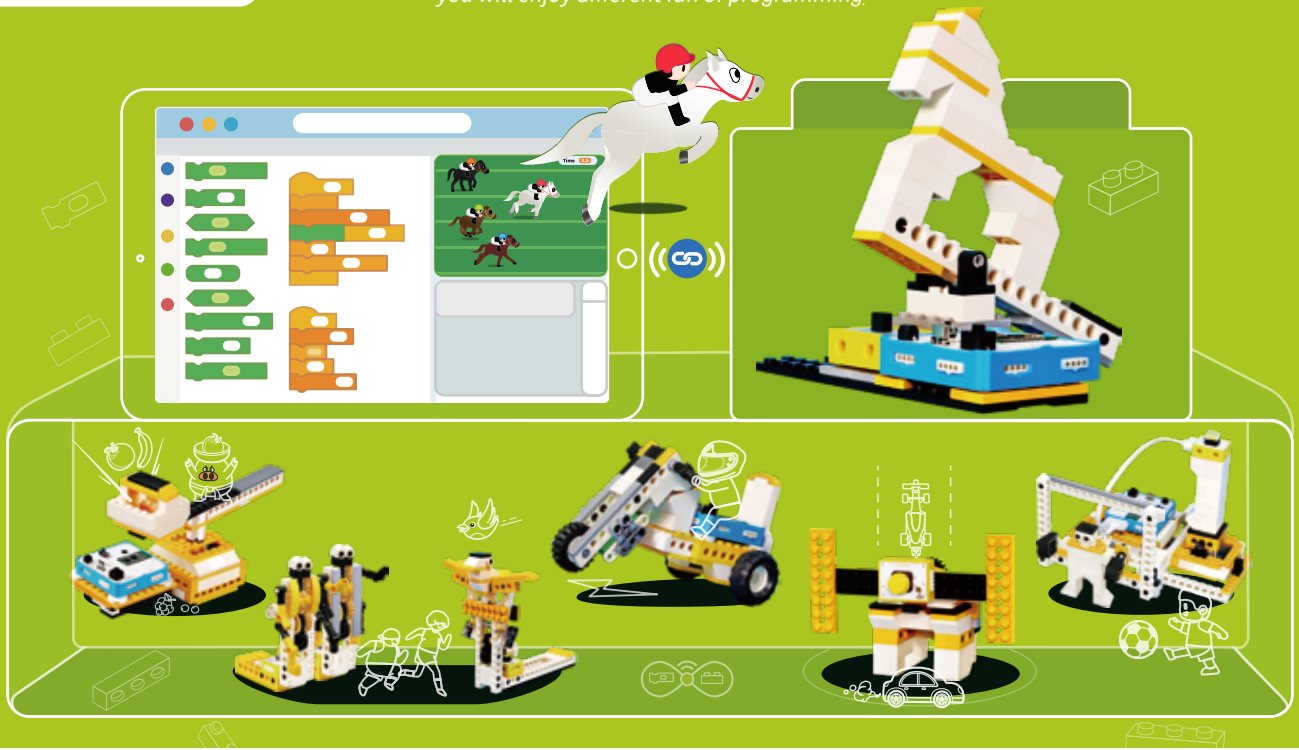
Crowbits
by Elecrow

Creator Kit

Combining hardware (sensor) and games.
you will enjoy different fun of programming.



USER MANUAL V1.1

www.elecrow.com

Directory

About Crowbits

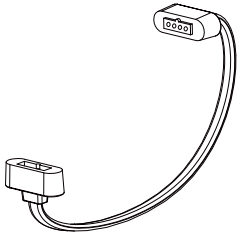
Crowbits is a STEM educational equipment specially designed for young technology enthusiasts. The equipment includes three mainstream main control boards (Micro:bit, Arduino, ESP32) and hundreds of electronic sensors and modules. The modules are perfectly packaged and connected by magnet, truly "plug and play". Standard Lego holes are designed on the bottom and sides of the module, which can be perfectly connected with Lego. Modules can realize programming-free and programming functions at the same time, kids can quickly get started, stimulate their creative ability.

"Creator Kit" is the primary programming kit of the Crowbits series. It uses Arduino as the mainboard console, combines electronic modules and Letscode which is a visual graphics programming software. Through simple drag-and-drop programming, kids can create many interactive game projects, which greatly lowers the threshold of artificial intelligence learning. "Creator Kit" encourages kids to create games actively and design animations. There is also code programming such as Arduino and Python for you to explore. In this way, kids will keep their interest in learning and improve logical thinking ability continuously.

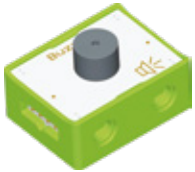
- **Modules List of Creator Kit**

RGB Matrix x1

Crowbits-UNO x1



Magnetic Cable x3



Buzzer x1

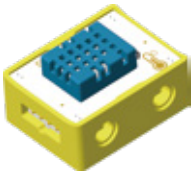
LED Bar x1

Vibration Motor x1

LED x1

IR Reflective x1

Button x1



DHT11 Sensor x1

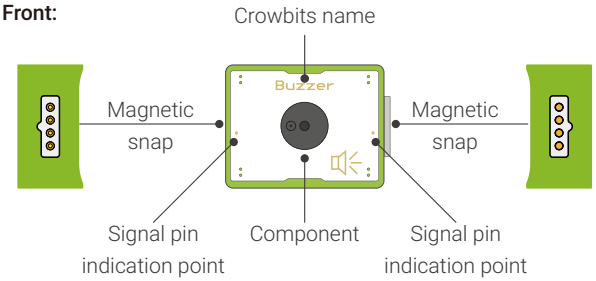
PIR Sensor x1

Linear Potentiometer x1

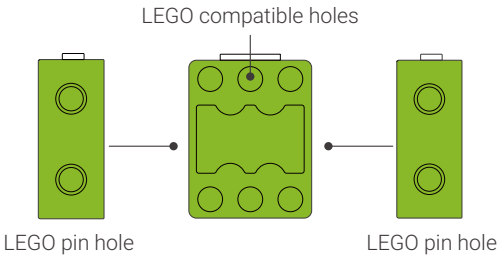
• Get to Know Crowbits modules

Crowbits module structure (take the buzzer as an example):

Front:



Back:



Crowbits name: Crowbits consists many types of modules, so we marked the name for you to recognize easily.

Component: The main component of Crowbits is placed there.

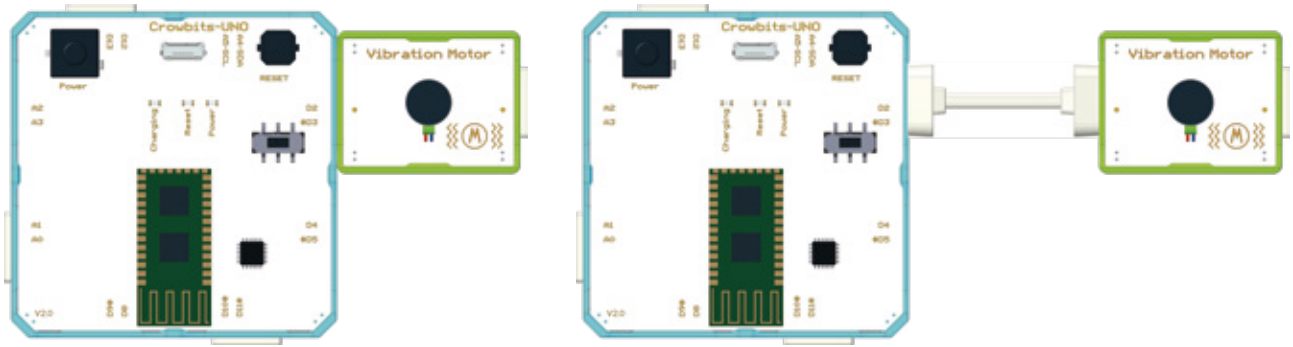
Magnetic snap: It's a magical magnetic pogo-pin, and the magnetic snap allows you to connect Crowbits together with magnet.

Signal pin indication point: This point represents the position of the signal input/output pins of the module. When connected to the main console or other modules, only the pins at this point can control the module. The main console uses the pin number to label the corresponding input/output pins.

A number of Lego holes are reserved on the back and side of Crowbits electronic modules, which are perfectly compatible with Lego rods or pins, and the connection is extremely convenient.

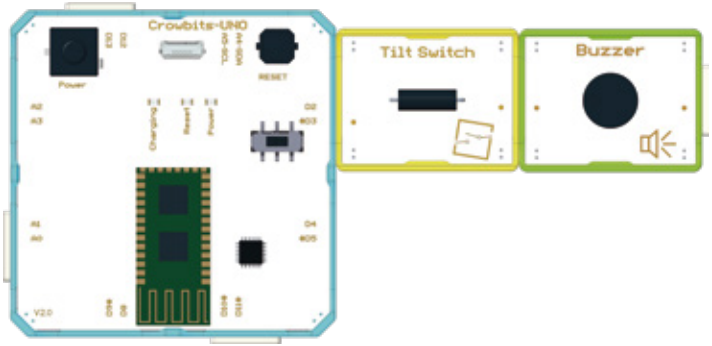
• Connection method

The modules can be directly connected by magnet or connected by magnetic cables.

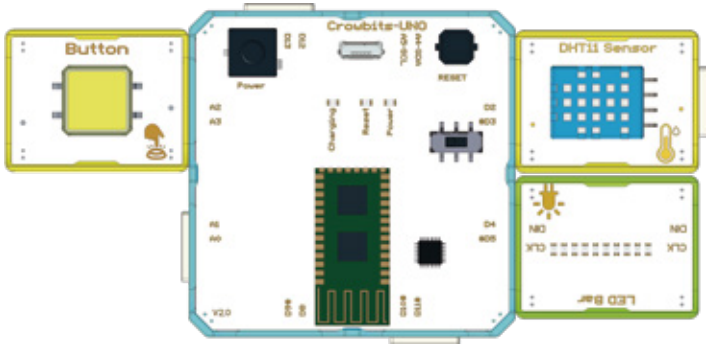


• Connection rules

The programming-free mode of the modules must include the power supply (the mainboard console expansion board can also be used as a power supply), input module and output module, and the output module must be connected to the right side of the input module to ensure that the output module is controlled by the input module.



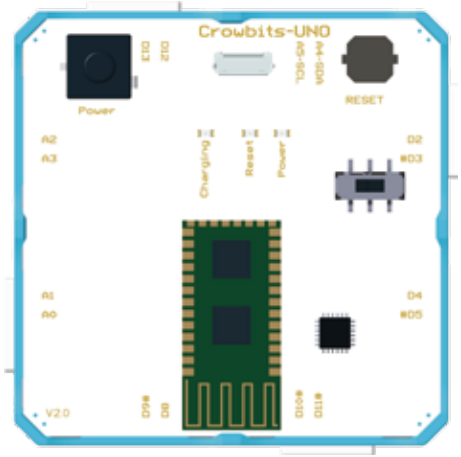
The programming mode needs to download the program to the Crowbits-UNO mainboard console. The mainboard console can control the input/output module through the signal pins, which can be connected at will, but it cannot be cross-connected.



• Crowbits-UNO

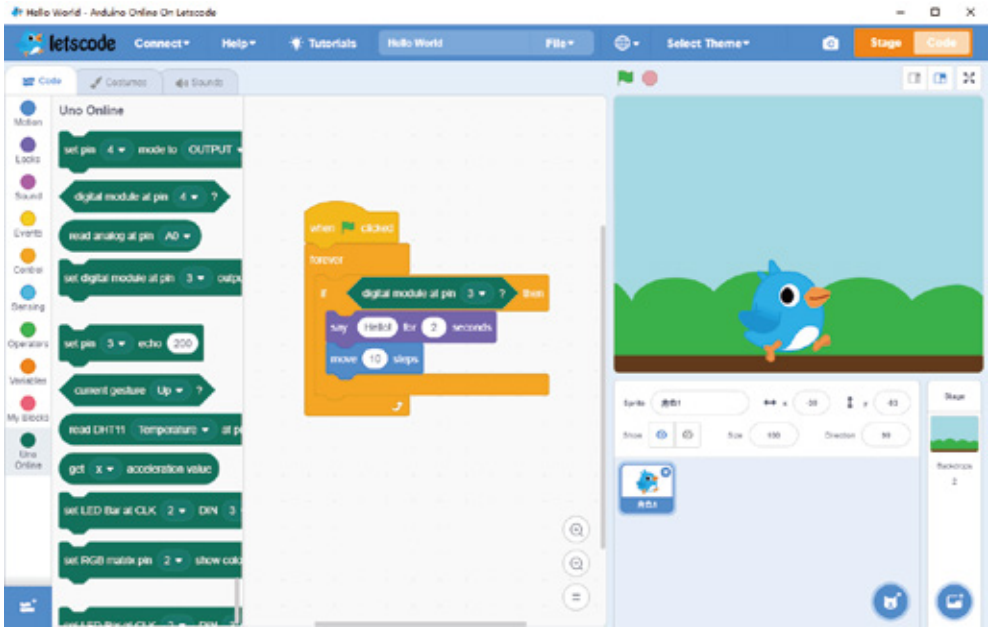
The mainboard of Crowbits-UNO is Arduino, and it integrates a Bluetooth module, six-axis gyroscope and other functions. Arduino is a convenient, flexible, and easy-to-use open source electronic prototyping platform. It can perfectly realize signal output and input control, and quickly design your creative works. We have also developed customized Letscode software and extension modules for the Crowbits series. Users can connect with the software via Bluetooth to realize the wireless software and hardware interaction function.

Turn on the power switch when using.



Letscode Software Introduction

Letscode is graphical programming software which created mainly for the global needs of science and technology innovation education. With letscode, we can code the computer programs by only dragging the code blocks with the mouse. It can help you to learn the program of game design and robot control easily. Here are some highlighted functions of Letscode.

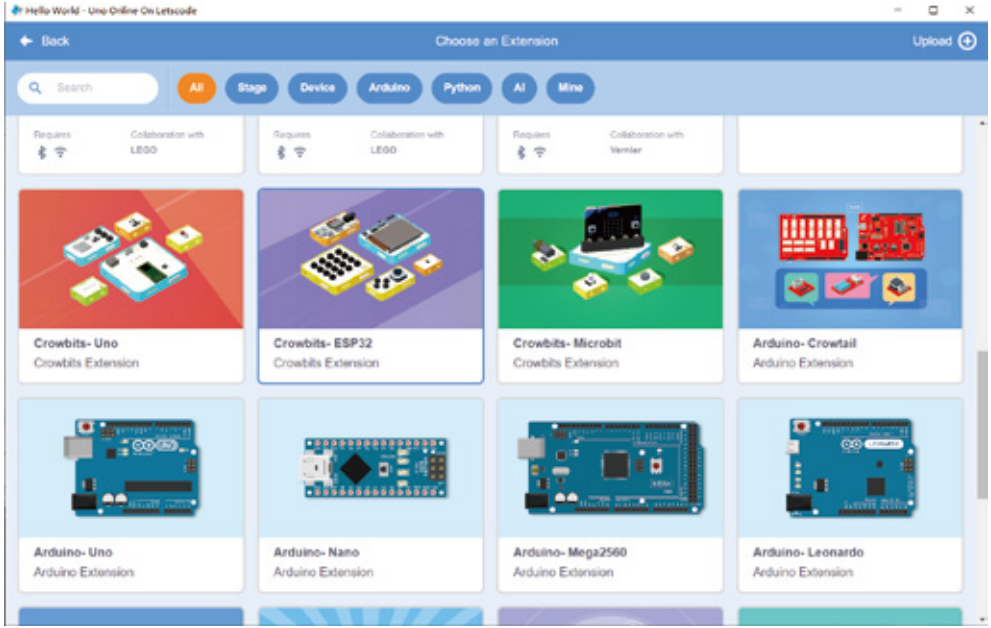


1. Well-compatible with various hardware devices like WeDo, LEGO, Arduino, Micro:bit, ESP32, DevkitC and so on.

2. Letscode can program and control hundreds of hardware modules like sensors, actuators, displays, communication modules and other functional modules.

3. Letscode also supports AI and IoT learning. With open-source and abundant user libraries, users also can create extended libraries for themselves.

4. Except for the graphical programming, you can also interact with animation game projects in real-time through hardware modules.



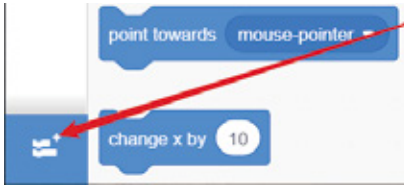
• Here we will mainly introduce the use of Crowbits-UNO Online extension module.

The way to get the letscode software: Logging in to the official website: **forum.elecrow.com**.
Then searching the download area, finding and downloading the Letscode software package then double-clicking it to install.

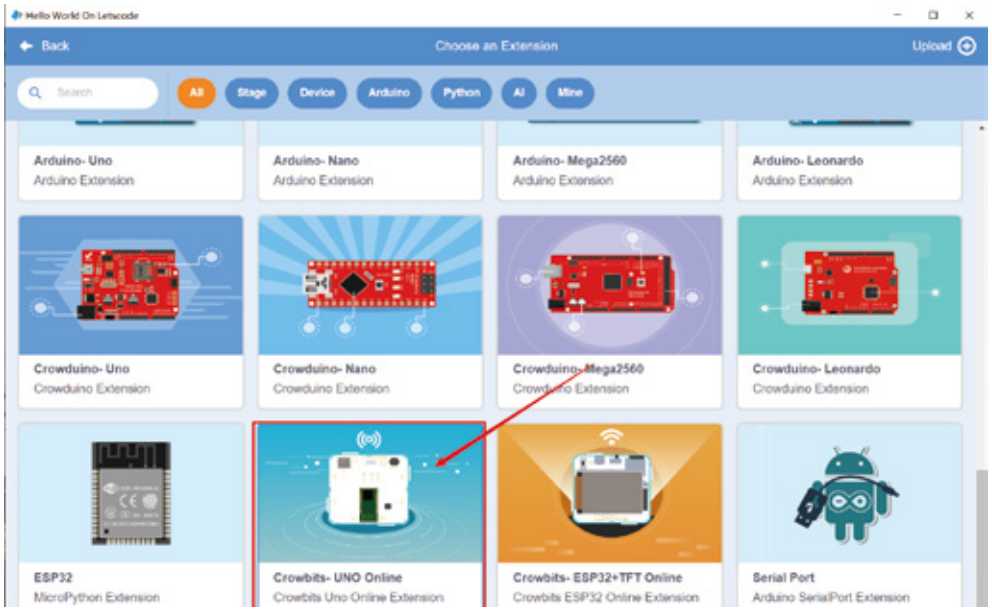
1. Opening the Letscode and Connecting the Devices

1. Double-clicking Letscode to open it on computer desktop.

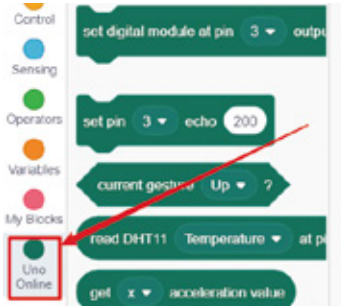
2. Clicking " Add Extension"



3. Dropping down the right scroll bar to find Crowbits-UNO Online module and click it.



Then UNO-Online module showing in the left sidebar “Code”, which means we have imported the Arduino-UNO Online module successfully.



