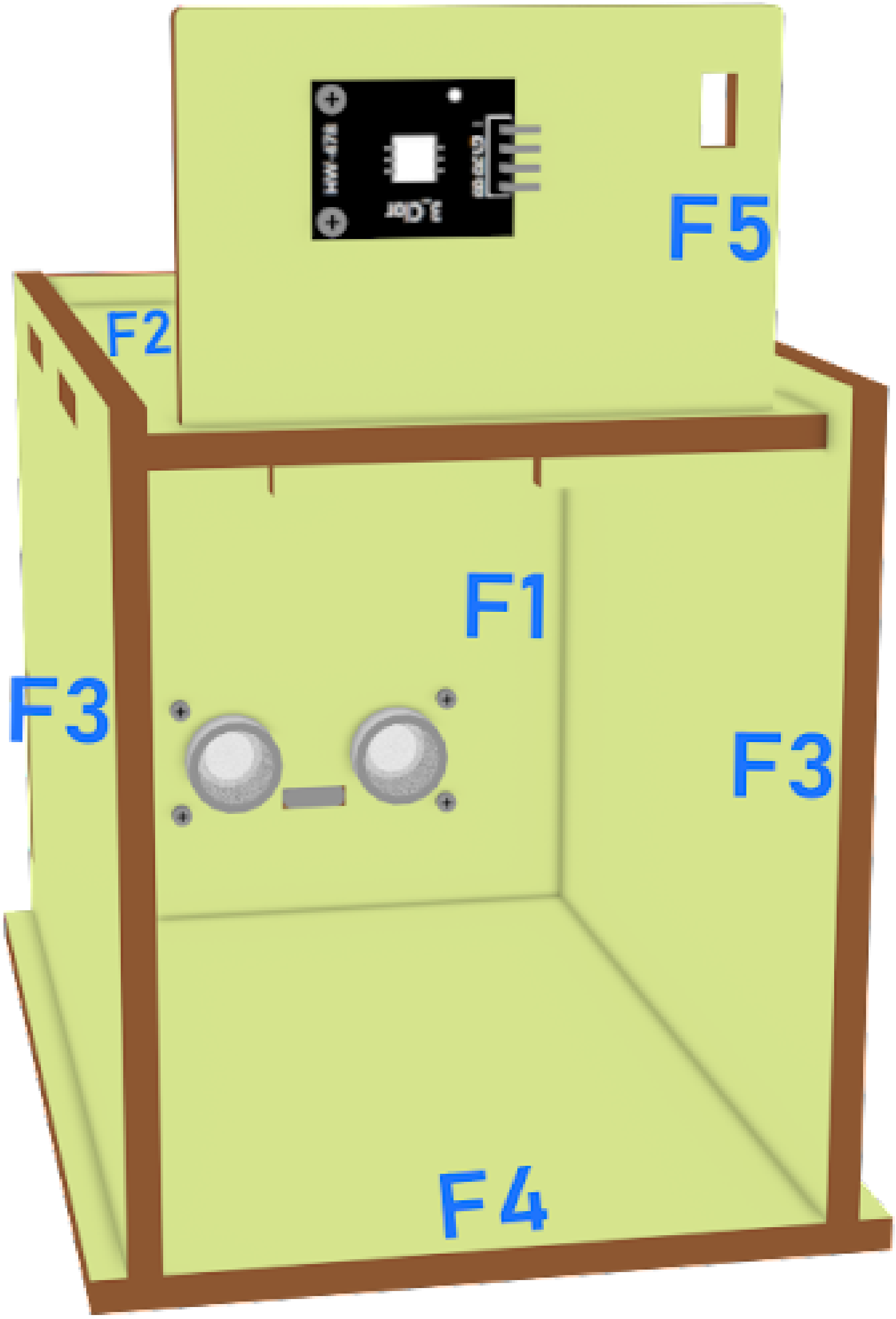



F3
F1
F3
F4

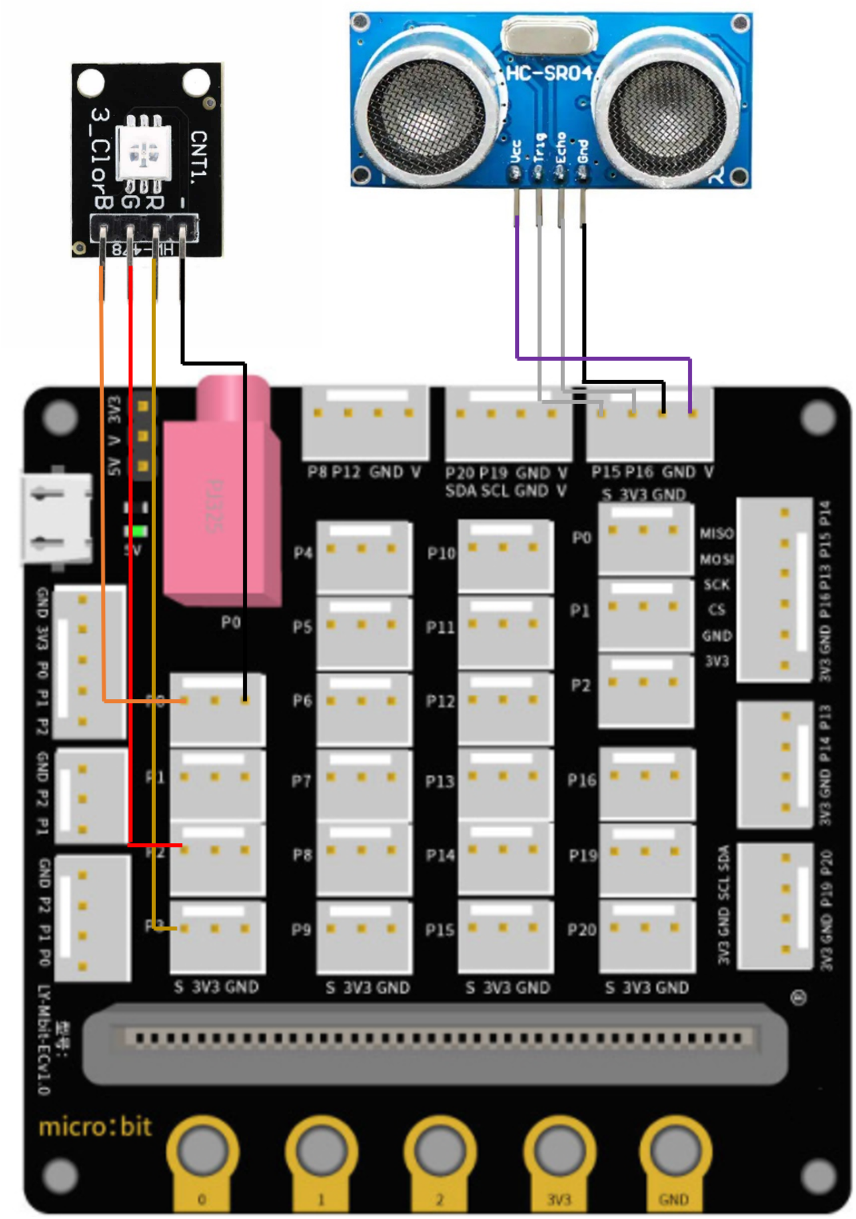
Step3



Step 4



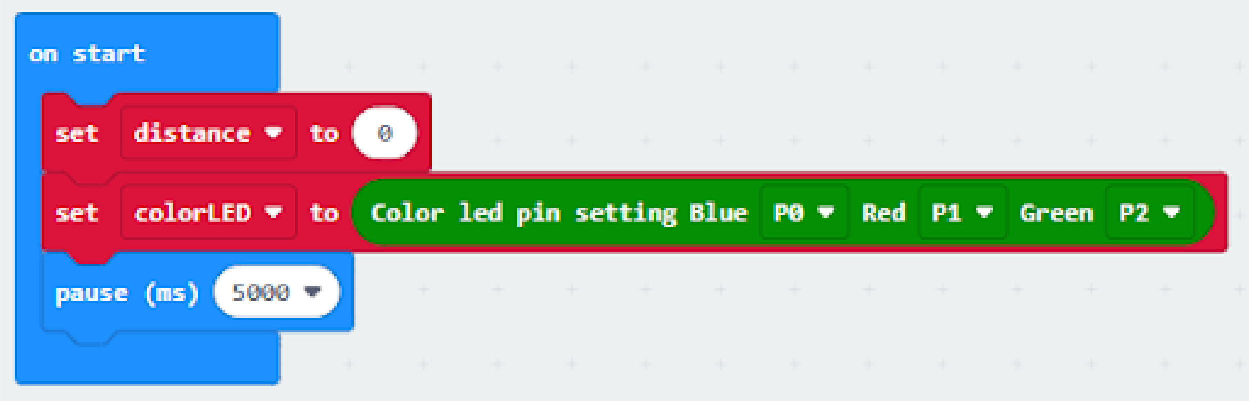
Hardware connect



Programming (MakeCode)

Step 1. Set variable and initialize multi-colour LED

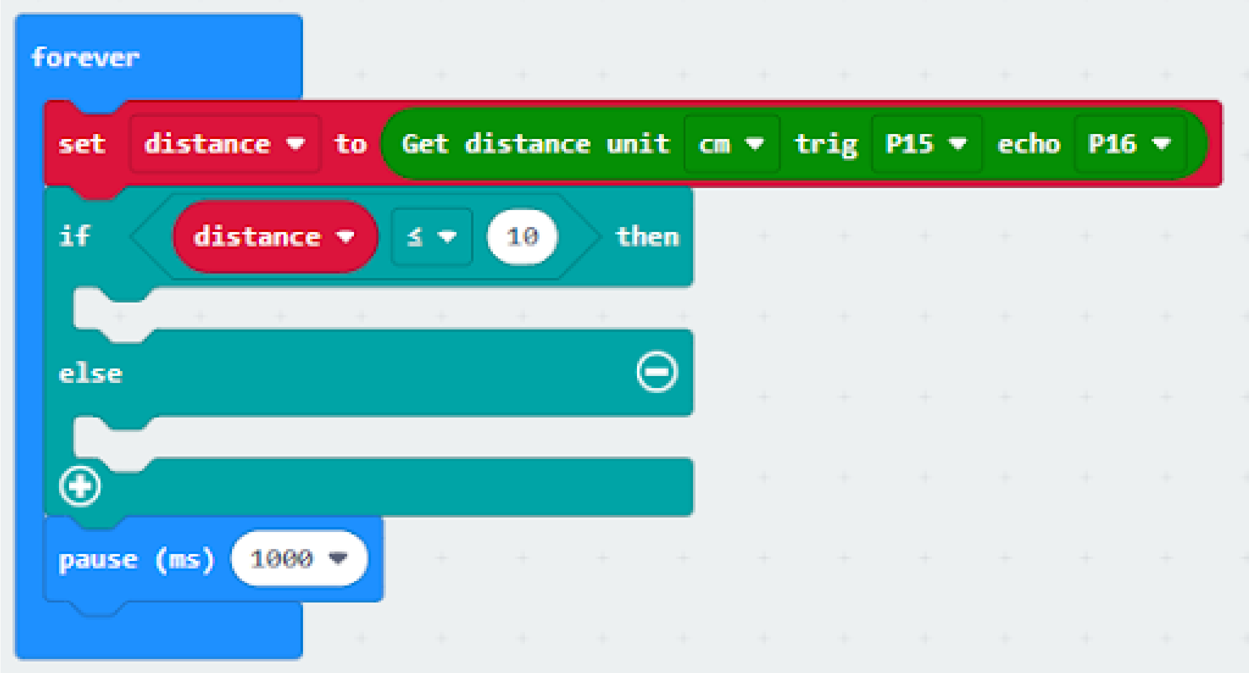
- Inside on start, snap set variable distance to 0 from variables
- Snap set colorLED to color pin setting. . . .
- Snap pause to wait 5 seconds



```
on start
    set distance ▾ to 0
    set colorLED ▾ to Color led pin setting Blue P0 ▾ Red P1 ▾ Green P2 ▾
    pause (ms) 5000 ▾
```

Step 2. Get distance value

- Inside block forever. Set distance to get distance unit cm trig P15 echo P16, that's say get the distance value by connecting the distance sensor to P15 and P16
- Snap if statement into forever, set distance 10 into if statement
- Snap Pause to the loop to wait 1 second for next checking



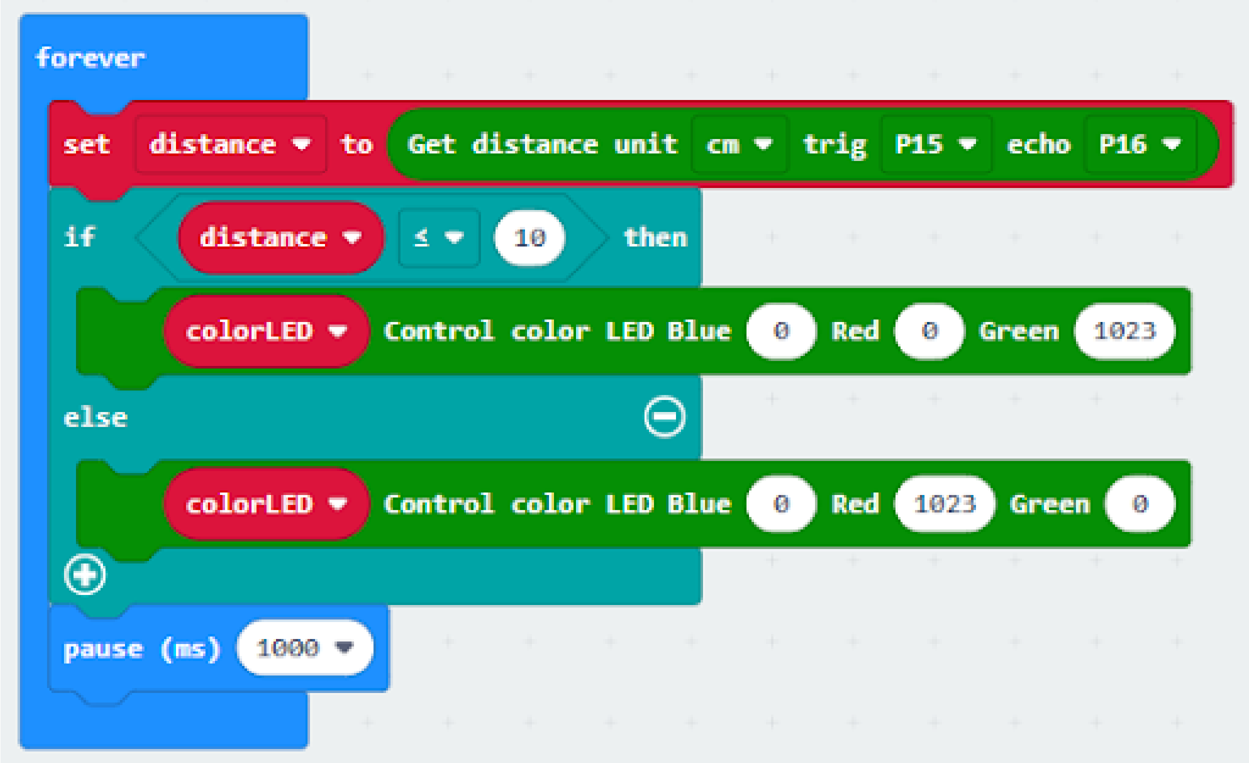
```
forever
    set distance ▾ to Get distance unit cm ▾ trig P15 ▾ echo P16 ▾
    if distance ▾ ≤ ▾ 10 then

    else ⊖

    ⊕
    pause (ms) 1000 ▾
```

Step 3. Show indicating colours with distance value

- If distance 10, then strip show color green, else strip show color red

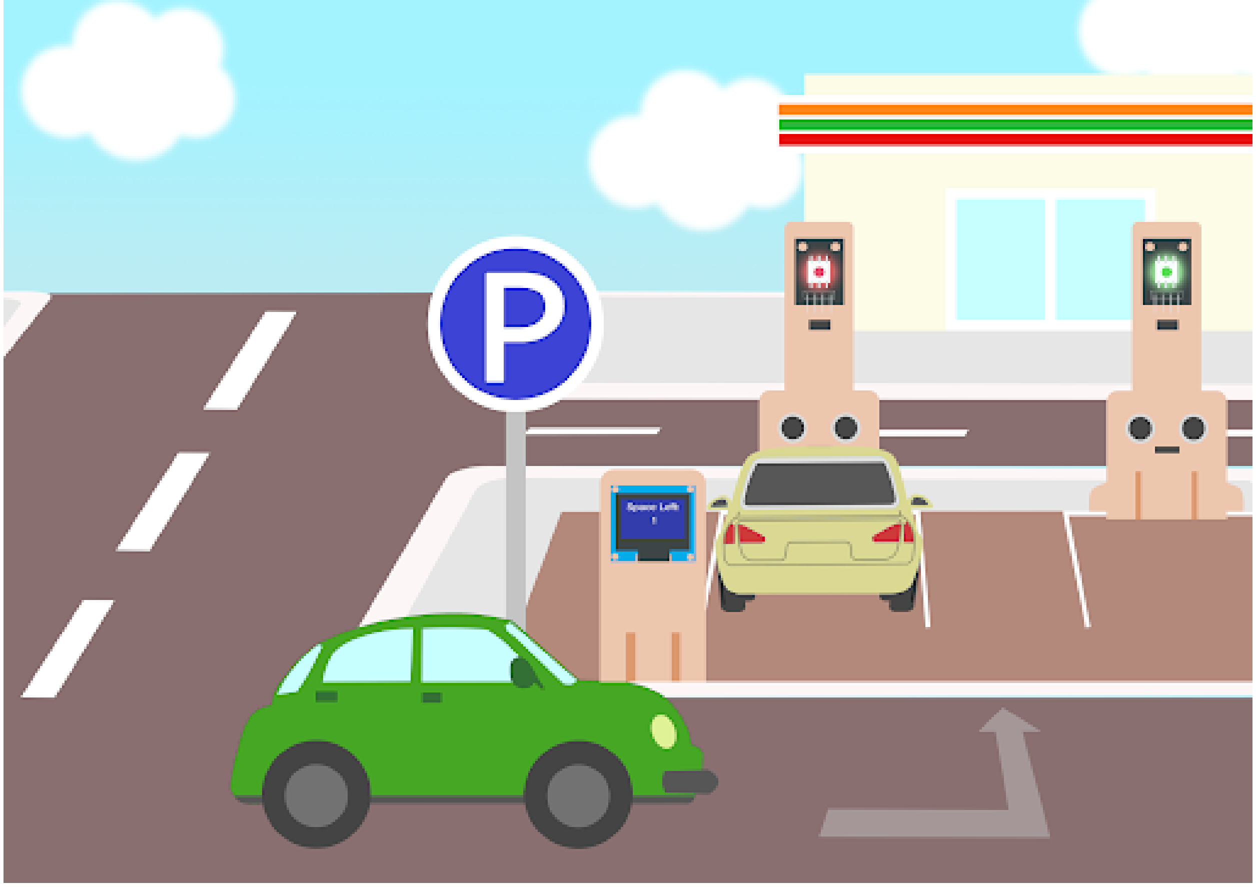


```
forever
  set distance ▾ to Get distance unit cm ▾ trig P15 ▾ echo P16 ▾
  if distance ▾ ≤ ▾ 10 then
    colorLED ▾ Control color LED Blue 0 Red 0 Green 1023
  else ⊖
    colorLED ▾ Control color LED Blue 0 Red 1023 Green 0
  ⊕
  pause (ms) 1000 ▾
```

Result

Think

1.2.6 Smart Car Park Access Barrier 1: Car Park Monitoring System

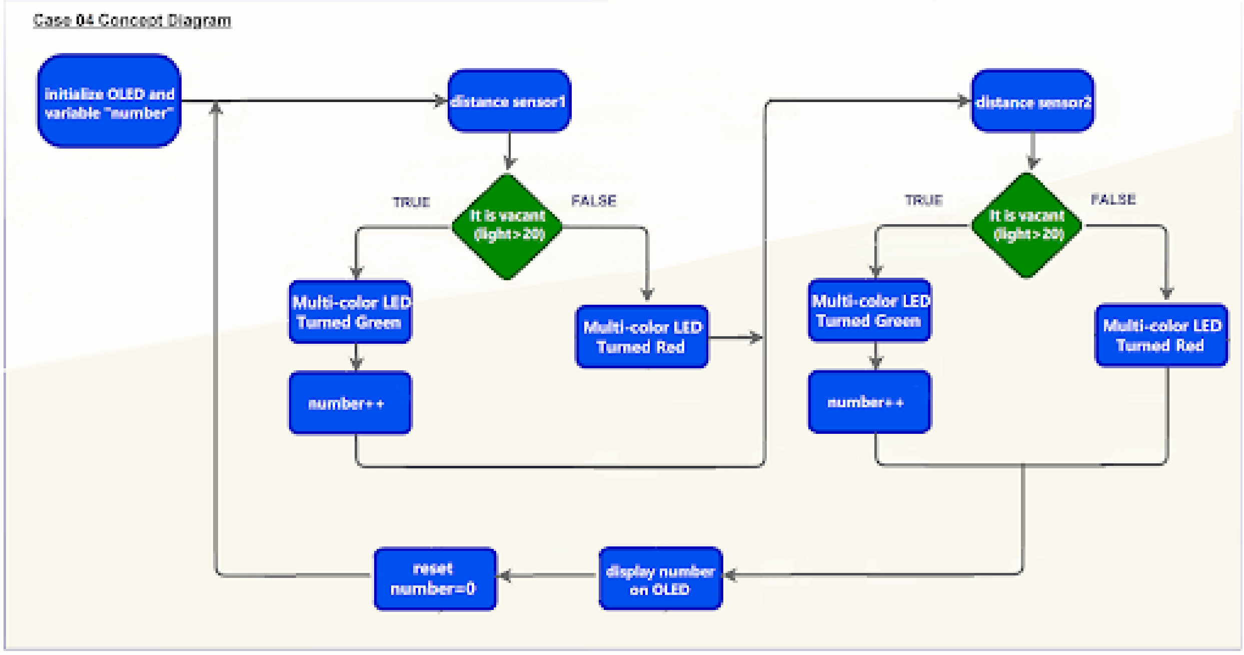


Goal

Background

What is an Smart car park monitoring systems?

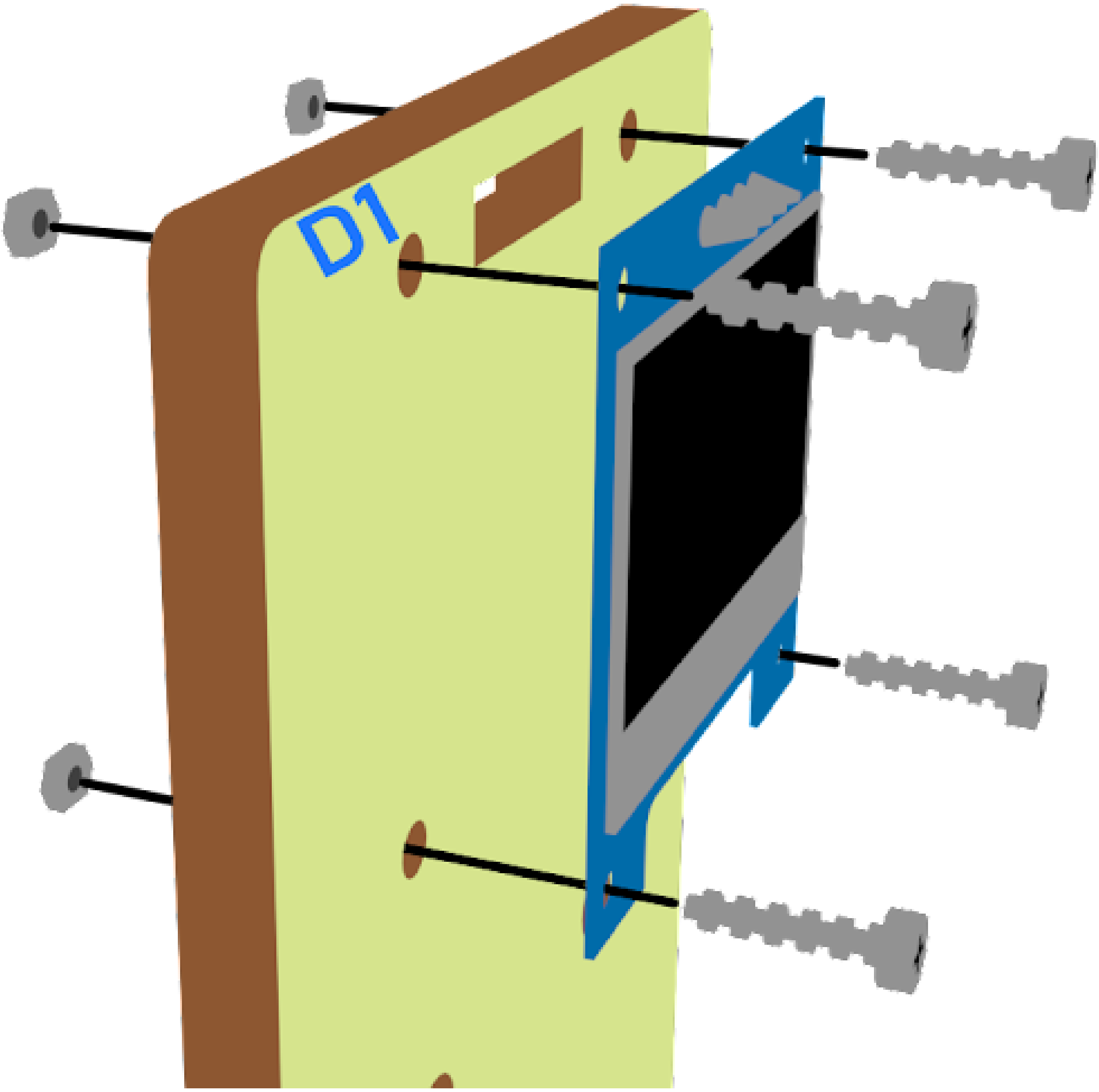
Smart car park monitoring systems operation