There are five areas for stage mode: Command Area, Programming Area, Stage Area, Sprite Area and Menu Area.

Command Area: Many code blocks offered with different colors and shapes for different functions.

Programming Area: Dragging code blocks to programming area. And then matching them logically to implement the program.

Stage Area: The display area of the sprites controlled by programming. The game project will be displayed here.

Sprite Area: The sprites included in the project are displayed in this area. Click one of the sprites to see all the attributes of it.

Menu Area: Include some basic operation of software, such as file operation, device connection, etc.

2. Connecting the Devices

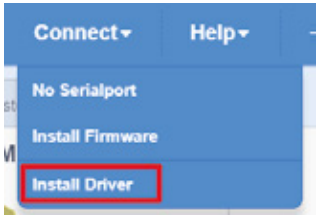
Connect the Device: There are two methods of connection, wired connection and Bluetooth wireless connection.

Wired Connection: Use a mini-USB cable to connect Crowbits-UNO main board and computer.

Turn the switch on the motherboard to the USB_DL end.

Clicking the button “Connect” with the mouse, then you will see COM in the drop-down list (here showing COM19, which may change according to the different computer you connected). Then clicking “COM” port, if “√” showing, that means connect succeeded.

If no serialport number is displayed, click Install Driver.



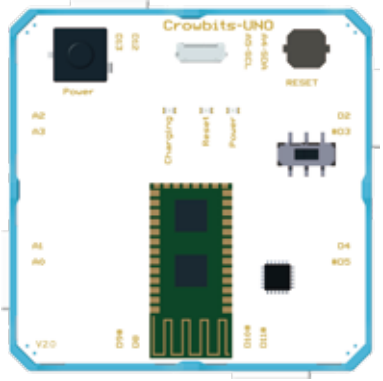
Bluetooth Connection: Connect Arduino UNO main board to the computer's Bluetooth firstly.

At this point, the switch on the motherboard should be turned to the BT_DL end.

Open computer settings to add a Bluetooth device, and search for Arduino UNO main board Bluetooth device:

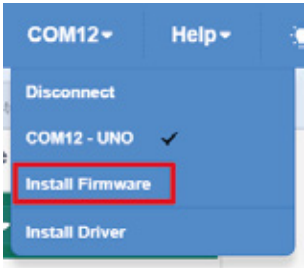
Find the Bluetooth "HC-02", enter the PIN: 1234, and click “connect”:

After the Bluetooth connection is successful, click button “Connect” with the mouse, then you will see two consecutive COM ports in the drop-down list ports (here showing COM31 and COM32, which may change according to the different computers you connected). Then clicking “COM31” button(usually choose the port with the smaller number), if “√” showing and the Bluetooth light of the main board changes from flashing to steady on, that means connect succeeded.



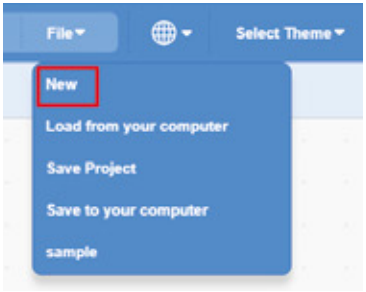
3. Install Firmware

The first time you use the Crowbits-UNO module, you need to install the firmware.Click the serial port location, click Install Firmware, and wait for the installation to succeed.

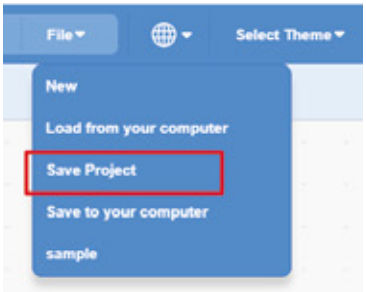


4. Instruction of Program Files

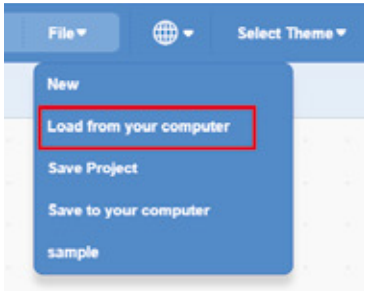
1. Program Files New-building: Clicking button "New" in the drop-down list of "File" to create a new program file.



2. Files Saving: For the first time to save your files, you can click button "Save Project" then choose save path and give it a name to save it. Clicking "Save Project" again, the files will be updating saved based on the previous files. If another file version needed, click "Save to your computer" button and set the save path and file name to store it.



3. File Opening: When you want to open existing Scratch files, click "Load from your computer" button and find the file and click it.



Project 1 Horse Racing

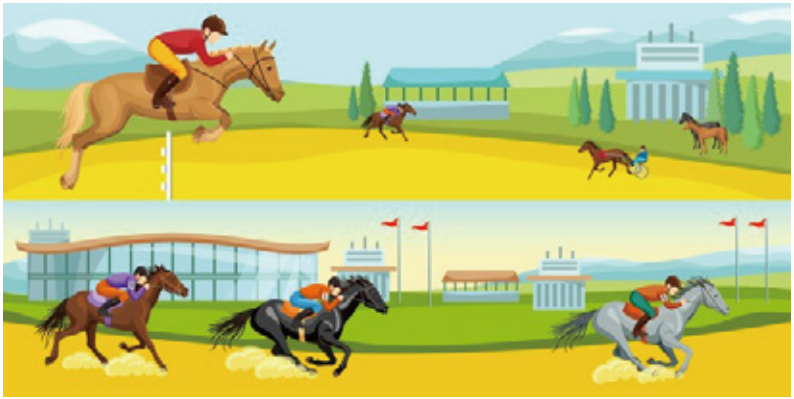
1. Learning Goals

1. Know the competition event: horse racing;
2. Learn the operating principle of IR reflective sensor and interactive programming control;
3. Complete the model building and game programming of horse racing project.

2. Application

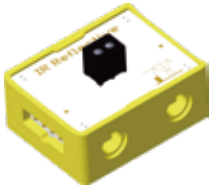
Originating in ancient Greece and Rome, horse racing has a long history. The heyday of the Roman Empire spawned a variety of horse racing events, such as harness racing, horse racing, Romanesque horse racing (the rider straddles on two horses), etc. But the basic rule of horse racing is to compete for speed.

Next, let's build a horse racing model and begin horse racing through software and hardware interaction!



3. Supplies List

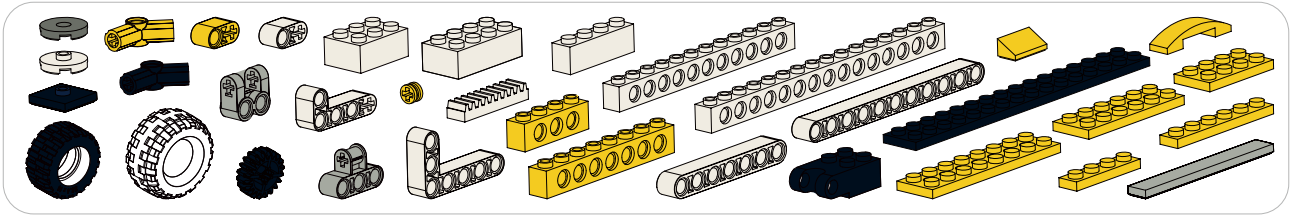
• **Electronic Supplies:**



IR Reflective Sensor x1

Crowbits-UNO x1

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**

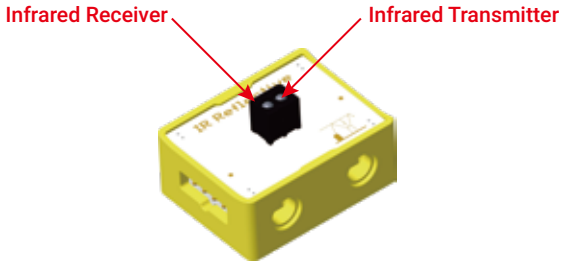


4. Knowledge Analysis

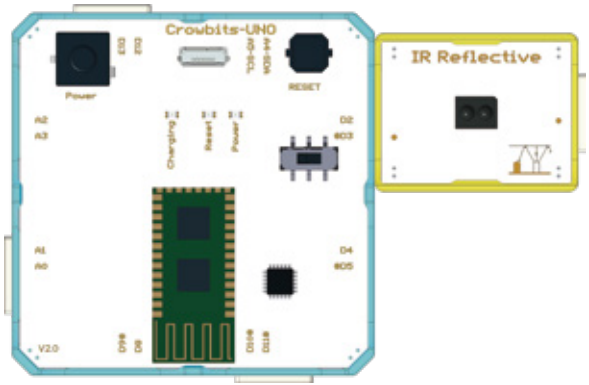
• **IR Reflective Sensor**

1. IR reflective sensor: It is a digital input module with a detection distance of about 1-2cm, which is composed of an infrared transmitter and receiver;

Operating principle: Infrared is invisible light, but it travels in a straight line like ordinary light, which can be reflected and absorbed. When the emitted infrared rays meet the reflective surface (not black), they will be reflected back, and the receiver will output high level 1 when receiving infrared rays. Conversely, if the infrared does not meet the reflective surface or is absorbed (black surface), the receiver will output low level 0;

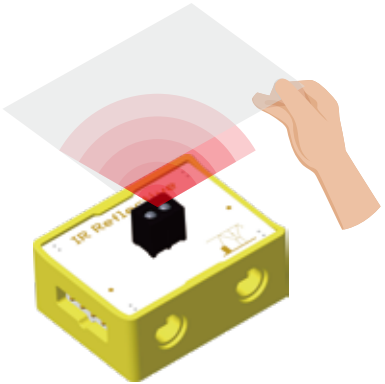


2. Circuit Connection:



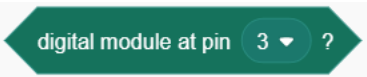
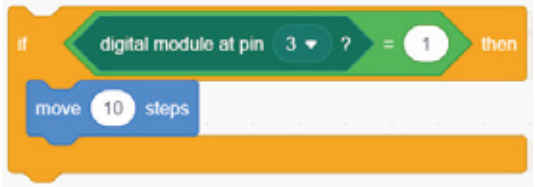
3. Program Driven:

You can program according to the situation listed in the following table:

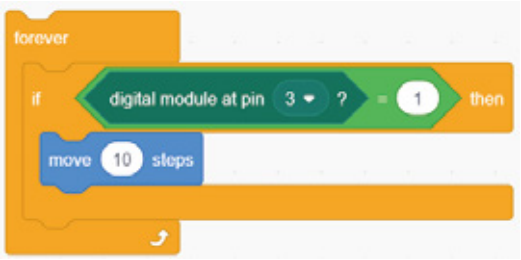


Object detected?	IR reflective sensor's signal level
Yes	high level/1
No	low level/0

Next, we will use the following blocks:

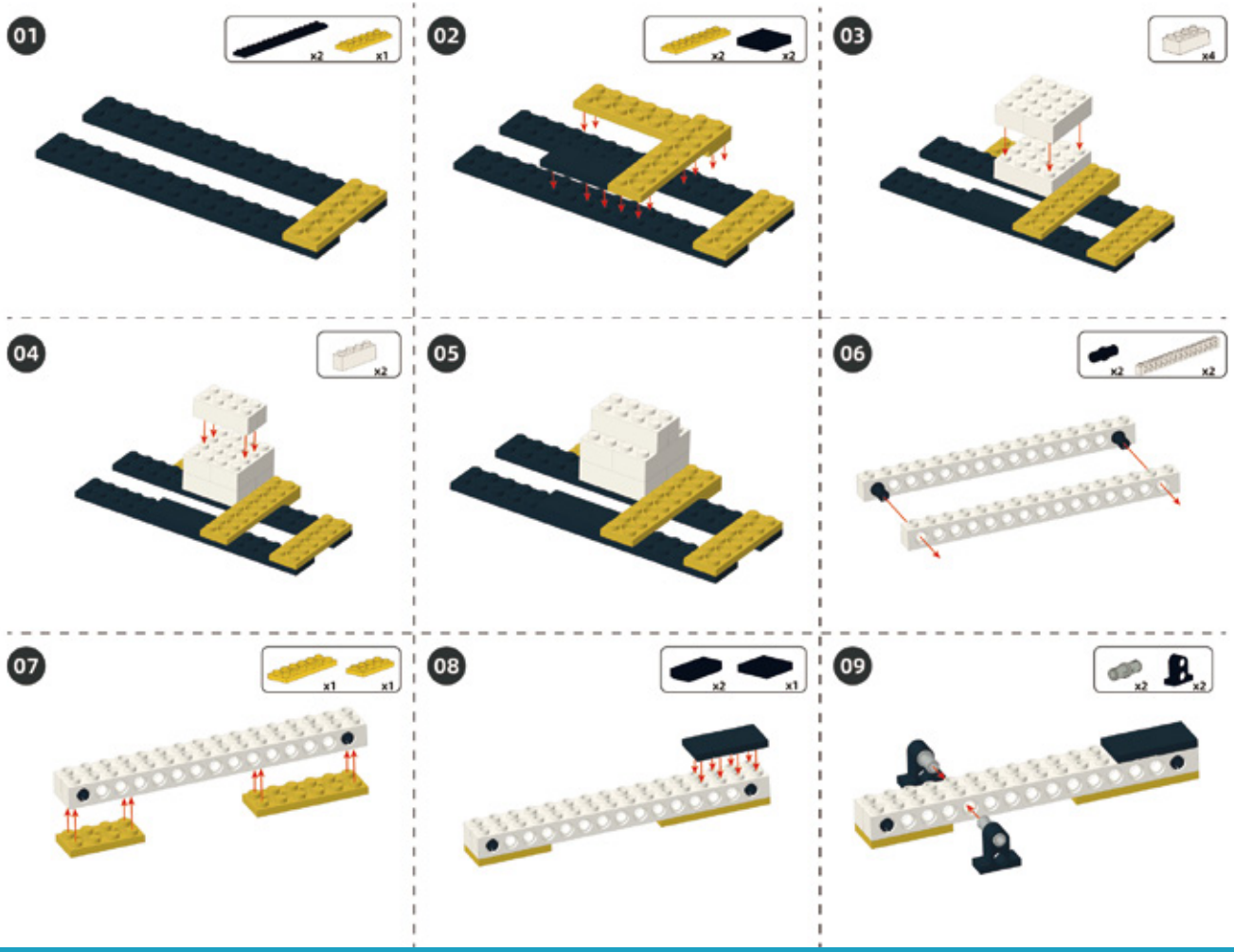
Block	Example
 Start the program: When the green flag is clicked, the program starts here. A program can have multiple green flag blocks	 Analysis: When the green flag is clicked, the program starts to execute and says "hello". Then the sprite moves 10 steps
 Judge whether they are equal: The two sides of the equal sign are the two objects to be compared	 Analysis: Judge whether the value obtained from the pin3 port is equal to 0
 Read the level signal of the digital port: If you read the port feedback signal of the IR reflective sensor, you will get high level "1" or low level "0"	
 Conditional judgment: If the condition is met, then execute internal programs	 Analysis: If the value of pin3 port is equal to 1, then the sprite moves 10 steps

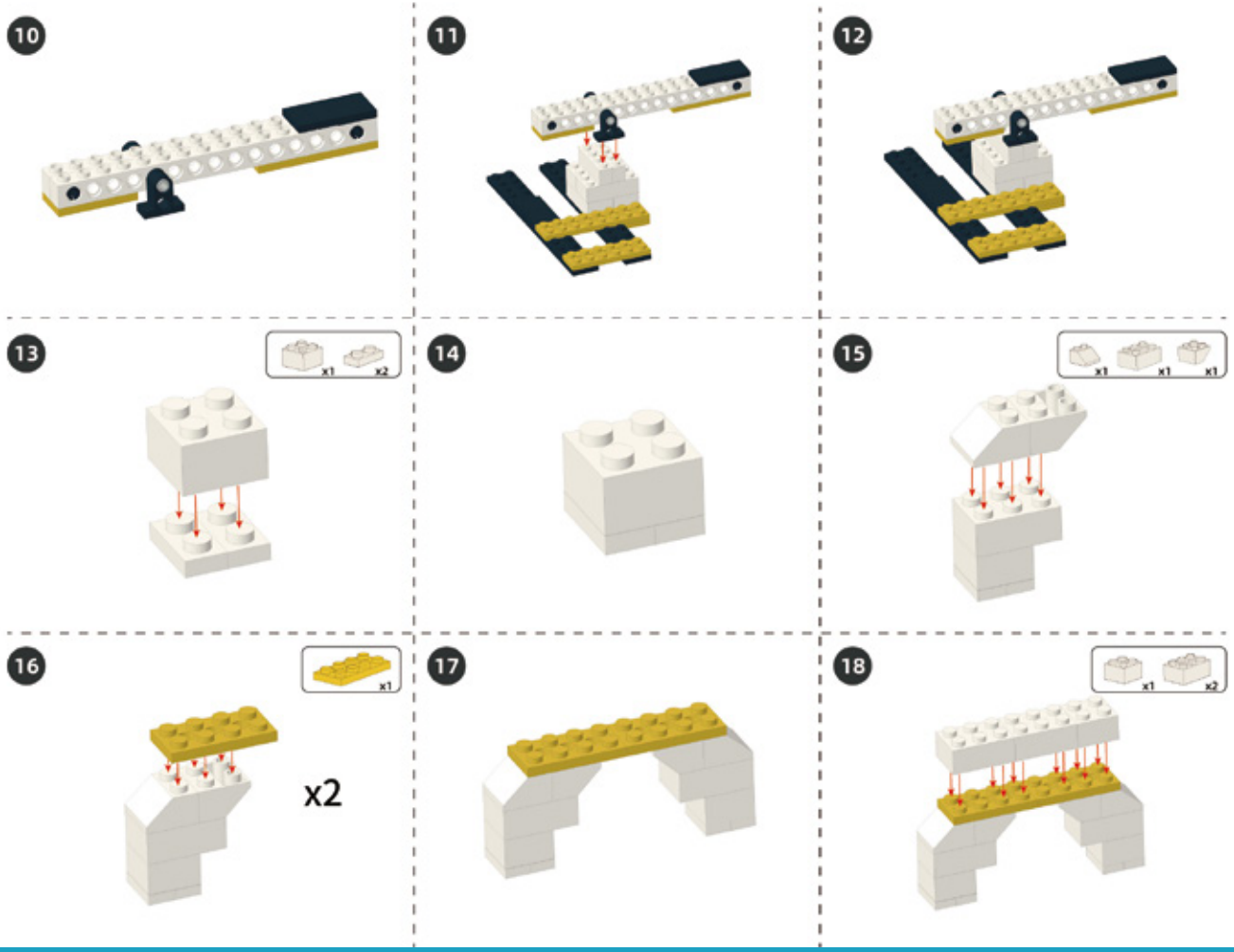
Continued

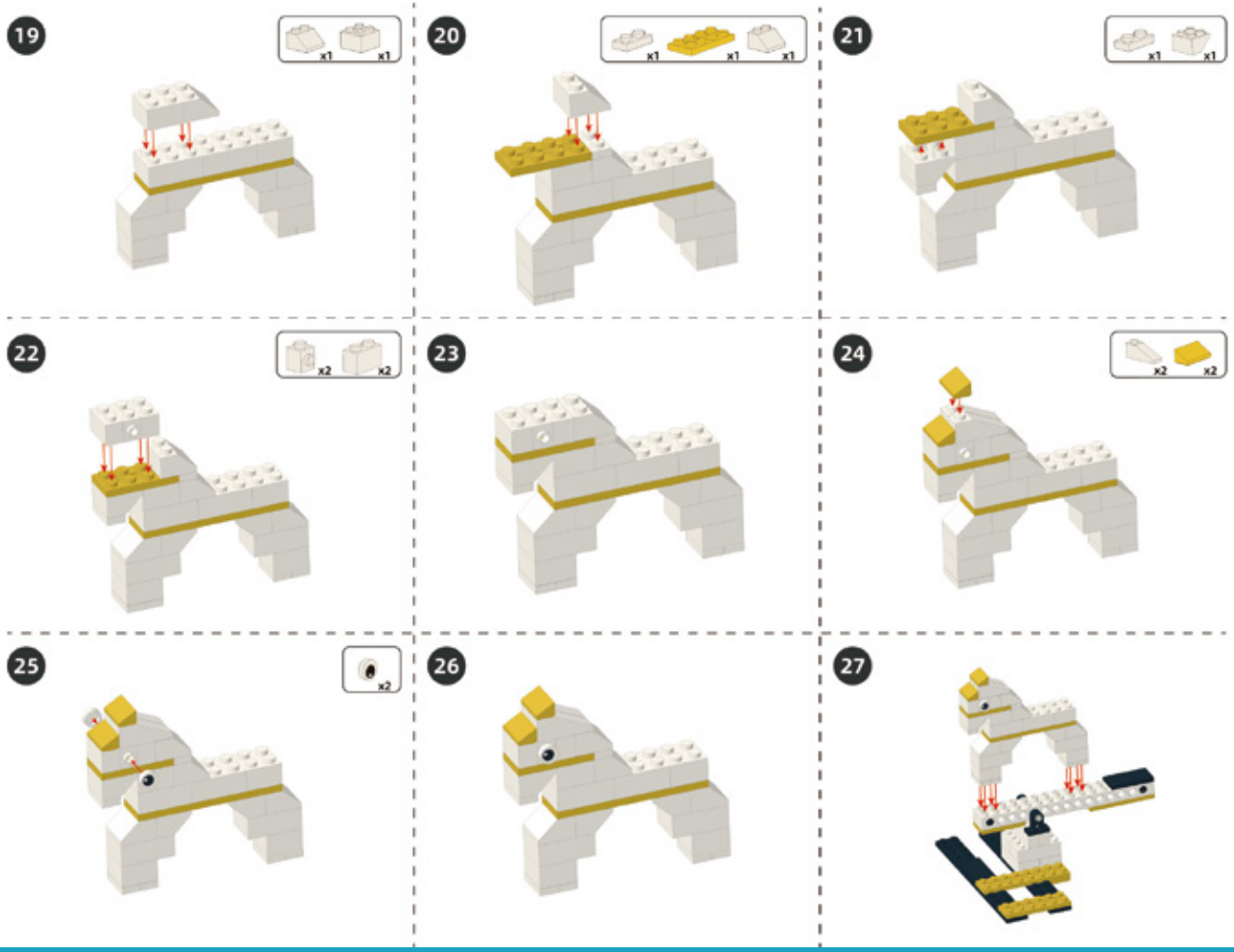
 Repeat execution: Repeat internal program	 Analysis: If the value of pin3 port is equal to 1, then the sprite moves 10 steps. It will always be repeated
 Repeat until: If the condition after "until" is not established, the internal program will be executed repeatedly. When the condition is established, the program will jump out of the loop and execute the following program	 Analysis: If the space bar is not pressed, then repeat the execution: move 10 steps and wait for 0. 5 seconds, otherwise it will jump out of the loop and execute other programs

5. Model Building

- **Building Blocks**


01
x2
x1
02
x2
x2
03
x4
04
x2
05
06
x2
x2
07
x1
x1
08
x2
x1
09
x2
x2

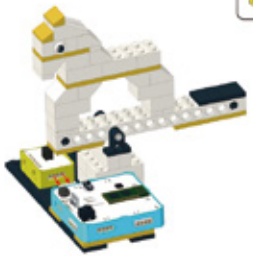

10
11
12
13
x1
x2
14
15
x1
x1
x1
16
x1
x2
17
18
x1
x2


19
x1
x1
20
x1
x1
x1
21
x1
x1
22
x2
x2
23
24
x2
x2
25
x2
26
27

28

29

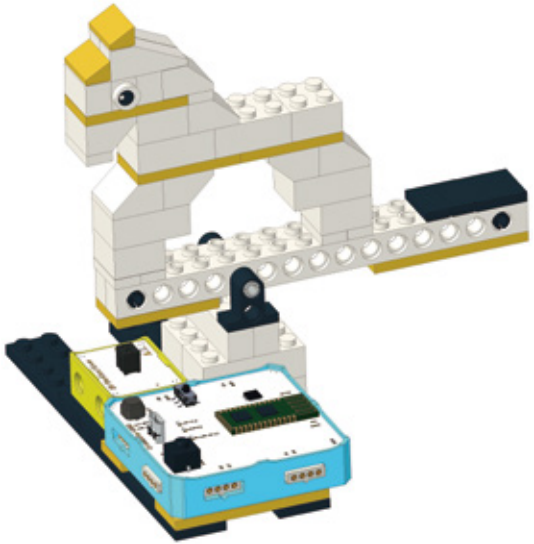
30



31

• **Circuit Connection**

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Horse racing training. Control the movement of the horse through the high and low level of IR reflective sensor.

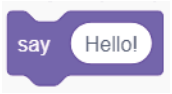
Task Analysis:

1. Detect the signal of IR reflective sensor;
2. Control the movement of the horse through the high and low level of IR reflective sensor.

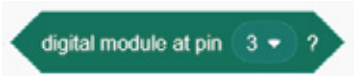
Task Steps:

Detect the signal of IR reflective sensor:

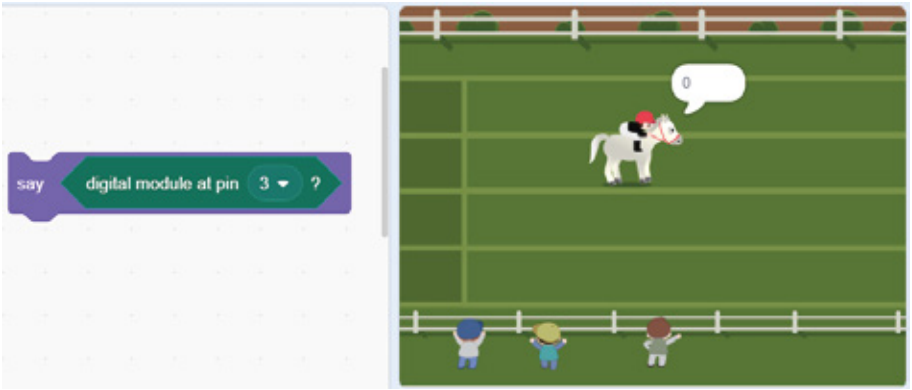
1. Power on the Arduino main board and switch to Bluetooth connection mode;
2. Search and connect to the Bluetooth serial port on the computer;
3. Drag the "say" code block **[say "Hello!"]** to the programming area;



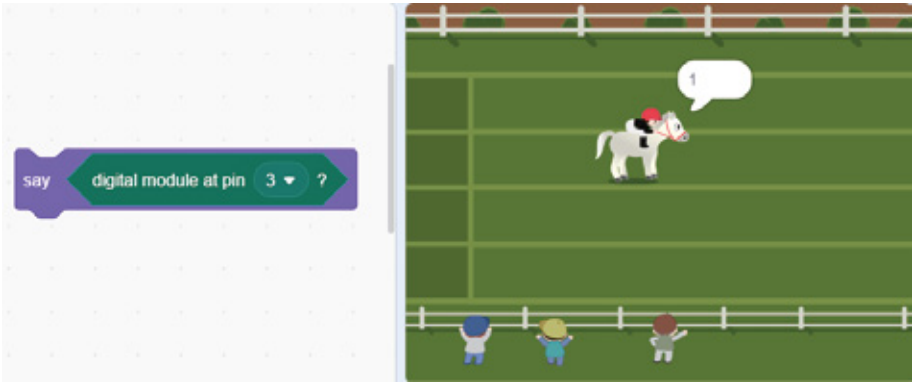
4. Add Arduino Online extension package and drag **[digital module at pin "3" ?]** in the "say" code block;



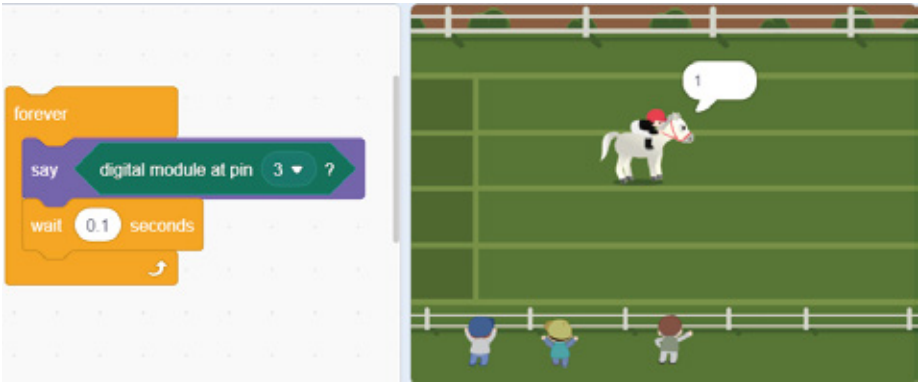
5. Click "say" code block by the mouse, if there is no object cover the IR reflective sensor, the signal is displayed as low level 0;



6. Place your hand about 1-2cm above the IR reflective sensor, and click "say" code block again, the signal would be high level 1;

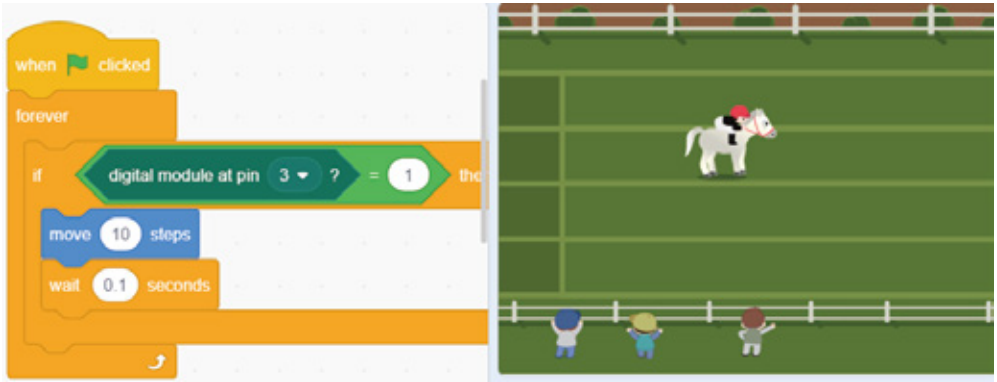


7. If you need to repeatedly detect the signal level, you can put the above program into the "forever" block, and add "when green flag clicked" block to start the program. Cover and stay away from the IR reflective sensor with your hand, and observe the change of the signal level.



Control the movement of the horse through the high and low level of IR reflective sensor:

1. When the signal level of IR reflective sensor is high level 1, the horse moves 10 steps; when the level is 0, no program is executed, that is, the horse does not move. In order to avoid instantaneous movements when the execution is too fast, a code block of waiting for 0.1 seconds can be added;

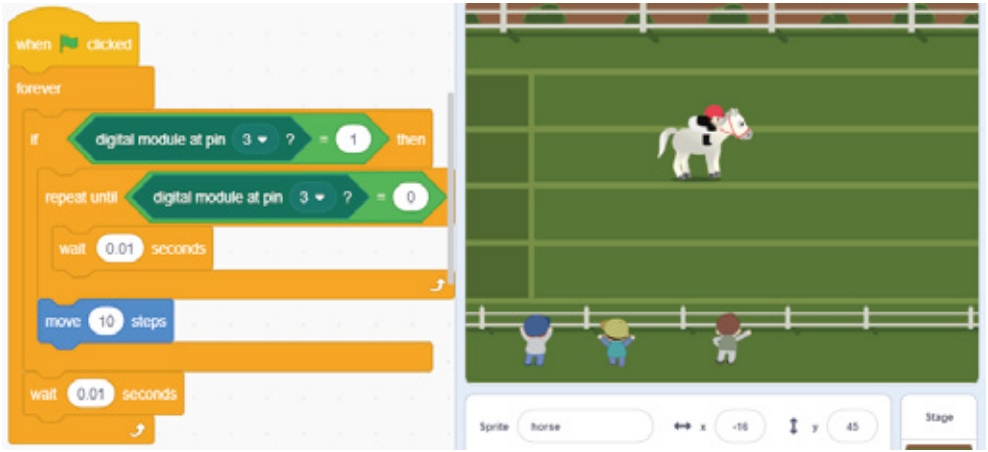


2. Try covering the IR reflective sensor with black or white objects to see if the horse is moving or not.

3. Have you found that when we keep our hands covering IR reflective sensor, the horse will always move forward, which loses the fun of the game. How to achieve the function that if triggered once, the horse only move once?

4. This function can be achieved by using the "repeated until" code block. When the signal level detected is high (that is, IR reflective sensor is blocked by the hand, the program enter the repeat execution to run the program loopy. Only when the signal changes to low level 0 (that is, remove your hand from IR reflective sensor), the program can jump out the loop and execute the horse racing program;

5. To determine the signal is high;



6. Awesome, you have learned how to control the horse racing forward, but the horse keeps the same costume and looks not like a real horse. Can you try to switch the next costume when the horse is moving?

We have provided a complete horse racing game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprite in the game require you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Let's invite your family or good friends to a horse racing competition and see who can get first place! You can also modify the program according to your own ideas to increase the difficulty and experience of the game.

8. Brainstorming

We mainly learned the IR reflective sensor in horse racing project. In fact, infrared technology is widely used in our life, such as infrared remote control (TV, air conditioner), infrared obstacle avoidance car and infrared tracking robot. Can you try to design an infrared anti-theft alarm with existing equipment? When an object covers IR reflective sensor within 1-2cm, the computer will sound the alarm.

Project 2 Catch the Fruit

1. Learning Goals

1. Know the classic game: catch the fruit;
2. Learn the operating principle of six-axis gyroscope and its graphical programming control;
3. Complete the model building and game programming of catch the fruit project.

2. Application

The ripe fruits in the forest are falling down. There are many types of fruits: apples, watermelons, grapes, oranges, etc. There is a pig waits for the fruit to fall under the tree and catches the fruit. However, in addition to delicious fruits, there are bombs falling, and the pig has to catch the fruits and avoid the bombs.

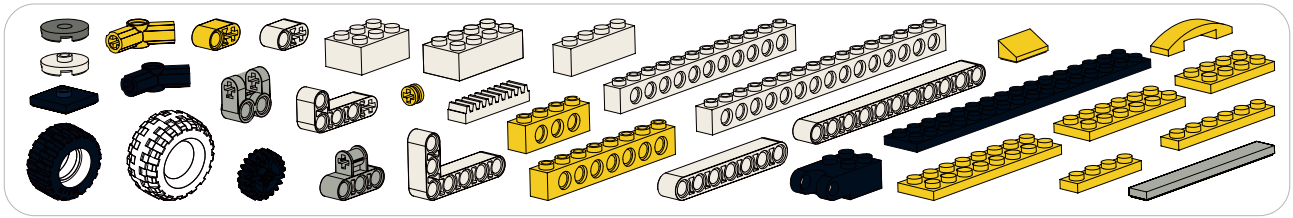
Next, we will use building blocks and electronic modules to build a fruit-picking basket, and control the pig on the game side through the basket model to achieve the interactive effect of catch the fruit game.