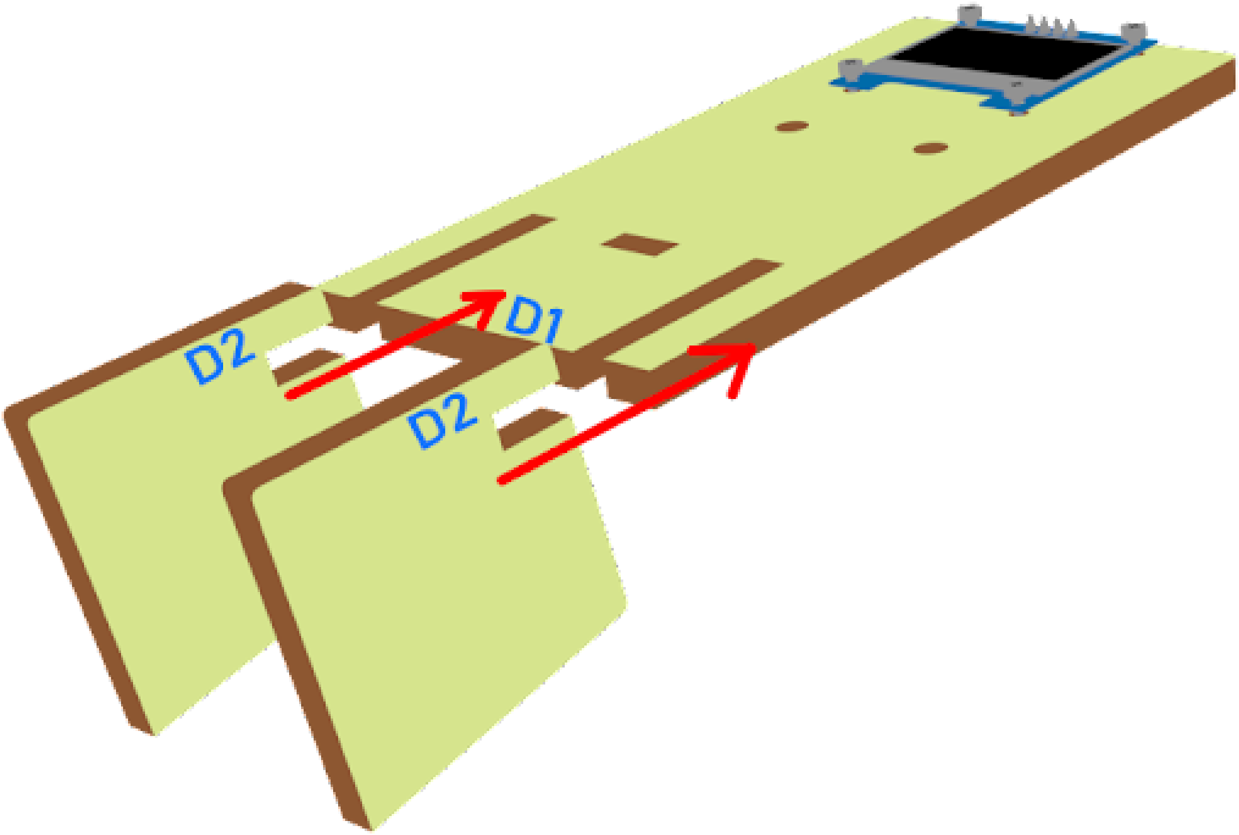
Part List

Assembly step

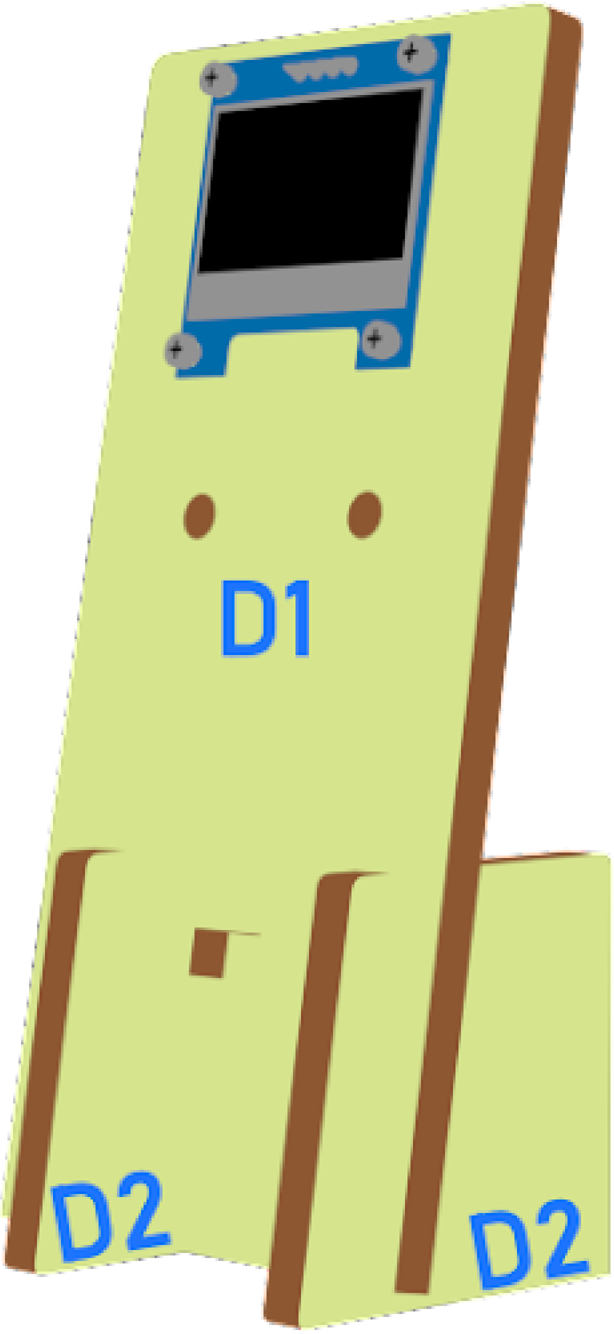
Step 1



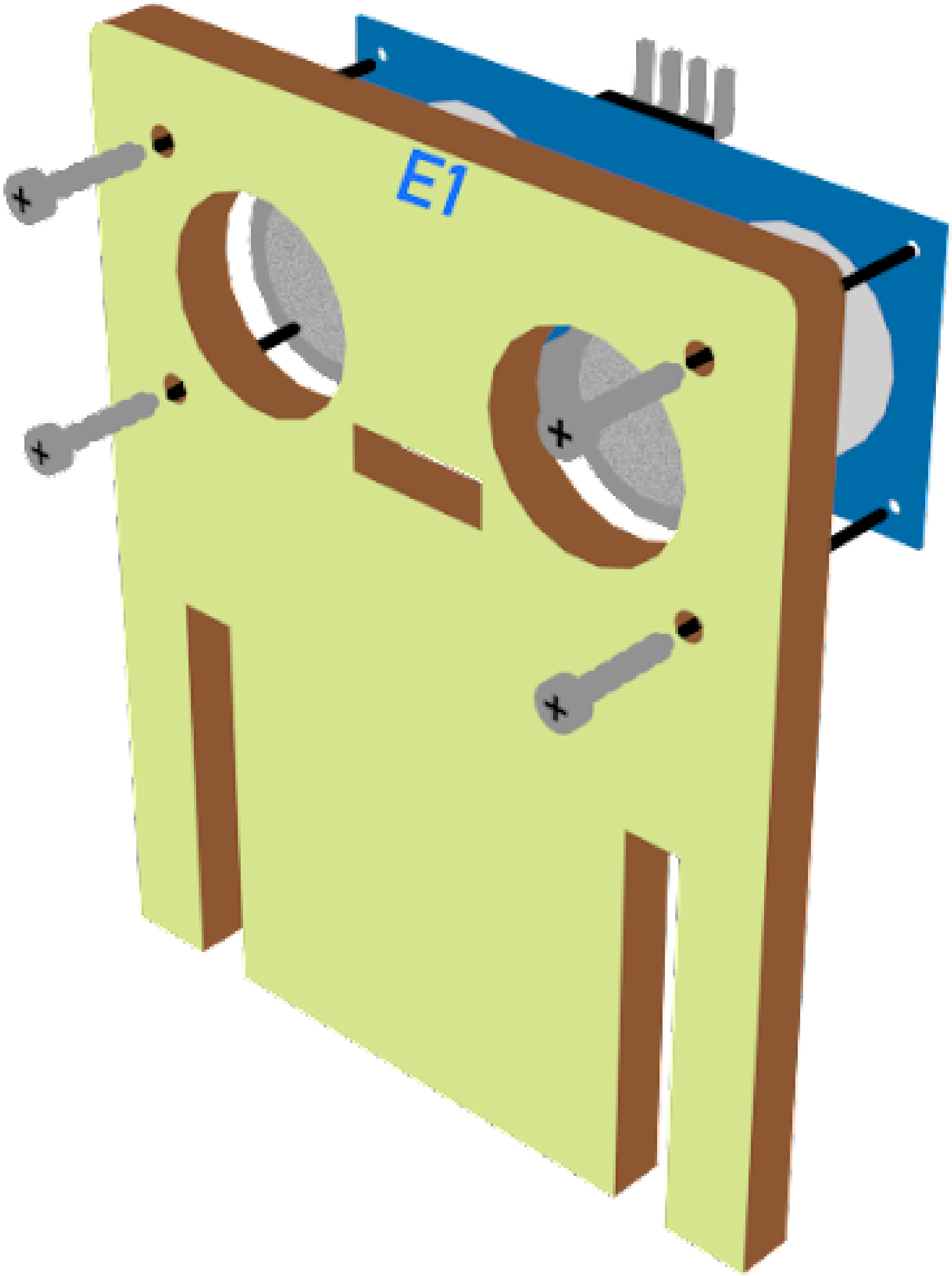
Step 2



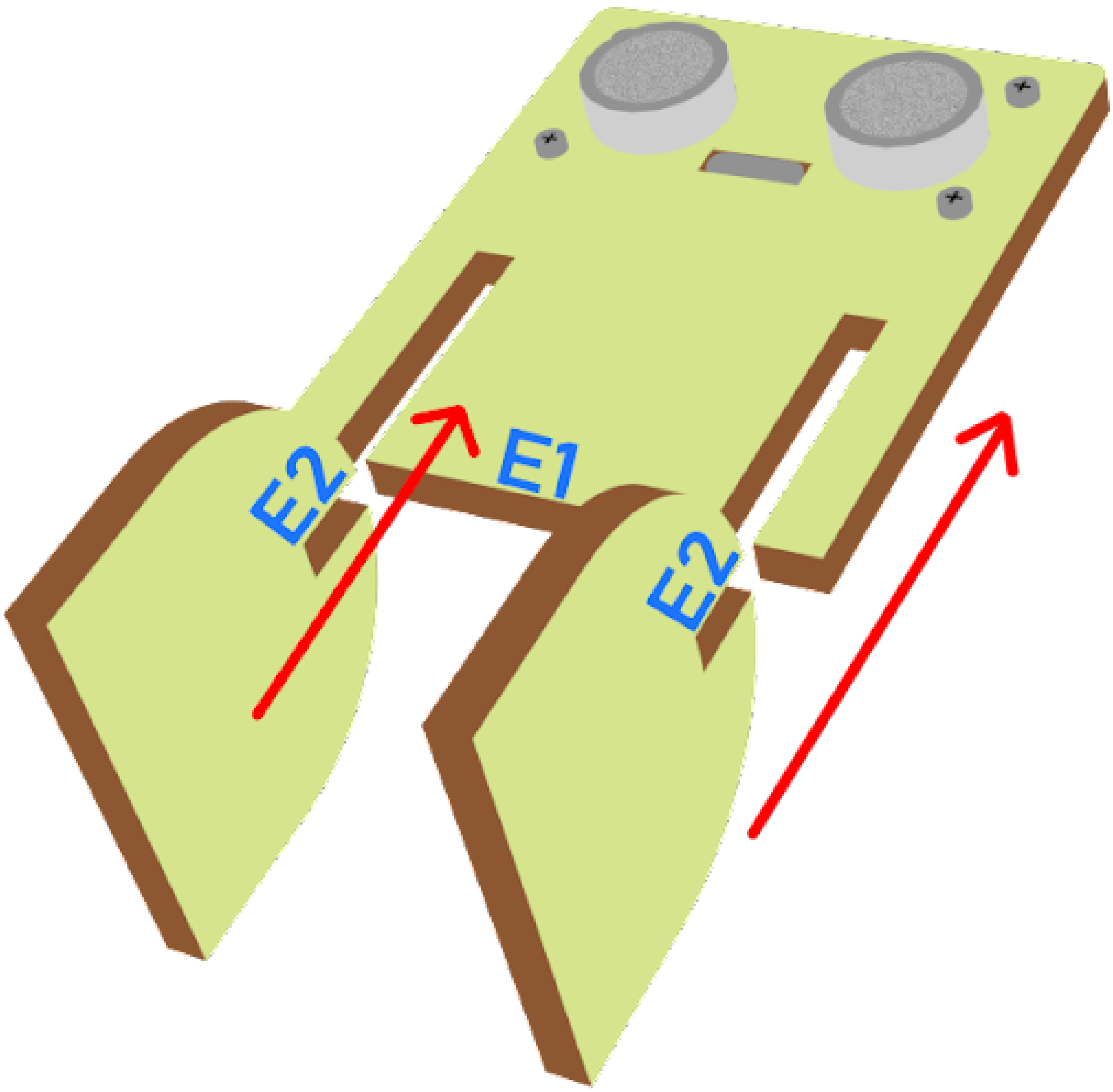
Step 3



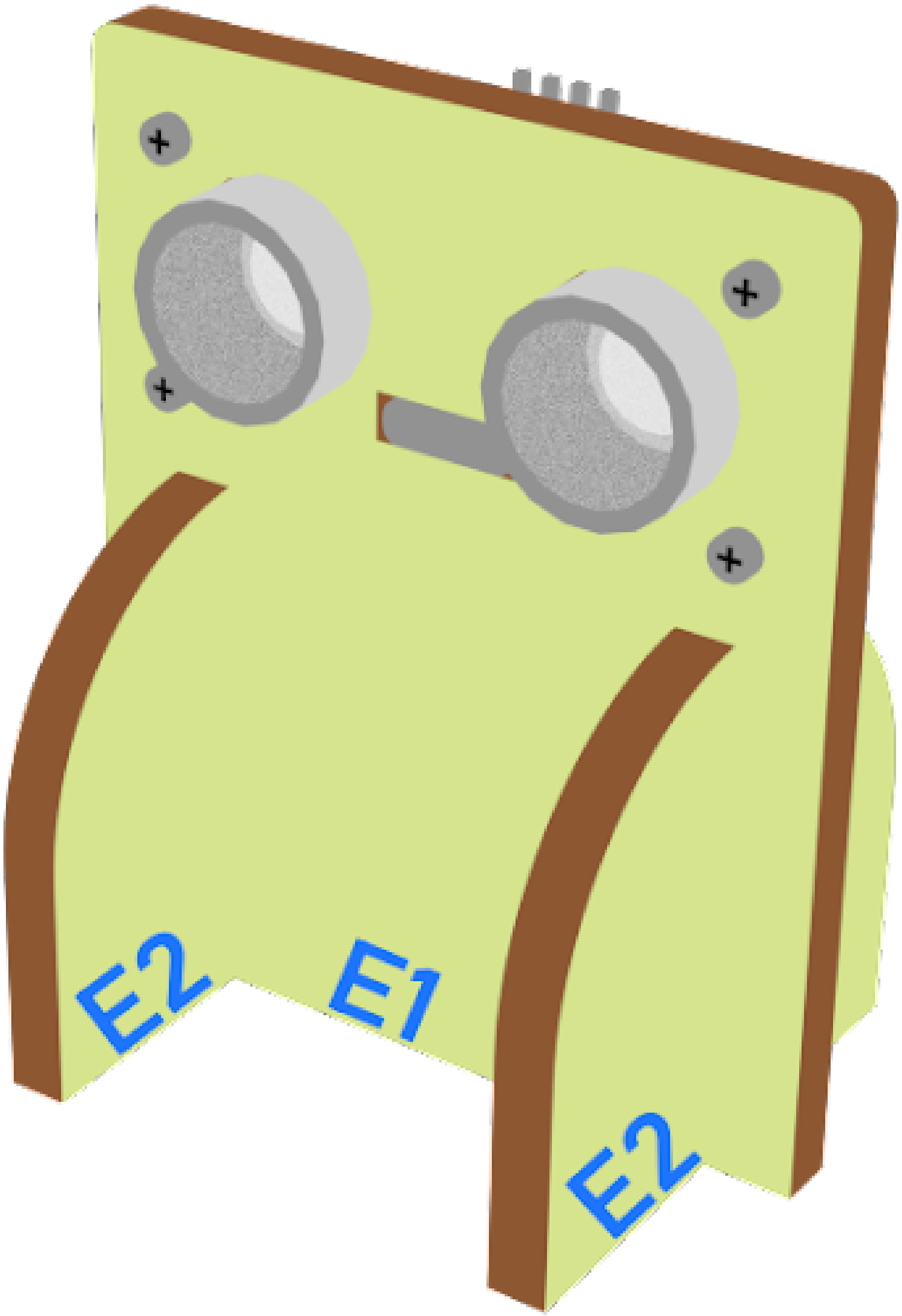
Step 4



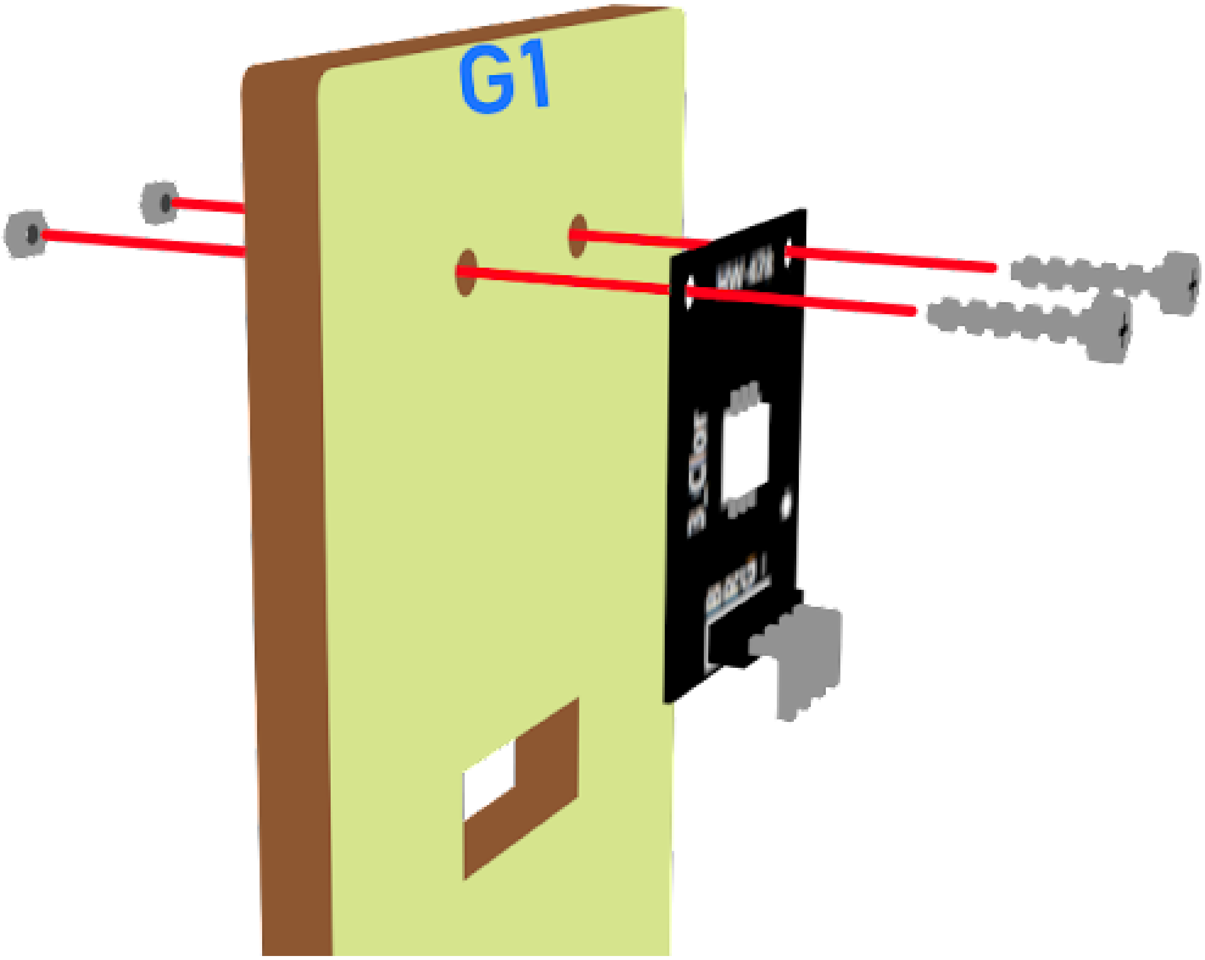
Step 5



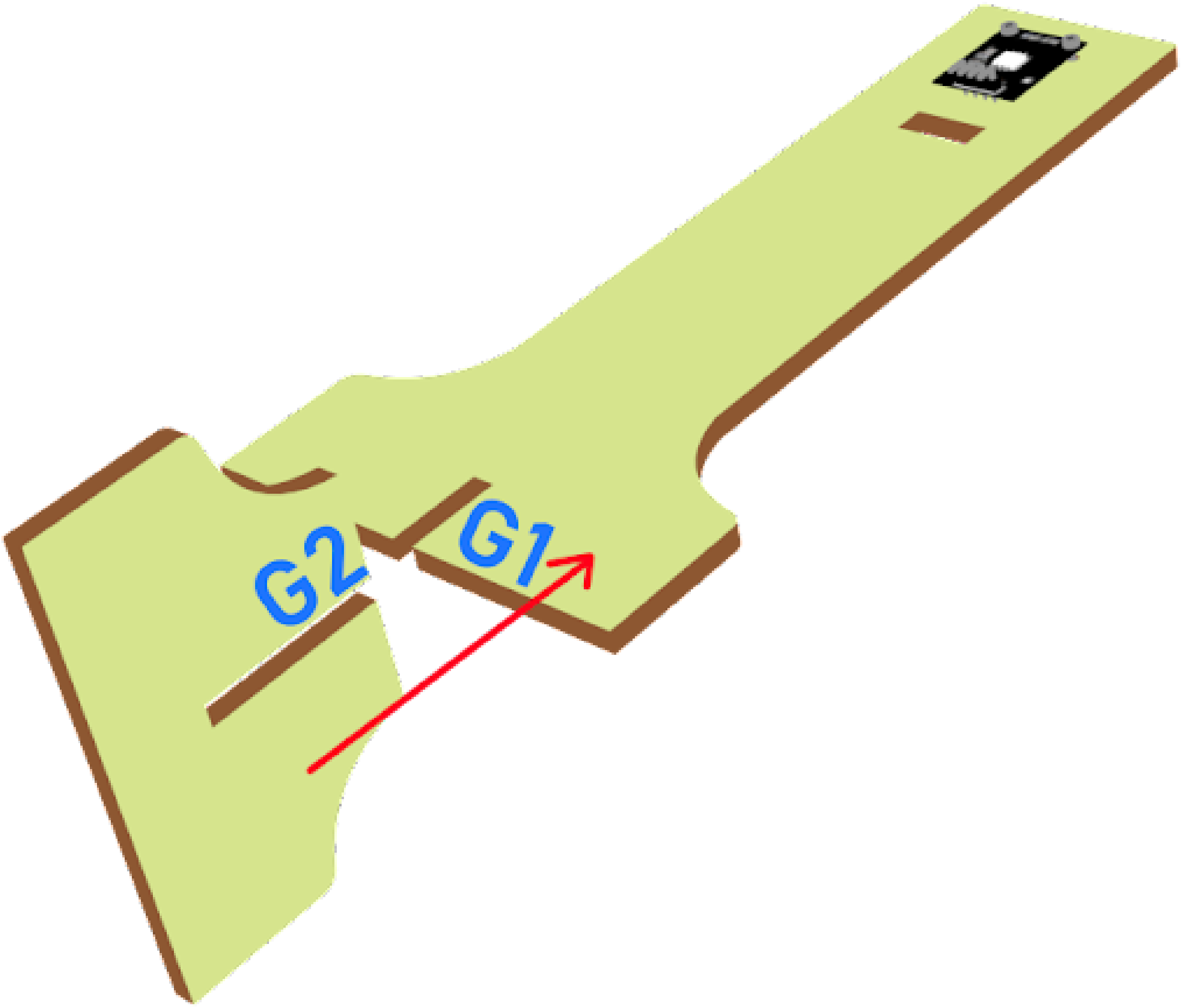
Step 6



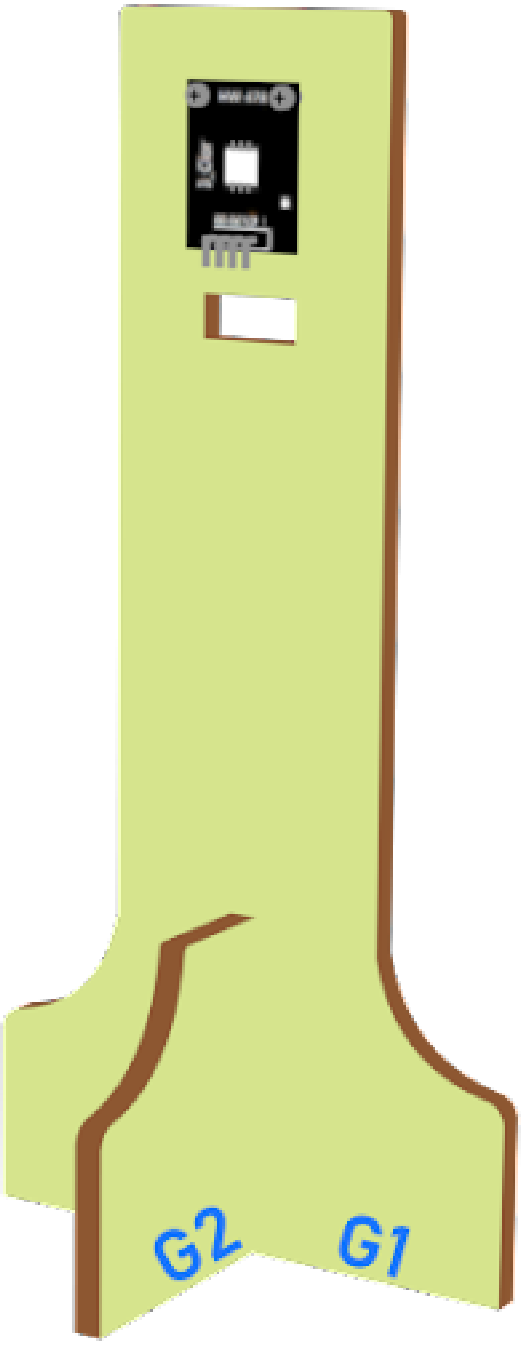
Step 7



Step 8



Step 9

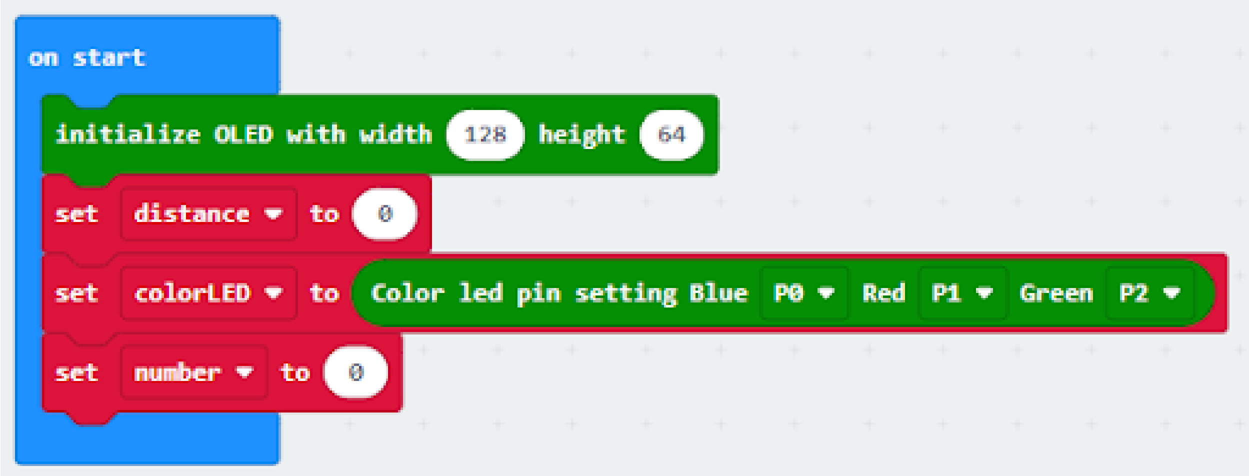


Hardware connect

Programming (MakeCode)

Step 1. Set variables and initialize multi-colour LED and OLED screen

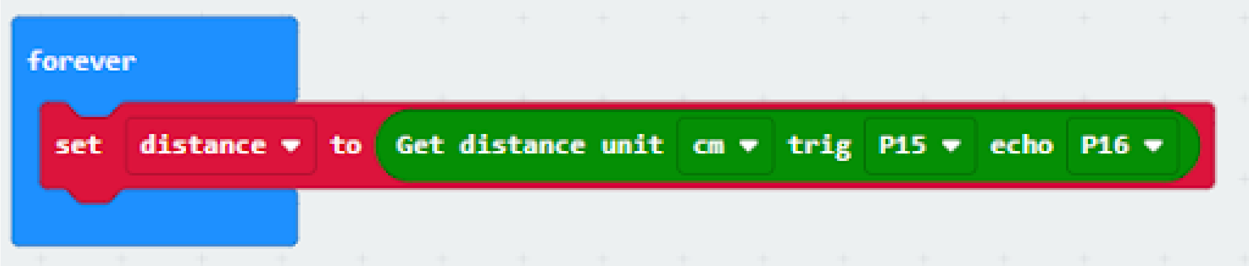
- Drag Initialize OLED with width:128, height: 64 to on start
- Inside on start, snap set variable distance to 0 and set number to 0 from variables.
- Snap set colorLED to color led pin setting … set colorLED to color led pin setting …



```
on start
    initialize OLED with width 128 height 64
    set distance ▾ to 0
    set colorLED ▾ to Color led pin setting Blue P0 ▾ Red P1 ▾ Green P2 ▾
    set number ▾ to 0
```

Step 2. Get distance

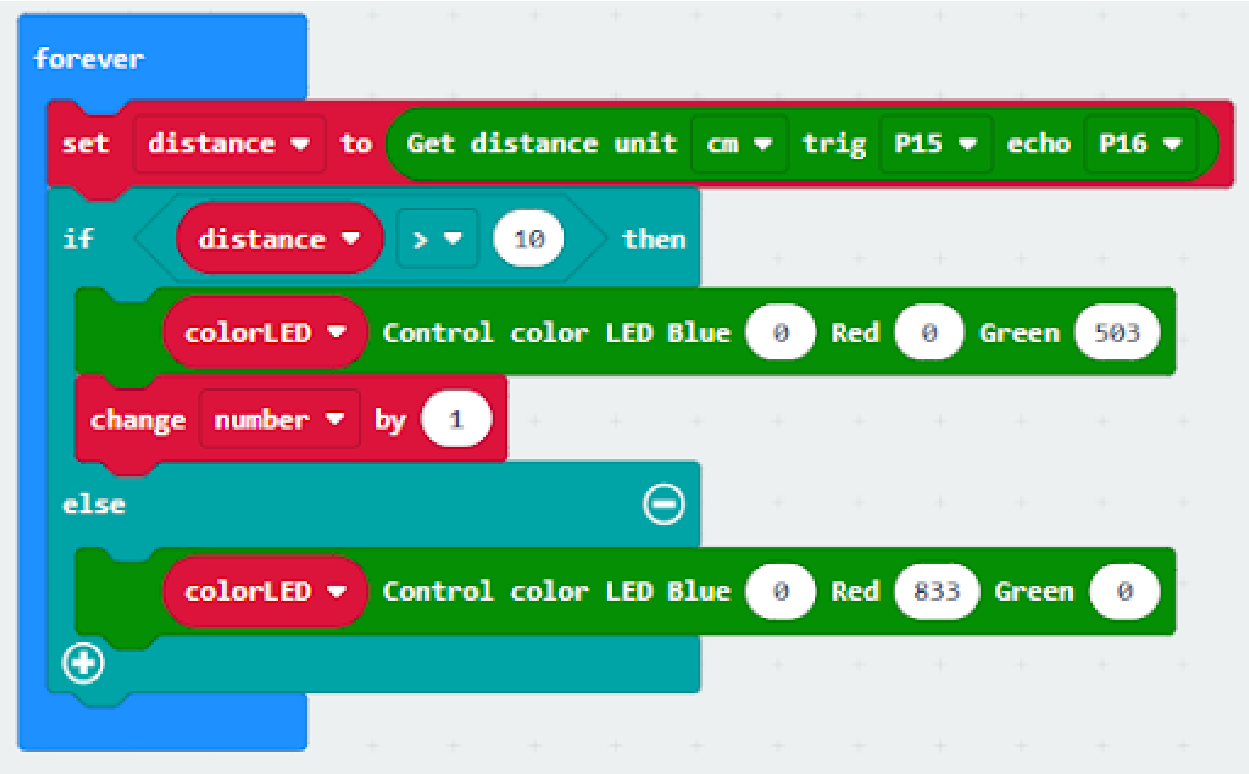
- Drag get distance to distance unit cm trig P15 echo P16, store the value to variable distance.



```
forever
    set distance ▾ to Get distance unit cm ▾ trig P15 ▾ echo P16 ▾
```

Step 3. Show indicating colours and count the number of vacancies

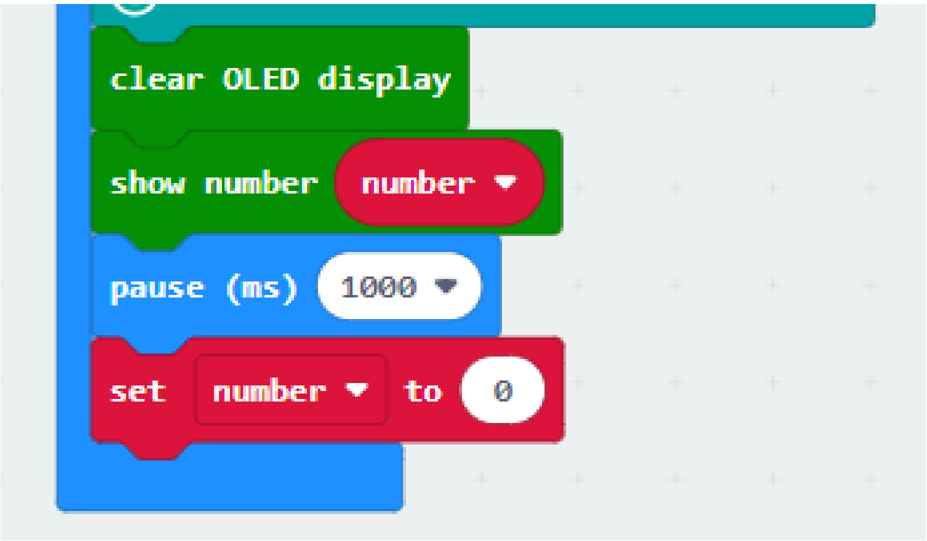
- Snap if statement into forever, set variable distance > 10
- If distance >10, then colorLED shows color green, else colorLED shows color red
- Snap change number by 1 if distance>10



```
forever
    set distance ▾ to Get distance unit cm ▾ trig P15 ▾ echo P16 ▾
    if distance ▾ > ▾ 10 then
        colorLED ▾ Control color LED Blue 0 Red 0 Green 503
        change number ▾ by 1
    else
        colorLED ▾ Control color LED Blue 0 Red 833 Green 0
```

Step 4 display on OLED

- Snap clear OLED display from OLED to avoid overlap
- Snap show number and show value of variables number
- Snap Pause to the loop to wait 1 second for next checking
- Reset number to 0 before next checking

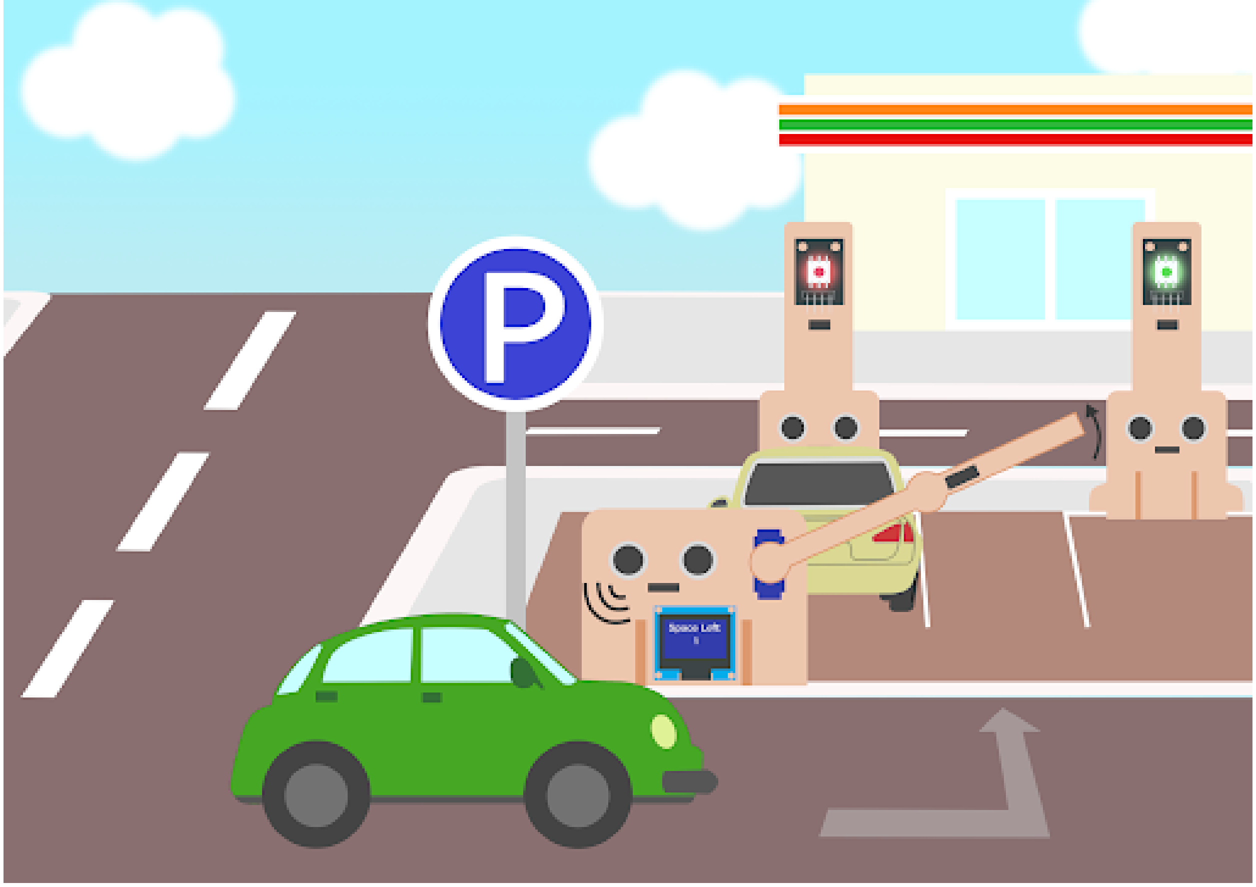


```
clear OLED display
show number number ▾
pause (ms) 1000 ▾
set number ▾ to 0
```

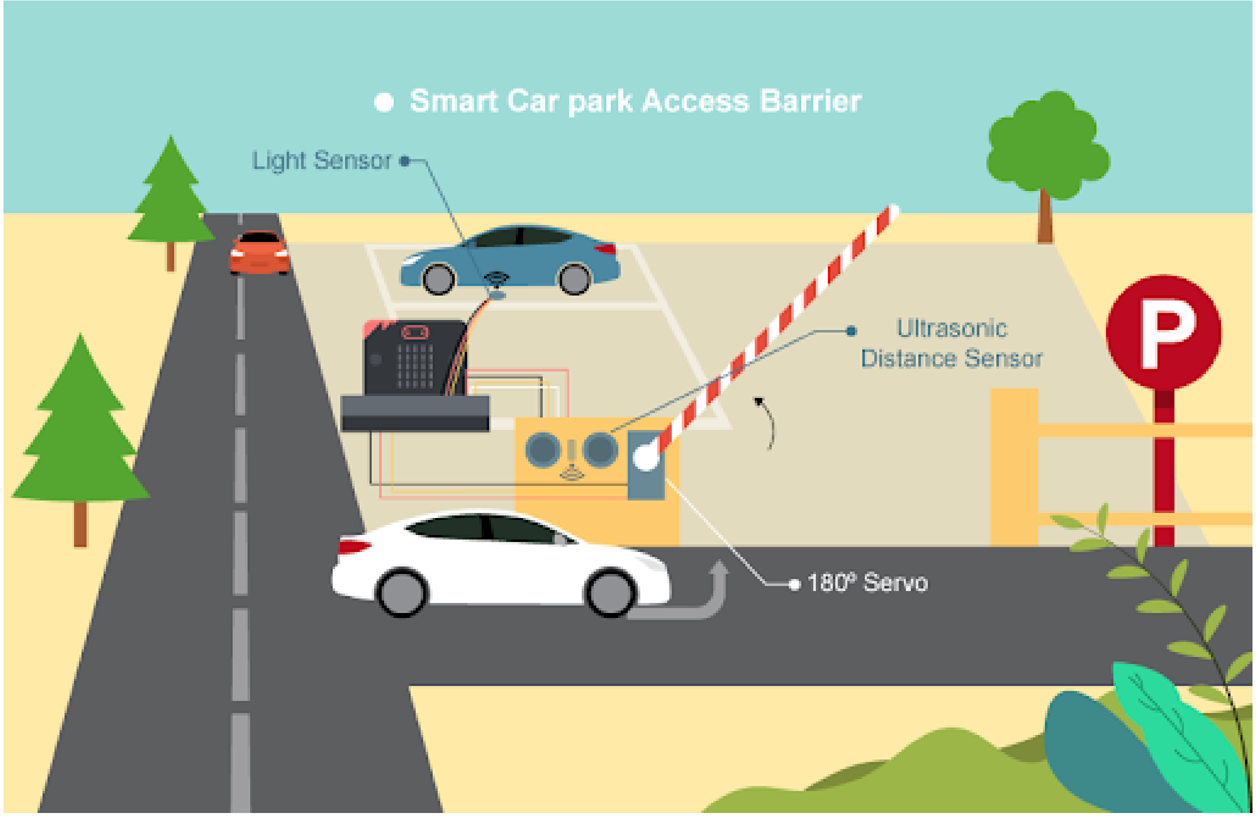
Result

Think

1.2.7 Smart Car Park Access Barrier 2: Car Park Access Barrier



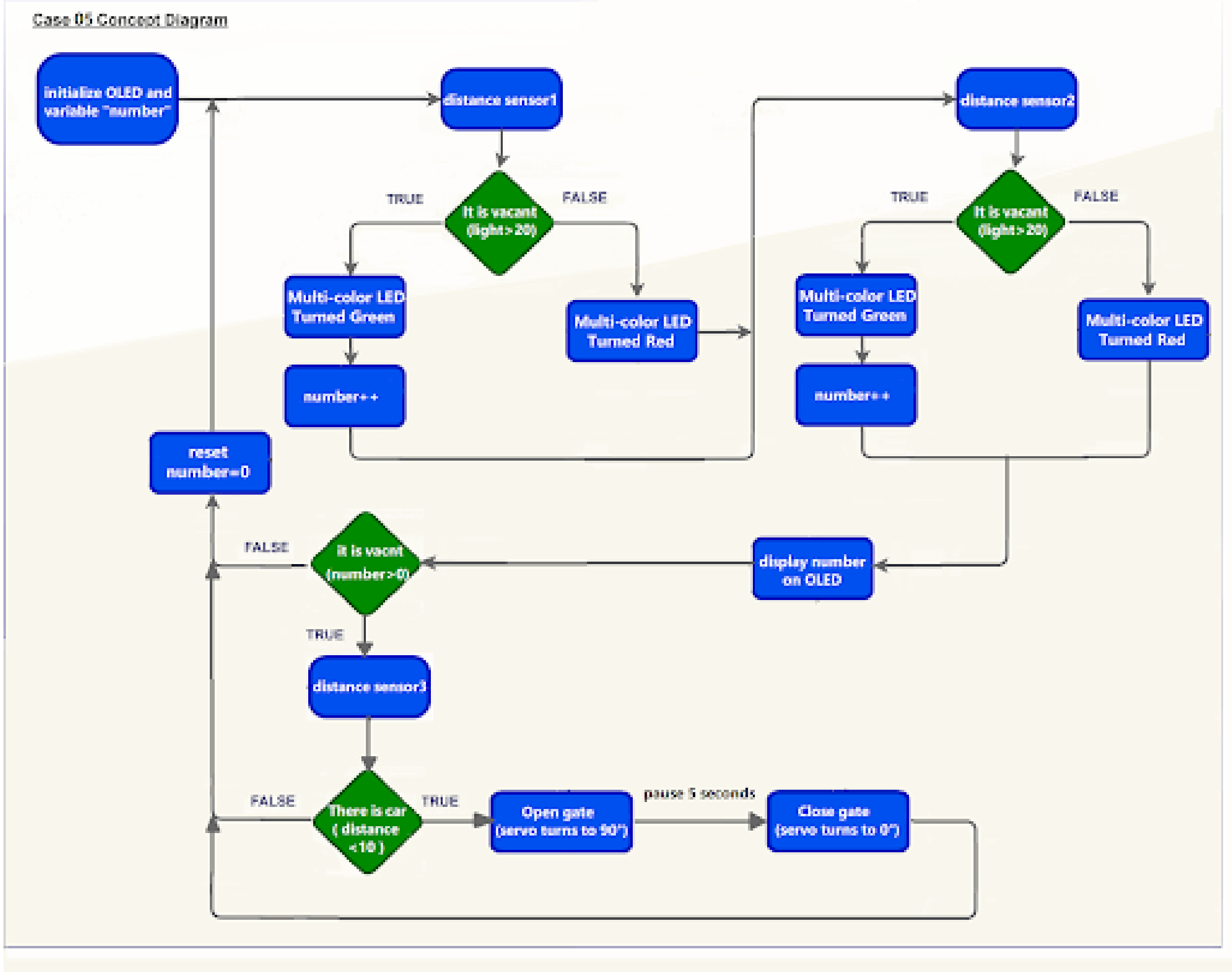
Goal



Background

What is a smart car park access barrier?

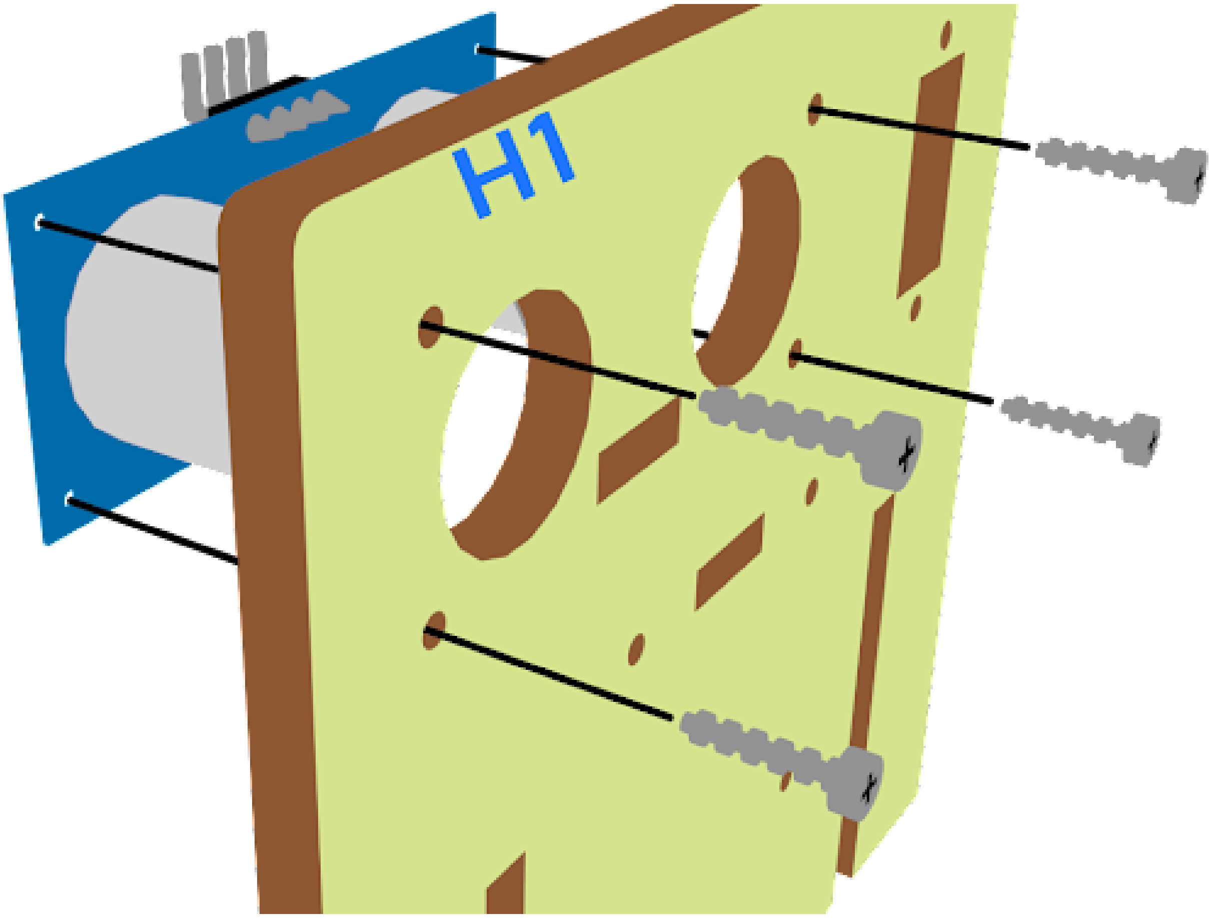
Car park access barrier operation



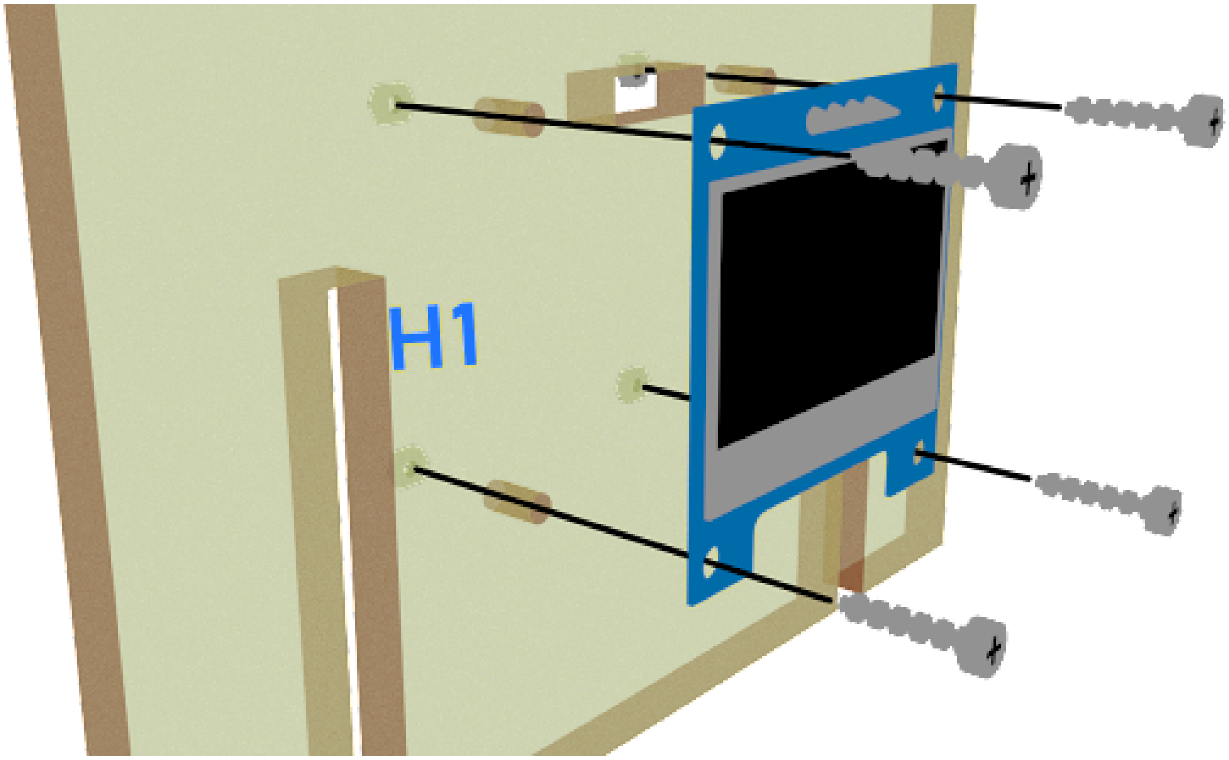
Part List

Assembly step

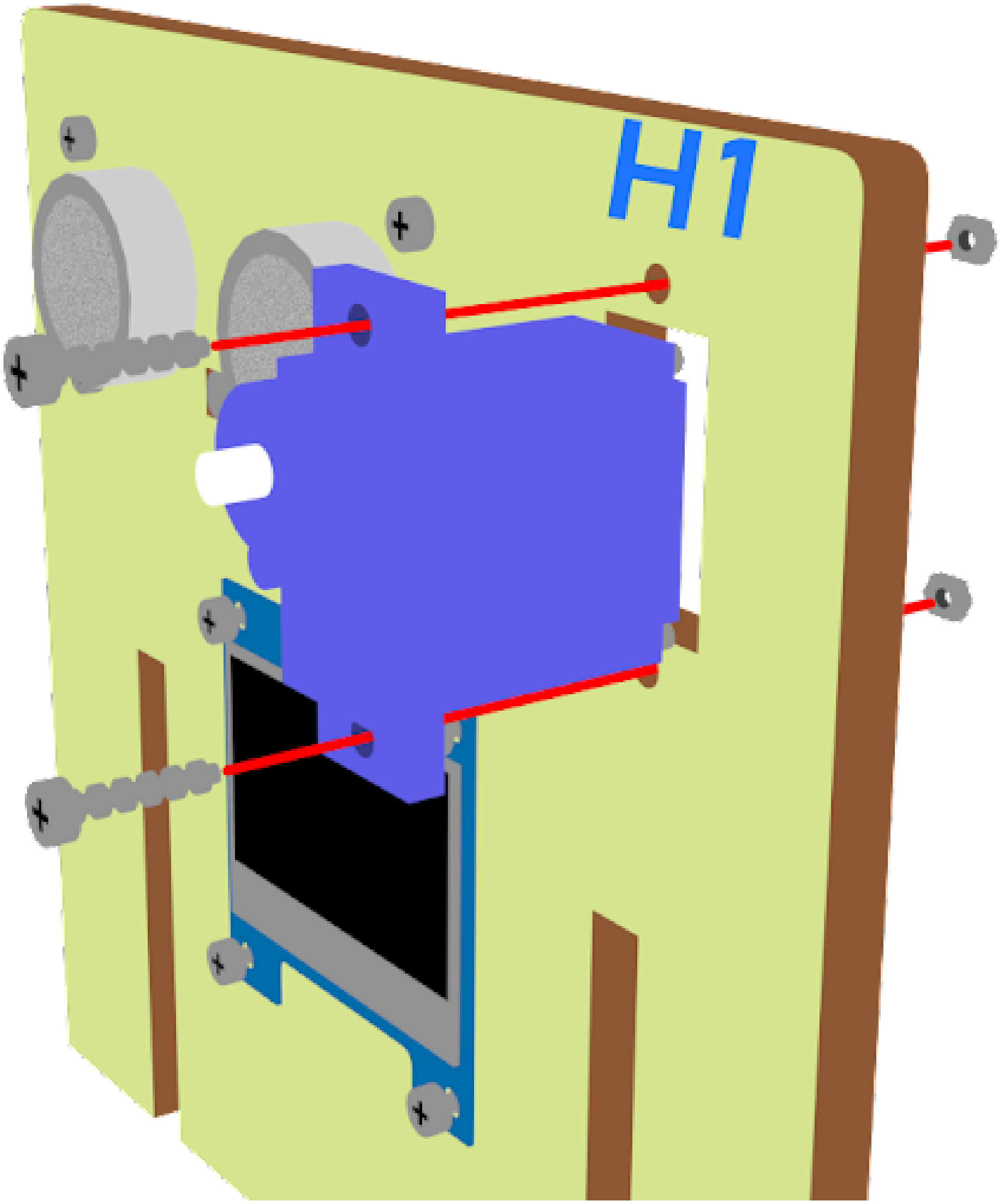
Step 1



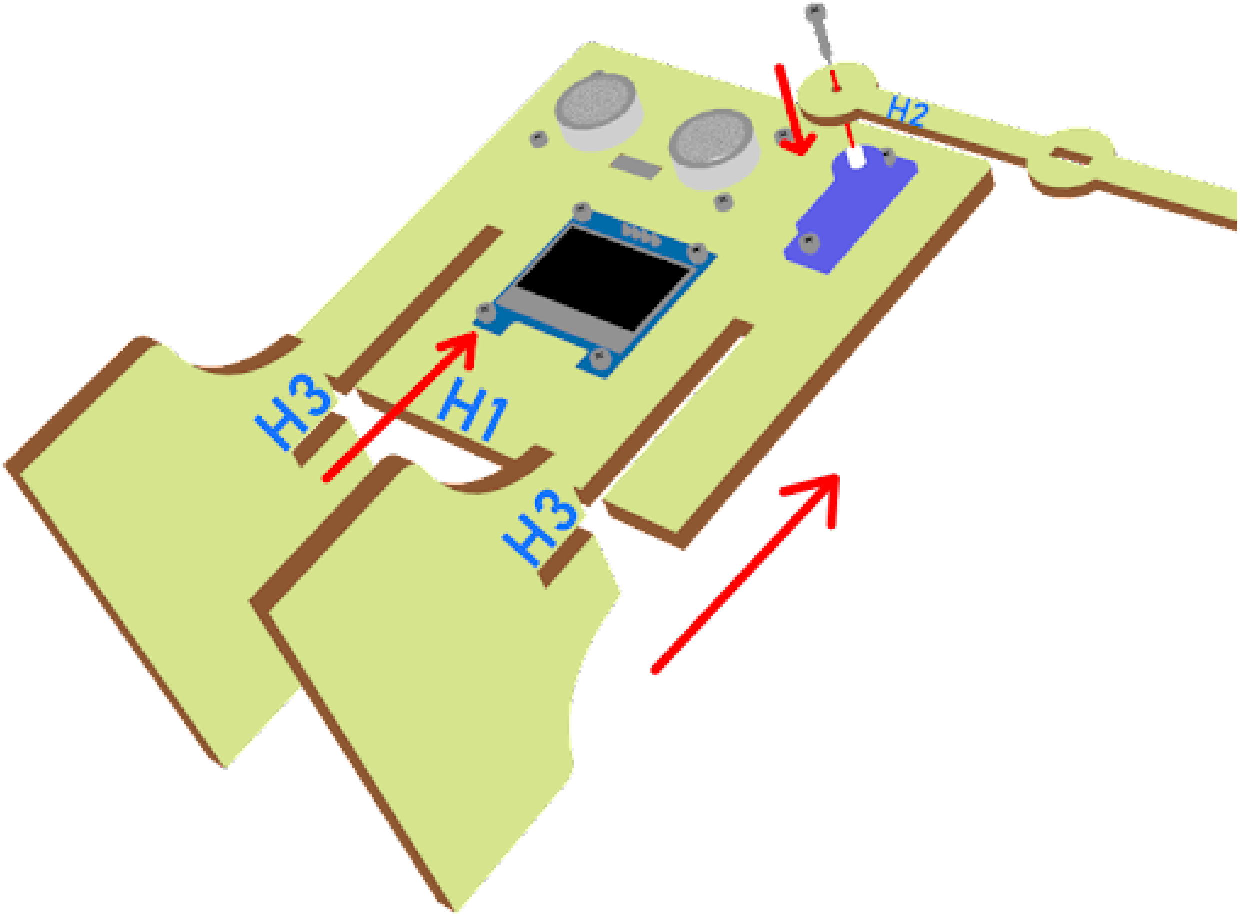
Step 2



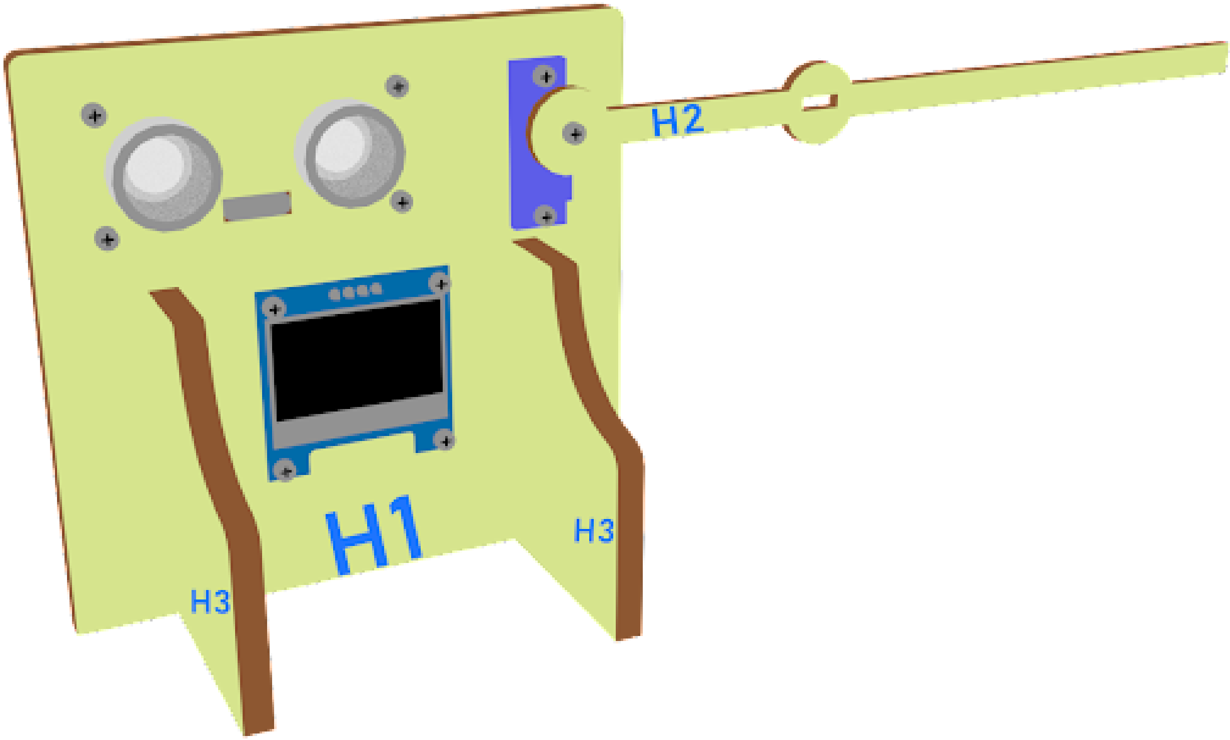
Step 3



Step 4



Step 5

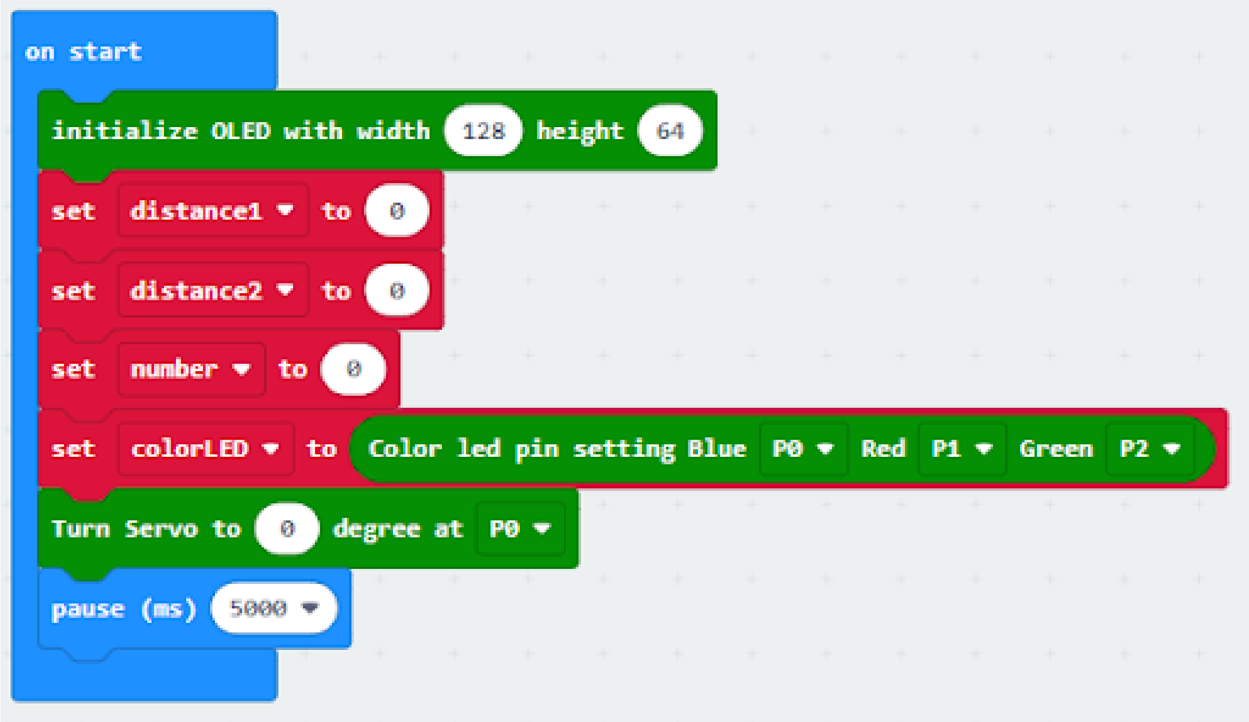


Hardware connect

Programming (MakeCode)

Step 1. Set variables, initialize OLED screen and servo at start position

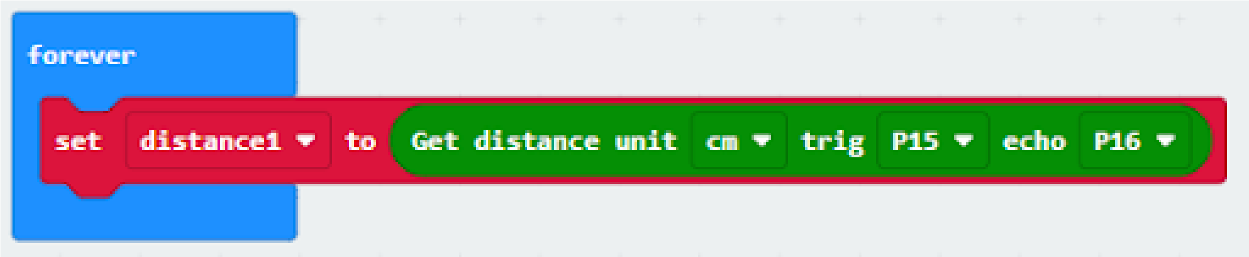
- Drag Initialize OLED with width:128, height: 64 to on start
- Inside on start, snap set variable distance1 to 0, set variable distance2 to 0 and set number to 0 from variables.
- Snap set colorLED to color led pin setting
- Snap Turn Servo to 0 degree at P0.
- Snap pause to wait 5 seconds



```
on start
    initialize OLED with width 128 height 64
    set distance1 ▾ to 0
    set distance2 ▾ to 0
    set number ▾ to 0
    set colorLED ▾ to Color led pin setting Blue P0 ▾ Red P1 ▾ Green P2 ▾
    Turn Servo to 0 degree at P0 ▾
    pause (ms) 5000 ▾
```

Step 2. Get distance

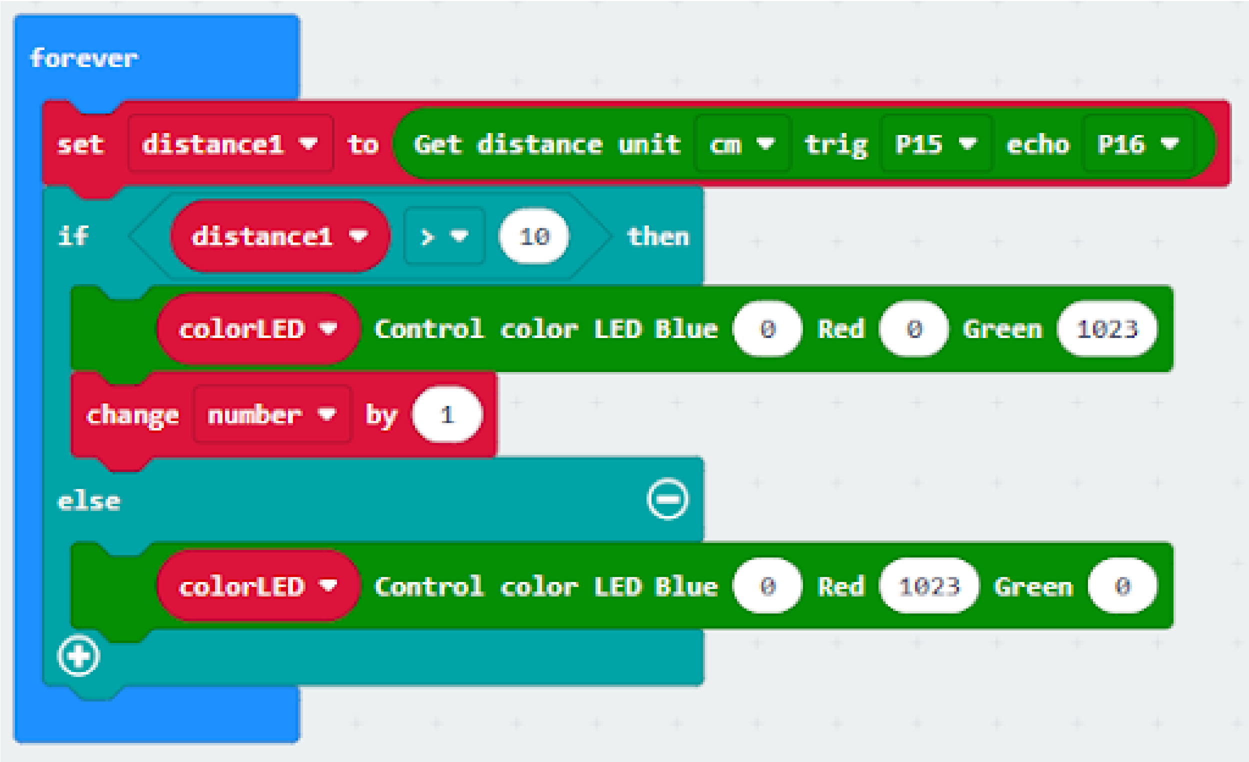
- Drag set distance1 to distance unit cm trig P15 echo P16, store the value to variable distance1.



```
forever
    set distance1 ▾ to Get distance unit cm ▾ trig P15 ▾ echo P16 ▾
```

Step 3. Show indicating colours and count the number of vacancies

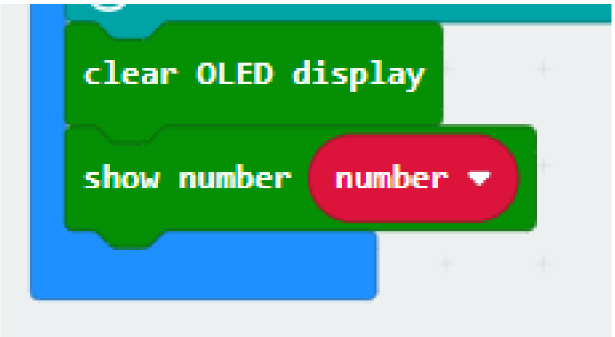
- Snap if statement into forever, set variable distance1 > 10
- If distance1 >10, then colorLED shows color green, else colorLED shows color red
- Snap change number by 1 if distance1>10



```
forever
    set distance1 ▾ to Get distance unit cm ▾ trig P15 ▾ echo P16 ▾
    if distance1 ▾ > ▾ 10 then
        colorLED ▾ Control color LED Blue 0 Red 0 Green 1023
        change number ▾ by 1
    else ⊖
        colorLED ▾ Control color LED Blue 0 Red 1023 Green 0
    ⊕
```

Step 4 display on OLED

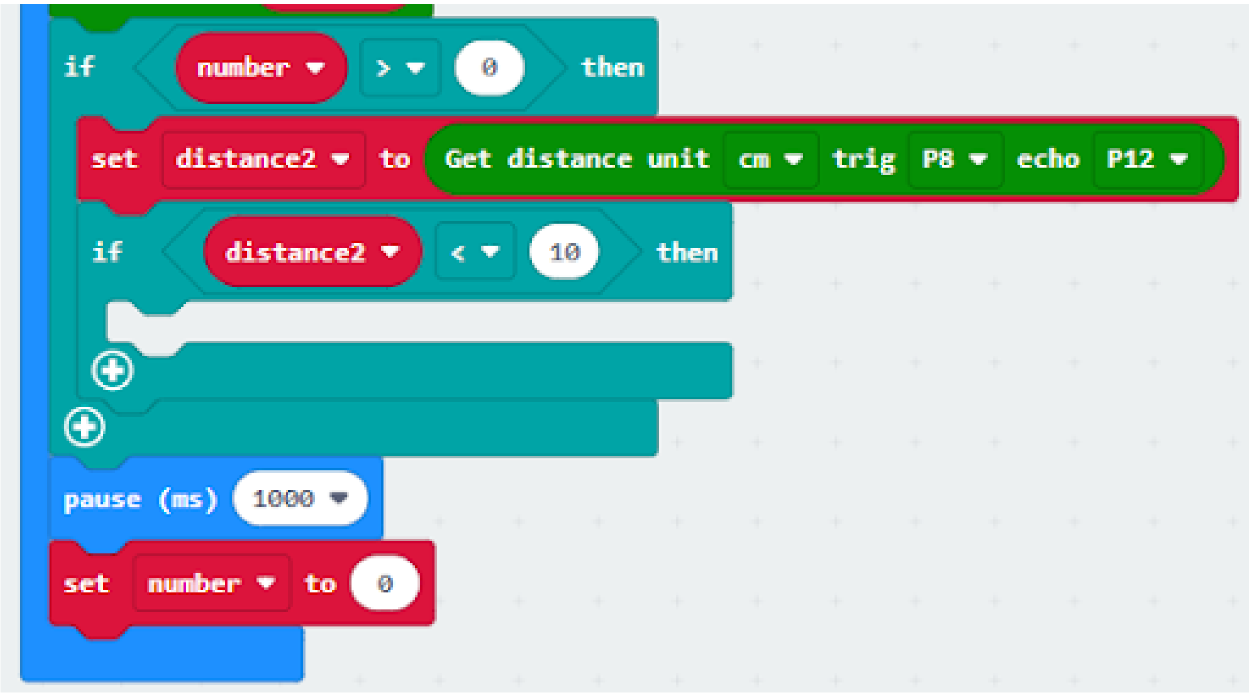
- Snap clear OLED display from OLED to avoid overlap
- Snap show number and show value of variables number



```
clear OLED display
show number number ▾
```

Step 5. Open/close gate with distance value

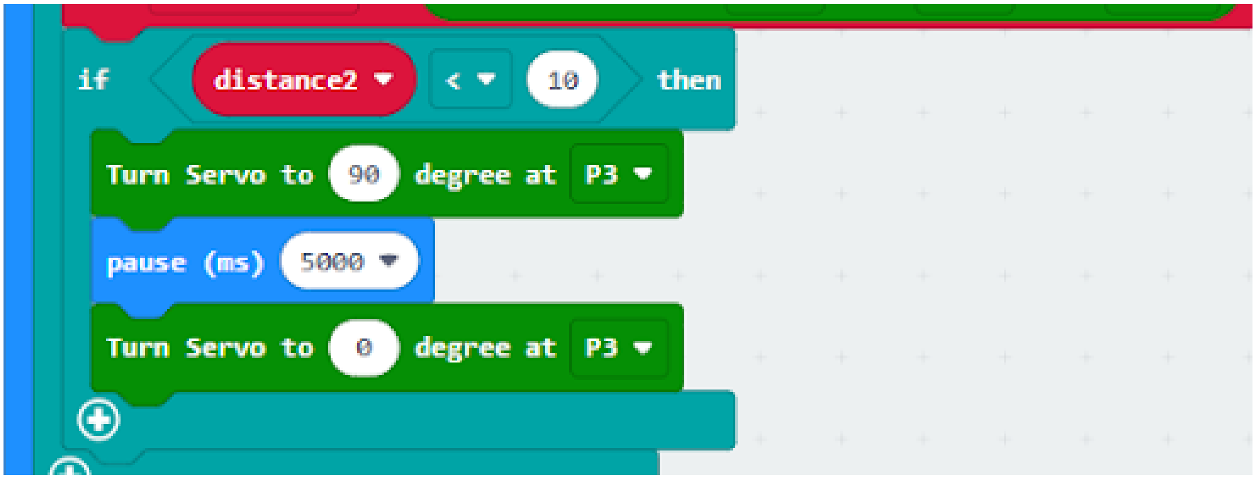
- Snap if statement into forever, set variable number>0
- Drag get distance2 to distance unit cm trig P8 echo P12, store the value to variable distance2.
- Snap if statement into forever, set variable distance2 < 10
- Snap Pause to the loop to wait 1 second for next checking
- Reset number to 0 before next checking



```
if number > 0 then
    set distance2 to Get distance unit cm trig P8 echo P12
    if distance2 < 10 then
pause (ms) 1000
set number to 0
```

Step 6. Set servo position

- Snap Turn Servo to 90 degree at P3 as the gate is opened.
- Snap pause to the loop to wait 5 seconds
- Snap Turn Servo to 0 degree at P3 as the gate is closed.

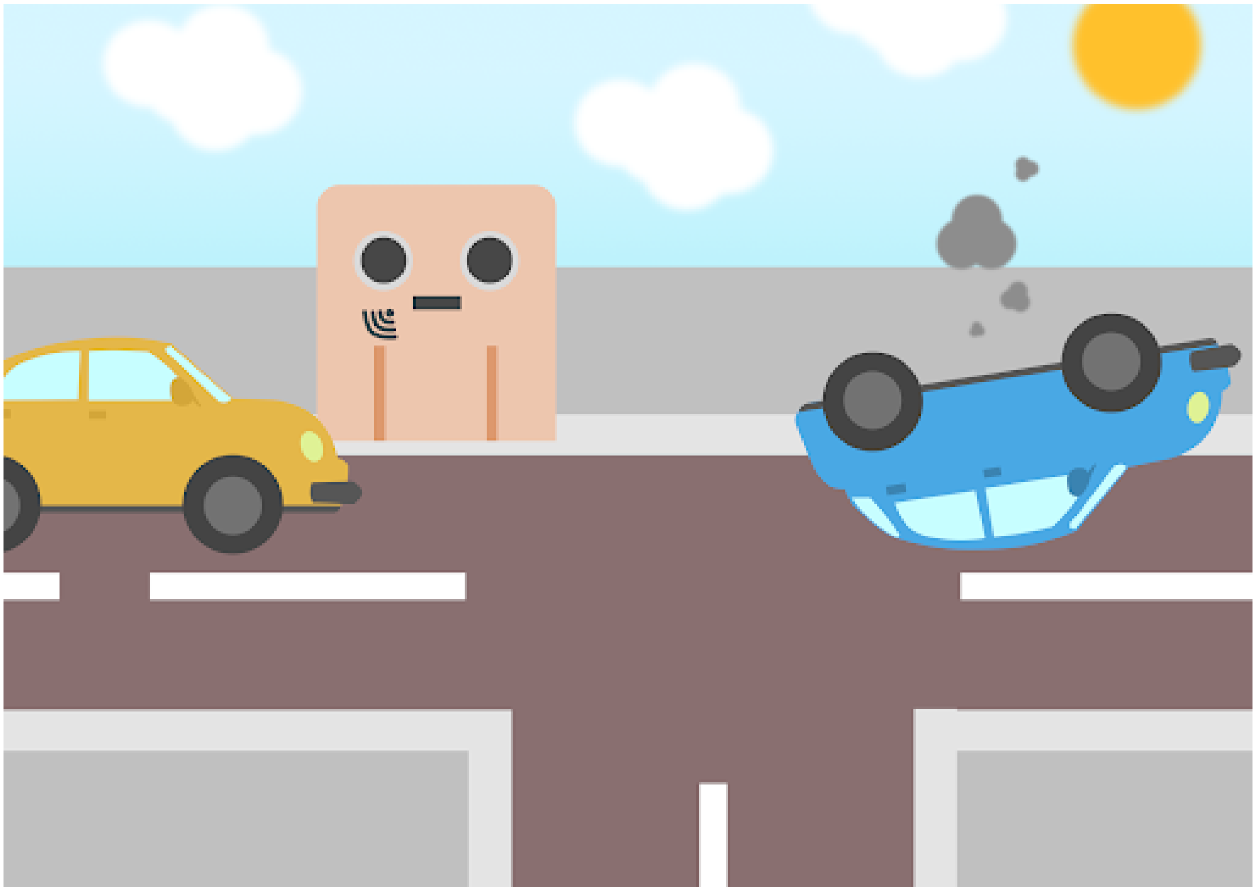


```
if distance2 < 10 then
    Turn Servo to 90 degree at P3
    pause (ms) 5000
    Turn Servo to 0 degree at P3
```

Result

Think

1.2.8 Broken Car

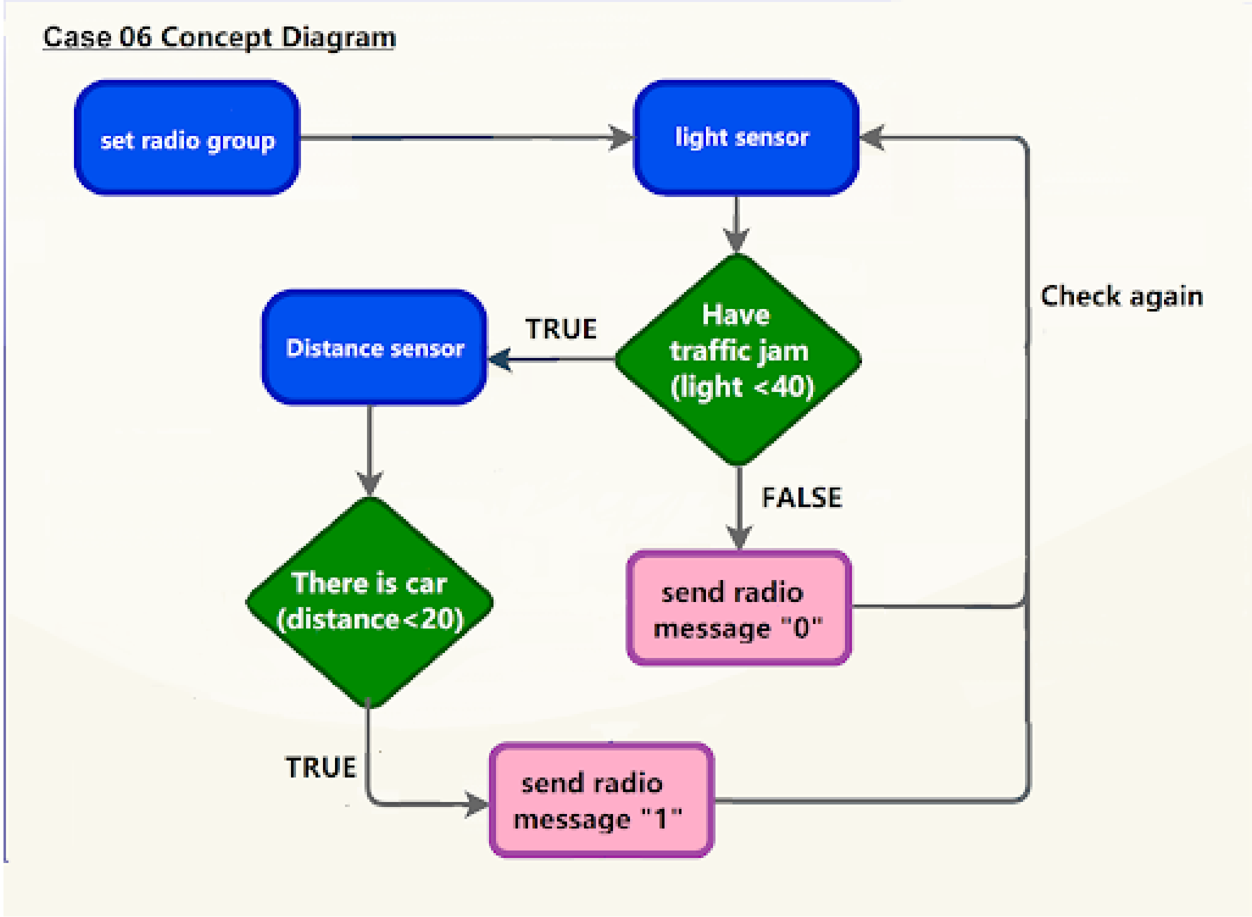


Goal

Background

What is a smart traffic system ?