3. Supplies List

• **Electronic Supplies:**

Crowbits-UNO x1

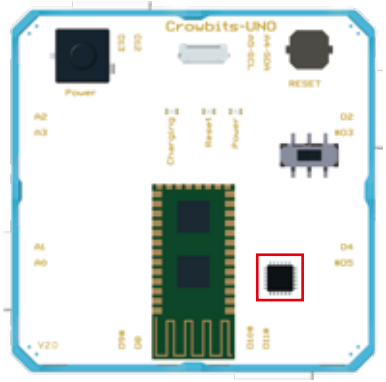
• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



4. Knowledge Analysis

• **Gyroscope Mpu6050**

1. Mpu6050 is the world's first integrated six-axis motion processing unit. It is a very popular space motion sensor chip that can obtain the current three acceleration components and three rotational angular velocities of the device. Due to its small size, powerful functions and high accuracy, it is not only widely used in industry, but also an artifact for model aircraft enthusiasts. It is installed on many kinds of aircraft to fly in the blue sky.

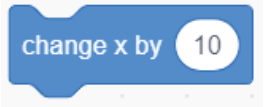
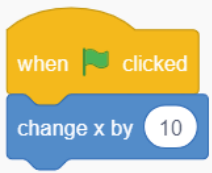
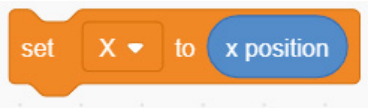


2. Program Driven:

Mpu6050 chip is installed on the front of the main board, and the deflecting direction of the main board is the same as that of the chip. You can program according to the situation listed in the following table:

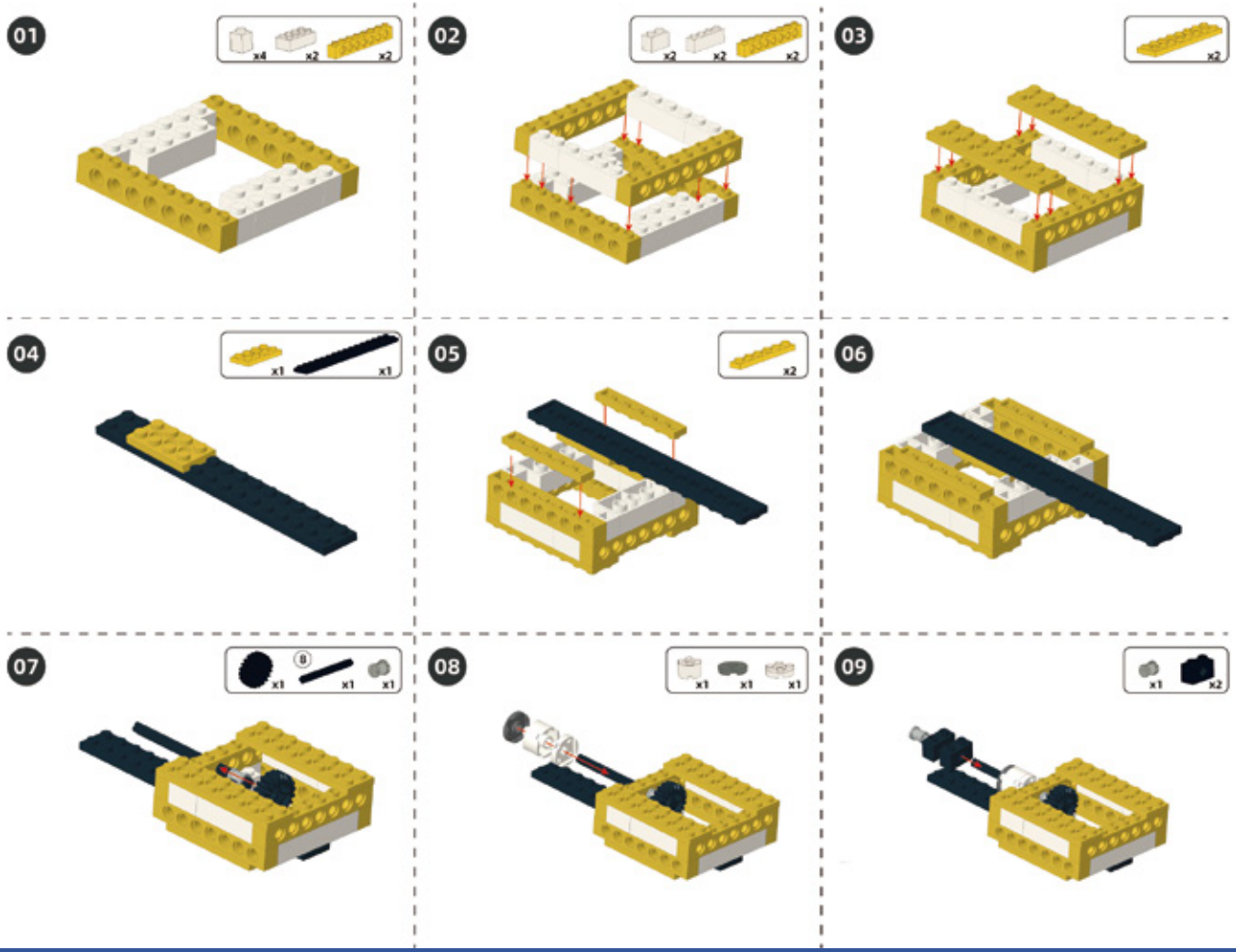
Block	Explanation
	When the main board module deflects upward, the return value is 1, otherwise the return value is 0
	When the main board module deflects downward, the return value is 1, otherwise the return value is 0
	When the main board module deflects to the left, the return value is 1, otherwise the return value is 0
	When the main board module deflects to the right, the return value is 1, otherwise the return value is 0

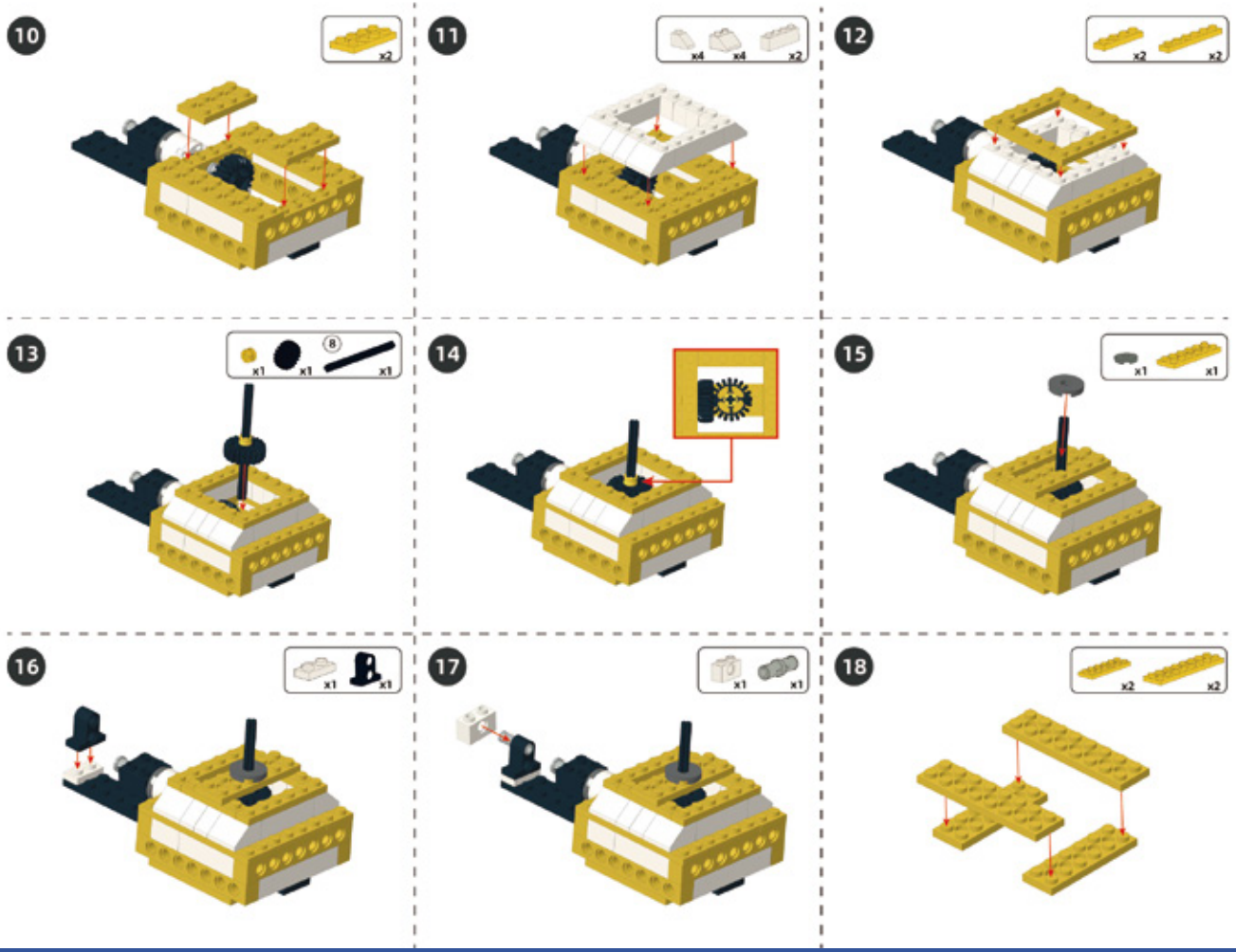
Next, we will use the following blocks:

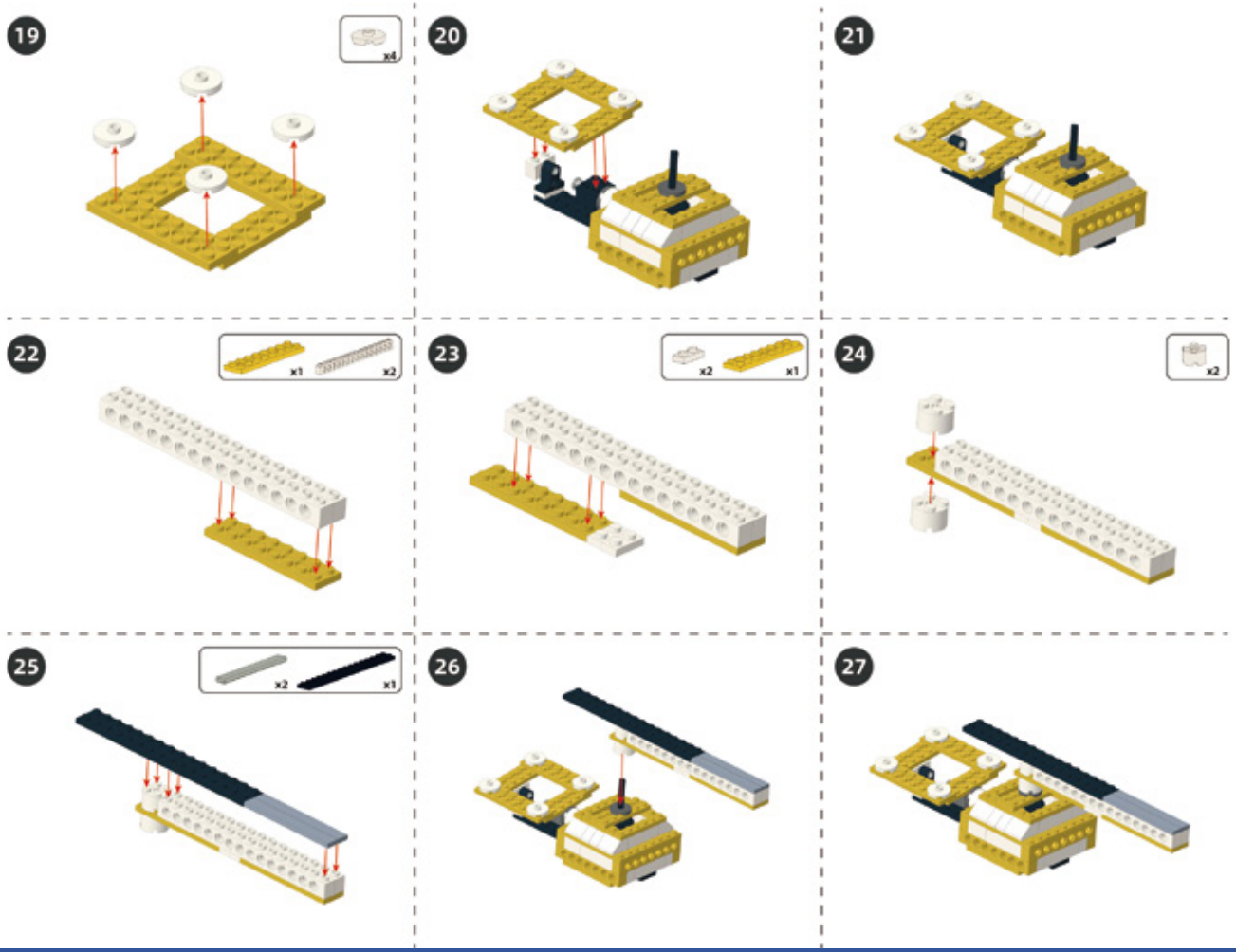
Block	Example
 Increase the X-position value: This block controls the change of the sprite's position on the X-axis. The number represents the distance moved each time, positive numbers move to the right, negative numbers move to the left	 Analysis: When the green flag is clicked, the sprite moves 10 units of coordinates to the right
 X-position value: represents the value of the x-coordinate of the current sprite, generally used with the calculation block	 Analysis: Set "X" to the value of the X-position
 Compare value: When the value on the left is greater than the value on the right, the result is true, the return value is '1' (true). When it is not true, the return value is '0' (false)	 Analysis: The left side of the greater than sign is the mathematical operation 2+2=4. 4>3, the expression is established, and the return value is 1 (true)

5. Model Building

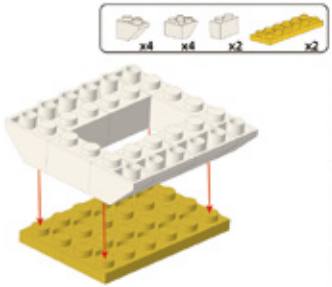
- **Building Blocks**


01
x4
x2
x2
02
x2
x2
x2
03
x2
04
x1
x1
05
x2
06
07
x1
x1
x1
08
x1
x1
x1
09
x1
x2

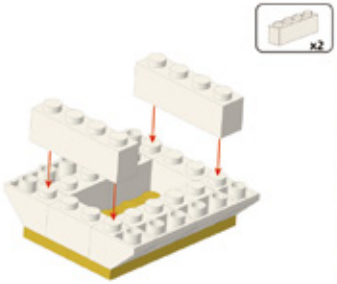

10
x2
11
x4
x4
x2
12
x2
x2
13
x1
x1
8
x1
14
15
x1
x1
16
x1
x1
17
x1
x1
18
x2
x2


19
x4
20
21
22
x1
x2
23
x2
x1
24
x2
25
x2
x1
26
27

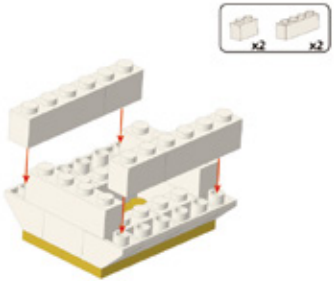
28



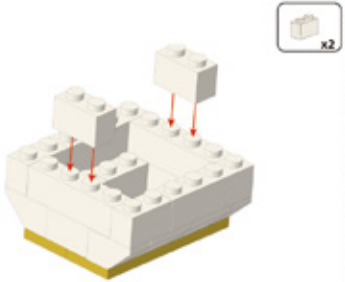
29



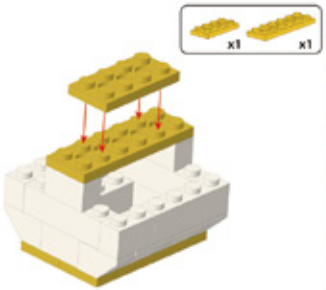
30



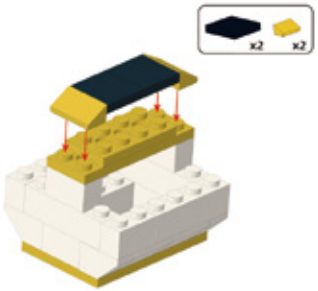
31



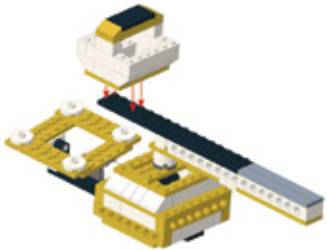
32



33

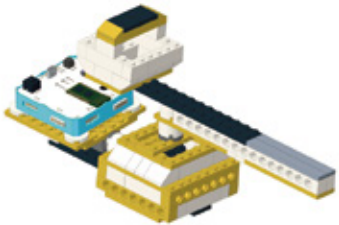


34



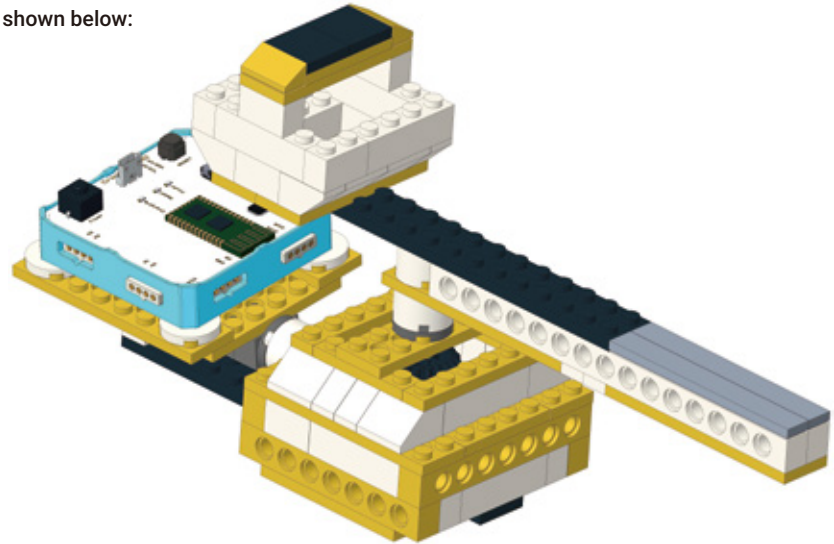
35

36



• Circuit Connection

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Use the six-axis gyroscope to control game sprite to move left and right.

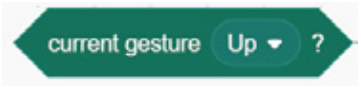
Task Analysis:

1. Detect the signal of the six-axis gyroscope's deflection;
2. According to the gyroscope's signal, the pig can move left and right;
3. Limit the moving range of the pig.