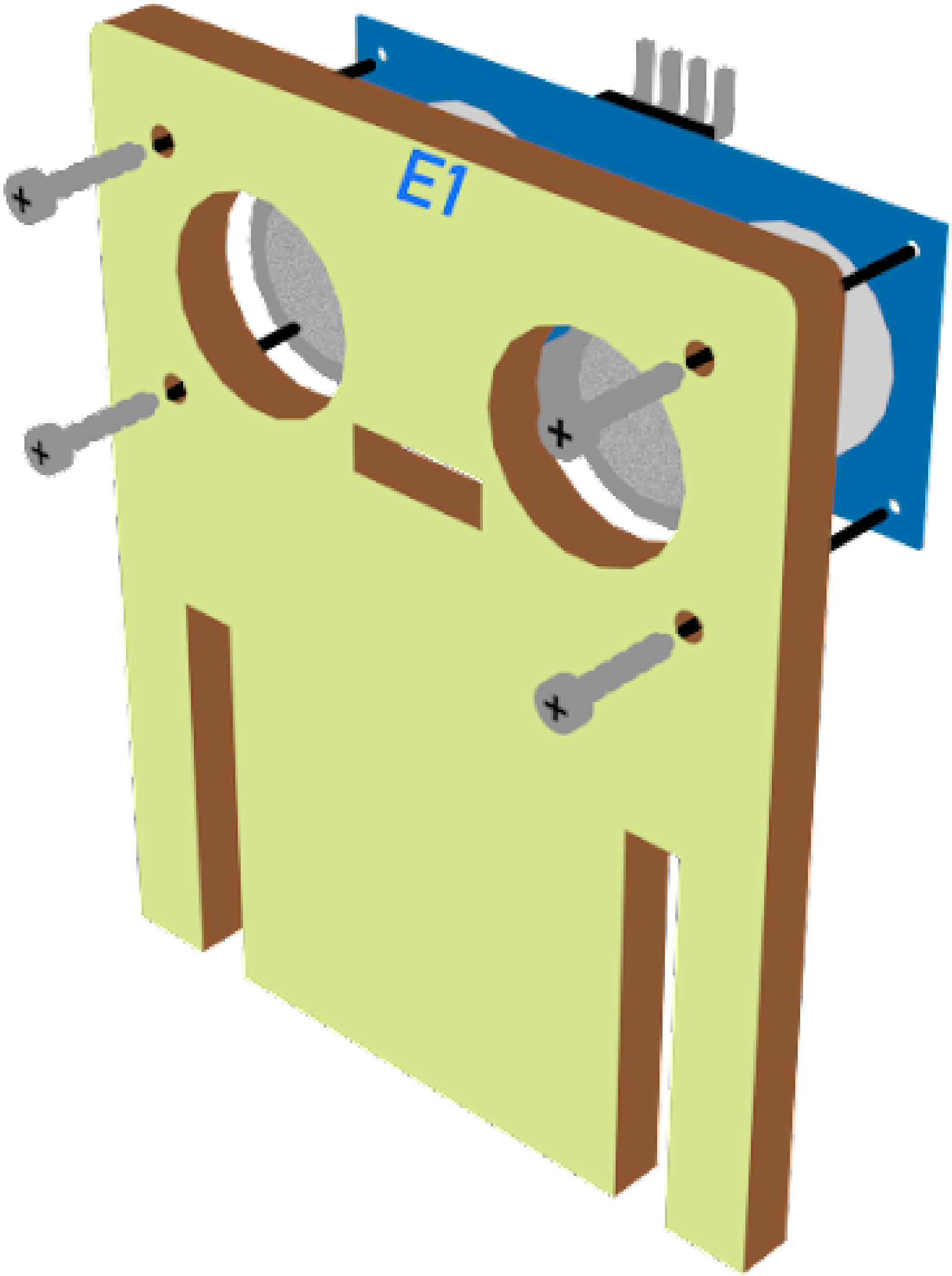
Smart traffic system Operation



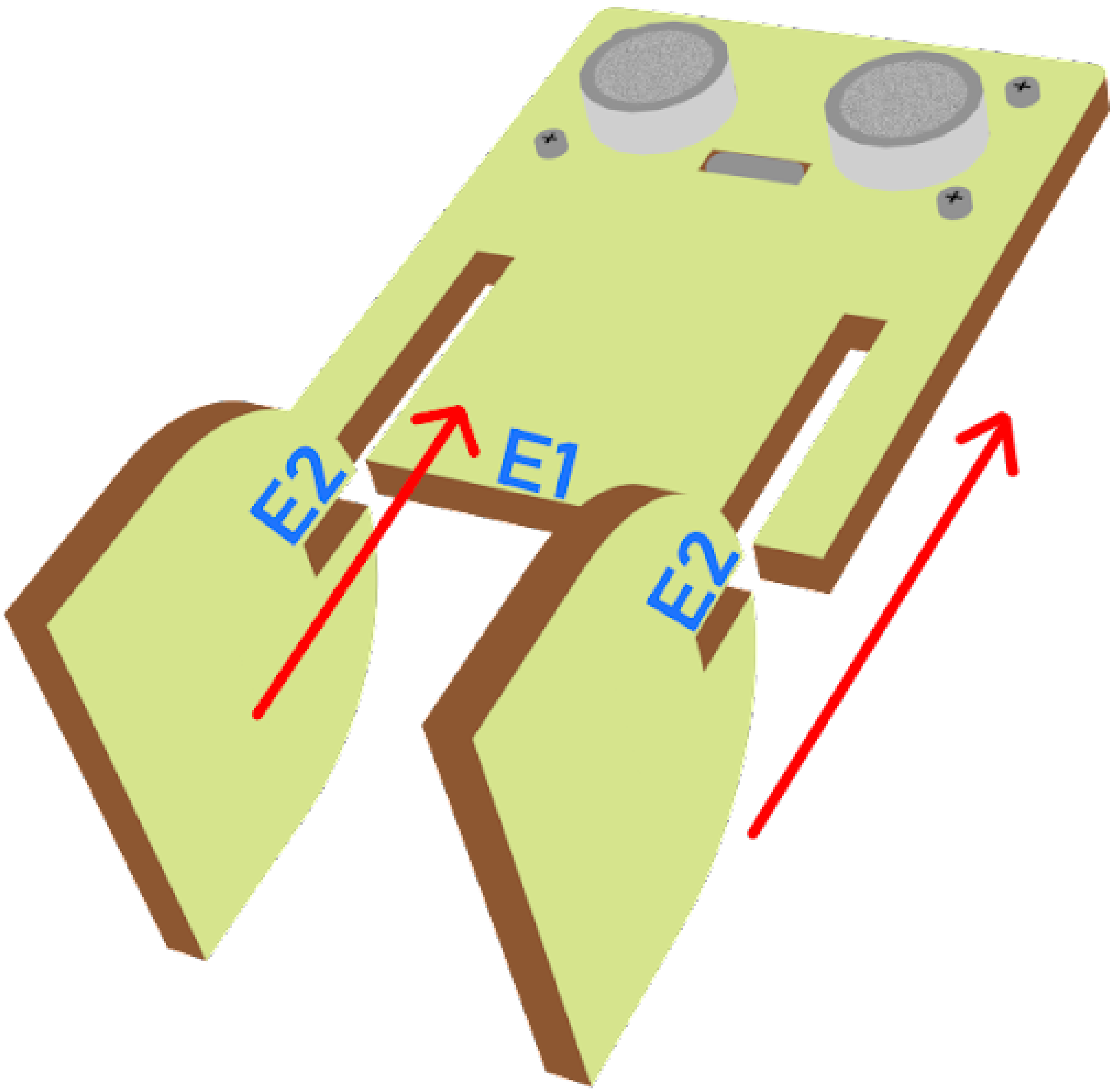
Part List

Assembly step

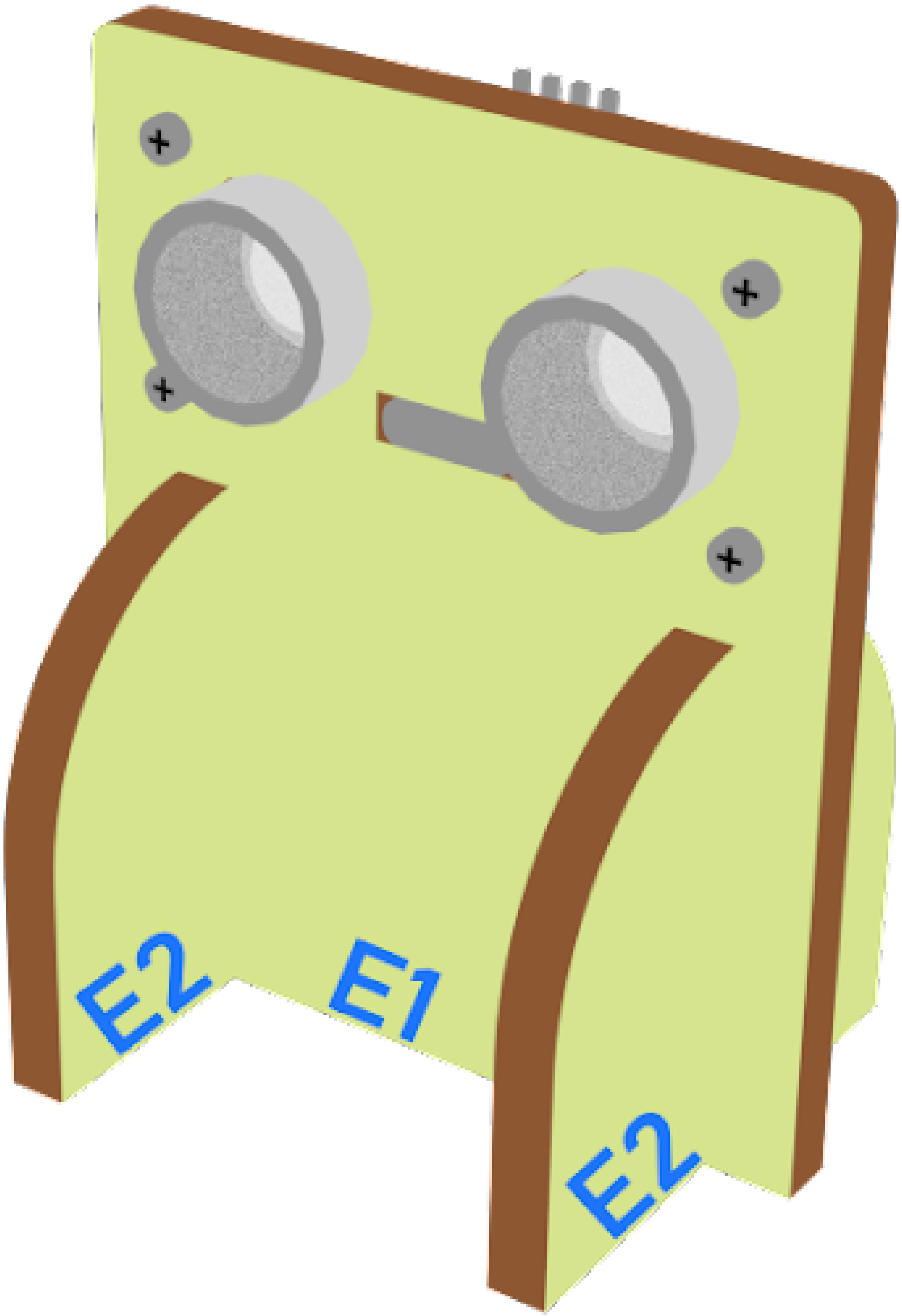
Step 1



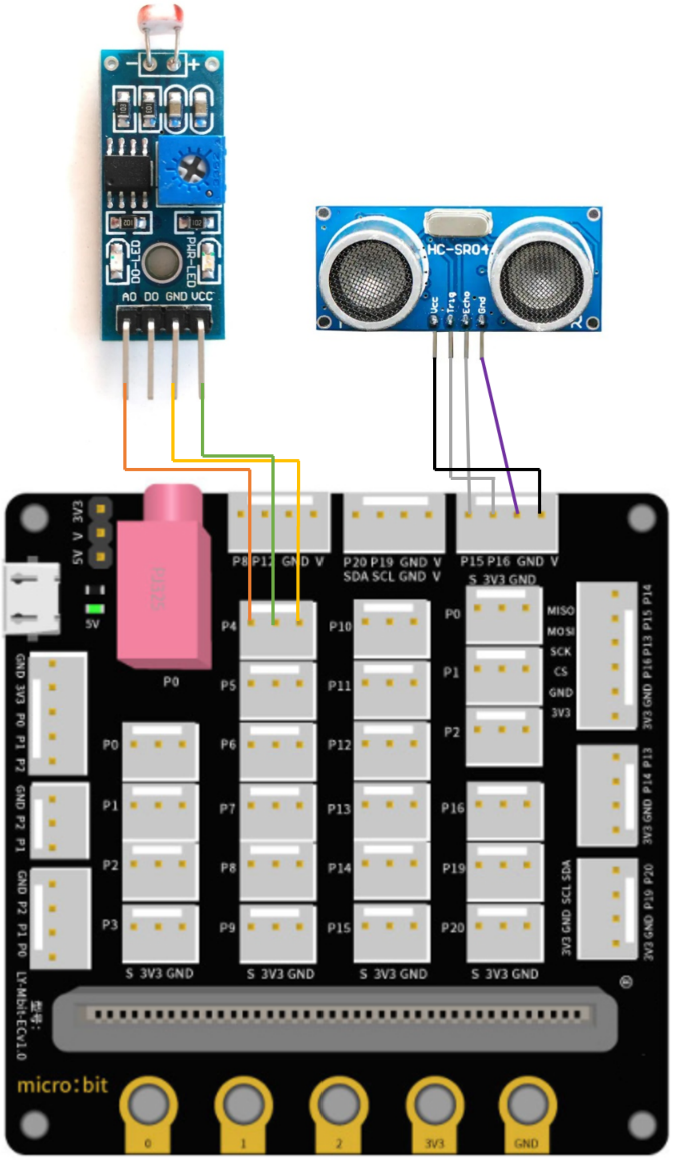
Step 2

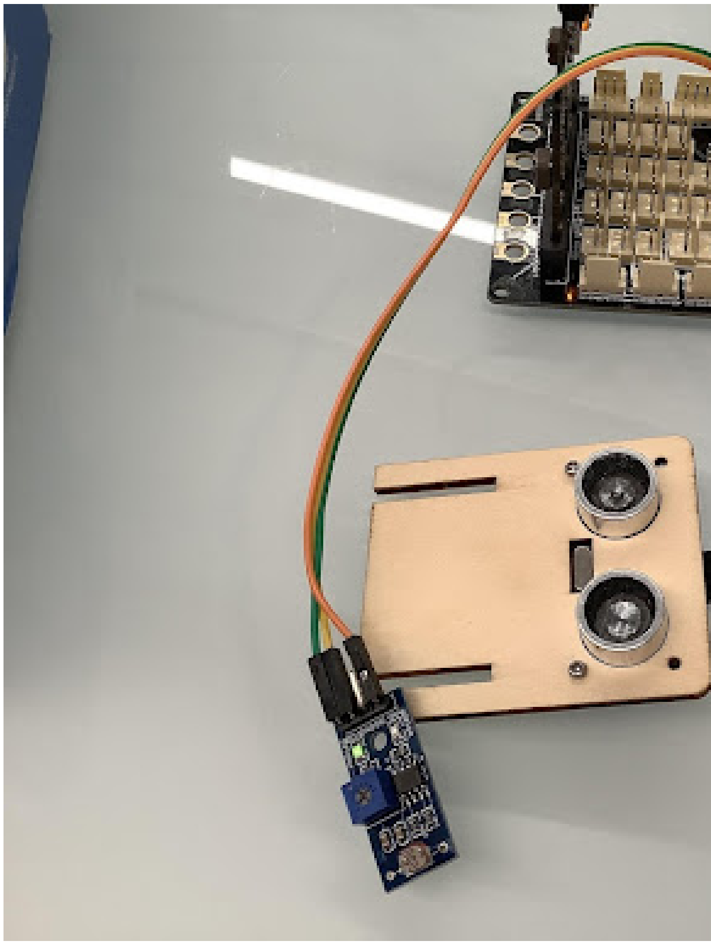


Step 3



Hardware connect



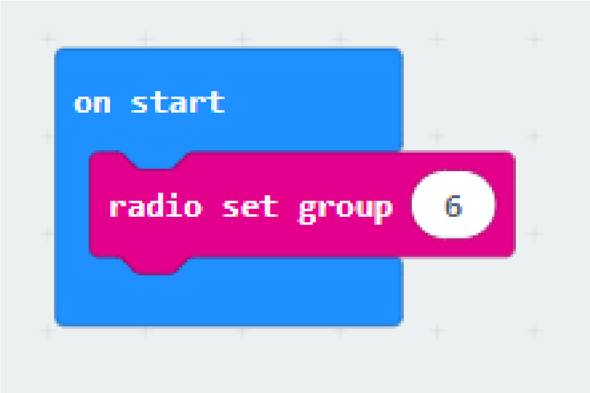


Programming (MakeCode)

Sender

Step 1. Set radio set group at start position

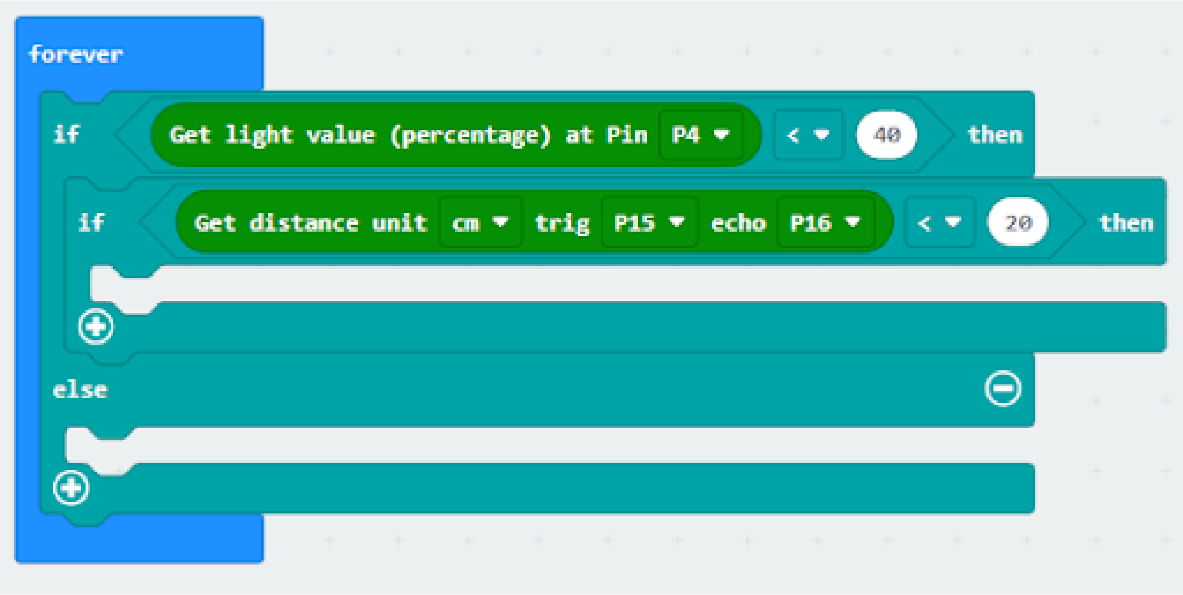
- Drag radio set group 6 to on start



```
on start
  radio set group 6
```

Step 2. Get light and distance value

- Snap if statement into forever, set get light value (percentage) at pin P4 < 40
- If get light value (percentage) at pin P4 < 40, and else if get distance unit cm trig P15 echo 16 < 20



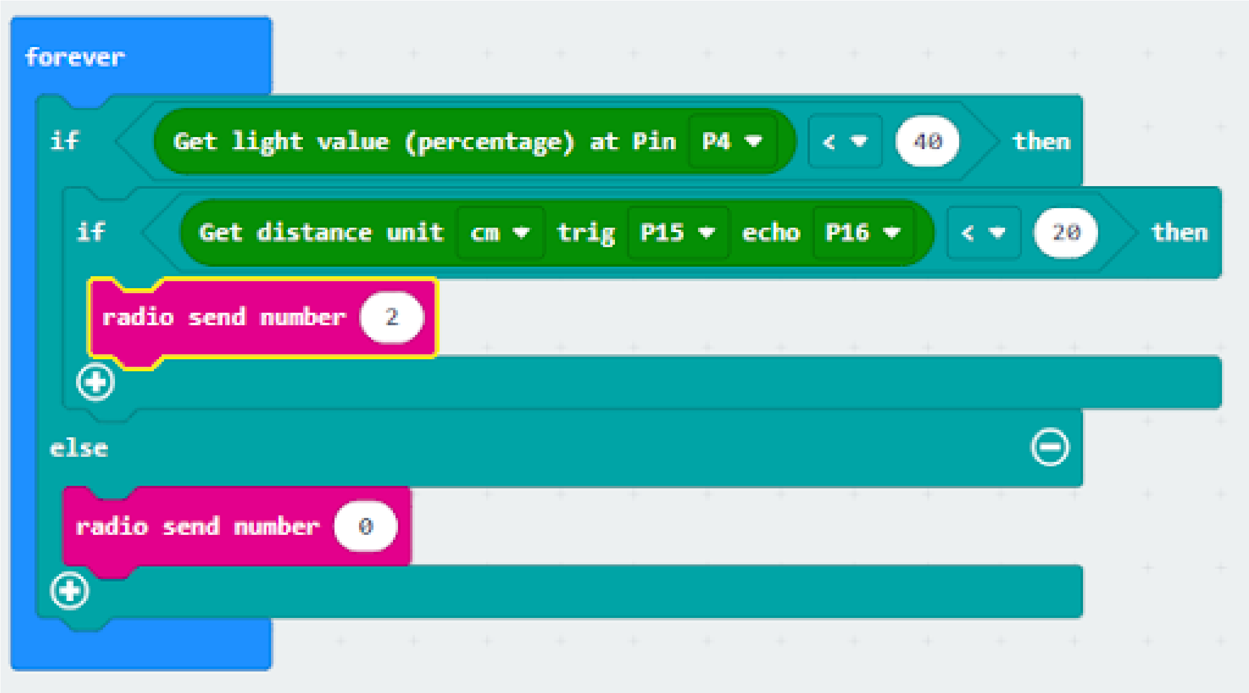
```
forever
  if Get light value (percentage) at Pin P4 < 40 then
    if Get distance unit cm trig P15 echo P16 < 20 then

  else

```

Step 3. Control the car by sending radio number

- Drag radio send number to 2 into if
- Drag radio send number to 0 into else

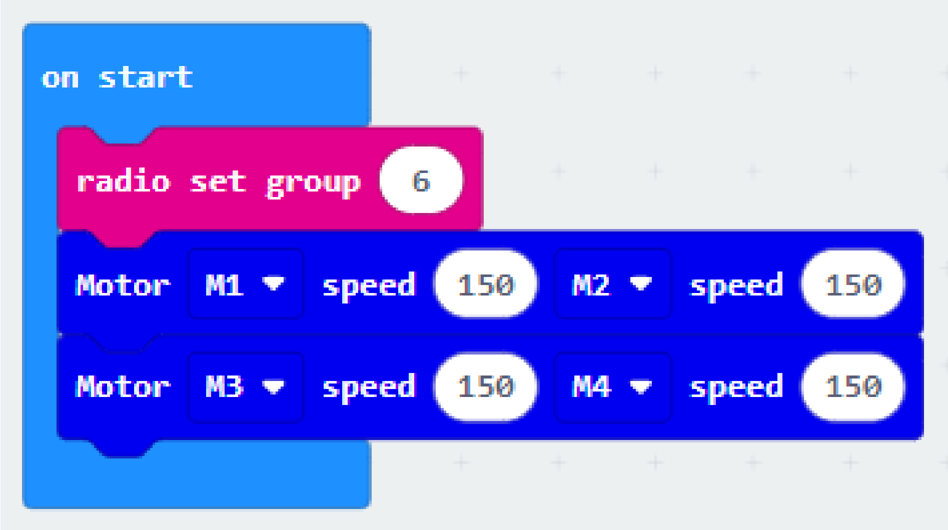


```
forever
  if  Get light value (percentage) at Pin P4 ▾  < ▾  40  then
    if  Get distance unit cm ▾ trig P15 ▾ echo P16 ▾  < ▾  20  then
      radio send number 2
    ⊕
  else ⊖
    radio send number 0
  ⊕
```

Receiver

Step 1. Set radio set group at start position

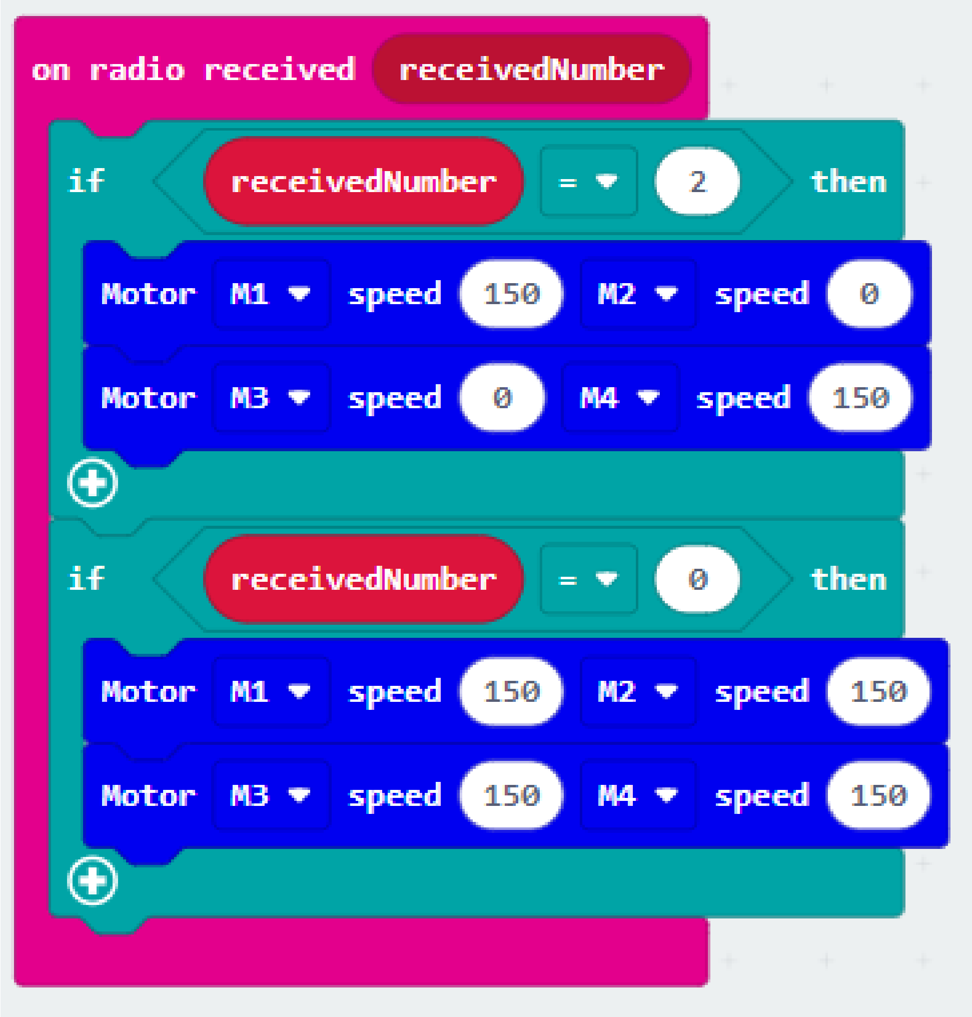
- Drag radio set group 6 to on start
- Initially, the car moves forward by default



```
on start
  radio set group 6
  Motor M1 ▾ speed 150 M2 ▾ speed 150
  Motor M3 ▾ speed 150 M4 ▾ speed 150
```

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =2 and make the car turn left
- Set receivedNumber=0 and make the car move forward

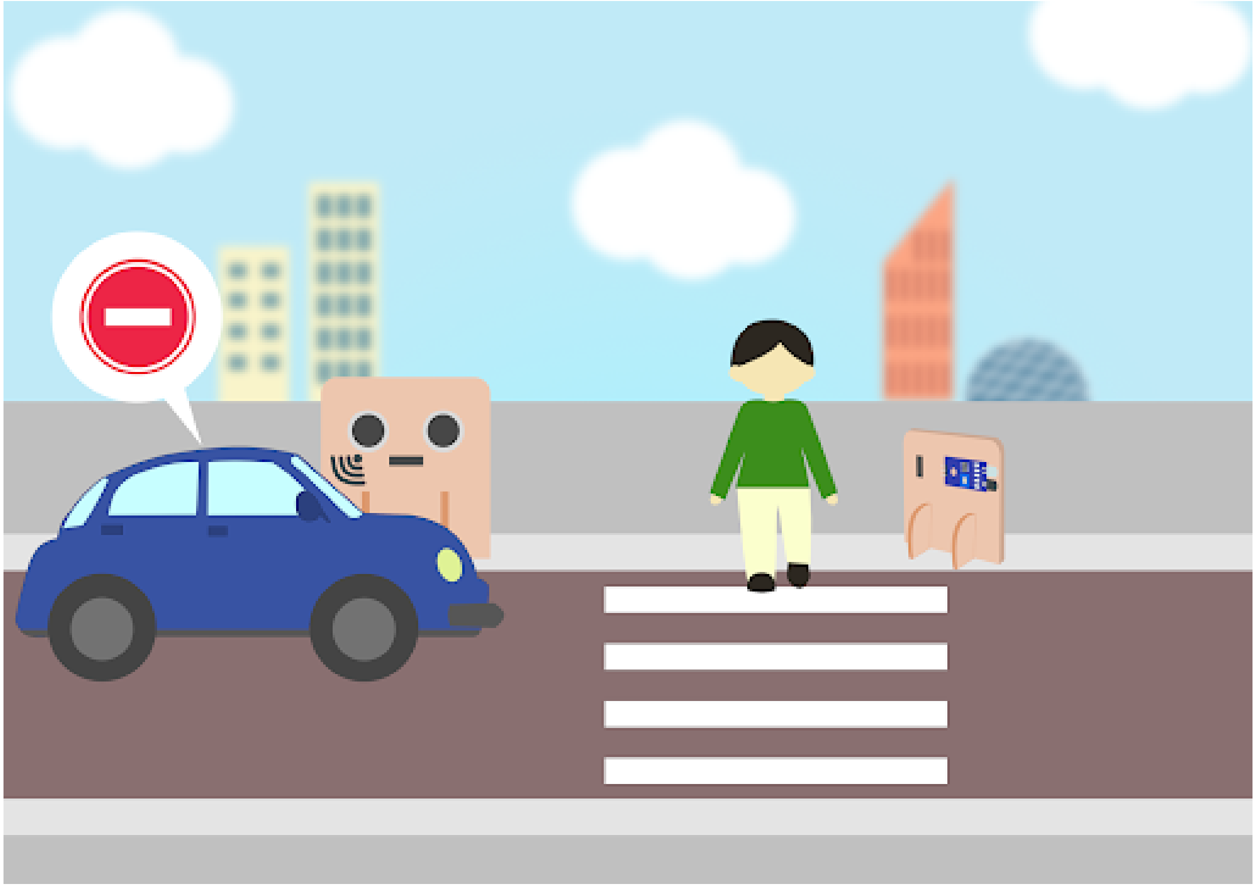


```
on radio received receivedNumber
  if receivedNumber = 2 then
    Motor M1 speed 150 M2 speed 0
    Motor M3 speed 0 M4 speed 150
  if receivedNumber = 0 then
    Motor M1 speed 150 M2 speed 150
    Motor M3 speed 150 M4 speed 150
```

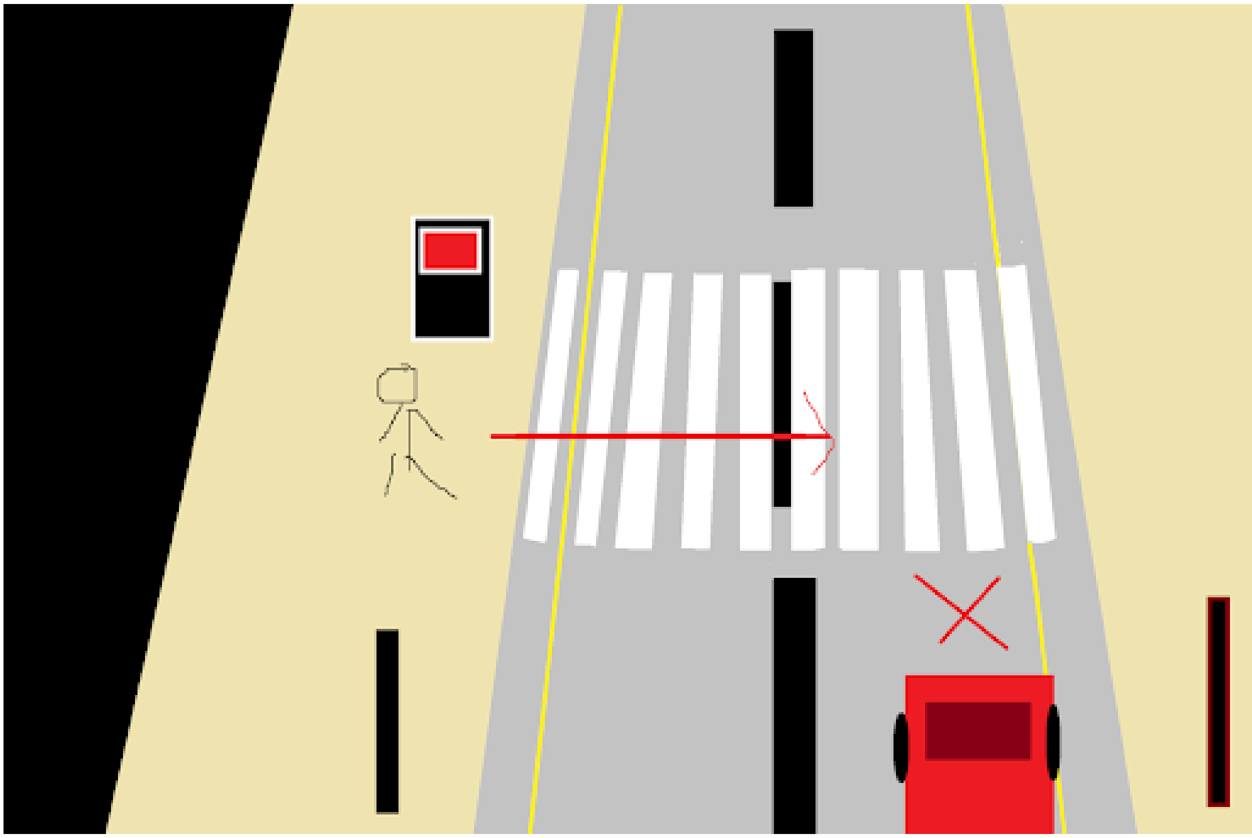
Result

Think

1.2.9 Crosswalk



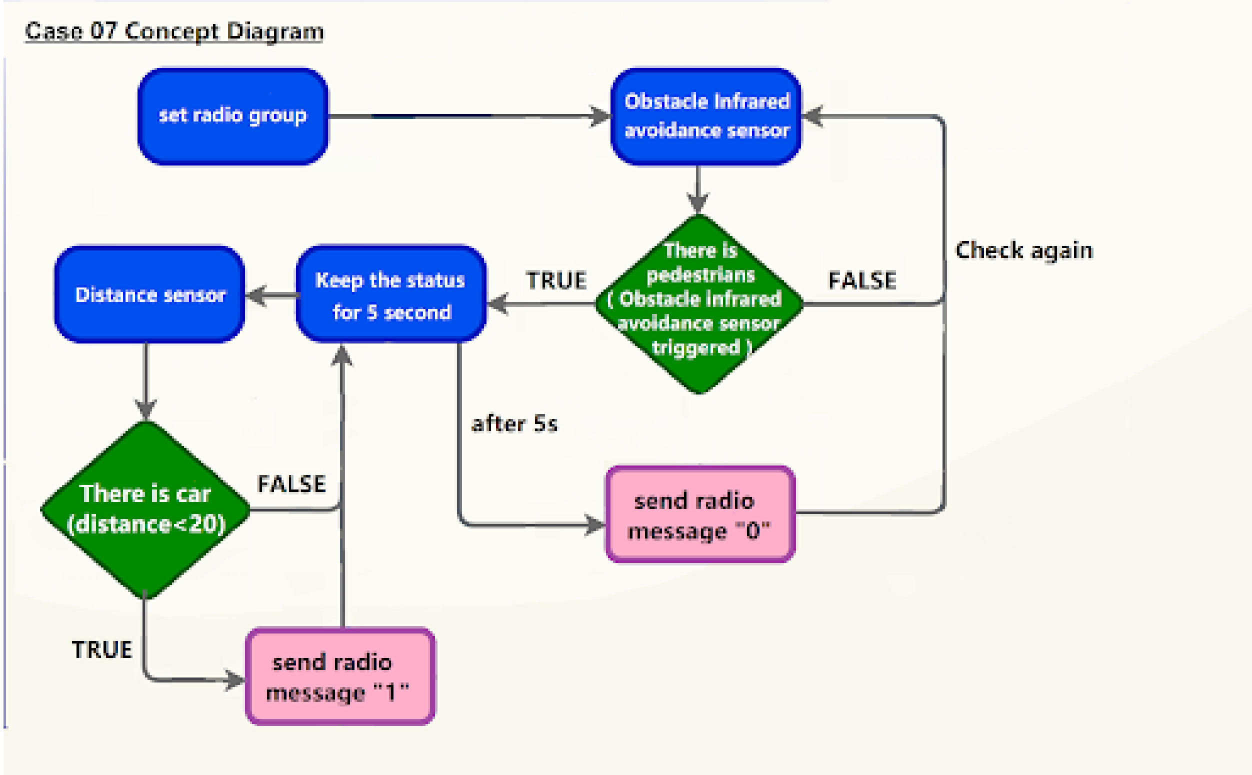
Goal



Background

What is a smart crosswalk system?

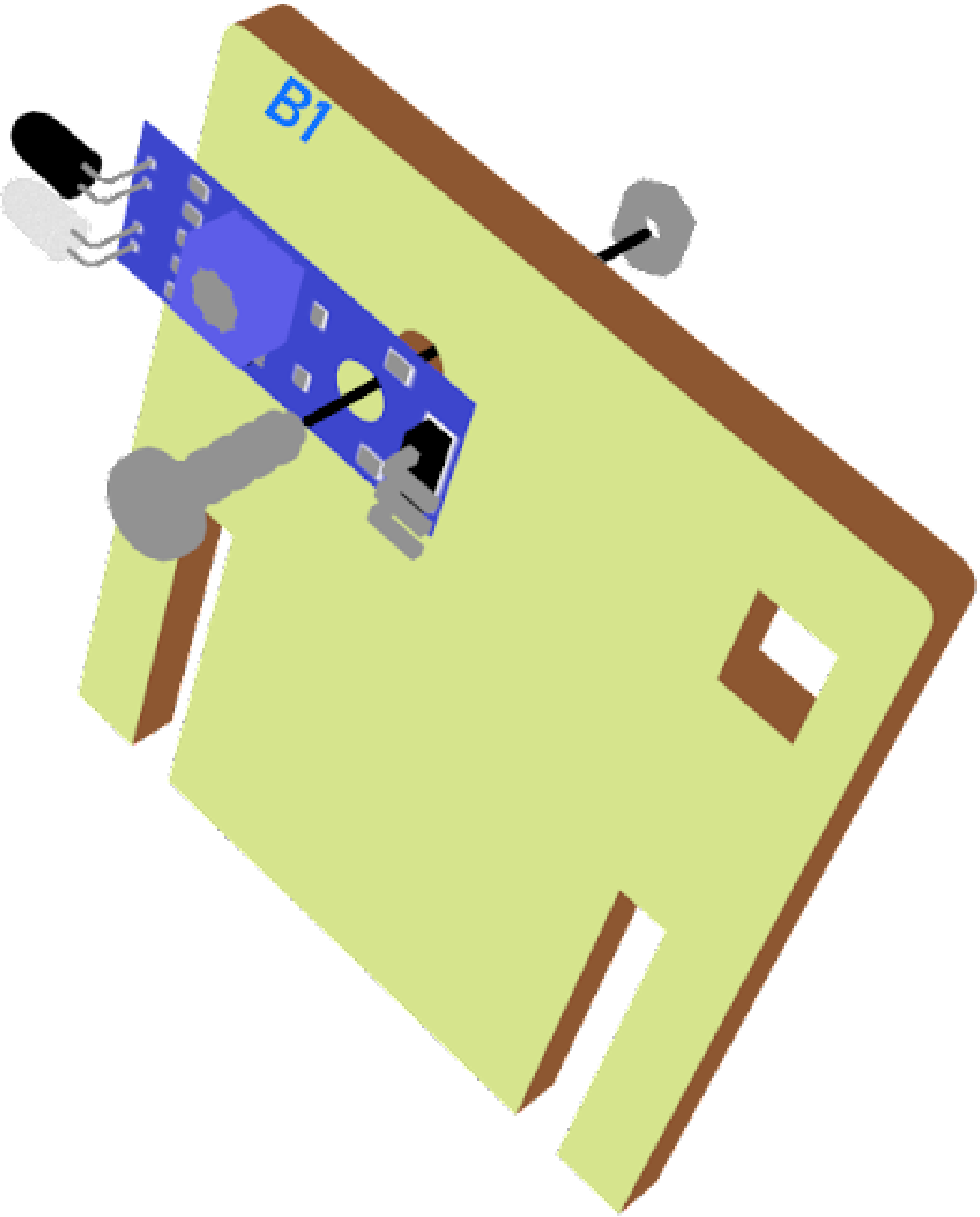
Smart crosswalk system operation


Case 07 Concept Diagram
set radio group
Obstacle Infrared avoidance sensor
There is pedestrians (Obstacle infrared avoidance sensor triggered)
TRUE
FALSE
Check again
Keep the status for 5 second
Distance sensor
after 5s
There is car (distance<20)
FALSE
TRUE
send radio message "0"
send radio message "1"

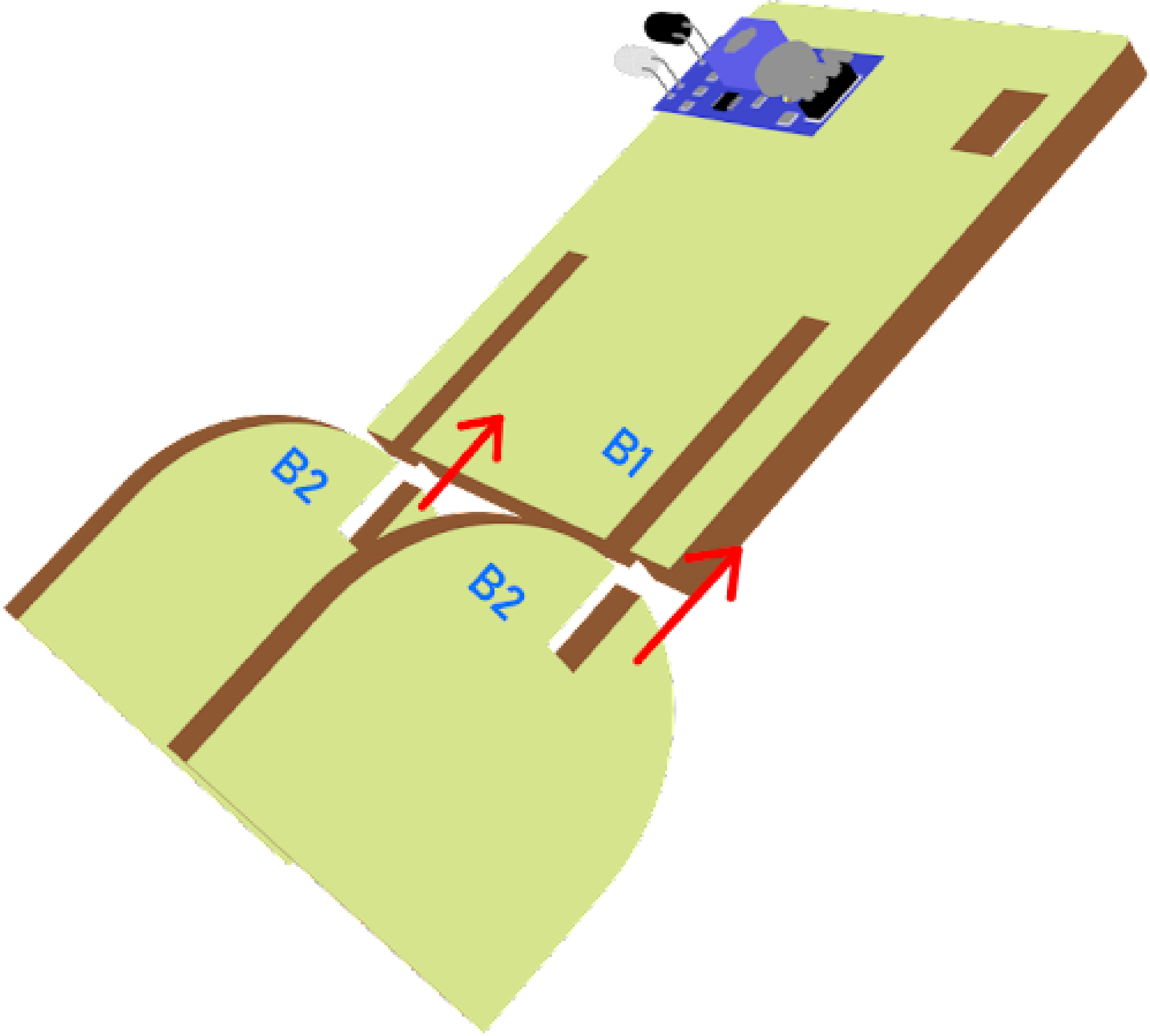
Part List

Assembly step

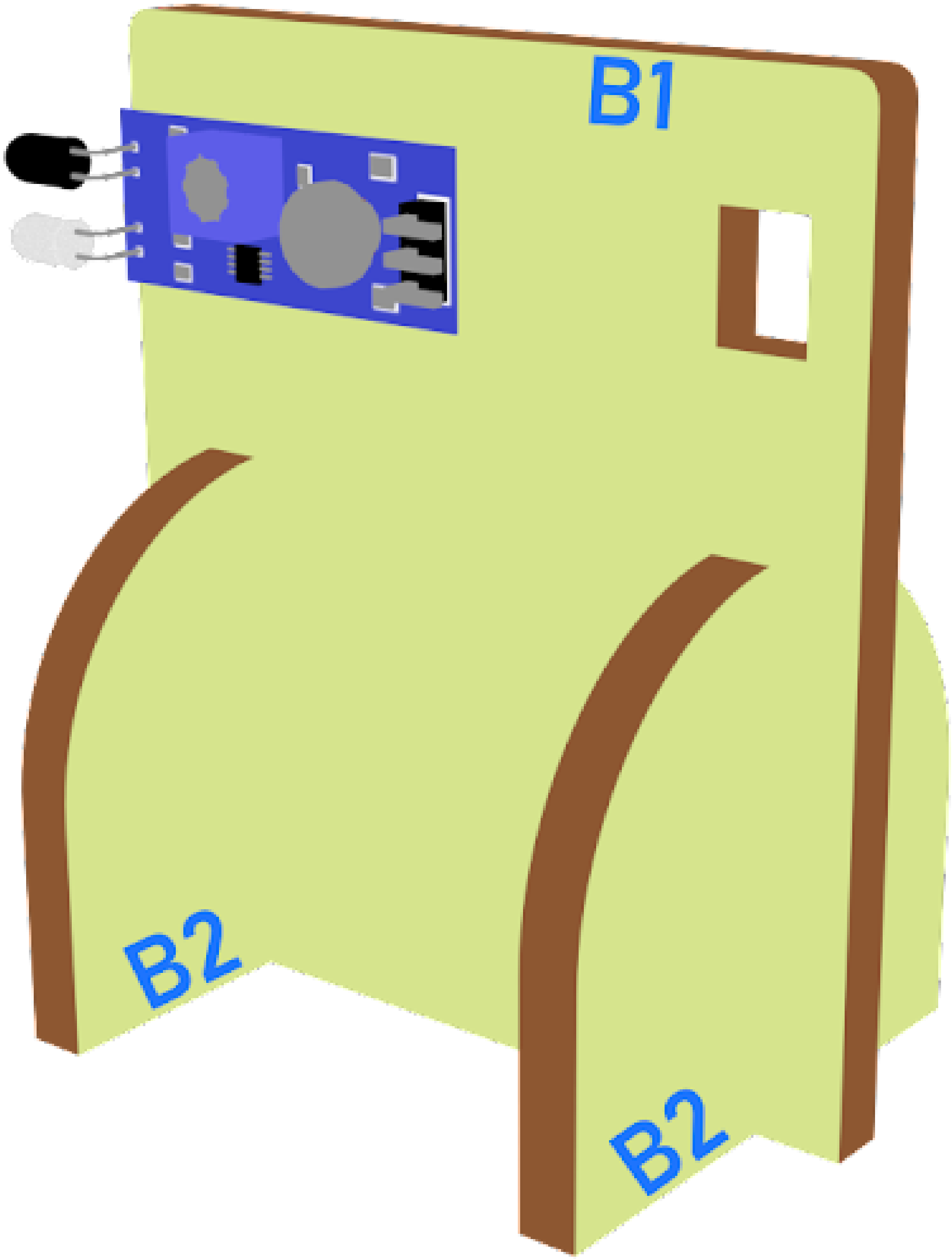
Step 1



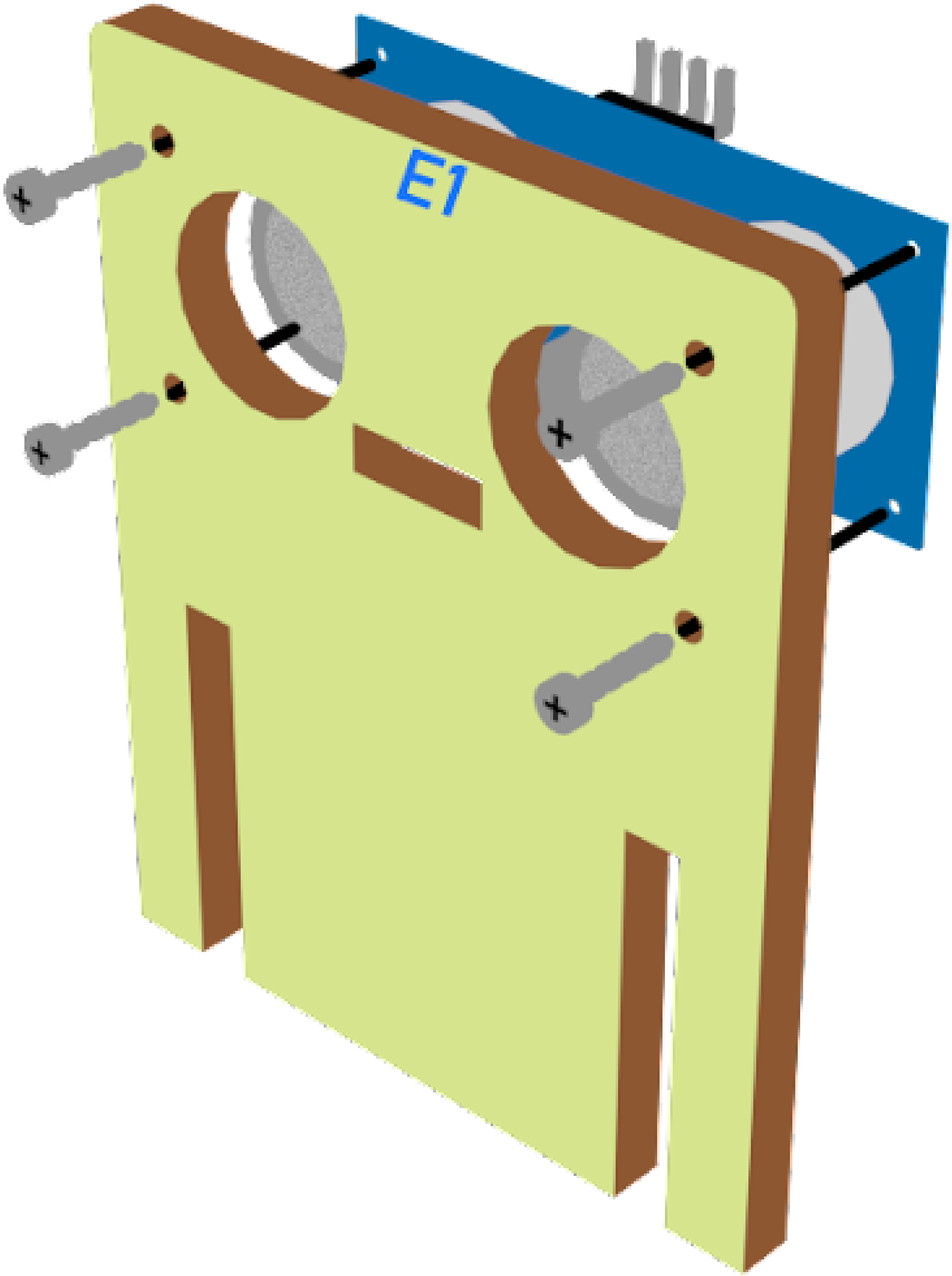
Step 2



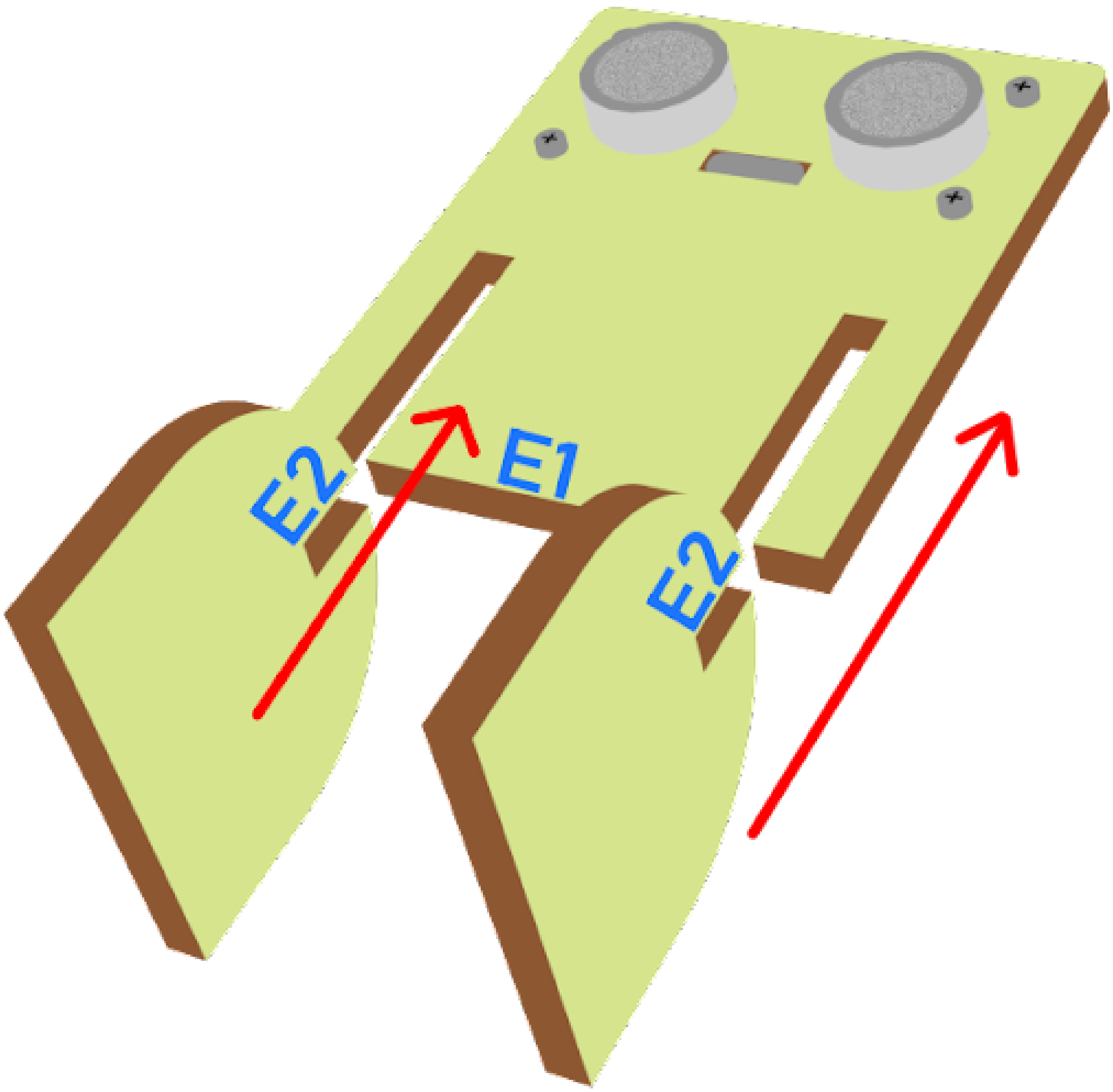
Step 3



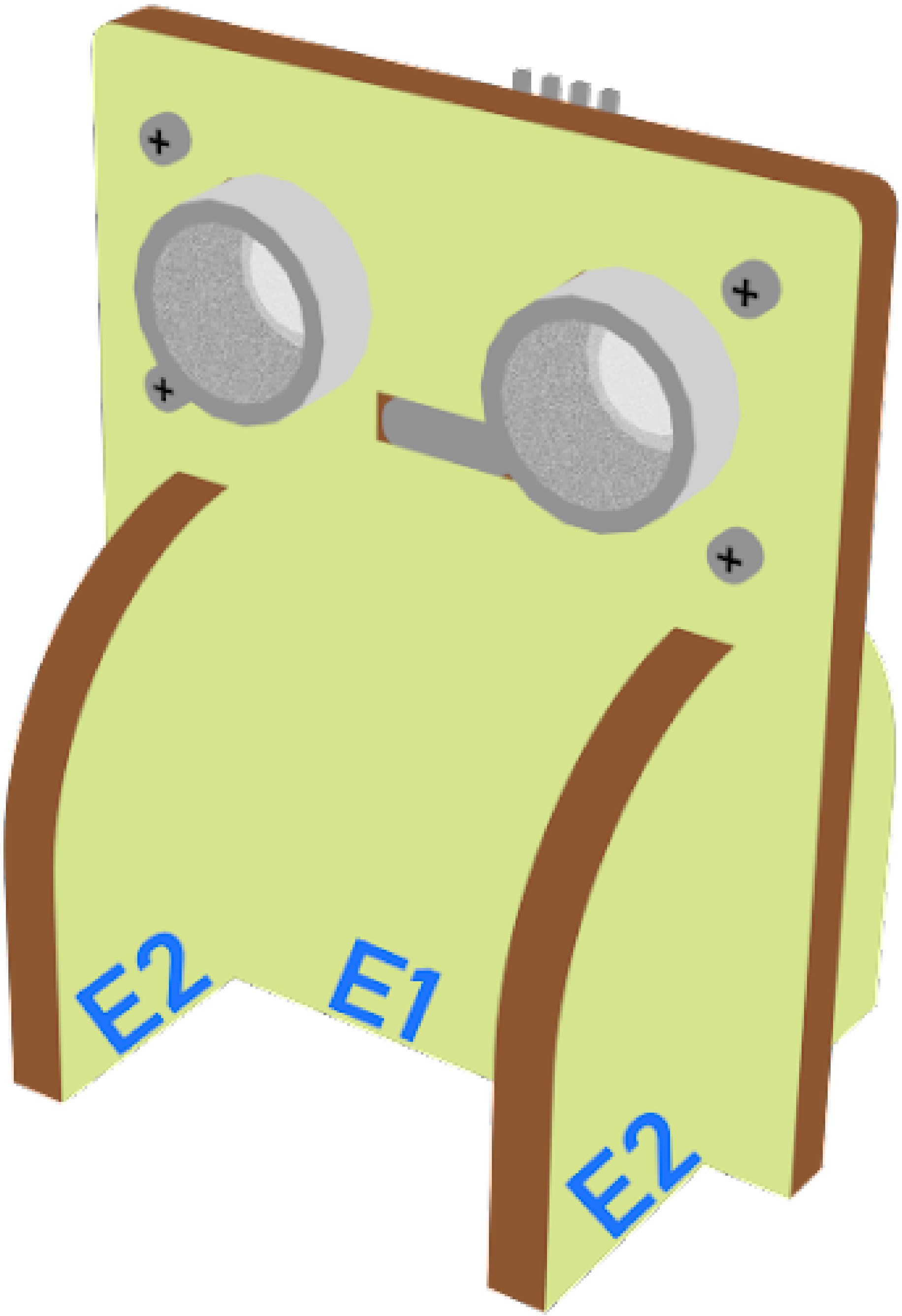
Step 4



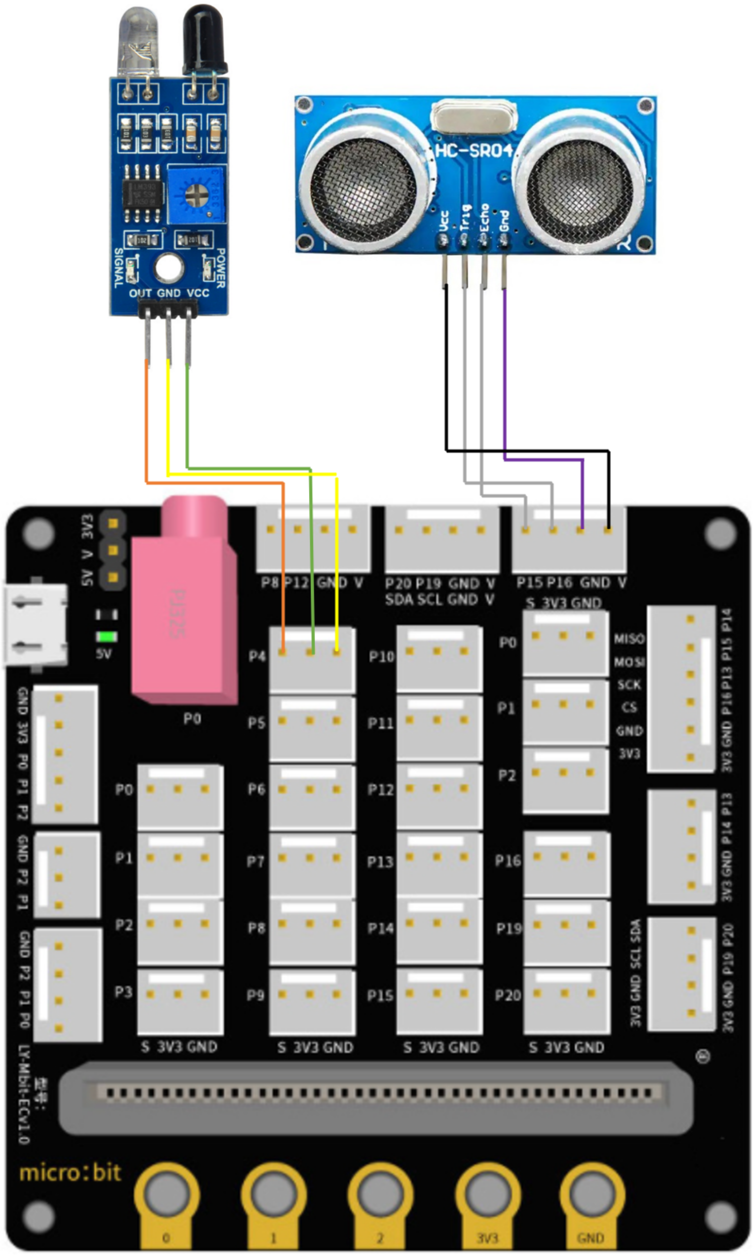
Step 5

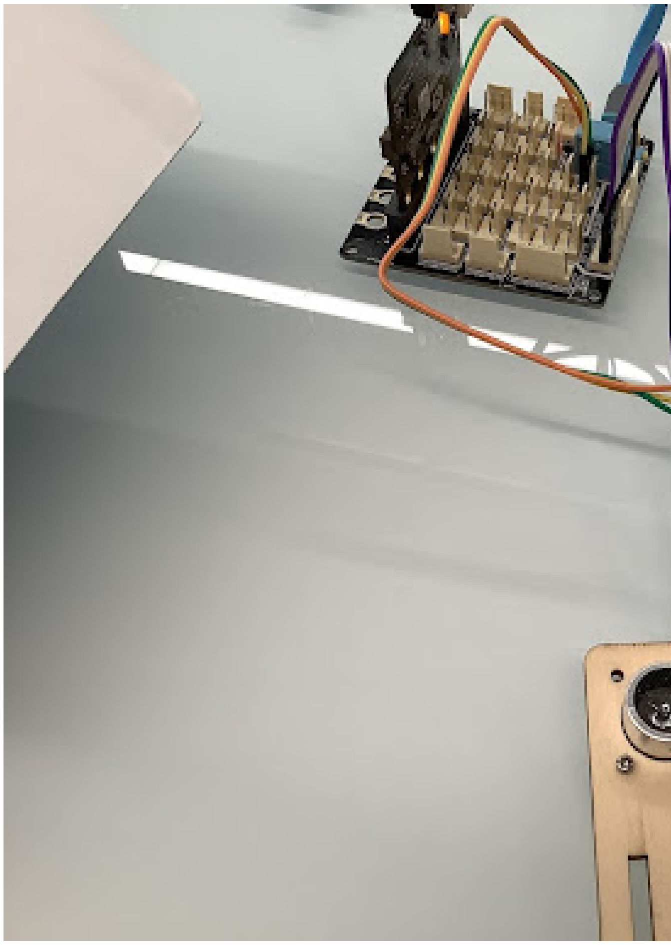


Step 6



Hardware connect



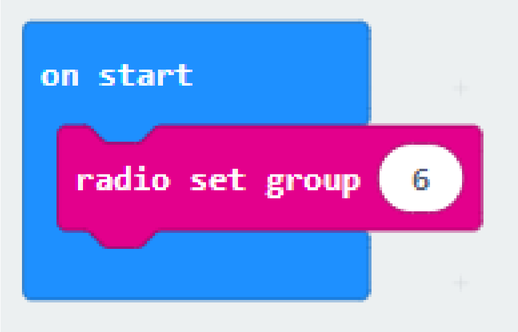


Programming (MakeCode)

Sender:

Step 1. Set radio set group at start position

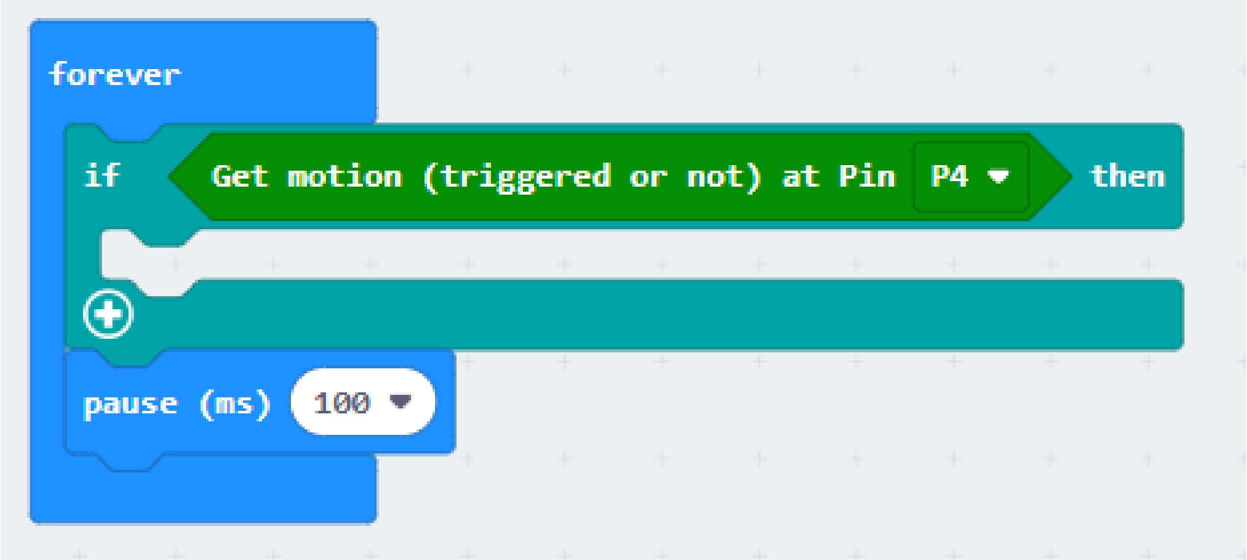
- Drag radio set group 6 to on start



```
on start
    radio set group 6
```

Step 2. When trigger a motion

- Snap if statement into forever, set get motion (triggered or not) at pin P4
- Snap pause to the loop to wait for 0.1 second for next checking

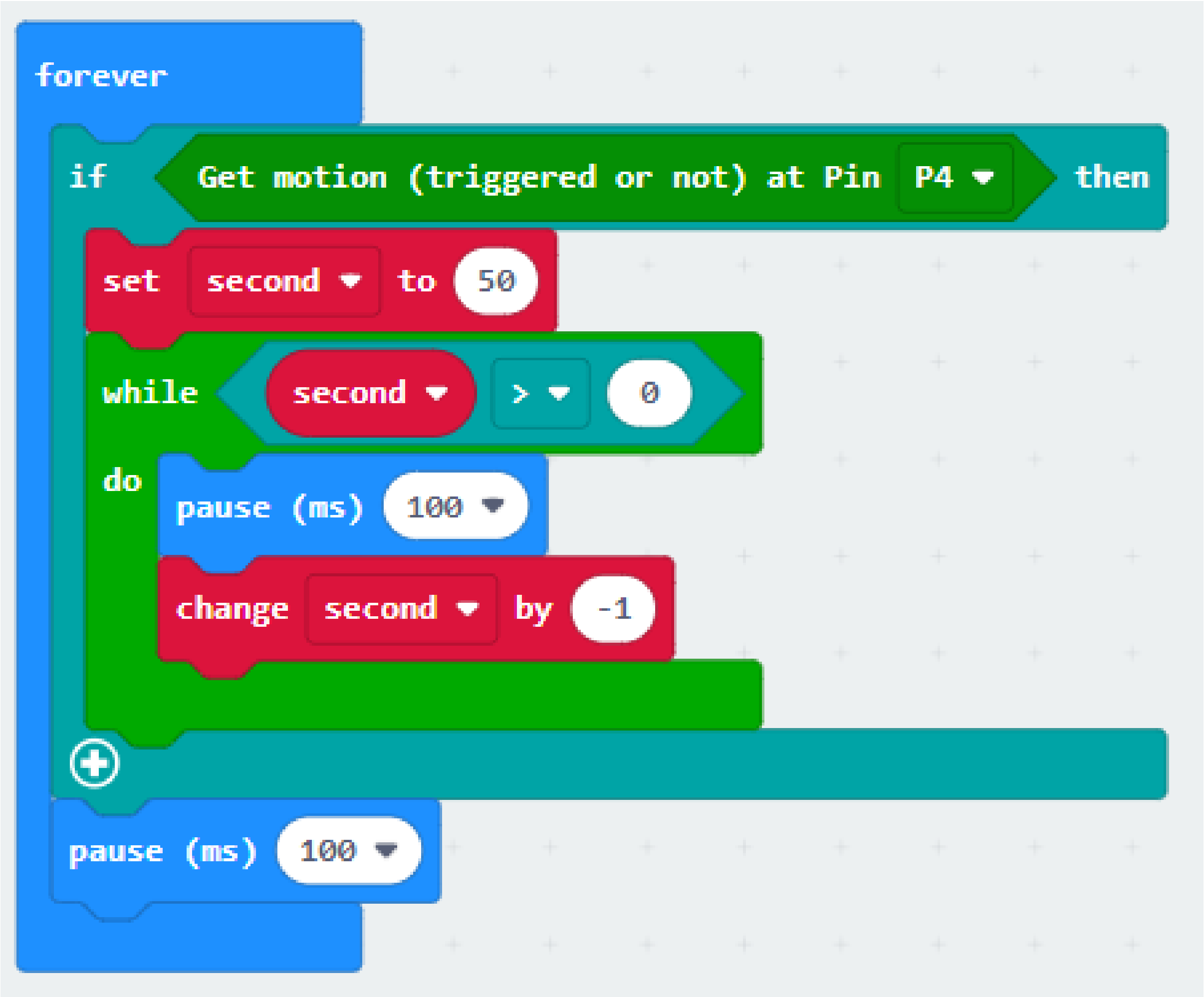


```
forever
    if Get motion (triggered or not) at Pin P4 then

    pause (ms) 100
```