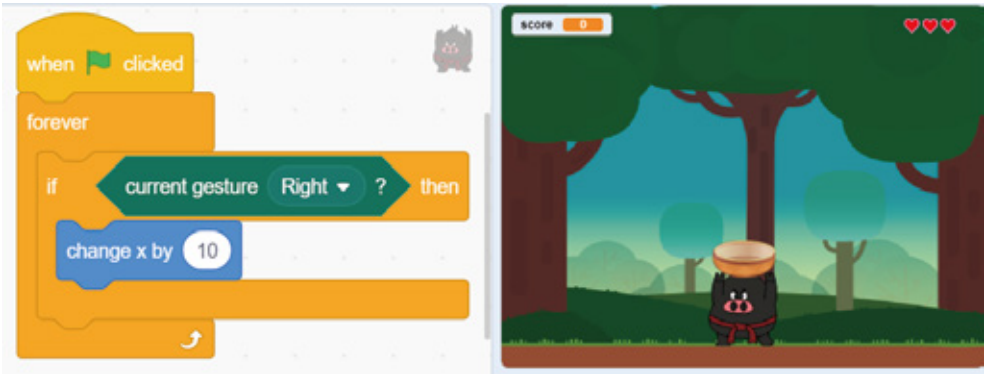
Task Steps:

Detect the signal of the gyroscope's deflection:

1. Power on the Arduino main board and switch to Bluetooth connection mode;
2. Search and connect to the Bluetooth serial port on the computer;
3. Drag the "say" code block to the programming area;
4. Add Crowbits-UNO Online extension package and drag **[current gesture "up" ?]** in the "say" code block and then repeat;

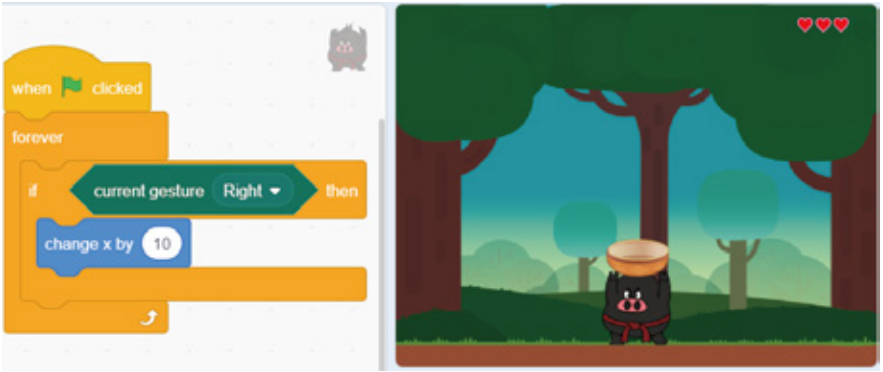


5. Click the green flag with the mouse to deflect the main board module upward, and try to modify the deflecting direction according to the signal changes. And then check the hardware stability.

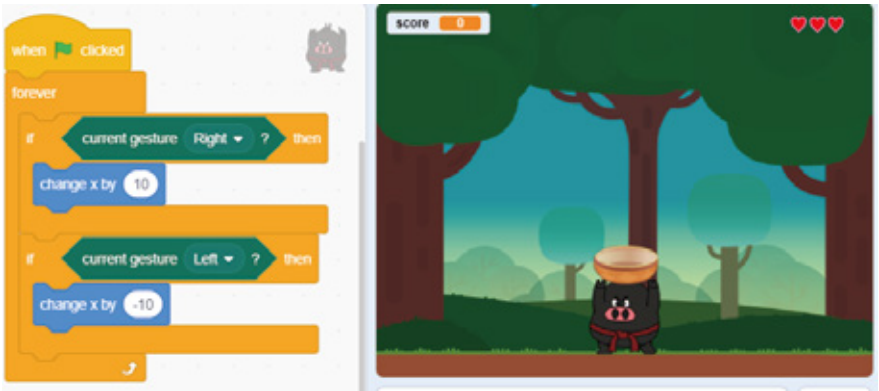


The gyroscope controls the sprite's movement:

1. When detecting the left and right deflection of the six-axis gyroscope position, let the pig in the stage area follow it. When the main board deflects to the right, the pig also moves to the right, that is, it moves 10 in the X-axis direction.



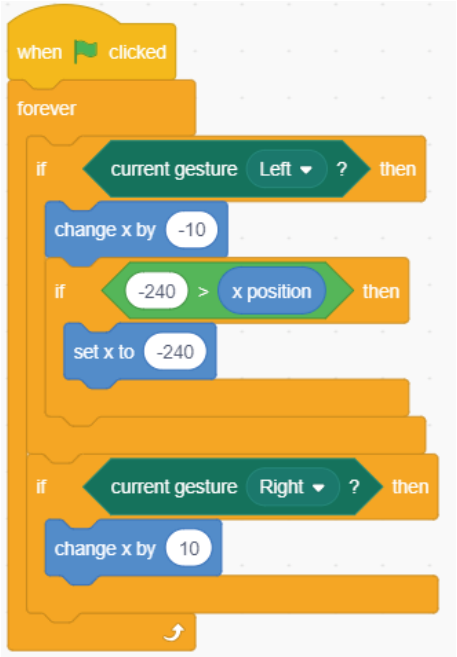
2. When the main board deflects to the left, the pig moves to the left, that is, it moves -10 in the X-axis direction.



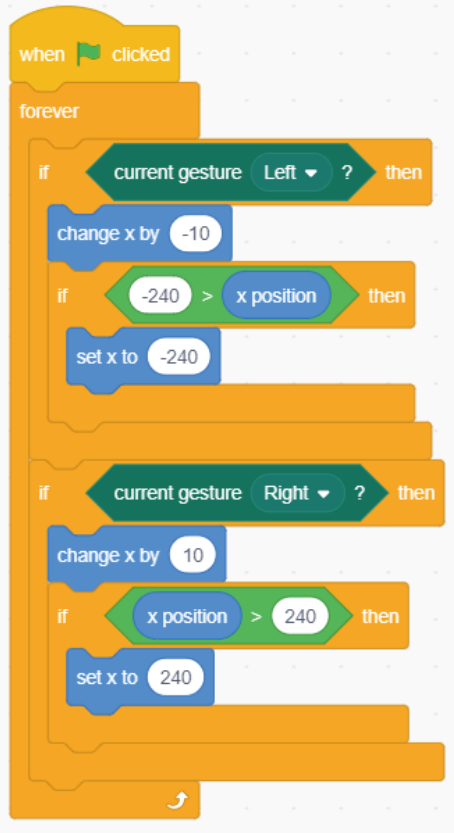
Limit the moving range of the pig:

1. If the pig moves to the left all the time, it will go out of the stage area due to the stage boundary. So we can't let the pig always moves to the left. When the coordinates are less than -240, the position of the pig should be limited to -240;

2. The pig must move firstly, then to walk out of the stage area. These two steps are sequential, so we can connect the two programs. In this way, when the main board deflects to the left, the pig will also move a short distance to the left, and when the pig moves beyond the boundary coordinates we defined, it will instantly move to the X-position -240;



3. The same is true for the right side, note that the X-position in the left and right directions are different;



4. Awesome! You have learned how to control the pig to move left and right, and restrict the pig from running out of the stage's edge. Now let's try to use the basket model to control the pig movement!

We have provided a complete "catch the fruit" game in the software resources. You can download it and try it out. There are many fruits inside, can you add more fruit?

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a catch the fruit competition and see who can get the highest score. You can modify the data in the game program to increase the difficulty of the game. Or you can increase the speed of fruits falling in the game, or increase the amount of blood deducted when you catch a bomb, to exercise your reaction ability.

8. Brainstorming

We used the six-axis gyroscope sensor as the trigger module in this project. The sensor is often used in smart devices, such as controlling the movement of sprite in games. You can try to use this electronic module to design a somatosensory remote control game.

Project 3 Car Racing

1. Learning Goals

1. Know the competition event: car racing;
2. Learn the operating principle of the linear potentiometer and its interactive programming control;
3. Complete the model building and game programming of the racing car steering wheel.

2. Application

The earliest racing competitions were held on the roads in the cities. The drivers control the cars to avoid obstacles and surpass other drivers to fight for the championship. Afterward, many drivers lose their lives because of its high danger on the road, then the professional racing tracks come into being.

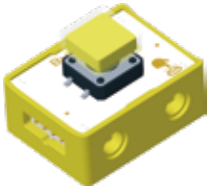
Next, we will make a race car steering wheel with the bricks and electronic modules and realize the interaction of car racing game through controlling the steering wheel modeling.

3. Supplies List

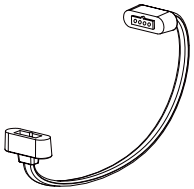
• **Electronic Supplies:**

Crowbits-UNO x1

Linear Potentiometer x1

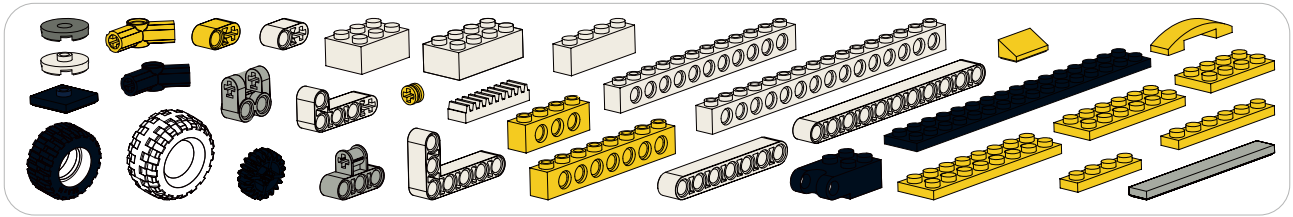


Button Module x1



Magnetic Cable x1

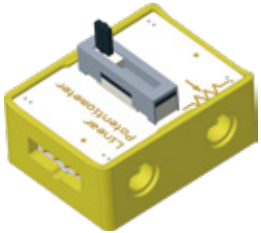
• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



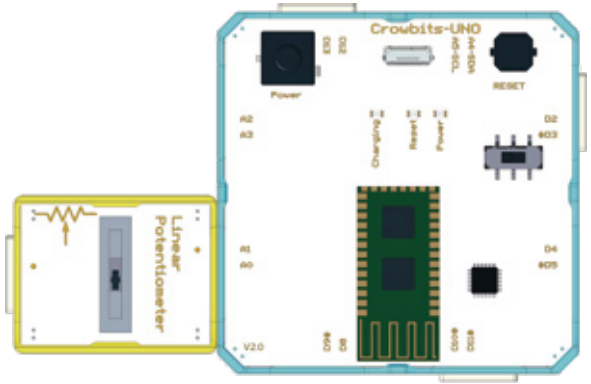
4. Knowledge Analysis

• Linear Potentiometer

1. Linear potentiometer is an analog input module, which is a kind of variable resistor. It is made of resistor and rotating or sliding system, that is moving on the resistor by a contacting point to obtain partial voltage output.

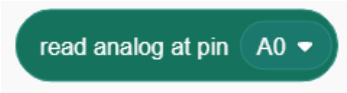


2. Connections:

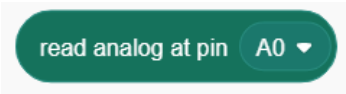


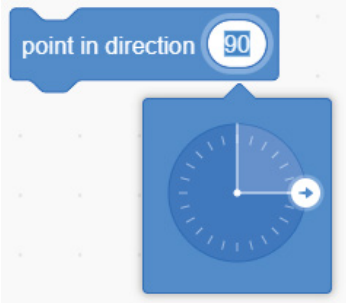
3. Program Driven

Because Linear potentiometer is an analog input module, we could use this code block **[read analog at pin "A0"]** to obtain analog value between 0-1023.



Next we will use the following code blocks:

Block	Example
 Analog input: Read the analog value of the corresponding pin	
 Judgment block: Judge whether the sprite touches the object	 Analysis: After running the program, the sprite would keep moving forward; when the sprite touches the edge of the stage, it would switch to the next costume

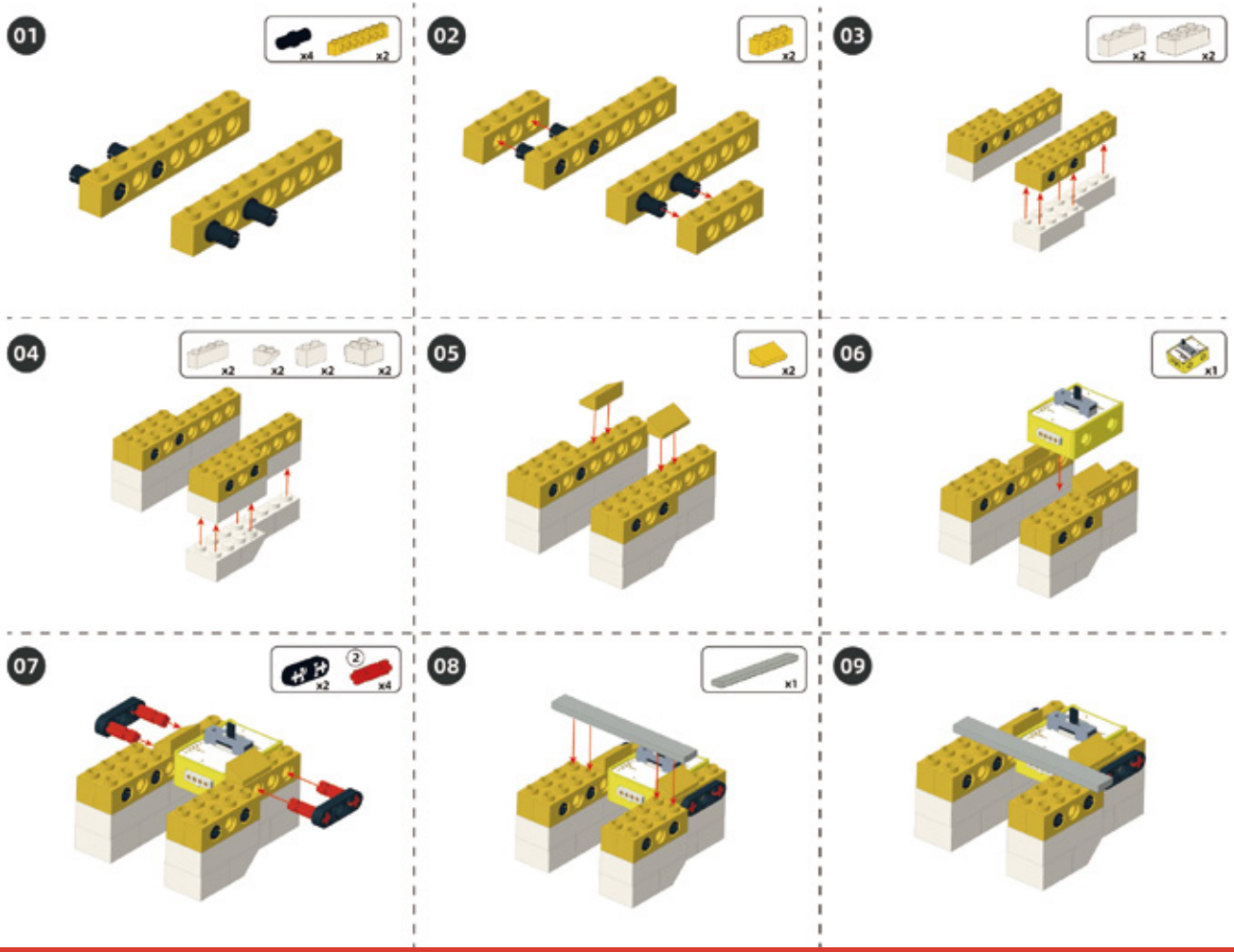


Point in direction: Set sprite's pointing direction, you could do that by typing values or adjusting the pointer

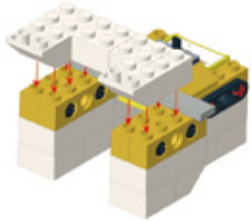
Analysis: The sprite moves forward to the right and then to the left