Step 3. Keep the status for 5 second

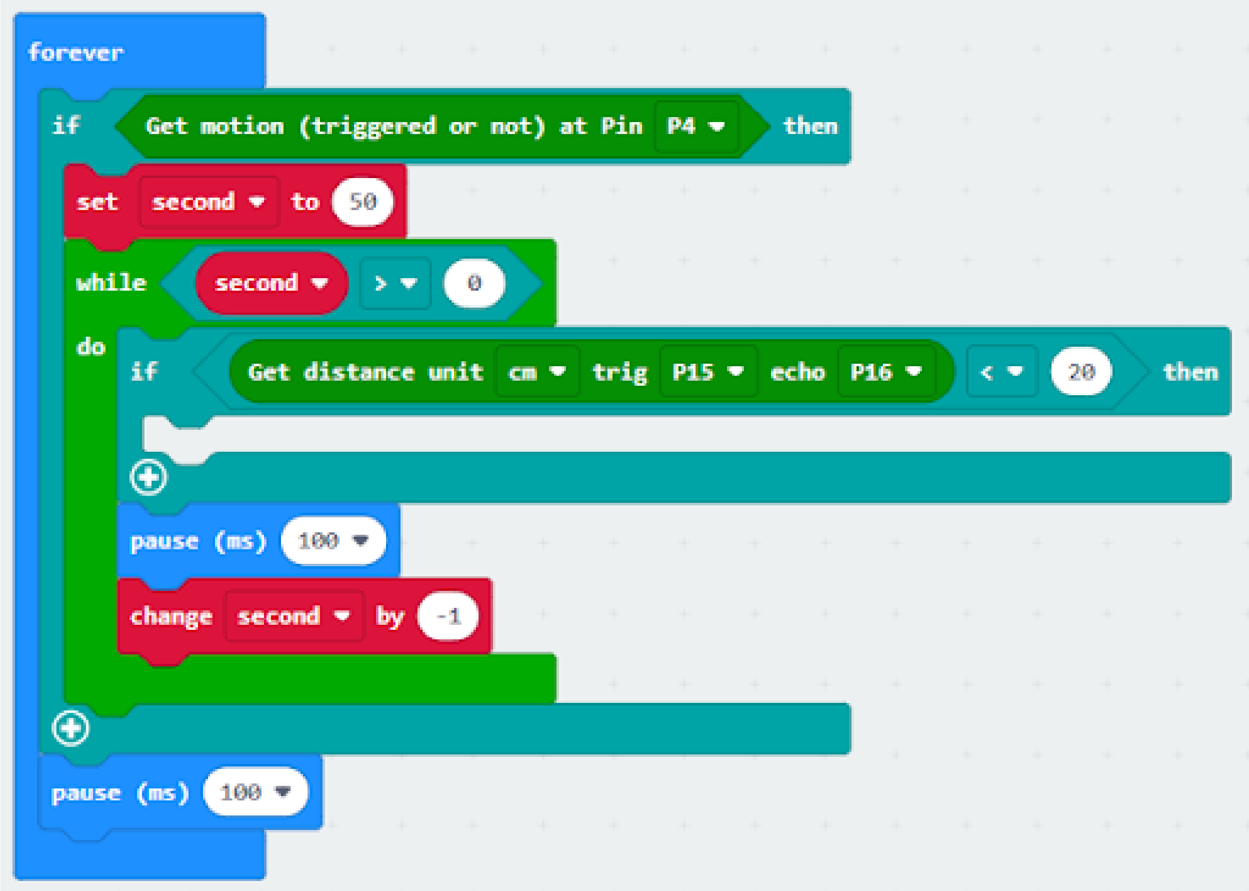
- set variable second to 50
- While second > 0, snap pause to 0.1 second and change second by -1.



```
forever
  if Get motion (triggered or not) at Pin P4 ▾ then
    set second ▾ to 50
    while second ▾ > ▾ 0
    do
      pause (ms) 100 ▾
      change second ▾ by -1
  ⊕
  pause (ms) 100 ▾
```

Step 4. Get distance value

- Snap if statement into while loop, set get distance unit cm trig P15 echo P16 < 20



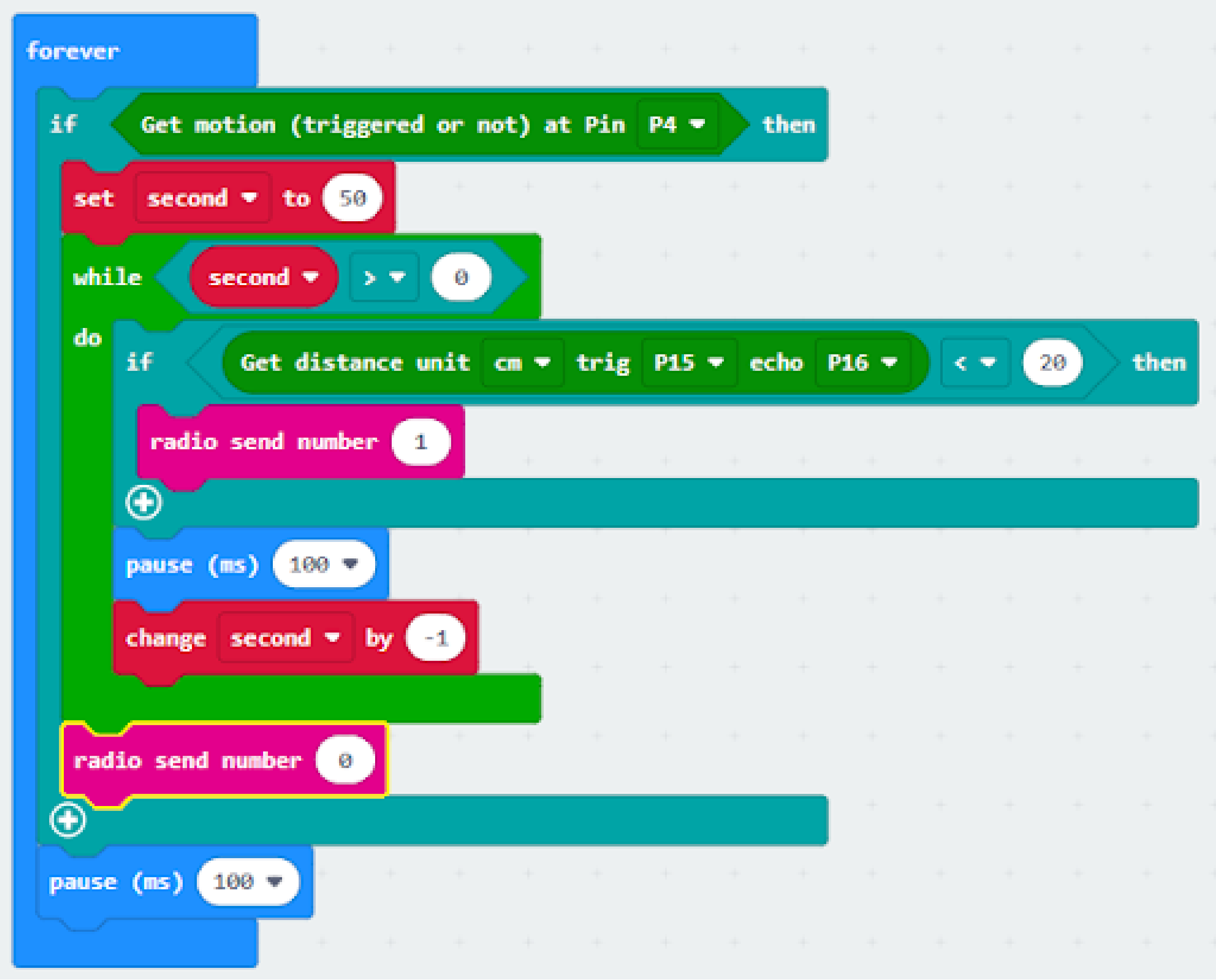
```
forever
    if Get motion (triggered or not) at Pin P4 then
        set second to 50
        while second > 0
        do
            if Get distance unit cm trig P15 echo P16 < 20 then

            pause (ms) 100
            change second by -1

    pause (ms) 100
```

Step 5. Control the car by sending radio number

- Drag radio send number to 1 into if 1 if

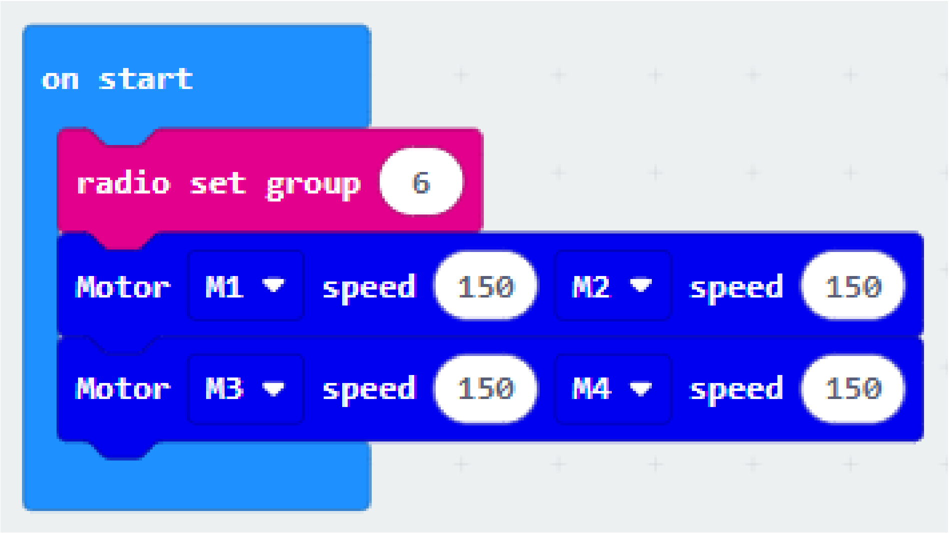


```
forever
  if Get motion (triggered or not) at Pin P4 then
    set second to 50
    while second > 0
    do
      if Get distance unit cm trig P15 echo P16 < 20 then
        radio send number 1
      pause (ms) 100
      change second by -1
    radio send number 0
  pause (ms) 100
```

Receiver

Step 1. Set radio set group at start position

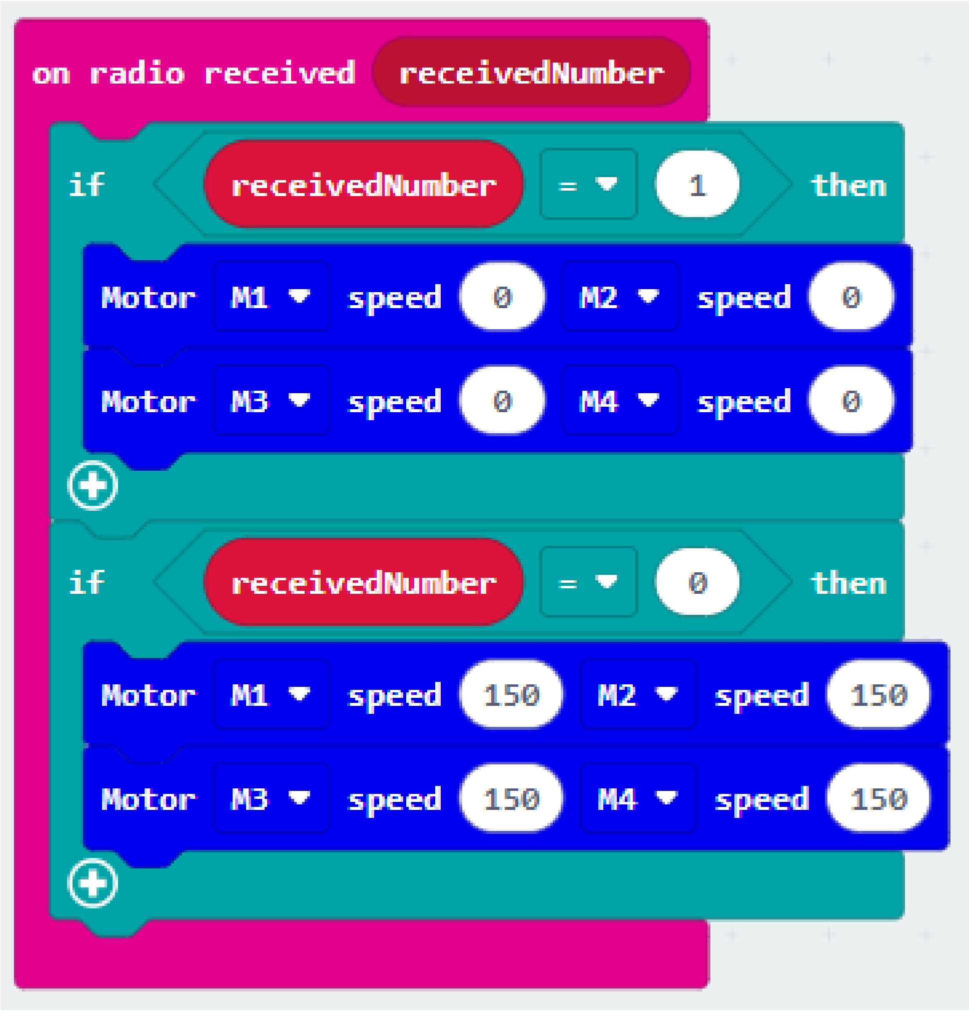
- Drag radio set group 6 to on start
- Initially, the car moves forward by default



```
on start
  radio set group 6
  Motor M1 speed 150 M2 speed 150
  Motor M3 speed 150 M4 speed 150
```

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward

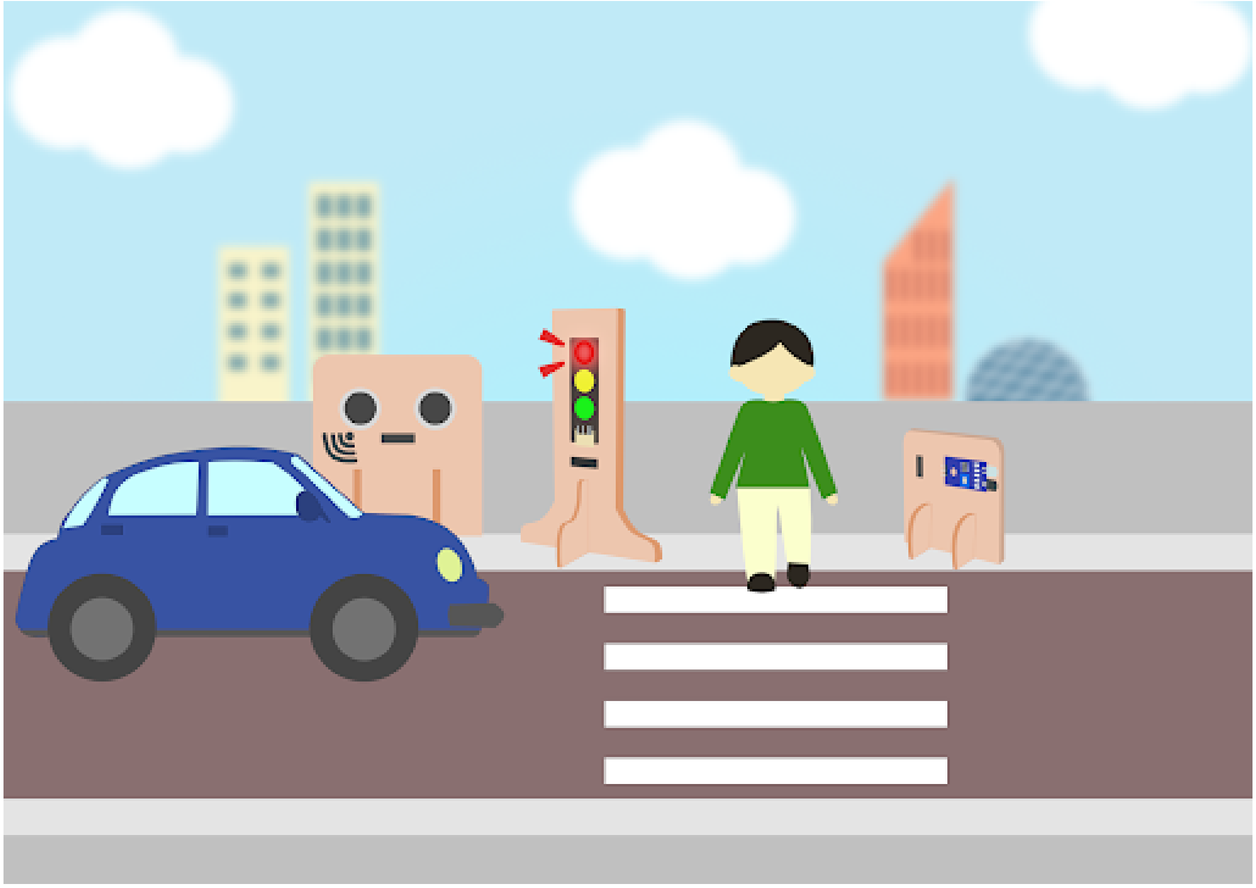


```
on radio received receivedNumber
    if receivedNumber = 1 then
        Motor M1 speed 0 M2 speed 0
        Motor M3 speed 0 M4 speed 0
    if receivedNumber = 0 then
        Motor M1 speed 150 M2 speed 150
        Motor M3 speed 150 M4 speed 150
```

Result

Think

1.2.10 Smart Traffic Lights

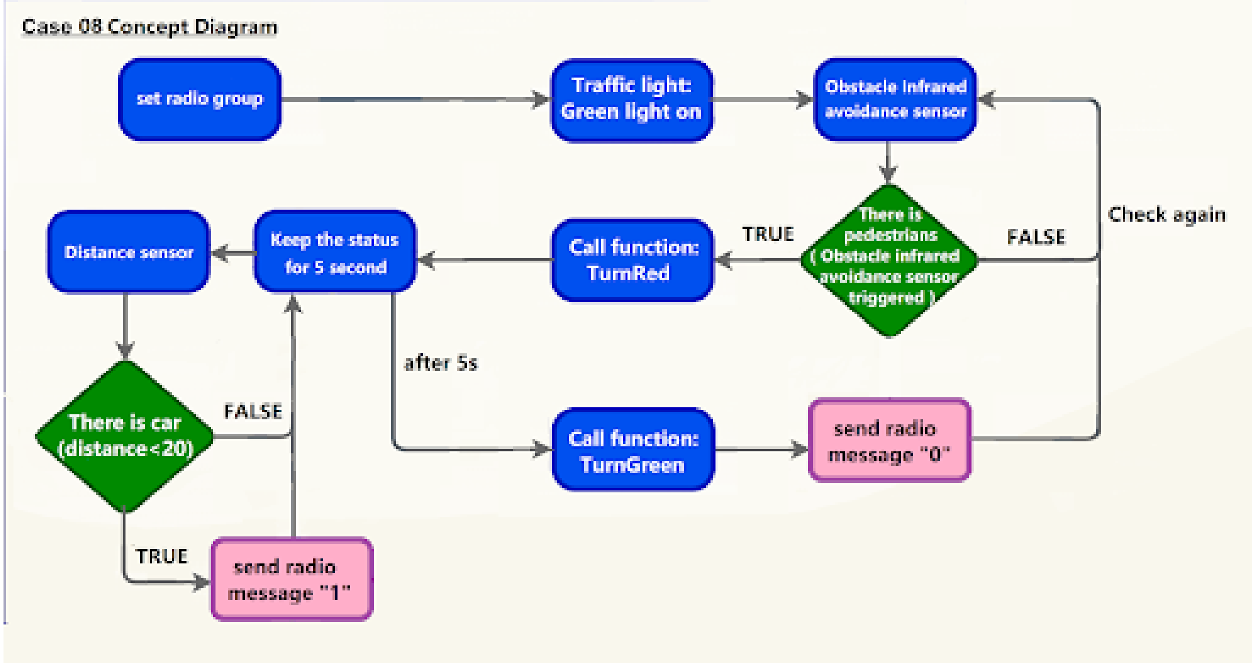


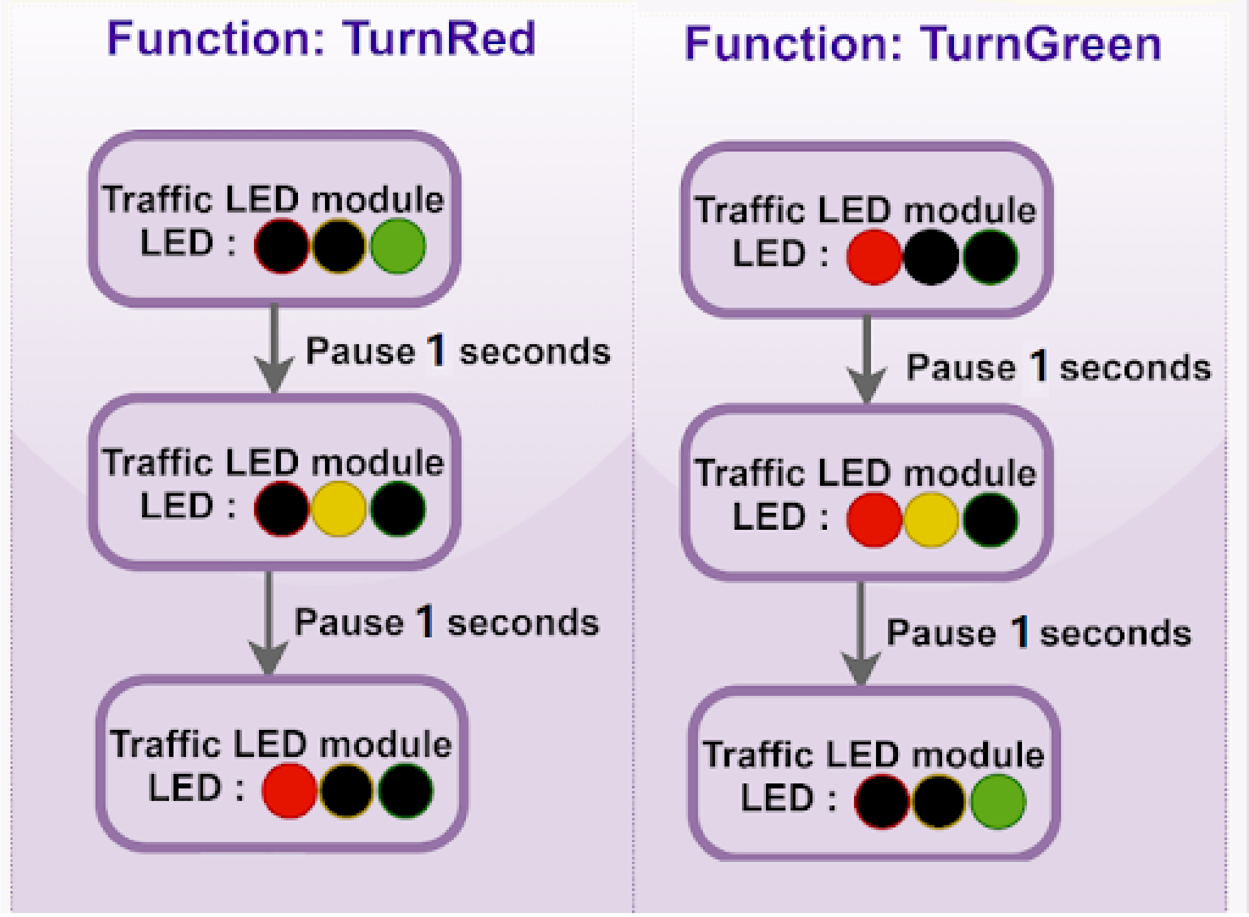
Goal

Background

What is a smart traffic light?

Smart traffic light operation

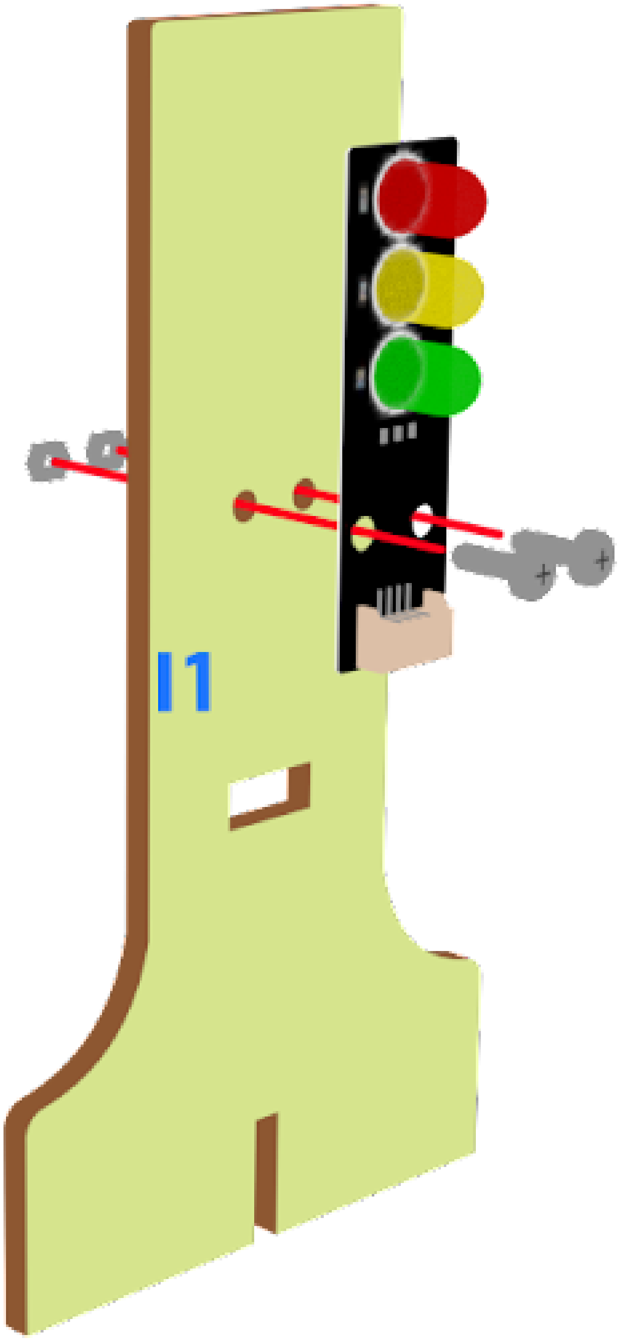

Case 08 Concept Diagram
set radio group
Traffic light: Green light on
Obstacle Infrared avoidance sensor
There is pedestrians (Obstacle infrared avoidance sensor triggered)
TRUE
FALSE
Check again
Call function: TurnRed
Keep the status for 5 second
Distance sensor
There is car (distance<20)
FALSE
TRUE
send radio message "1"
after 5s
Call function: TurnGreen
send radio message "0"



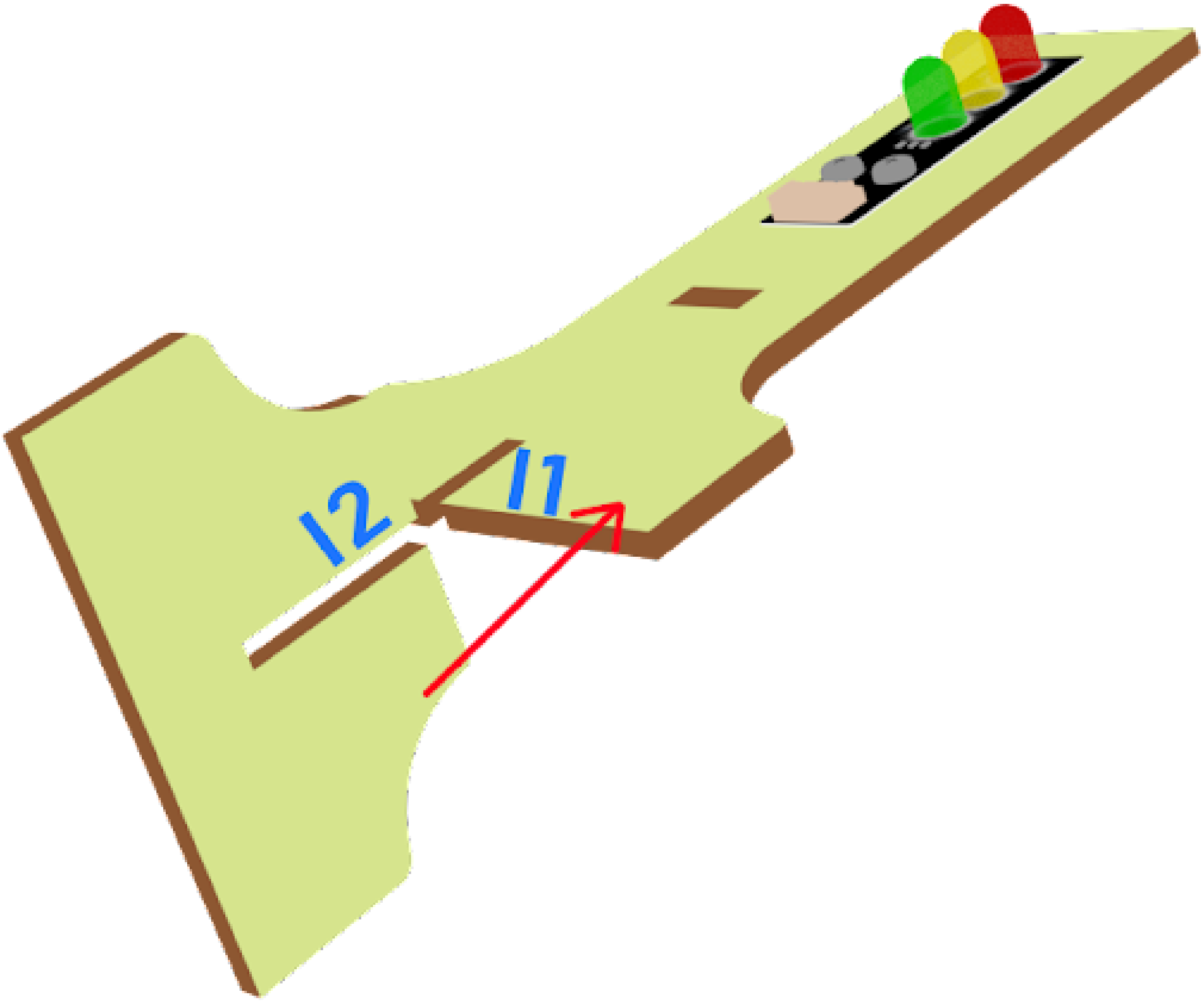
Part List

Assembly step

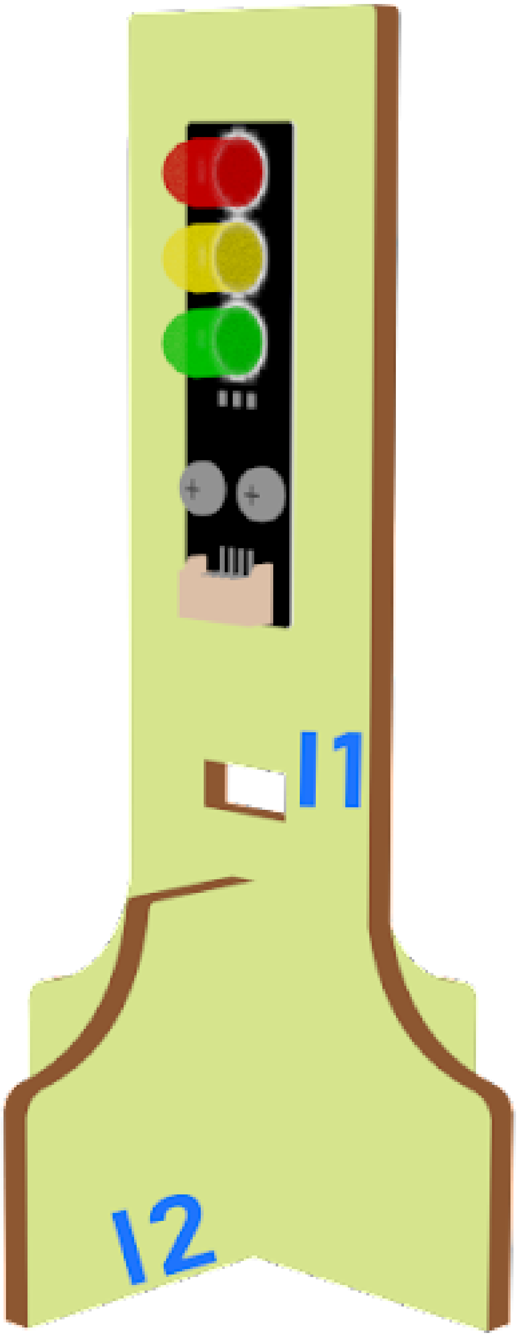
Step 1



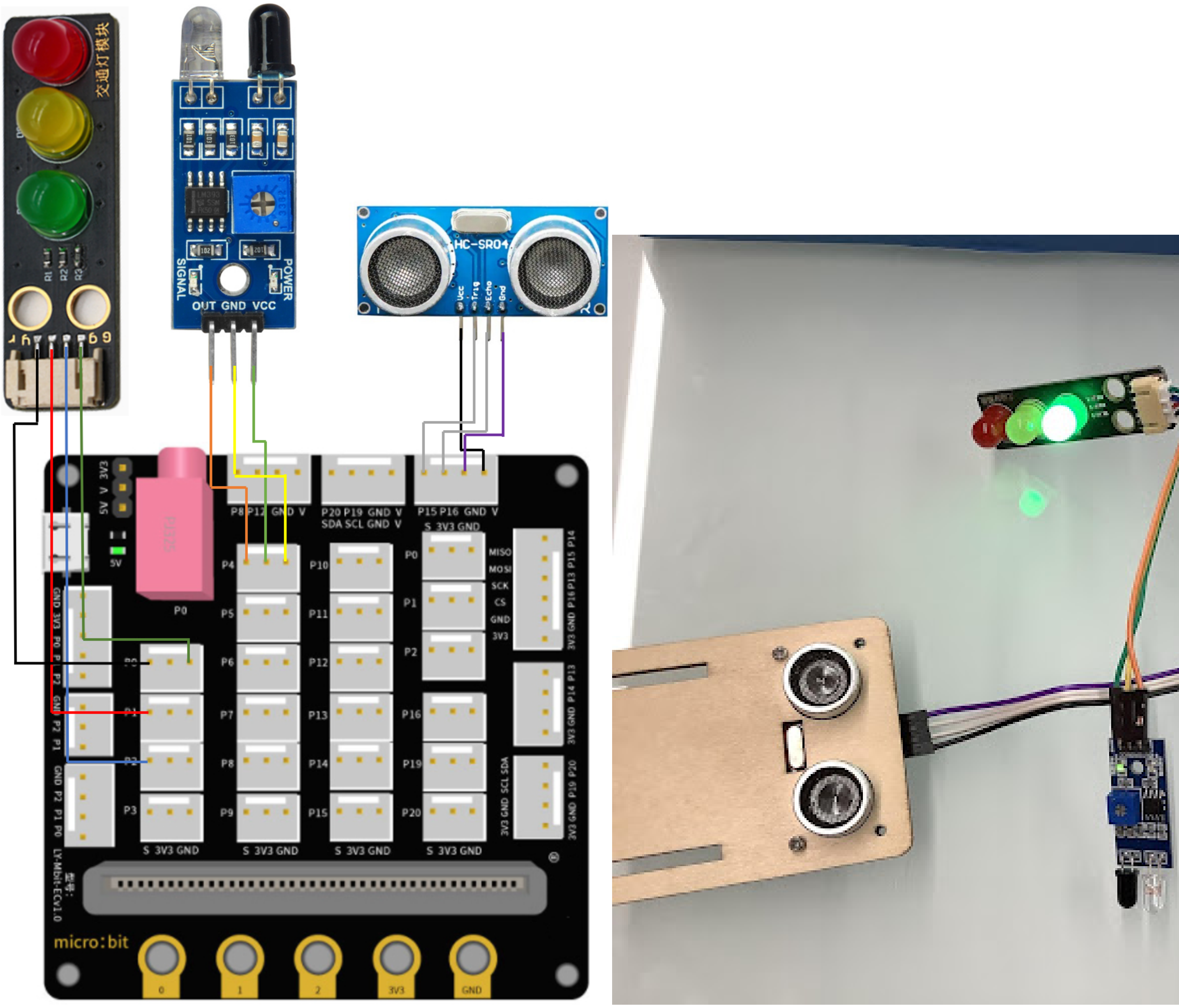
Step 2



Step 3



Hardware connect



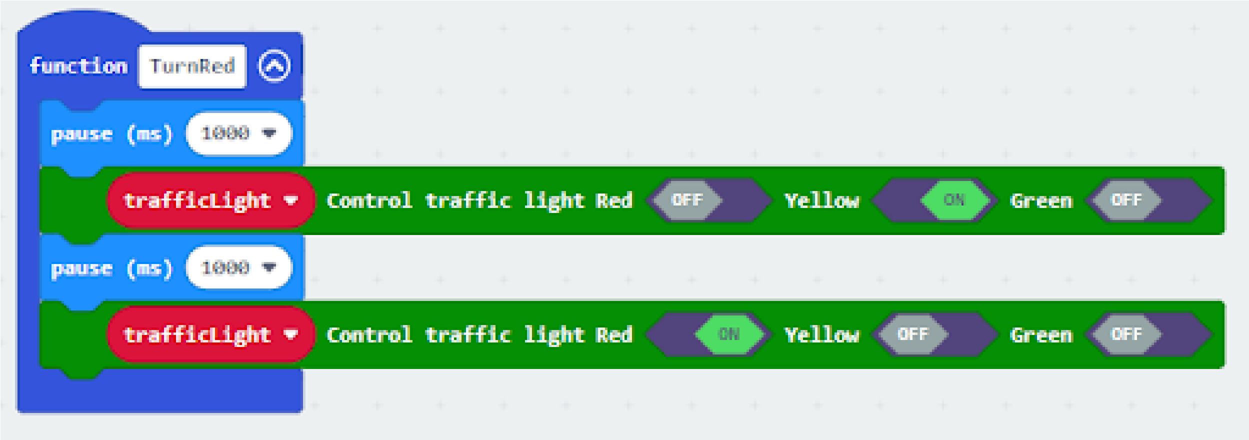
Programming (MakeCode)

Sender

Step 1. Set up a new function (TurnRed)

- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second

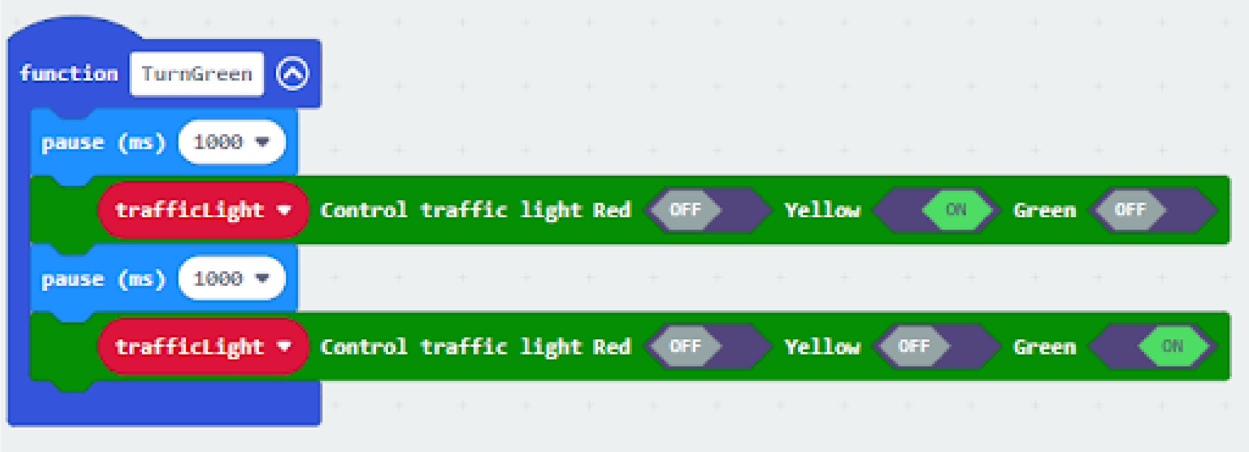
- Control traffic light red on



```
function TurnRed
  pause (ms) 1000
  trafficLight Control traffic light Red OFF Yellow ON Green OFF
  pause (ms) 1000
  trafficLight Control traffic light Red ON Yellow OFF Green OFF
```

Step 2. Set up a new function (TurnGreen)

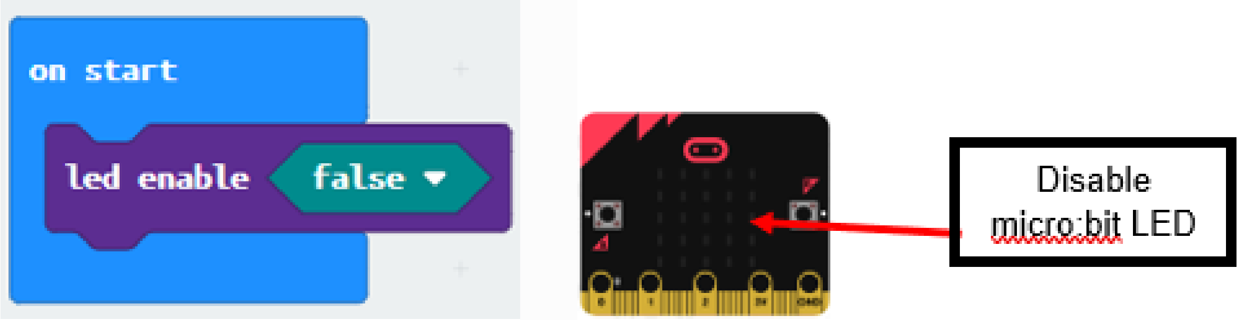
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light green on



```
function TurnGreen
  pause (ms) 1000
  trafficLight Control traffic light Red OFF Yellow ON Green OFF
  pause (ms) 1000
  trafficLight Control traffic light Red OFF Yellow OFF Green ON
```

Step 3. Disable micro:bit LED.