5. Model Building

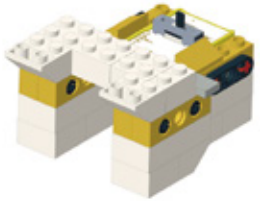
- **Building Blocks**


01
x4
x2
02
x2
03
x2
x2
04
x2
x2
x2
x2
05
x2
06
x1
07
x2
2
x4
08
x1
09

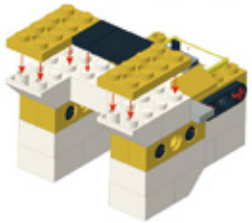
10



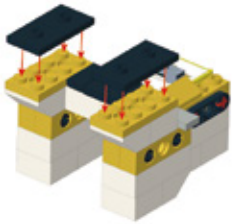
11



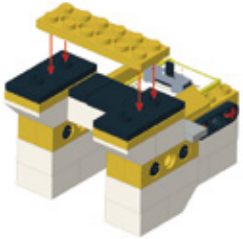
12



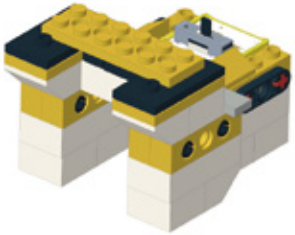
13



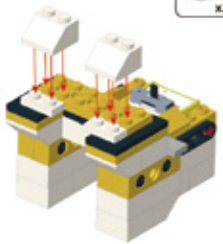
14



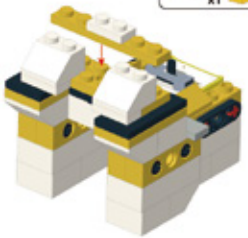
15



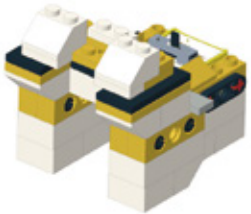
16

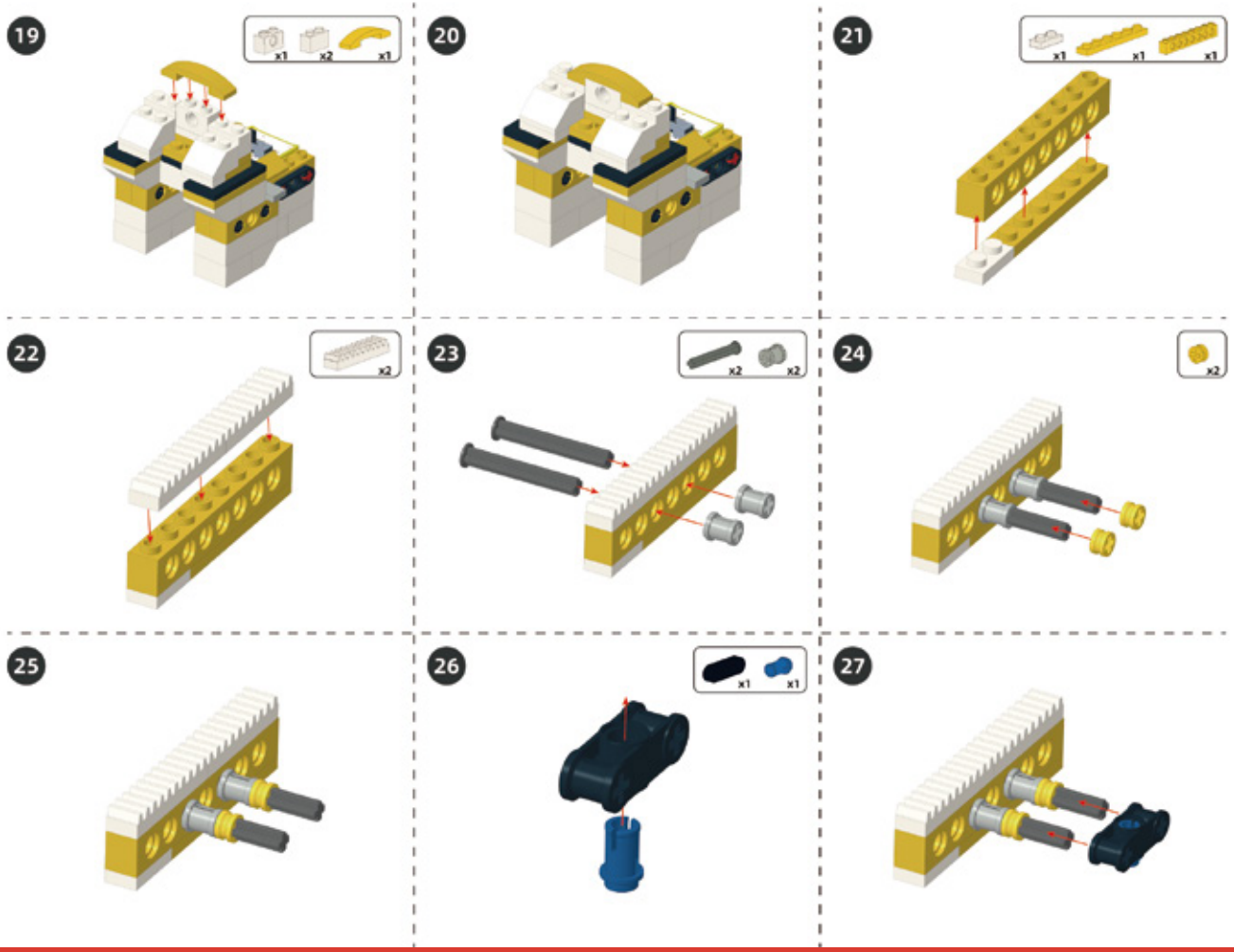


17

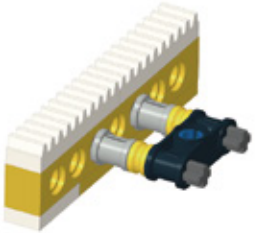


18




19
x1
x2
x1
20
21
x1
x1
x1
22
x2
23
x2
x2
24
x2
25
26
x1
x1
27

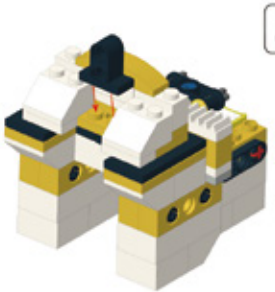
28



29

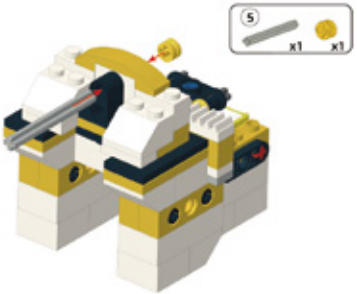
30

31

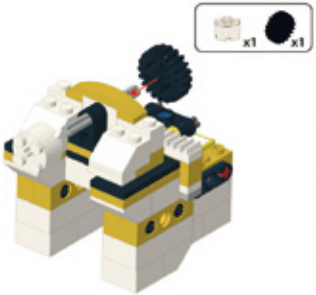


32

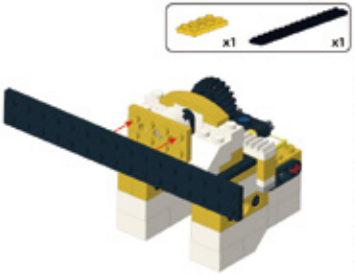
33



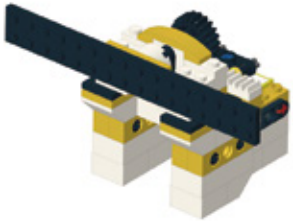
34

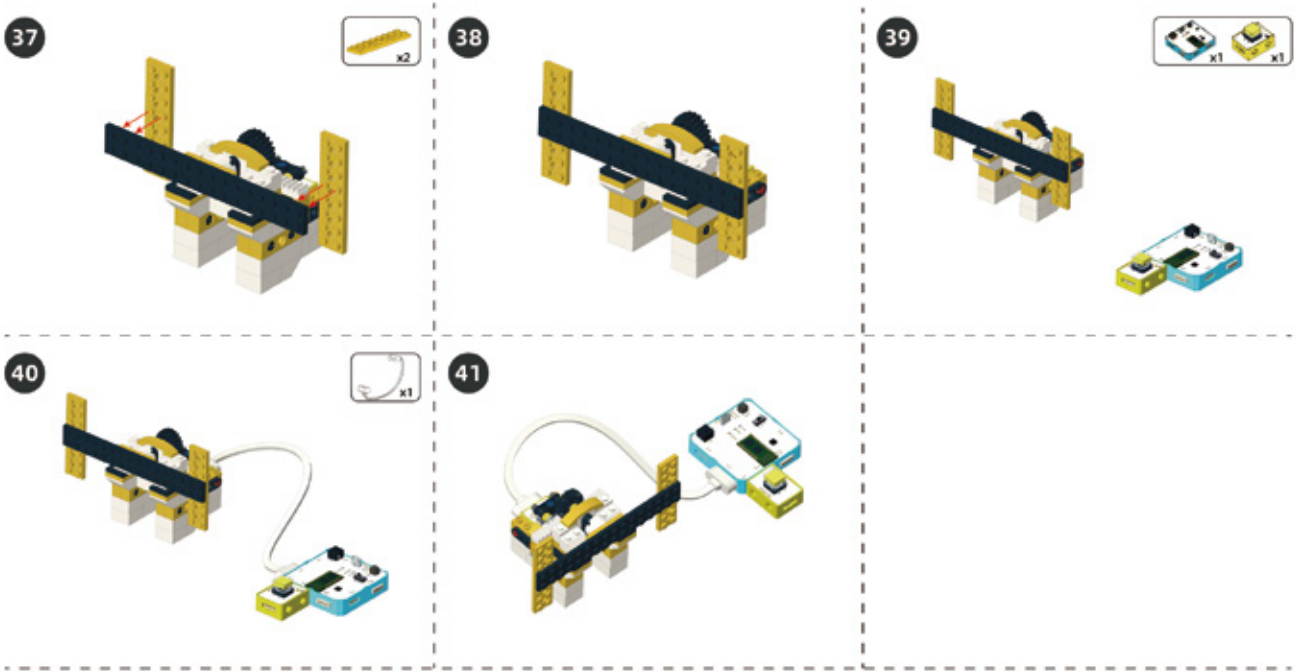


35



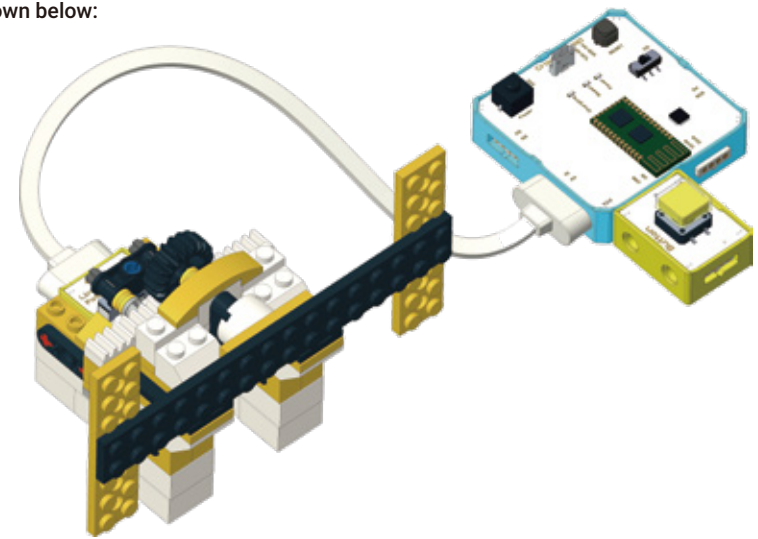
36




37
x2
38
39
x1
x1
40
x1
41

• **Circuit Connection**

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Car racing training. Control the car to avoid obstacles, make the truck appear randomly and move down, and simulate the car moving forward.

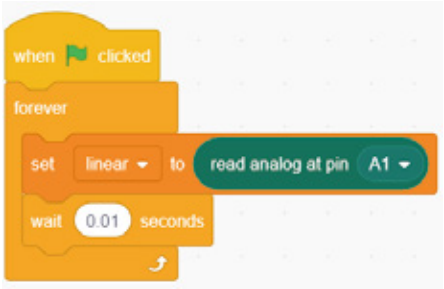
Task Analysis:

1. Control the racing car to move left and right;
2. Make the truck appear randomly and move down.

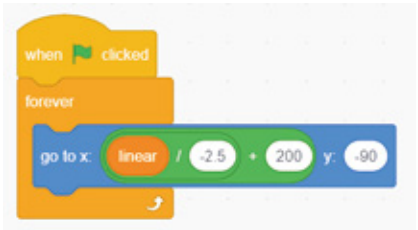
Task Steps:

Control the racing car to move left and right:

1. Stores the obtained analog value of linear potentiometer in a variable;



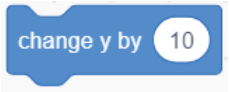
2. Map the obtained analog value to the racing car sprites to achieve the effect of moving left and right. The stage area range is -240 to 240, the value that potentiometer could obtain is from 0 to 1023, we need to do some simple math to get the one-to-one correspondence. Here we divide the value abtained from the linear potentiometer by -2.5 and plus 200, and then set the value as the x-coordinate of the racing car sprite.



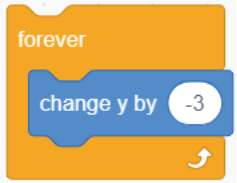
Try to slide the linear potentiometer to see whether the racing car will moves left and right?

Make the truck appear randomly and move down:

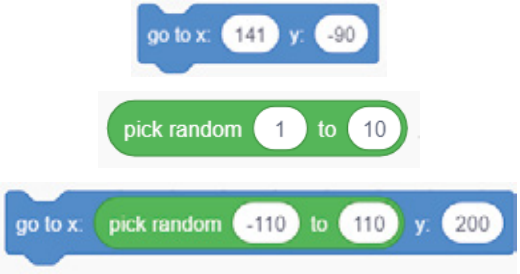
1. Programming for the truck sprite, the truck moves slowly from the top of the stage to the bottom, then the Y-position of the sprite would change, use the motion block **[change y by "10 "]** to change the number to a negative number, then sprite would move down.



2. In order to keep truck moving, the block **[forever]** would be needed, by connecting the two blocks, the truck is able to move continuously.



3. The truck appears from a random position at the top of the stage, its Y-position is set to 200; the truck should appear on roads in the game, so its X-position have a limit range. Here motion block **[go to x: 141, y: -90]** and operators block **[pick random "1" to "10"]** will be used, then set the parameters of X-axis and Y-axis.



4. The truck moves to the bottom of the stage and then appears randomly at the top of the stage, then it is necessary to check truck's position, motion block **[y position]** and operators block **[" " < "-200"]** would be used.

5. Combine the program above, then you could make the truck appear randomly at the top of stage and move down.

If you want to change the speed of the truck, which parameters you need to change?

We have provided a complete car racing game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprites in the game require you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Let's experience the work with your friends together!

Invite your family or good friends to a car racing competition and see who can get the highest scores! You can also modify the program according to your own ideas to increase the difficulty. For instance, increase the frequency of obstacle vehicles or speed of the racing car to increase the difficulty of the game, which can exercise your reaction and operating skills.

8. Brainstorming

In the car racing game, we use the linear potentiometer as a triggered module. Linear potentiometer is usually used in circuit to control the current. You could try to use this module to design a game that controls music volume by sliding.

Project 4 Crazy Bird

1. Learning Goals

1. Know the game: crazy bird;
2. Learn the operating principle of button and its interactive programming control;
3. Complete the model building and game programming of crazy bird project.

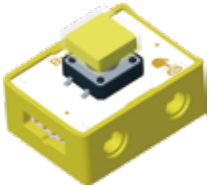
2. Application

The player controls the bird to flap its wings, keep flying, and cross different obstacles through the buttons on the model. Be careful not to let your bird fall down on the ground and hit obstacles. Let's see how far you can fly!

Next, we will use building blocks and electronic modules to build a bird, and control the bird on the game side through the bird model to achieve the interactive effect of crazy bird game.

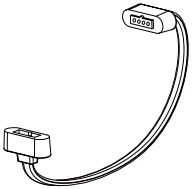
3. Supplies List

• **Electronic Supplies:**



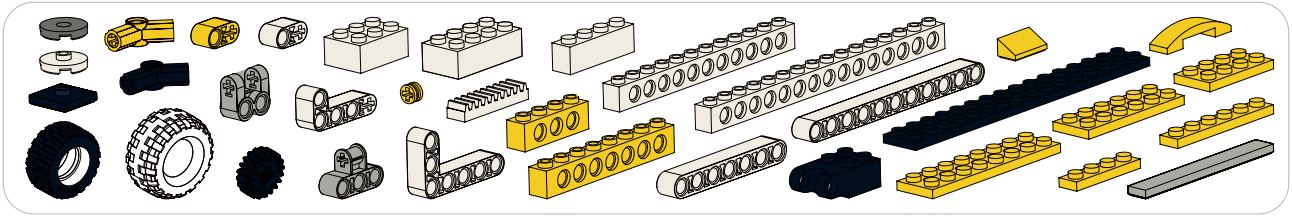
Button Module x1

Crowbits-UNO x1



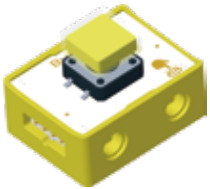
Magnetic Cable x1

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



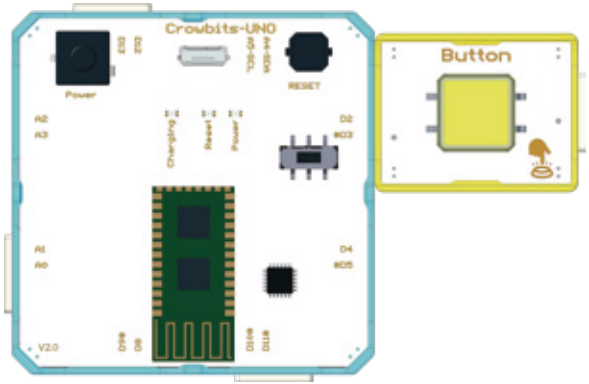
4. Knowledge Analysis

• **Button**



1. The button is a digital input module. When pressed and released, it will return to different levels (return 1 if pressed; return 0 if released) to judge whether the button is pressed.

2. Circuit Connection:

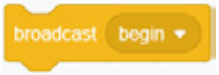
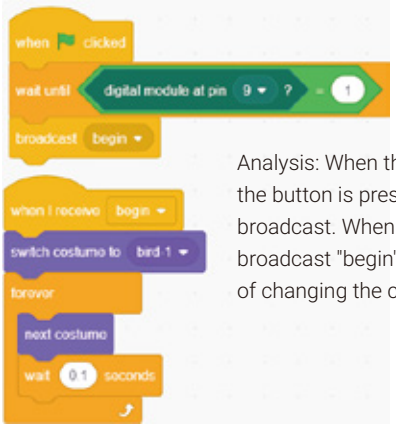
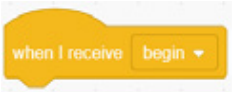


3. Program Driven:

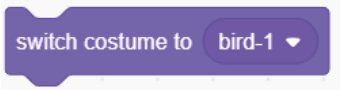
You can program according to the situation listed in the following table:

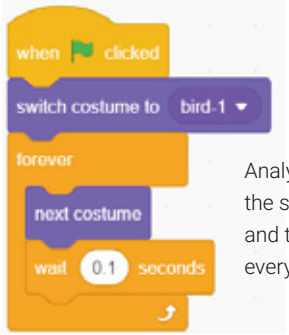
Pressed or not?	Button's signal level
Yes	high level/1
No	low level/0

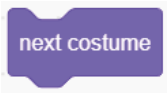
Next, we will use the following blocks:

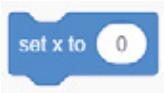
Block	Example
 Broadcast message: it's for all sprites and backgrounds, and use the "when I receive XX" broadcast message code block to start a new program	 Analysis: When the green flag is clicked and the button is pressed, the "begin" will be broadcast. When the sprite receives the broadcast "begin", it will start the program of changing the costume
 when I receive XX: when receiving XX broadcast message, it will start the following new program. The function is similar to "when green flag clicked", and both are program start commands	

Continued

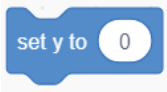
<table>
<tr>
<td> 

Costume switching: you can switch a specific costume of the sprite</td>
<td rowspan="2"> 

Analysis: When the green flag is clicked, the sprite will change to " bird-1" costume, and then change to the next costume every 0.1 seconds</td>
</tr>
<tr>
<td> 

Next costume: switch the currently displayed costume to the next one</td>
</tr>
<tr>
<td> 

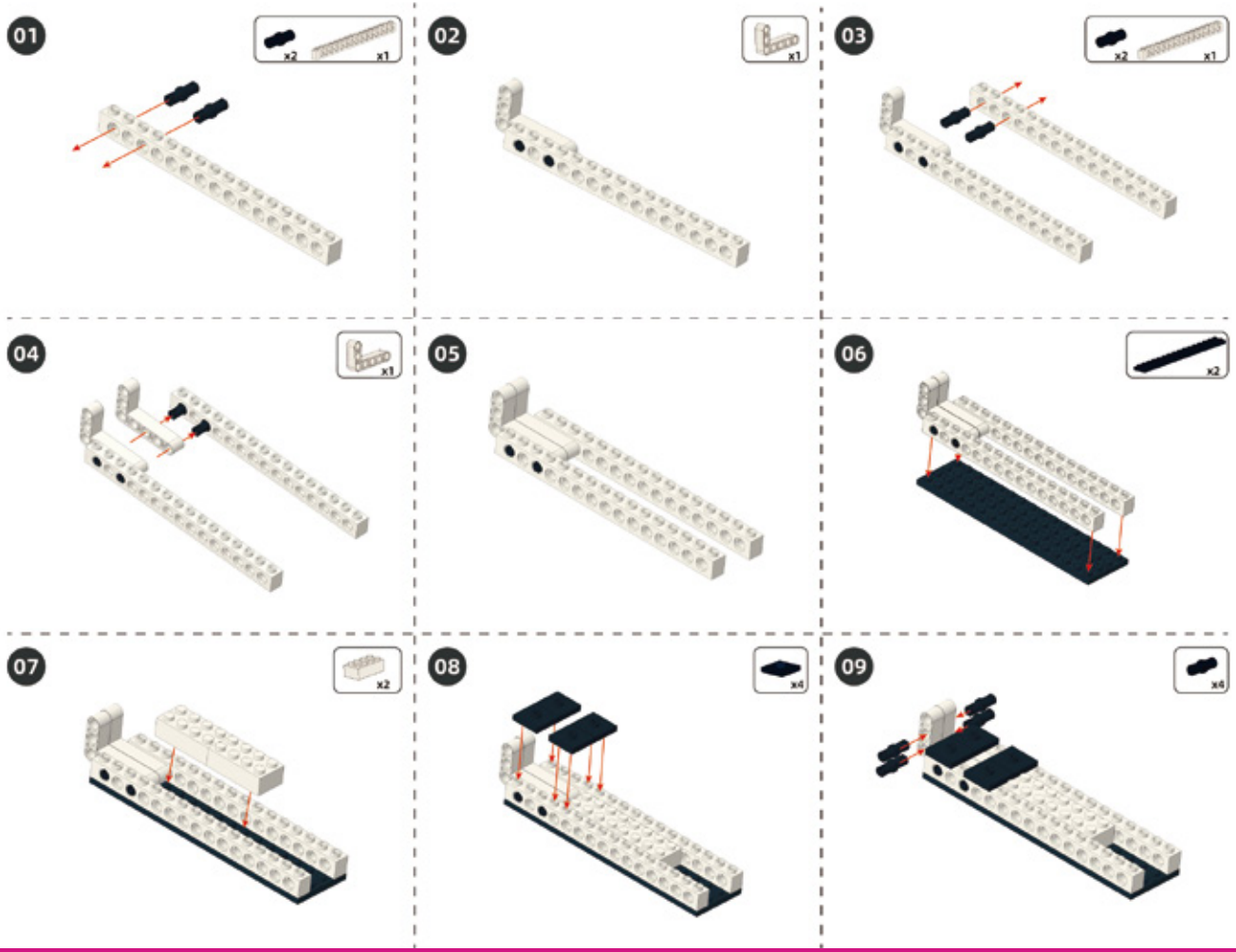
Set the X coordinate: Set the sprite's x coordinate value</td>
<td rowspan="2"> 

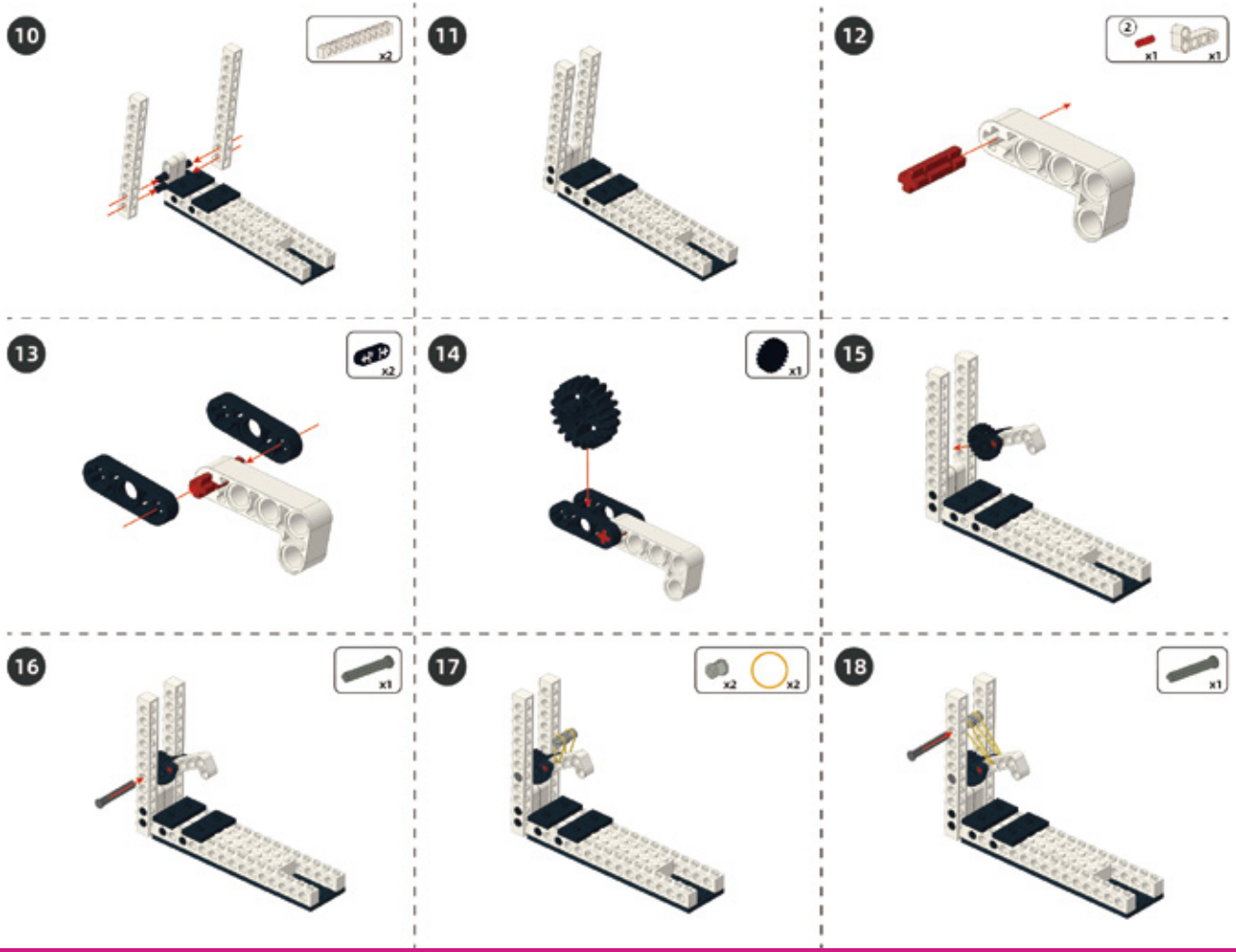
Analysis: When the green flag is clicked, the program will start. And the sprite will move to the position where the x coordinate is 100 and the Y-position is 100</td>
</tr>
<tr>
<td> 

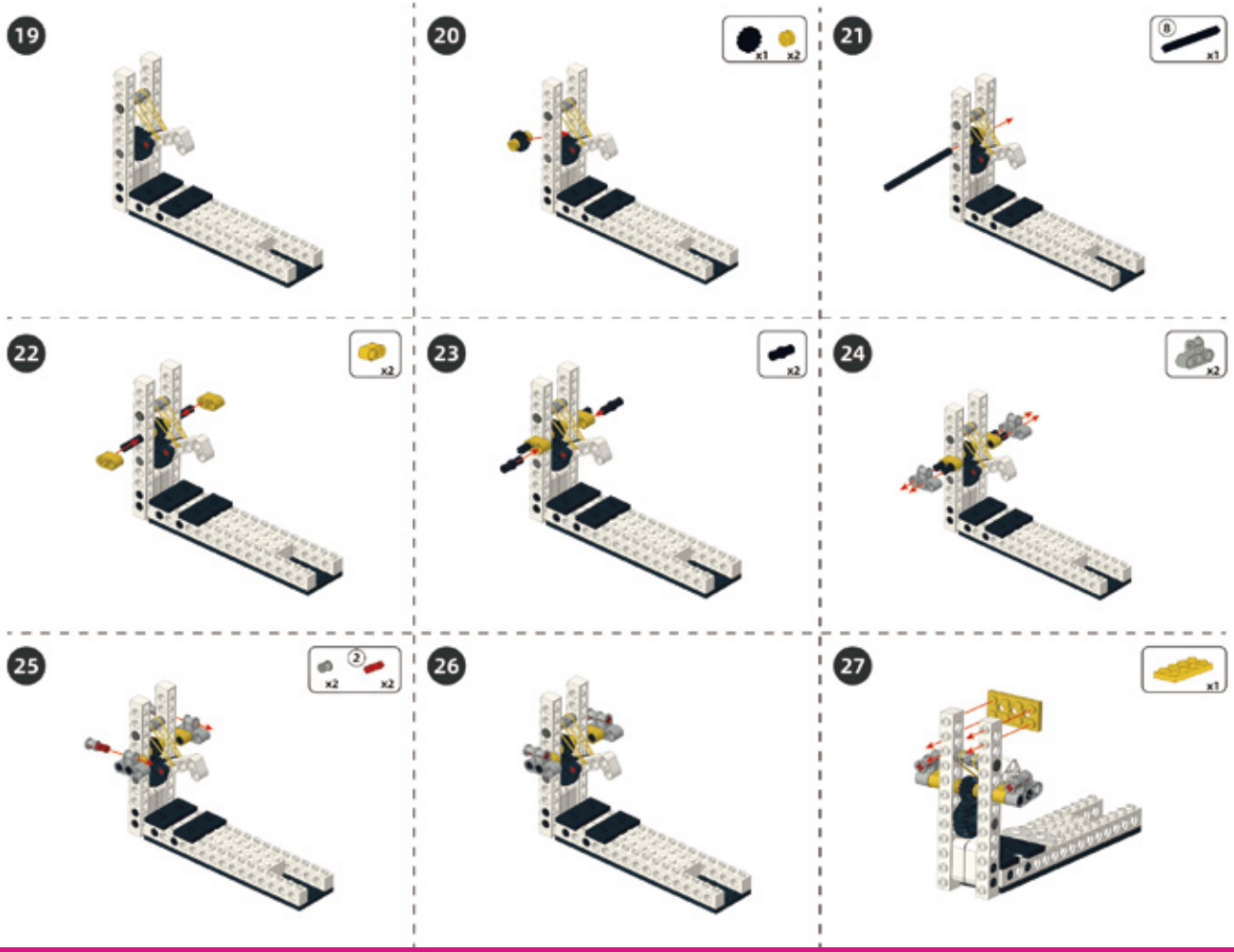
Set the Y-position: Set the sprite's Y-position value</td>
</tr>
</table>