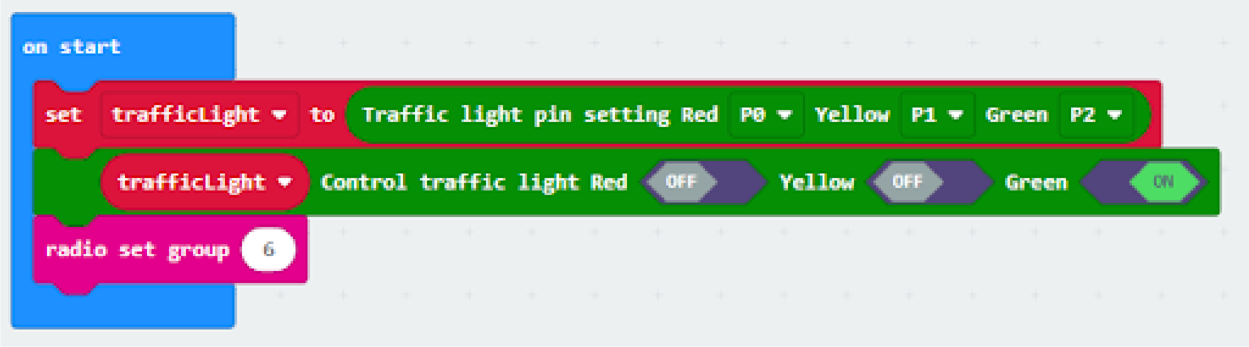
- Snap led enable false to on start
- Note that P3 is used as LED in default setting, LED need to be disable



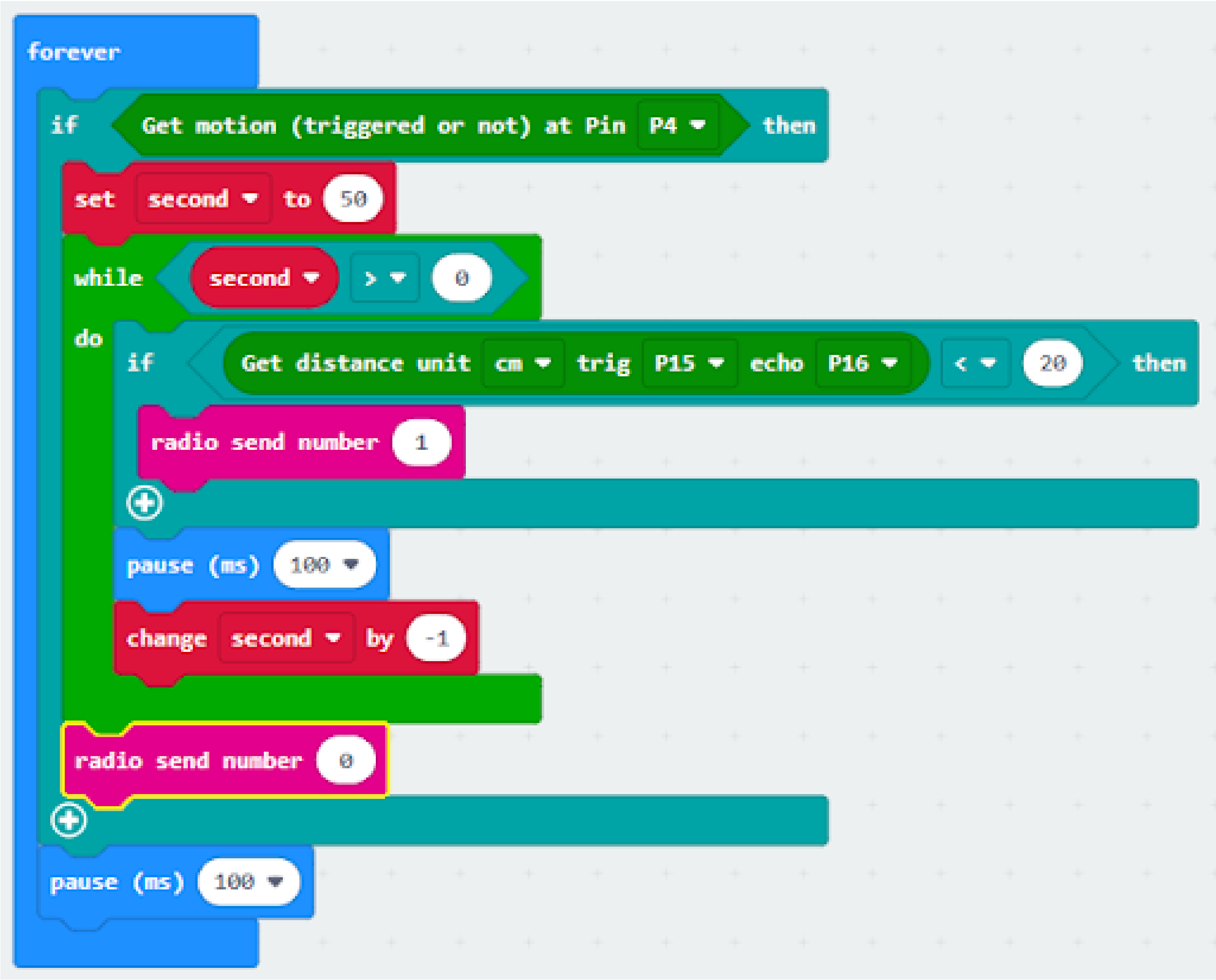
```
on start
    led enable false ▾
```

Step 4. Initialize the program similar as last lesson

- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Control traffic light green on
- Drag radio set group 6 to on start



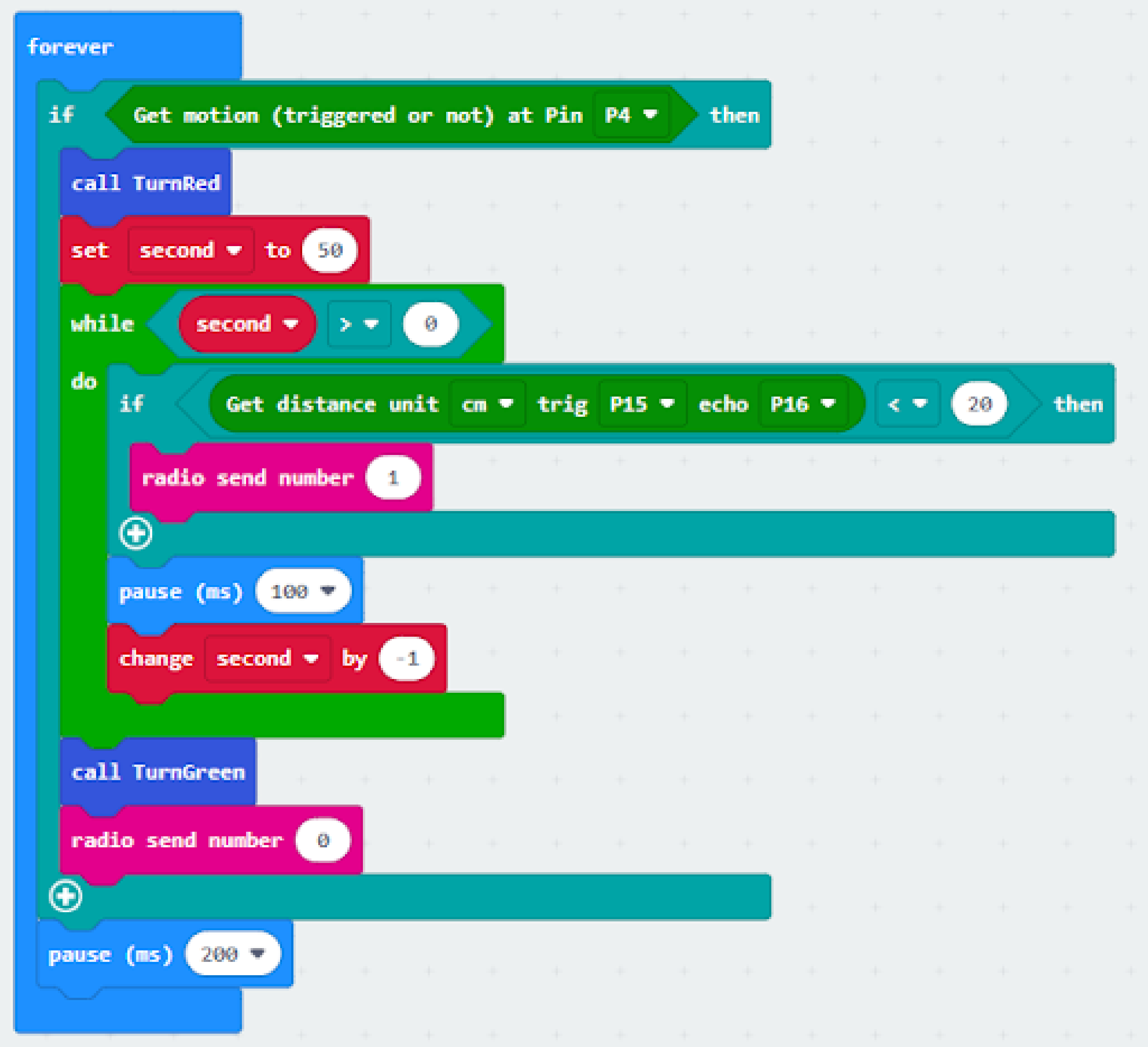
```
on start
    set trafficLight ▾ to Traffic light pin setting Red P0 ▾ Yellow P1 ▾ Green P2 ▾
    trafficLight ▾ Control traffic light Red OFF Yellow OFF Green ON
    radio set group 6
```



```
forever
    if Get motion (triggered or not) at Pin P4 then
        set second to 50
        while second > 0
        do
            if Get distance unit cm trig P15 echo P16 < 20 then
                radio send number 1
            pause (ms) 100
            change second by -1
        radio send number 0
    pause (ms) 100
```

Step 5. Call function

- Snap function TurnRed into if get motion (triggered or not) at pin P4 case
- Drag function TurnGreen after the while loop

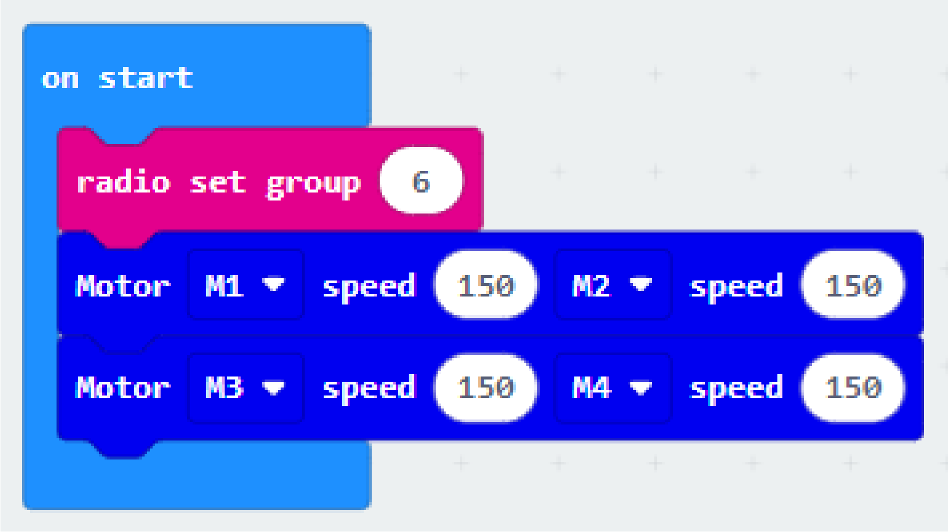


```
forever
    if Get motion (triggered or not) at Pin P4 then
        call TurnRed
        set second to 50
        while second > 0
        do
            if Get distance unit cm trig P15 echo P16 < 20 then
                radio send number 1
            pause (ms) 100
            change second by -1
        call TurnGreen
        radio send number 0
    pause (ms) 200
```

Receiver

Step 1. Set radio set group at start position

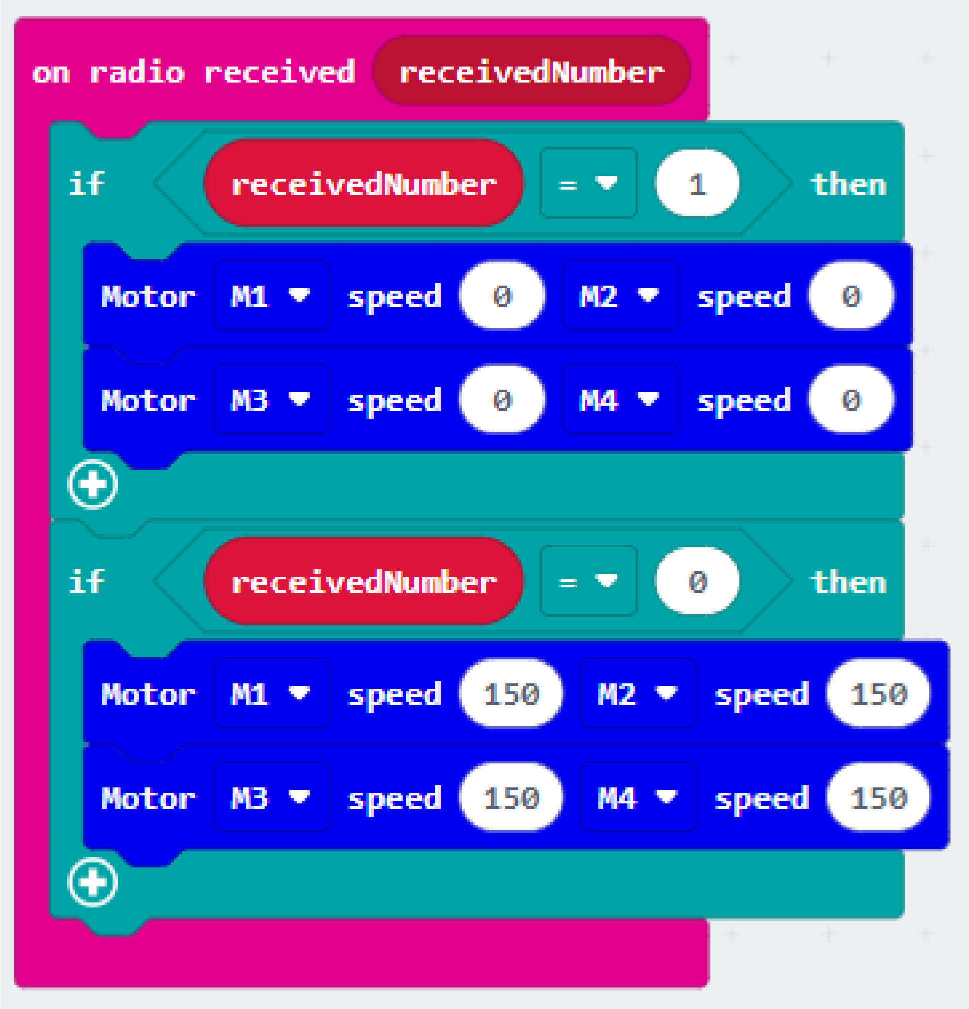
- Drag radio set group 6 to on start
- Initially, the car moves forward by default



```
on start
    radio set group 6
    Motor M1 speed 150 M2 speed 150
    Motor M3 speed 150 M4 speed 150
```

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward



```
on radio received receivedNumber
    if receivedNumber = 1 then
        Motor M1 speed 0 M2 speed 0
        Motor M3 speed 0 M4 speed 0
    if receivedNumber = 0 then
        Motor M1 speed 150 M2 speed 150
        Motor M3 speed 150 M4 speed 150
```

Result

Think

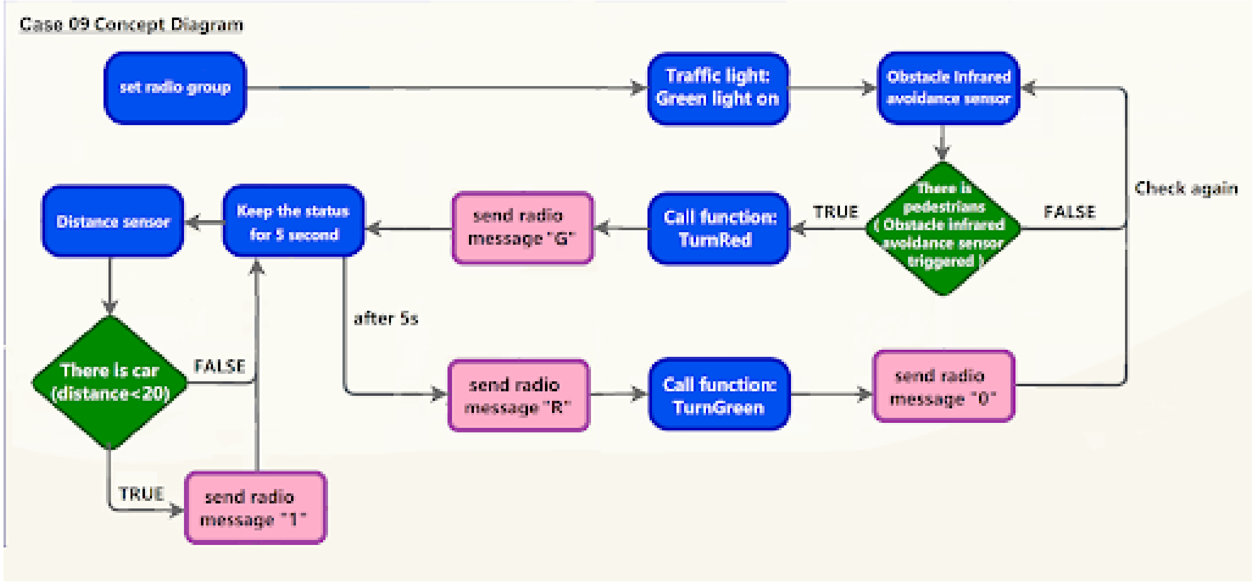
1.2.11 Smart Pedestrian Lights 2

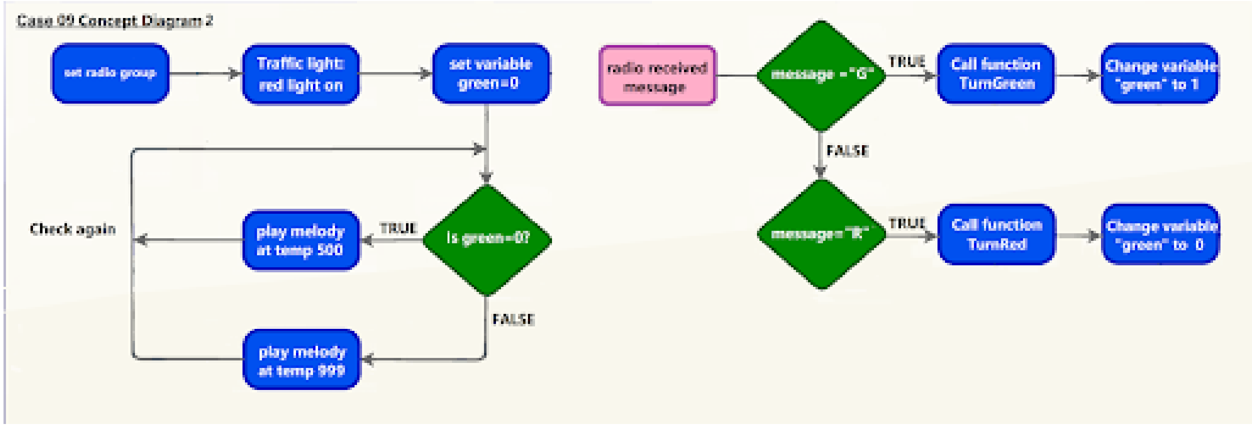
Goal

Background

What is a smart pedestrian light?

Smart traffic light operation

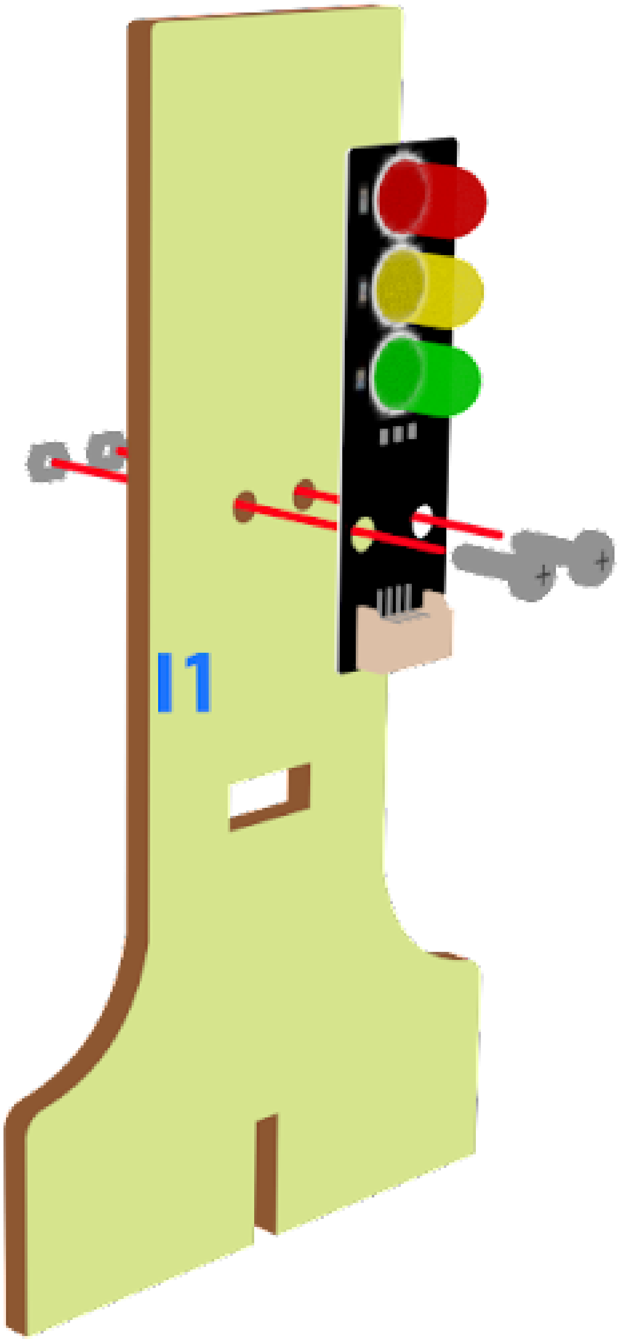




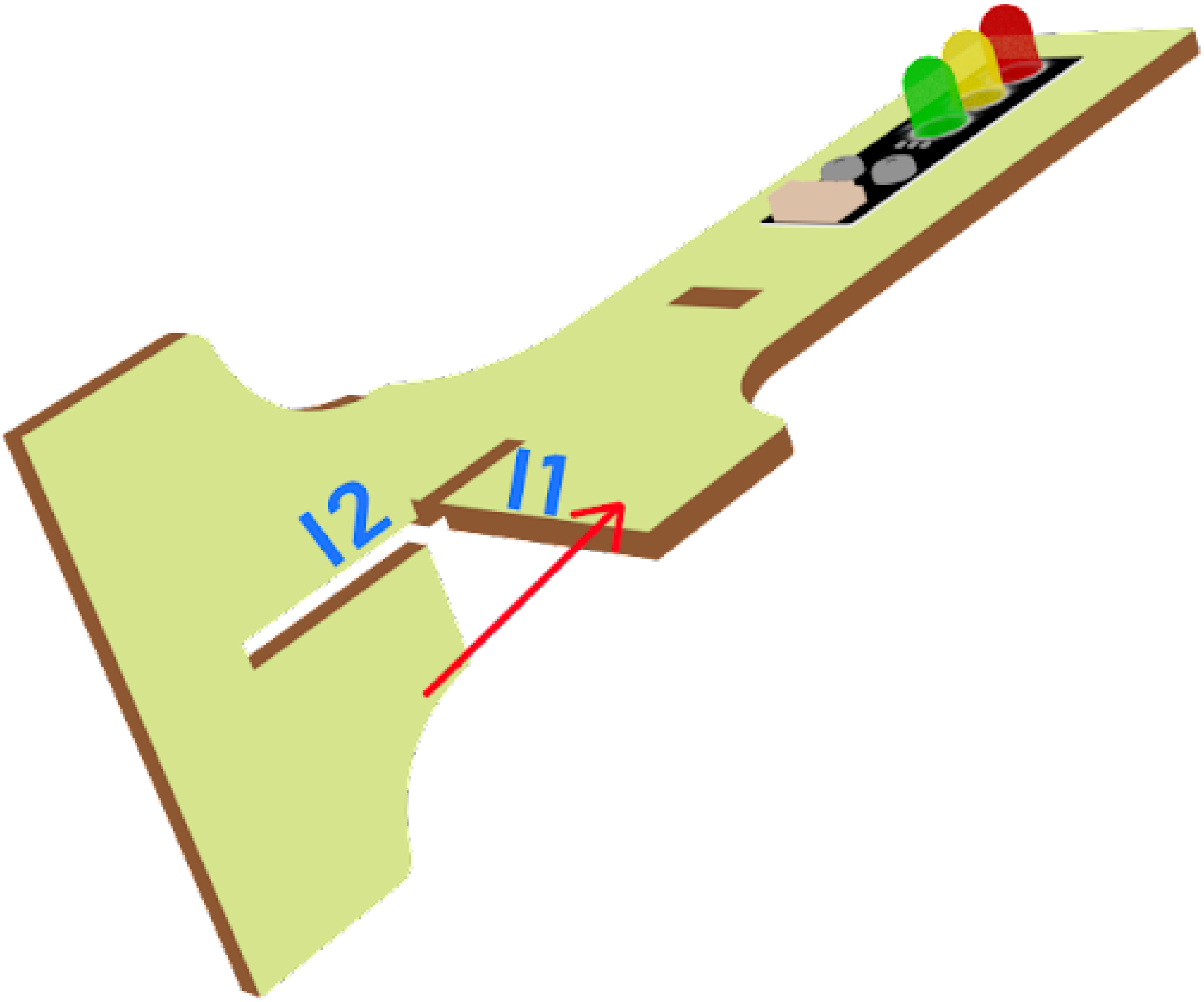
Part List

Assembly step

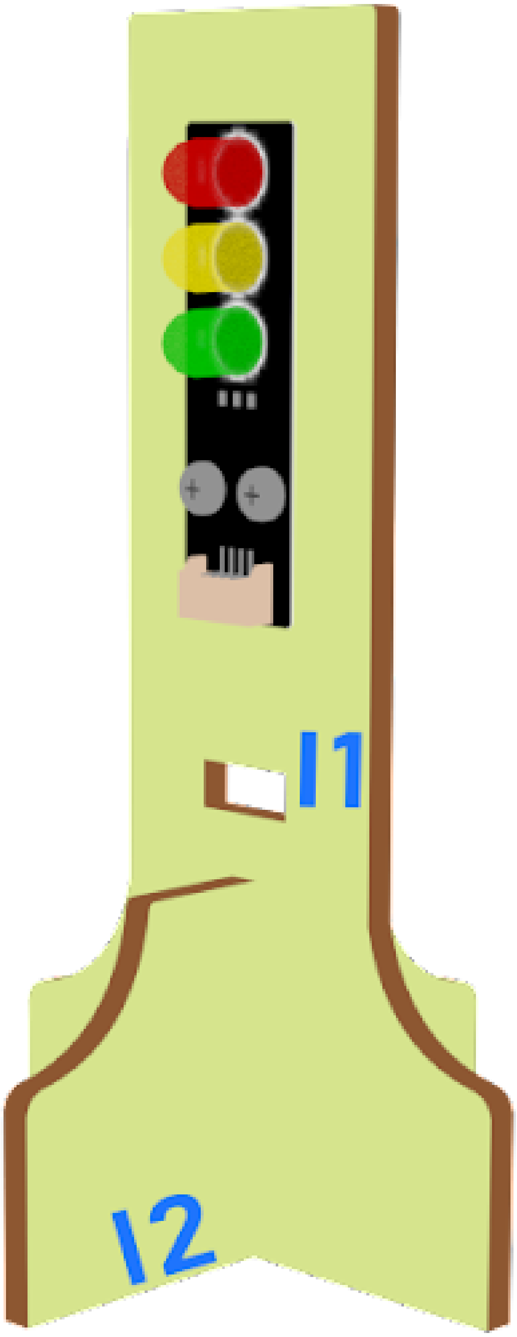
Step 1



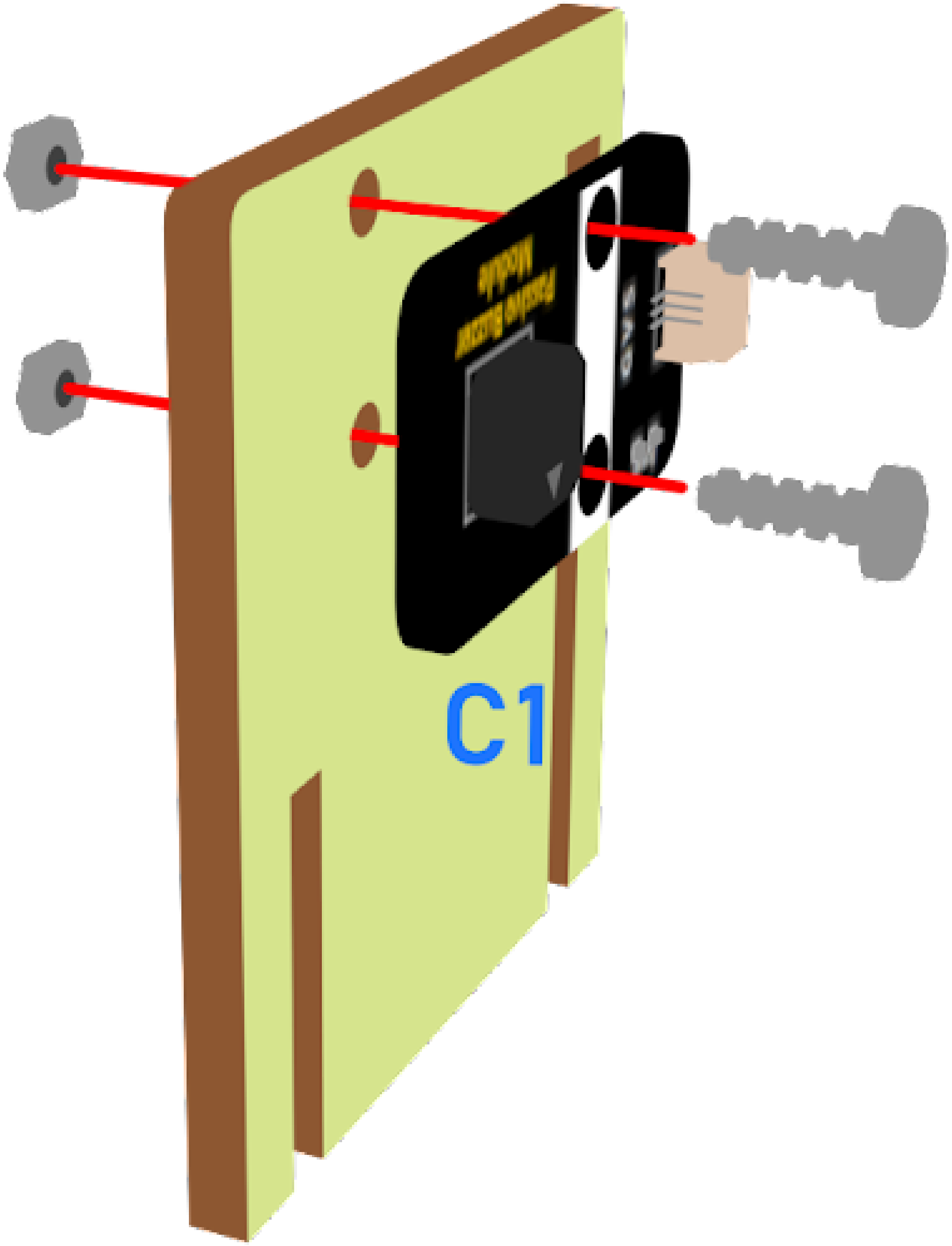
Step 2



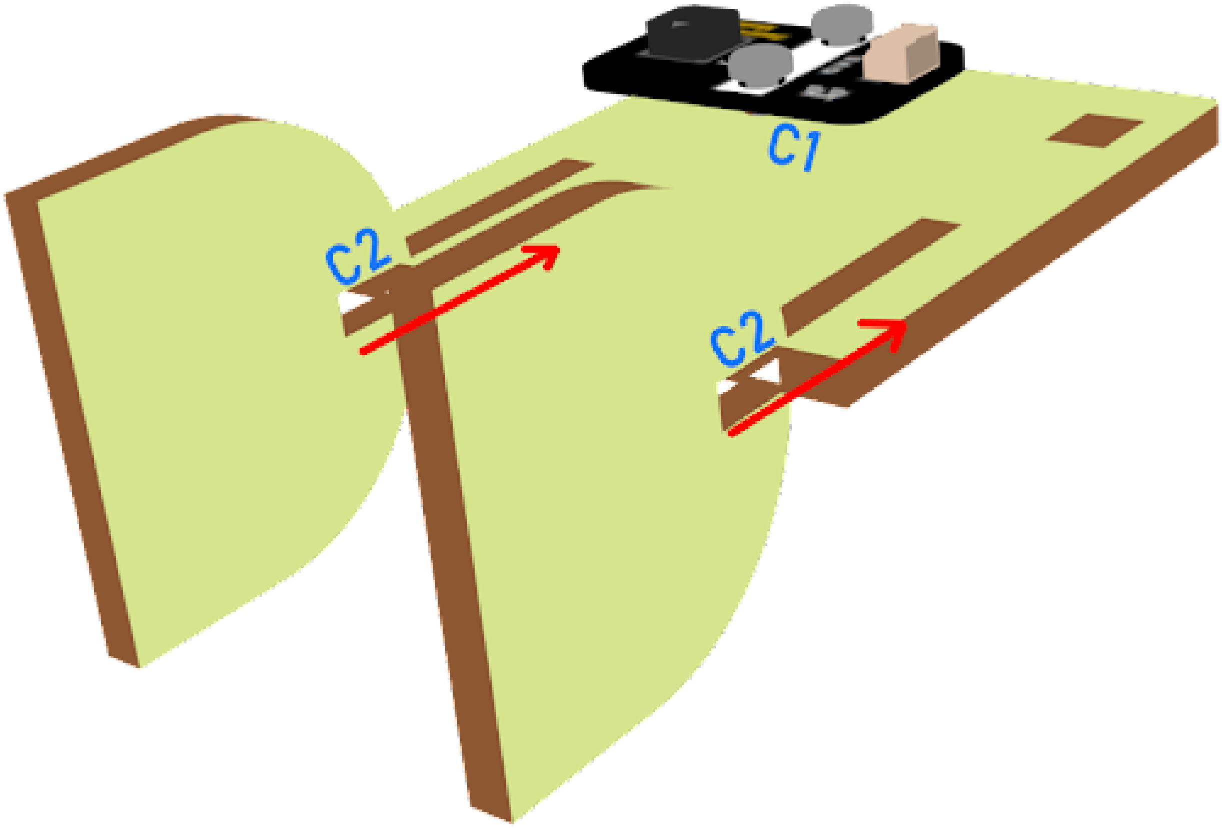
Step 3



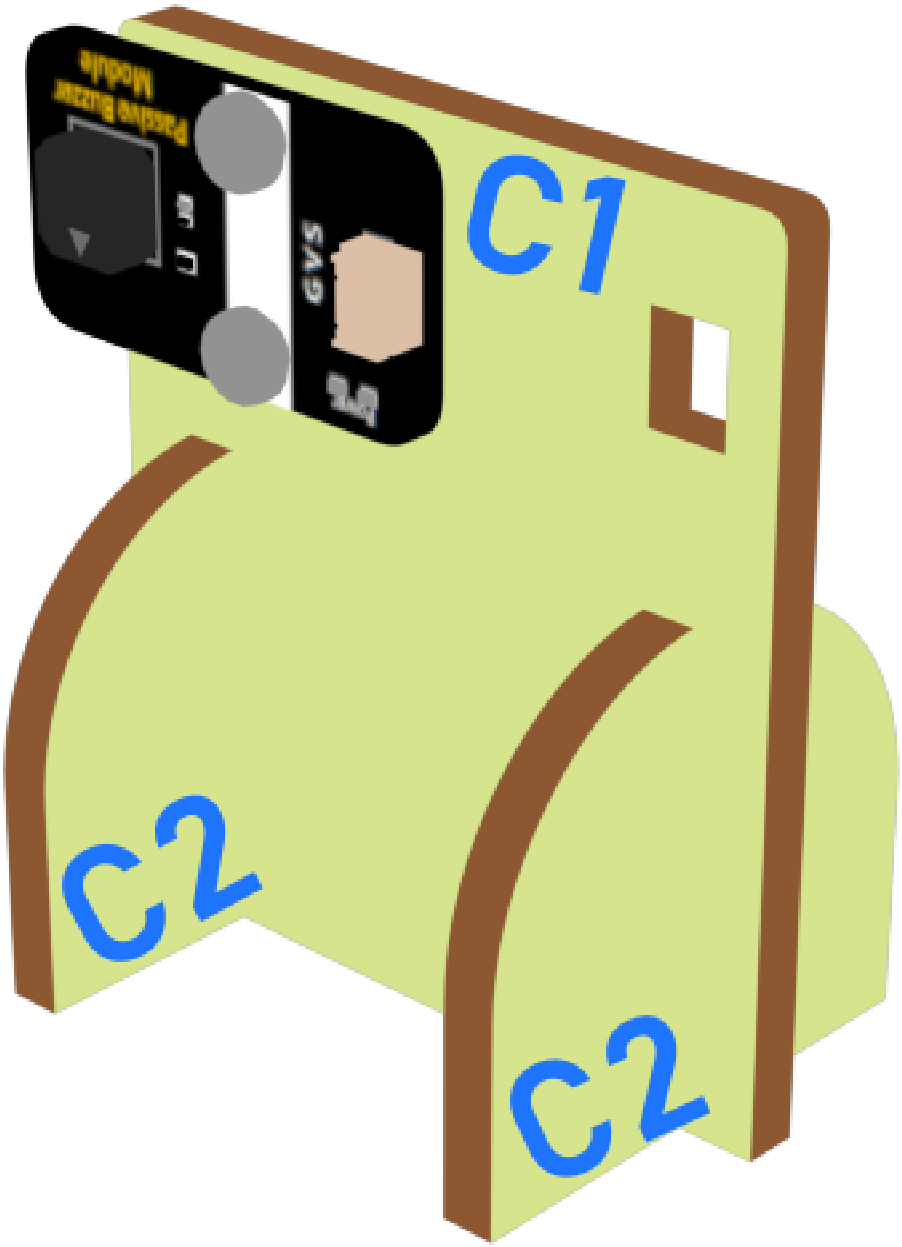
Step 4



Step 5



Step 6



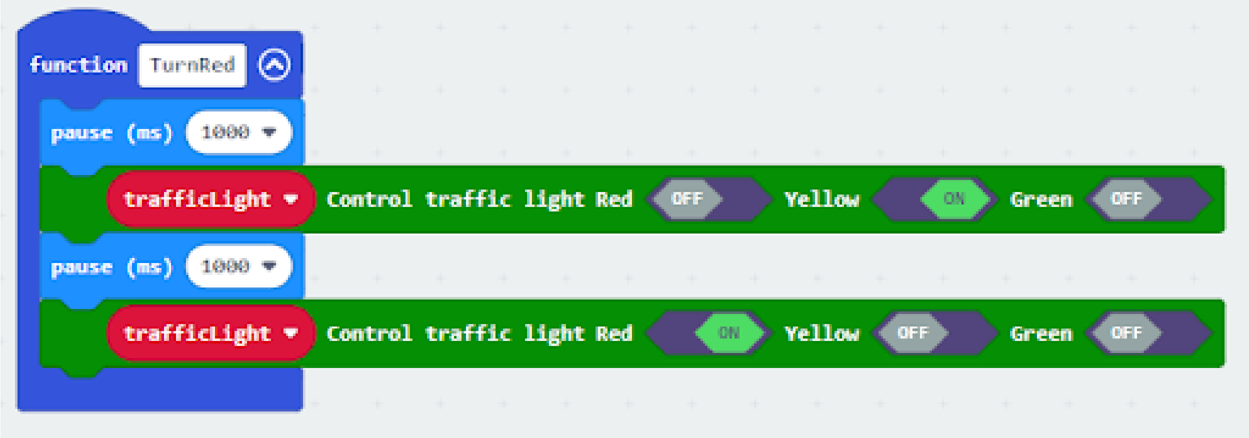
Hardware connect

Programming (MakeCode)

Traffic light 1

Step 1. Set up a new function (TurnRed)

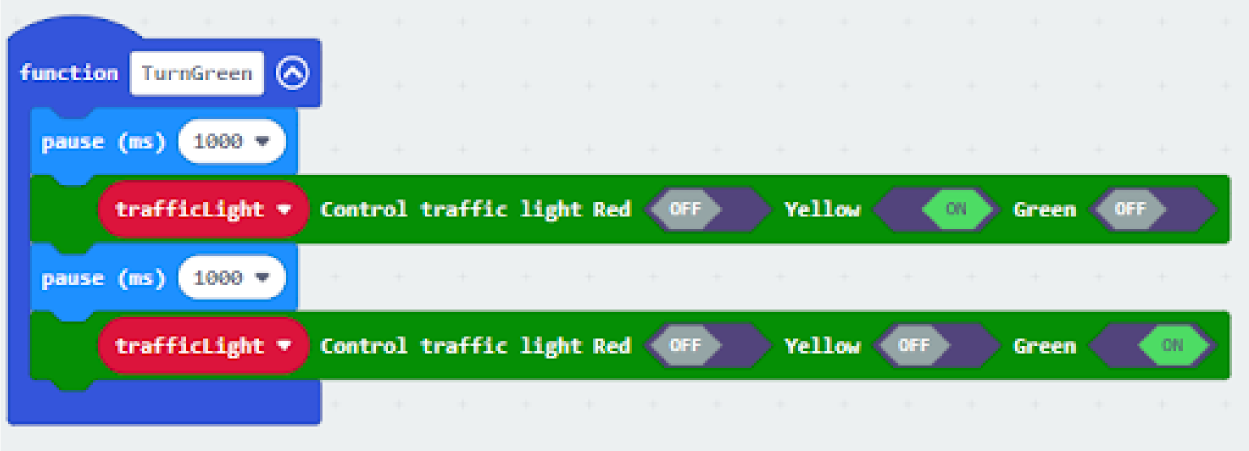
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light red on



```
function TurnRed
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red ON Yellow OFF Green OFF
```

Step 2. Set up a new function (TurnGreen)

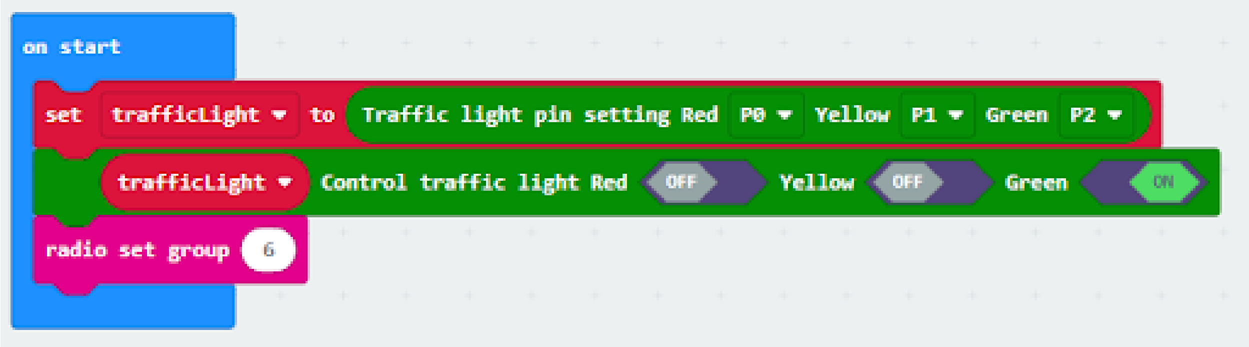
- Snap pause to wait 1 second
- Control traffic light yellow on
- Snap pause to wait 1 second
- Control traffic light green on



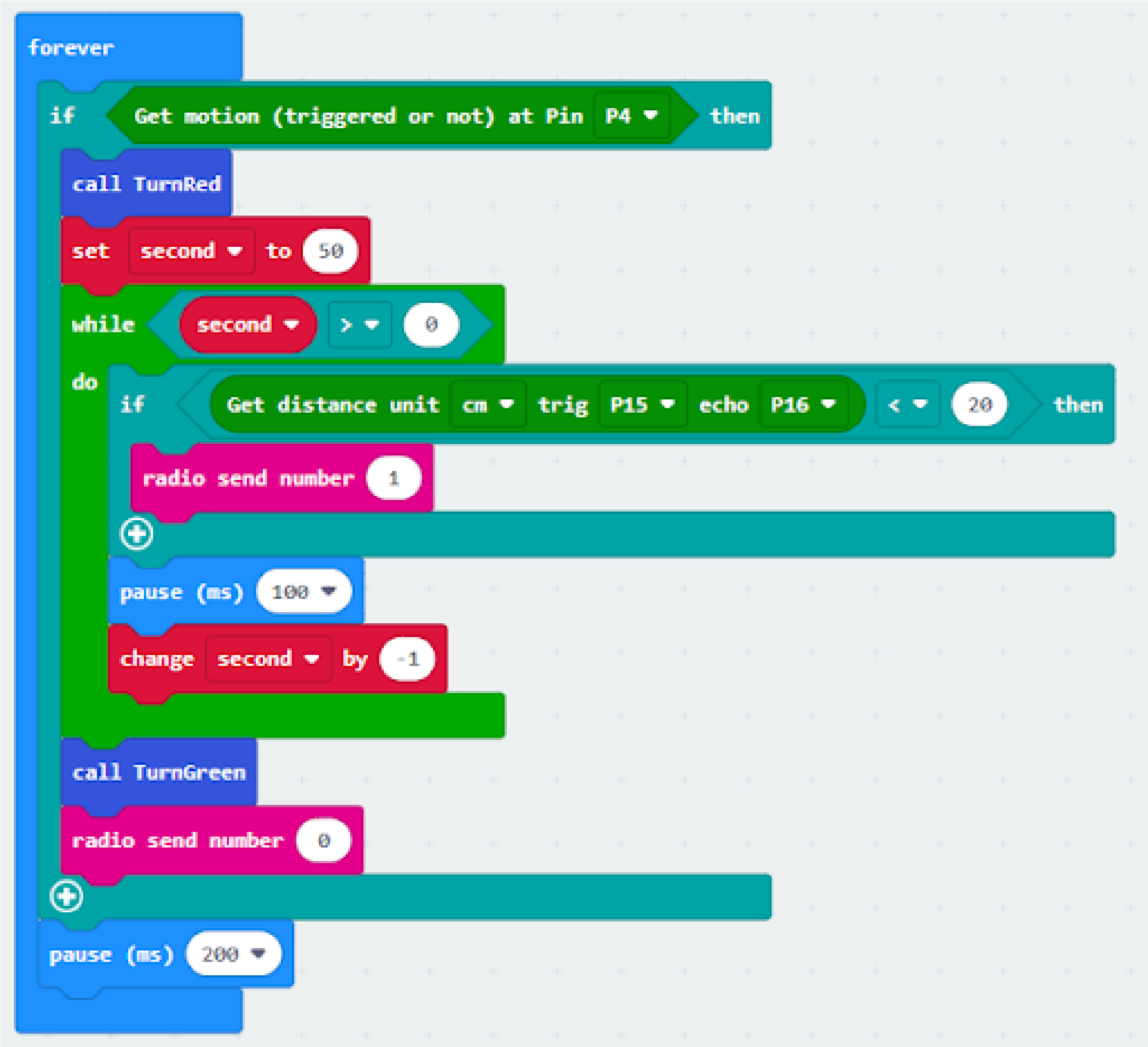
```
function TurnGreen
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow OFF Green ON
```

Step 3. Initialize the program

- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Control traffic light green on
- Drag radio set group 6 to on start
- In forever, snap function TurnRed into if get motion (triggered or not) at pin P4 case
- Drag function TurnGreen after the while loop



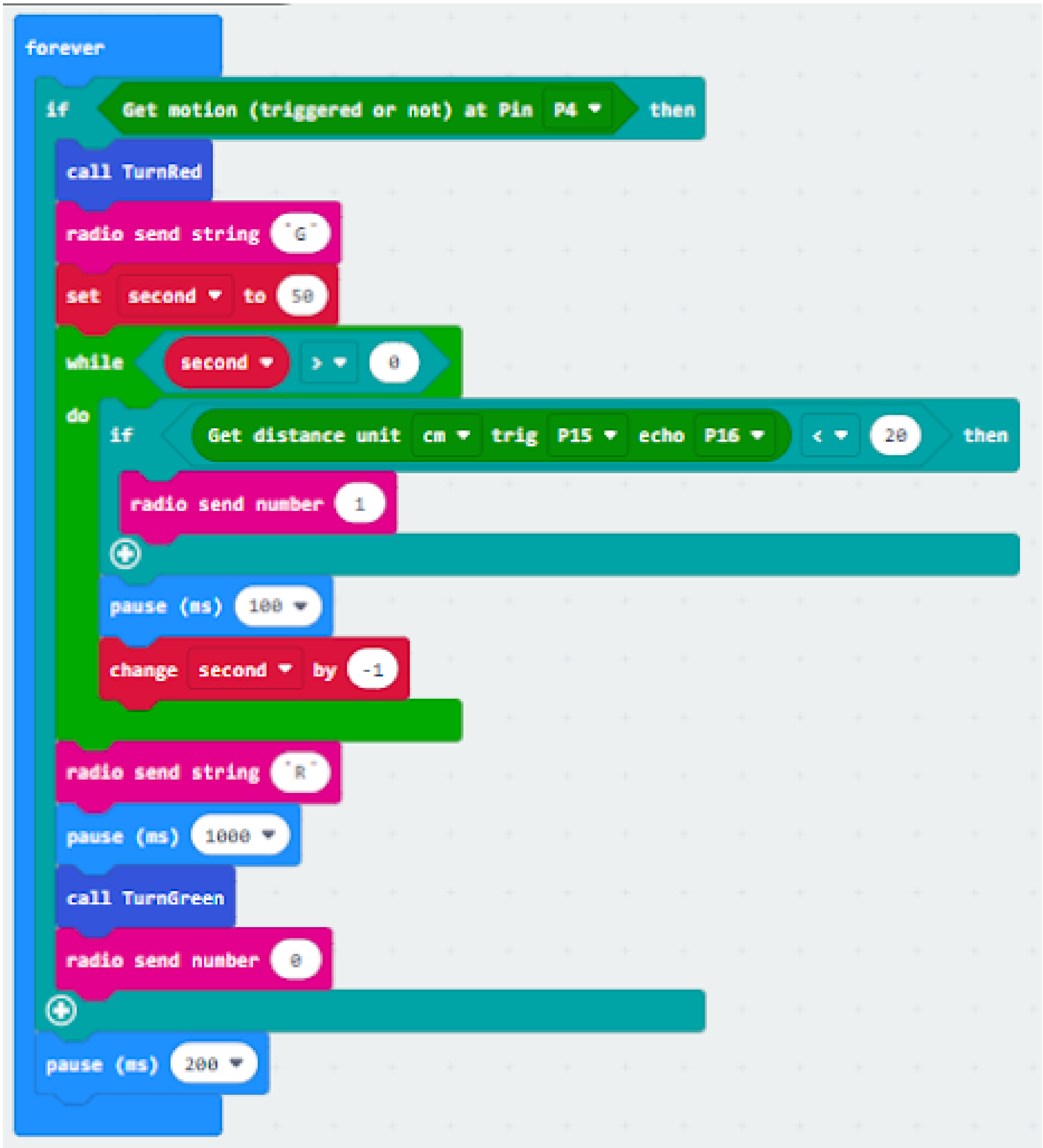
```
on start
  set trafficLight ▾ to Traffic light pin setting Red P0 ▾ Yellow P1 ▾ Green P2 ▾
  trafficLight ▾ Control traffic light Red OFF Yellow OFF Green ON
  radio set group 6
```



```
forever
    if Get motion (triggered or not) at Pin P4 then
        call TurnRed
        set second to 50
        while second > 0
        do
            if Get distance unit cm trig P15 echo P16 < 20 then
                radio send number 1
            ⊕
            pause (ms) 100
            change second by -1
        call TurnGreen
        radio send number 0
    ⊕
    pause (ms) 200
```

Step 4. Control traffic light 2 by sending radio string

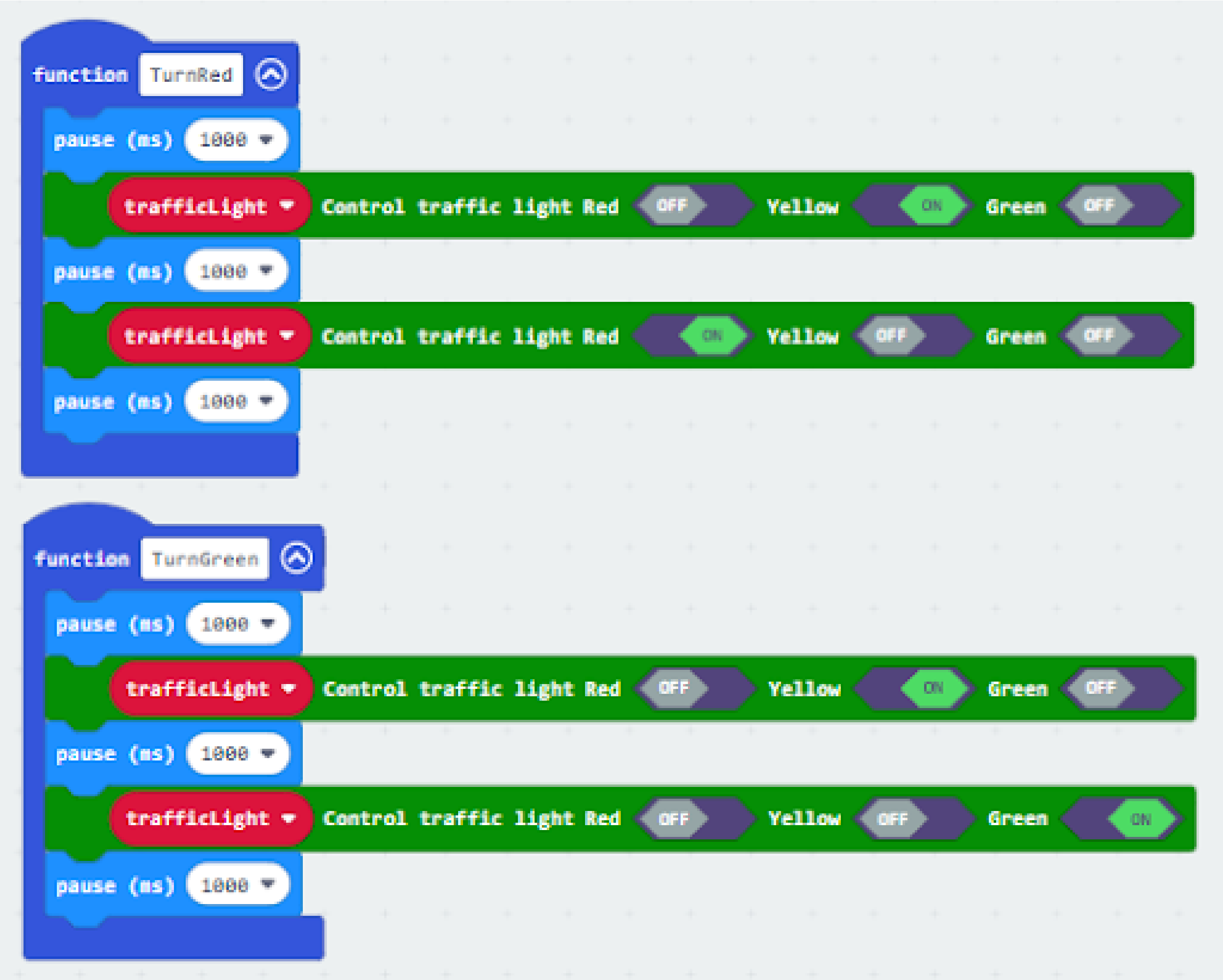
- Drag radio send string “R” before TurnGreen
- Drag radio send string “G” after TurnRed



```
forever
  if Get motion (triggered or not) at Pin P4 then
    call TurnRed
    radio send string "G"
    set second to 50
    while second > 0
    do
      if Get distance unit cm trig P15 echo P16 < 20 then
        radio send number 1
      ⊕
      pause (ms) 100
      change second by -1
    radio send string "R"
    pause (ms) 1000
    call TurnGreen
    radio send number 0
  ⊕
  pause (ms) 200
```

Traffic light 2

Step 1. Set up new functions

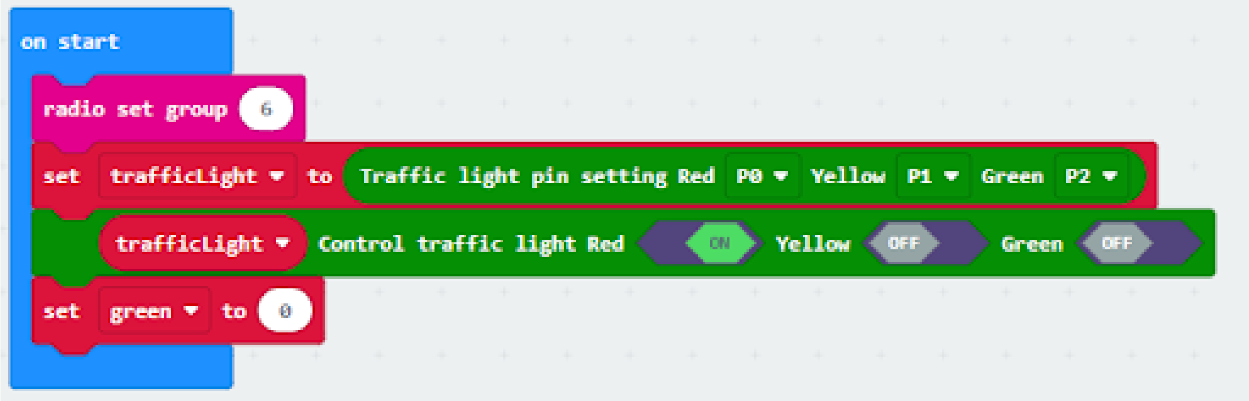


```
function TurnRed
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red ON Yellow OFF Green OFF
    pause (ms) 1000

function TurnGreen
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow ON Green OFF
    pause (ms) 1000
    trafficLight Control traffic light Red OFF Yellow OFF Green ON
    pause (ms) 1000
```

Step 2. Initialize the program

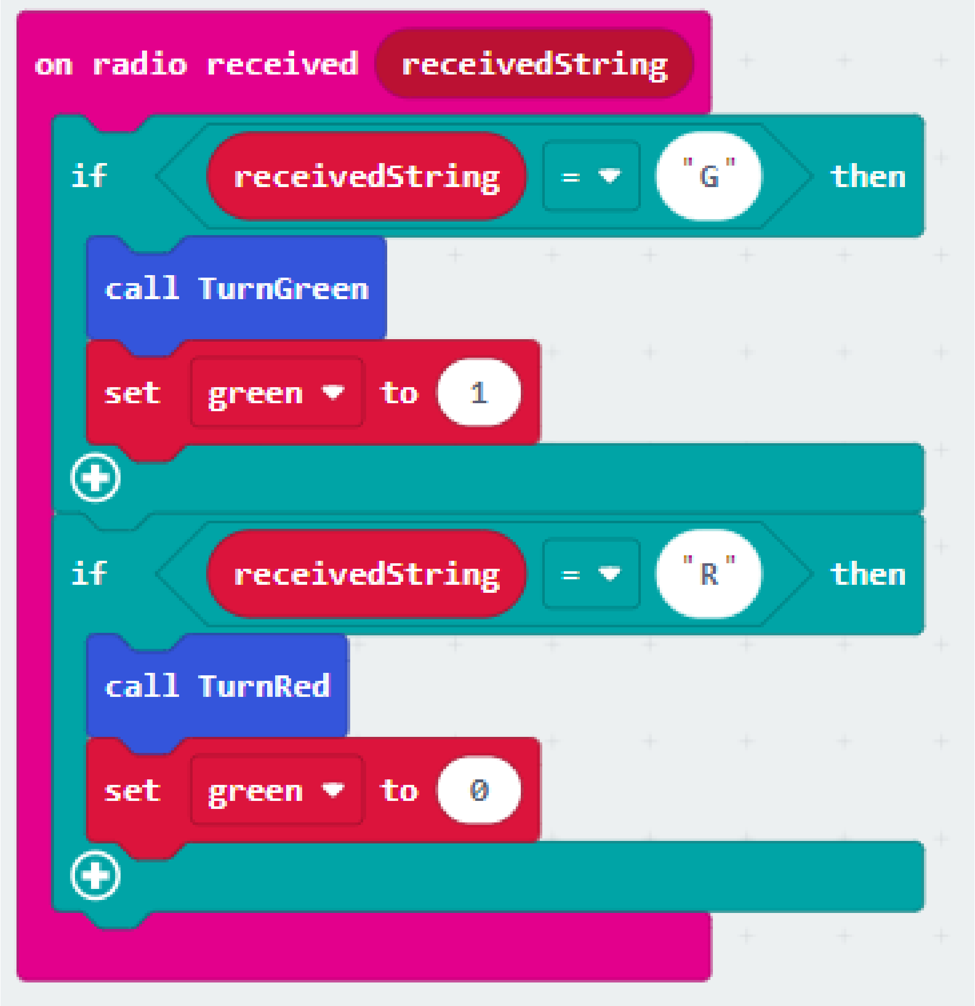
- Drag set variable trafficLight to Traffic light pin setting Red P0 Yellow P1 Green P2 to on start
- Drag radio set group 6 to on start
- Control traffic light green on
- Set a variable green=0



```
on start
    radio set group 6
    set trafficLight ▾ to Traffic light pin setting Red P0 ▾ Yellow P1 ▾ Green P2 ▾
    trafficLight ▾ Control traffic light Red ON Yellow OFF Green OFF
    set green ▾ to 0
```

Step 3. Control traffic light by receiving different number

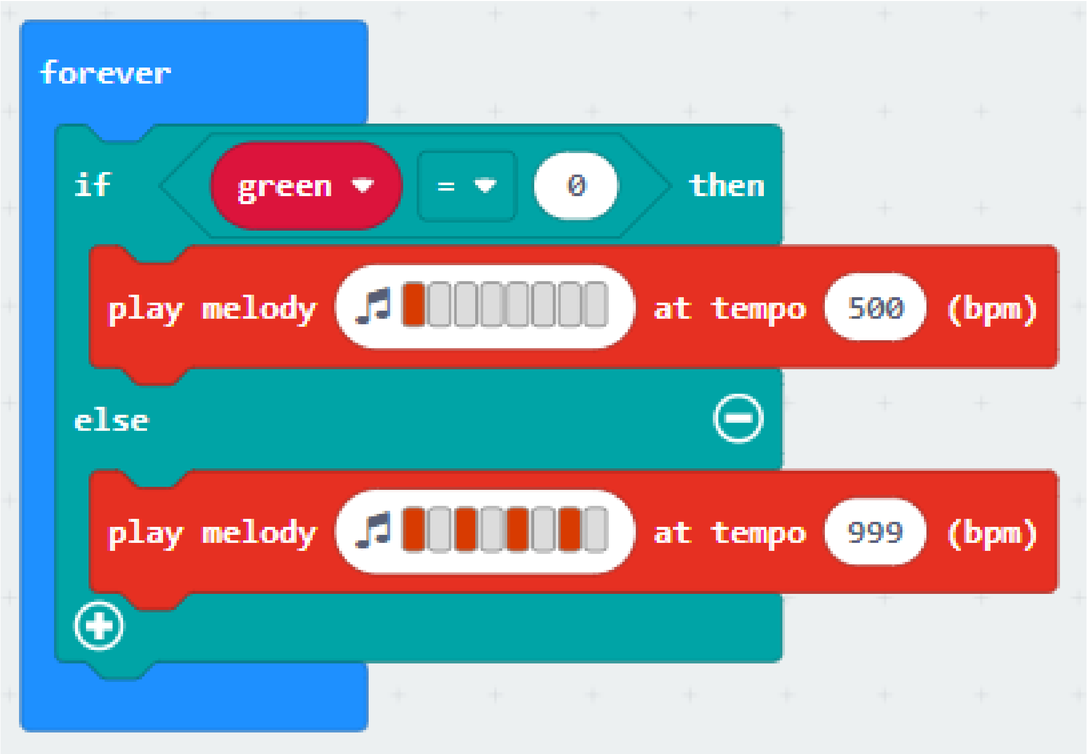
- Snap if statement into on radio received receivedString
- Set receivedString =“R” and call TurnRed
- Set receivedString =“G” and call TurnGreen
- Change variable green depend on the light



```
on radio received receivedString
    if receivedString = "G" then
        call TurnGreen
        set green to 1
    if receivedString = "R" then
        call TurnRed
        set green to 0
```

Step 4. Play sound effect depend on the light status

- Snap if statement into forever
- Play melody with different tempo

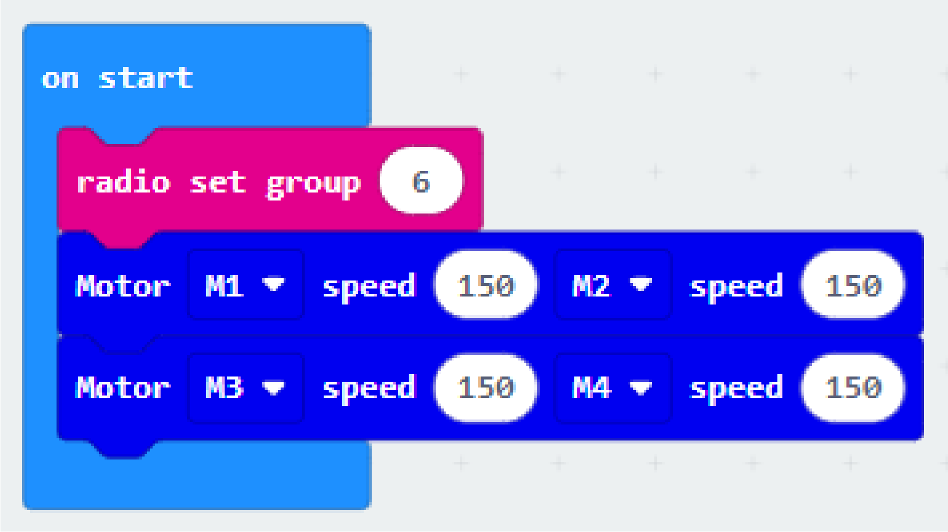


```
forever
    if green = 0 then
        play melody ♬ at tempo 500 (bpm)
    else
        play melody ♬ at tempo 999 (bpm)
```

Receiver

Step 1. Set radio set group at start position

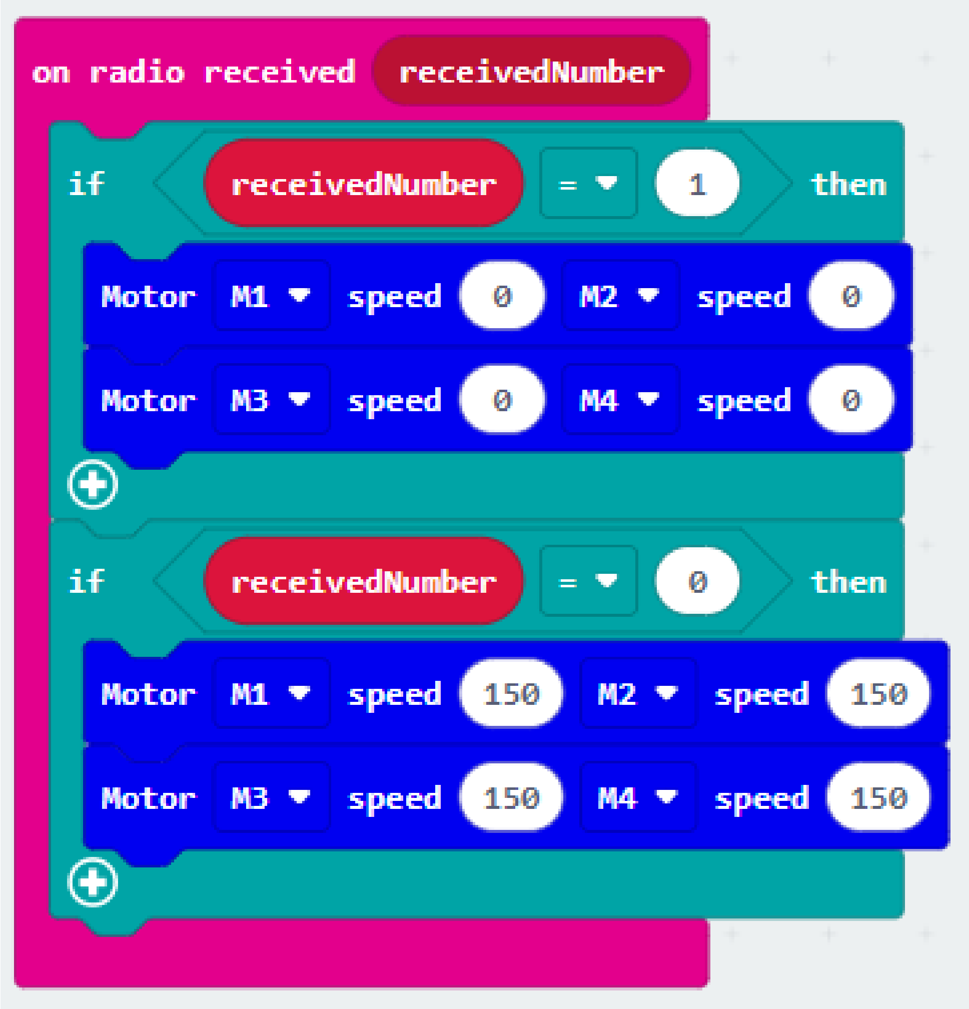
- Drag radio set group 6 to on start
- Initially, the car moves forward by default



```
on start
    radio set group 6
    Motor M1 speed 150 M2 speed 150
    Motor M3 speed 150 M4 speed 150
```

Step 2. Control car by receiving different number

- Snap if statement into on radio received receivedNumber
- Set receivedNumber =1 and make the car stop
- Set receivedNumber=0 and make the car move forward



```
on radio received receivedNumber
    if receivedNumber = 1 then
        Motor M1 speed 0 M2 speed 0
        Motor M3 speed 0 M4 speed 0
    if receivedNumber = 0 then
        Motor M1 speed 150 M2 speed 150
        Motor M3 speed 150 M4 speed 150
```

Result

Think