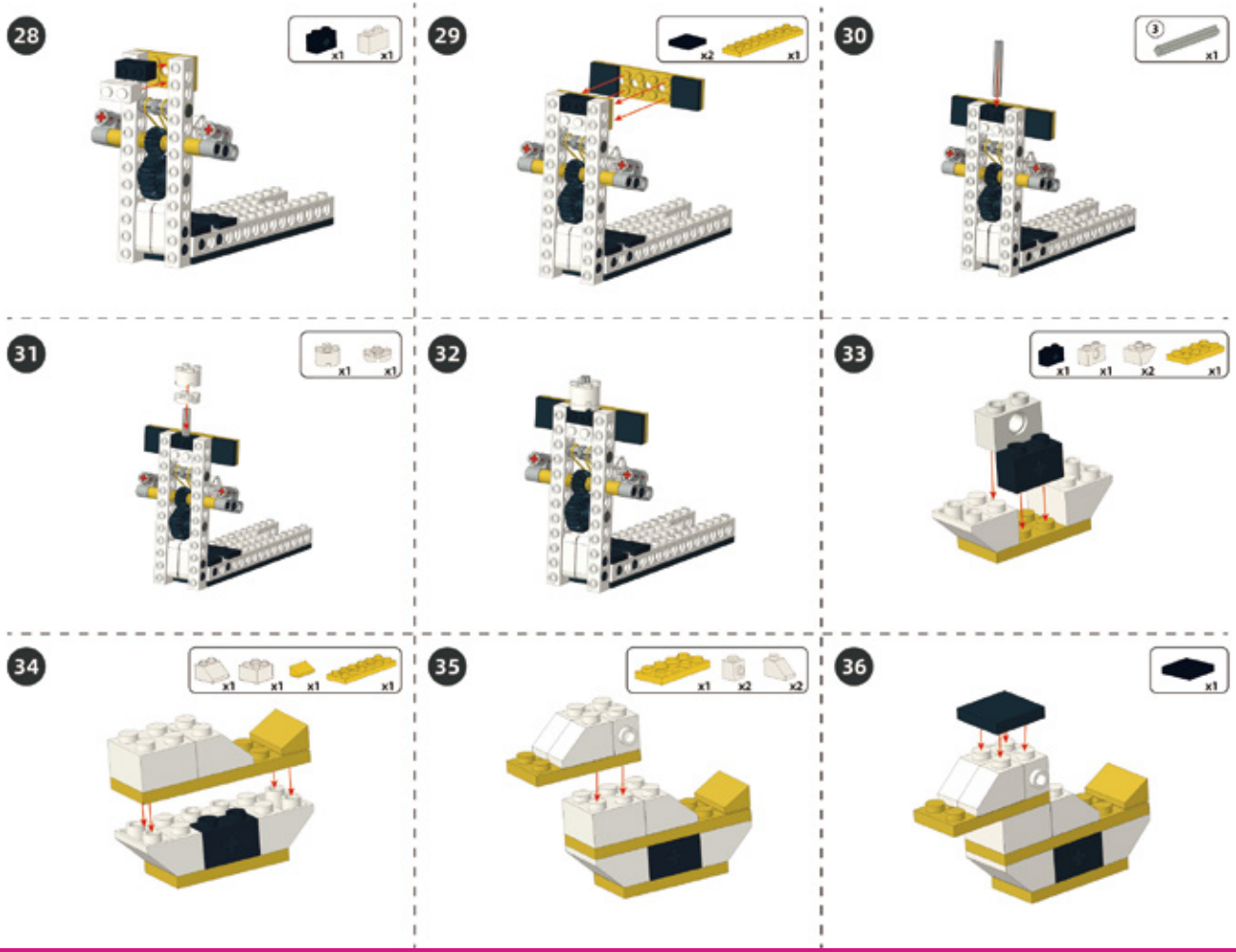
5. Model Building

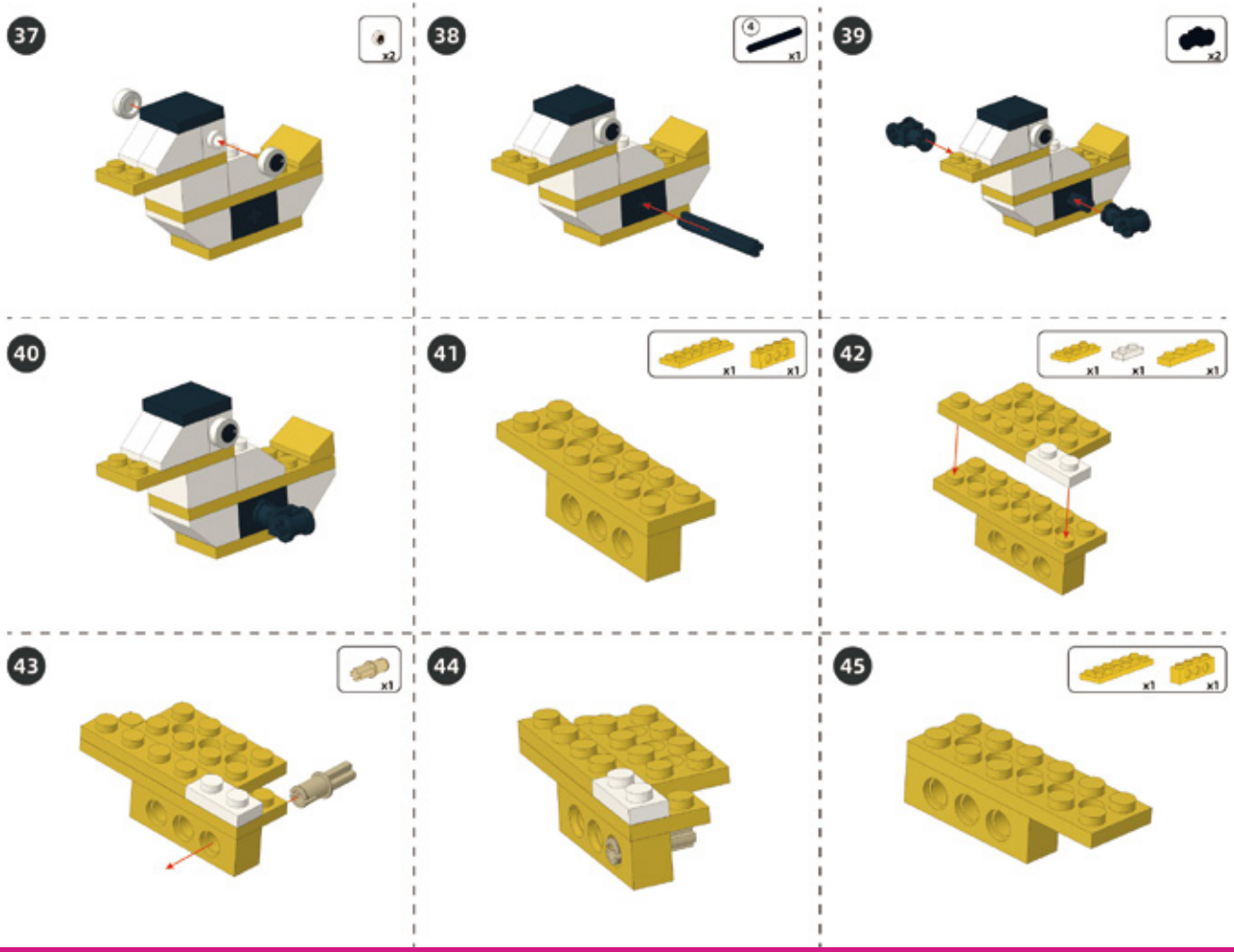
- **Building Blocks**

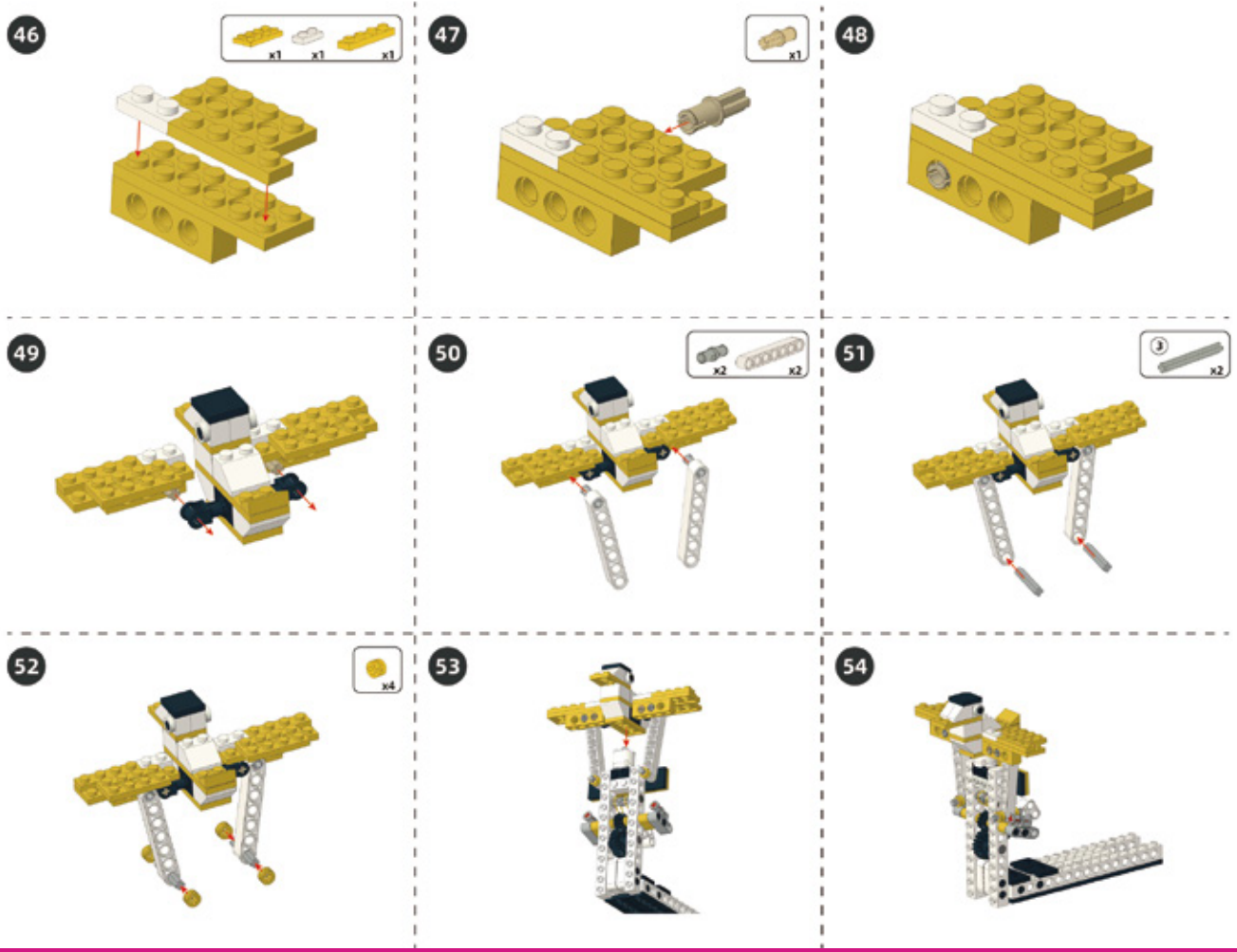

01
x2
x1
02
x1
03
x2
x1
04
x1
05
06
x2
07
x2
08
x4
09
x4


10
x2
11
12
2
x1
x1
13
x2
14
x1
15
16
x1
17
x2
x2
18
x1


19
20
x1
x2
21
x1
22
x2
23
x2
24
x2
25
x2
x2
26
27
x1


28
x1
x1
29
x2
x1
30
3
x1
31
x1
x1
32
33
x1
x1
x2
x1
34
x1
x1
x1
x1
35
x1
x2
x2
36
x1


37
x2
38
x1
39
x2
40
41
x1
x1
42
x1
x1
x1
43
x1
44
45
x1
x1


46
x1
x1
x1
47
x1
48
49
50
x2
x2
51
x2
52
x4
53
54

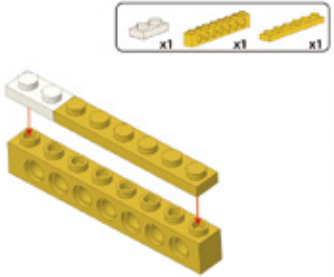
55

56

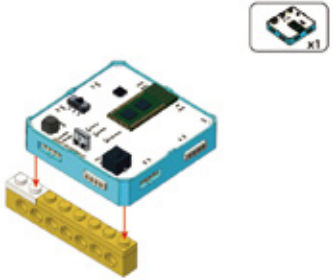
57

58

59



60



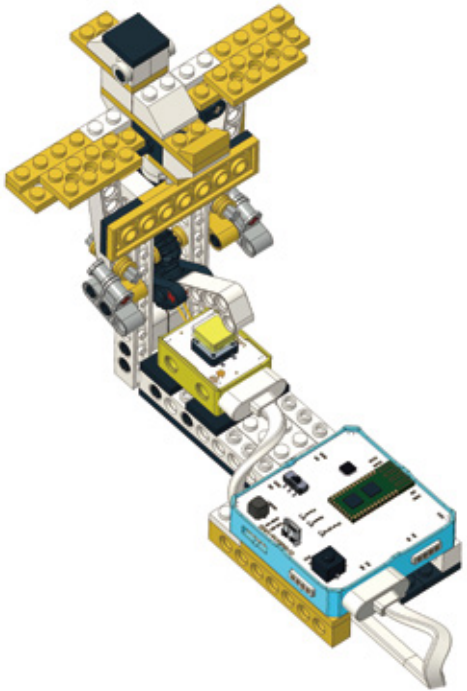
61

62

63

• Circuit Connection

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Let the bird fly. Control the bird to flap its wings and switch the flight attitude to avoid obstacles through the high and low level signals of the button.

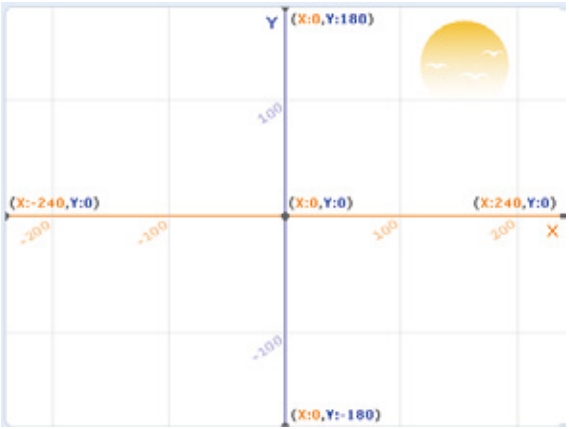
Task Analysis:

1. Understand the coordinate axis and movement;
2. Control the bird's flight by the high and low level signals of the button;
3. Make the obstacles move to the left.

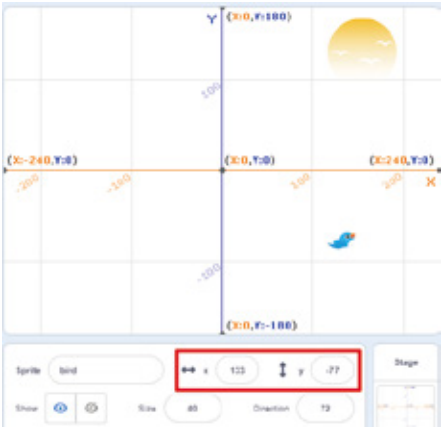
Task Steps:

Understand the coordinate axis and movement:

Movement means the position changes. The following figure shows the coordinate system of Letscode stage. Any point on the stage can be represented by (x, y). The coordinate value in the middle of the stage is (0, 0); the x-axis represents the horizontal coordinate value (range -240 to 240), and the y-axis represents the vertical coordinate value (range -180 to 180).



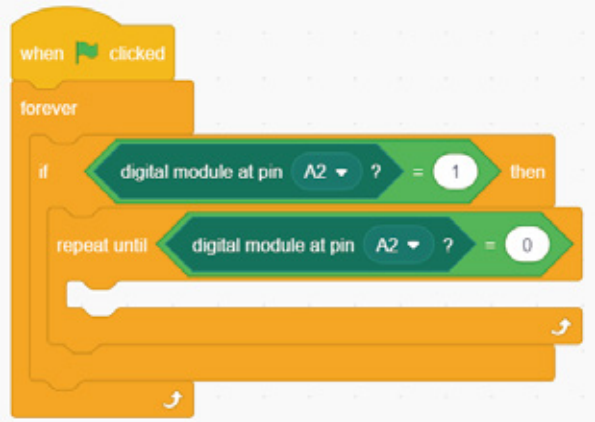
For ease of understanding, let's take a look at how its coordinate changes when the bird at different positions.



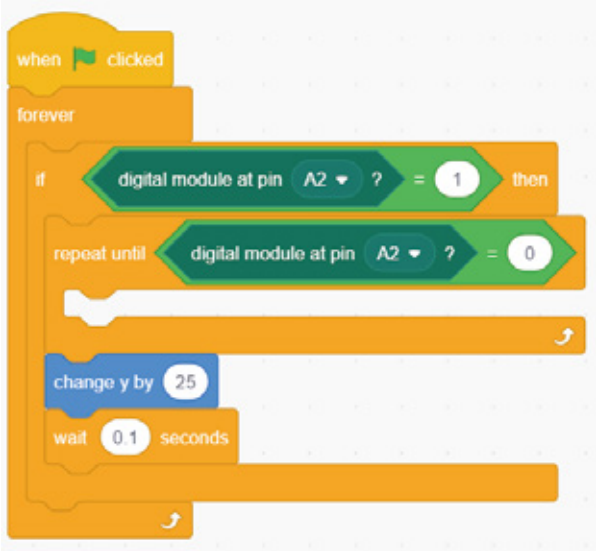
As shown in the figure above, the X and Y-position of the bird are different in different positions. As long as we change the coordinates of the sprite, we can change its position. Let's try to modify the coordinate values to make the sprite in a different position.

Control the bird's flight by the high and low level signals of the button:

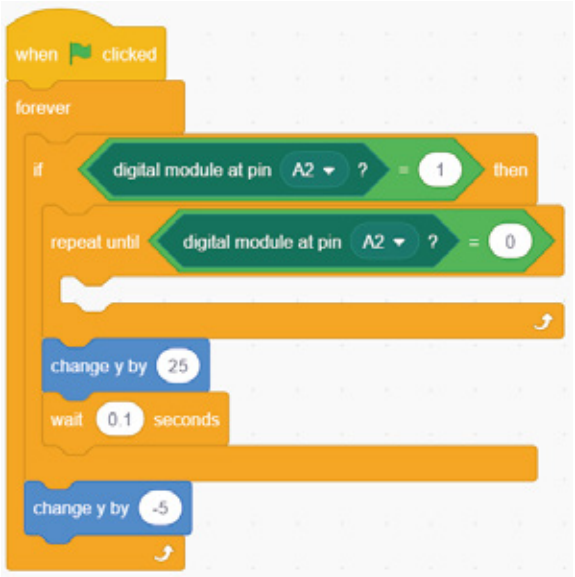
1. Same as the previous horse racing project, the button module completes a signal level change, and returns to the low level (released state), which can be considered as a trigger condition. Therefore, the program must firstly judge whether the button is pressed (that is, whether the level is 1), and let it repeat the empty program until it is released (that is, the level is 0), and then it will be considered as a trigger;



2. When pressed and released, the y-axis of the bird will increase by 25. In order to avoid instantaneous movements in a short period of time, a code block of waiting for 0.1 seconds can be added;



3. When the trigger condition is not established, the y-axis of the bird will increase -5, that is, the bird will continue to fall when the button is not pressed; in order to prevent the bird from falling down on the ground, we need to keep pressing and releasing the button.

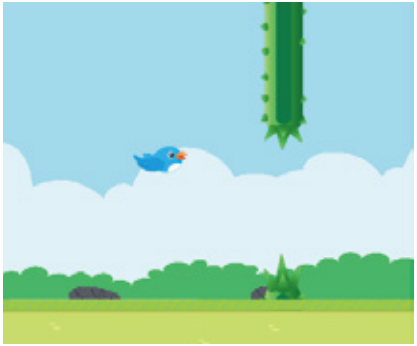


Make the obstacles move to the left:

In the game, from a visual point of view, the bird is moving to the right (the positive direction of the X-position), but in fact, the X-position of the bird does not change. How is this achieved?

1. It turns out that other sprites are moving to the left, and visually the bird is moving to the right. How to realize the sprite keeps moving to the left?

2. Firstly, set the sprite's X-position at the edge, that is, the position where the X-position is 240, and continue to move to the left. At this point you can see the bird flying forward.



There is no judgment about whether the bird hits an obstacle or not. Can you try to make the bird fall down on the ground directly after hitting an obstacle?

We have provided a complete crazy bird game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprite in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a crazy bird competition and see who can get the highest score. You can also modify the program according to your own ideas to increase the difficulty of the game. For example, you can increase the speed at which the bird falls, reduce the speed at which the bird rises each time, and increase the speed of obstacles to change the difficulty of the game.

8. Brainstorming

We used a button as a trigger module in this project. Buttons are often used in circuits, such as controlling switches. You can try to use this electronic module to design a virtual room light. When the button is pressed, the room light in the software is turned on, and when it is pressed again, the room light is turned off.

Project 5 Crazy Motorcycle

1. Learning Goals

1. Know the competition event: motocross;
2. Learn the operating principle of vibration motor and its interactive programming control;
3. Complete the model building and game programming of motorcycle.

2. Application

Motocross is a closed competition project carried out on natural obstacles or complex terrain. The route includes special terrain such as steep slopes, undulating roads, sandy and muddy roads. The scoring method is based on athletes participating in consecutive competitions held in different countries throughout the year, and accumulating the scores of each competition, and finally determining the world champion.

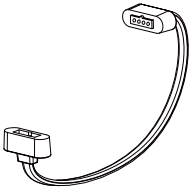
Next, we will use building blocks and electronic modules to build a motorcycle, and control the motorcycle on the game side through the motorcycle model to achieve the interactive effect of the crazy motorcycle game.

3. Supplies List

• **Electronic Supplies:**

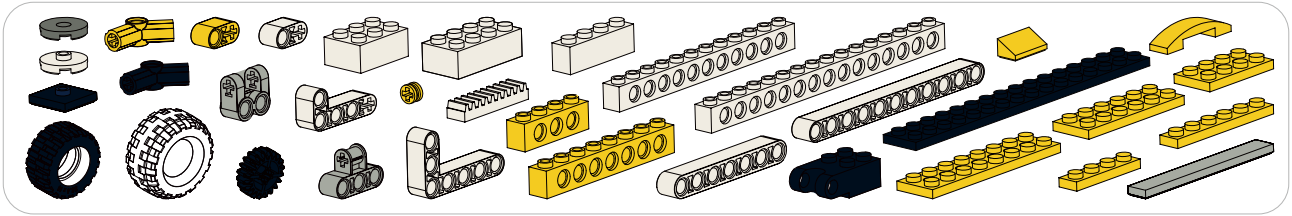
Vibration Motor x1

Crowbits-UNO x1



Magnetic Cable x1

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



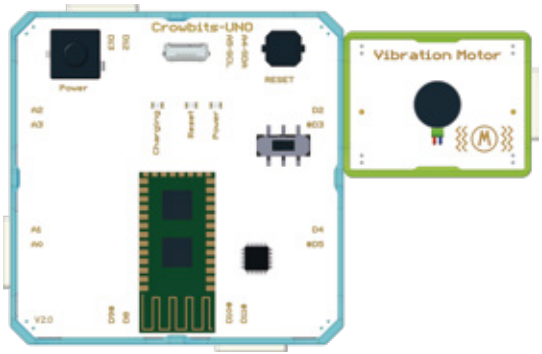
4. Knowledge Analysis

• **Vibration Motor**

1. The vibration motor is a digital output module with a miniature motor inside and an eccentric wheel on the motor shaft.

Working principle: When receiving a high level, the motor rotates and drives the eccentric wheel to rotate. The center mass point of the eccentric wheel is not on the rotation center of the motor so that the motor is constantly out of balance and cause vibration due to inertia. When the low level is received, the motor stops vibrating.

2. Circuit Connection:

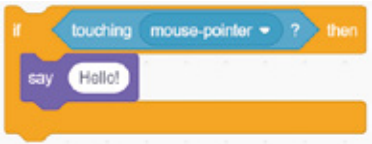
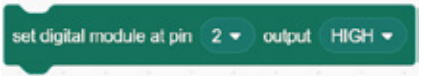
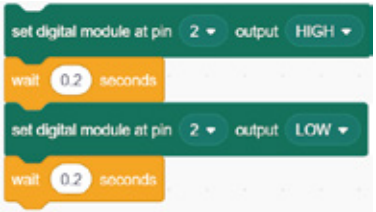


3. Program Driven:

Vibration motor programming control logic:

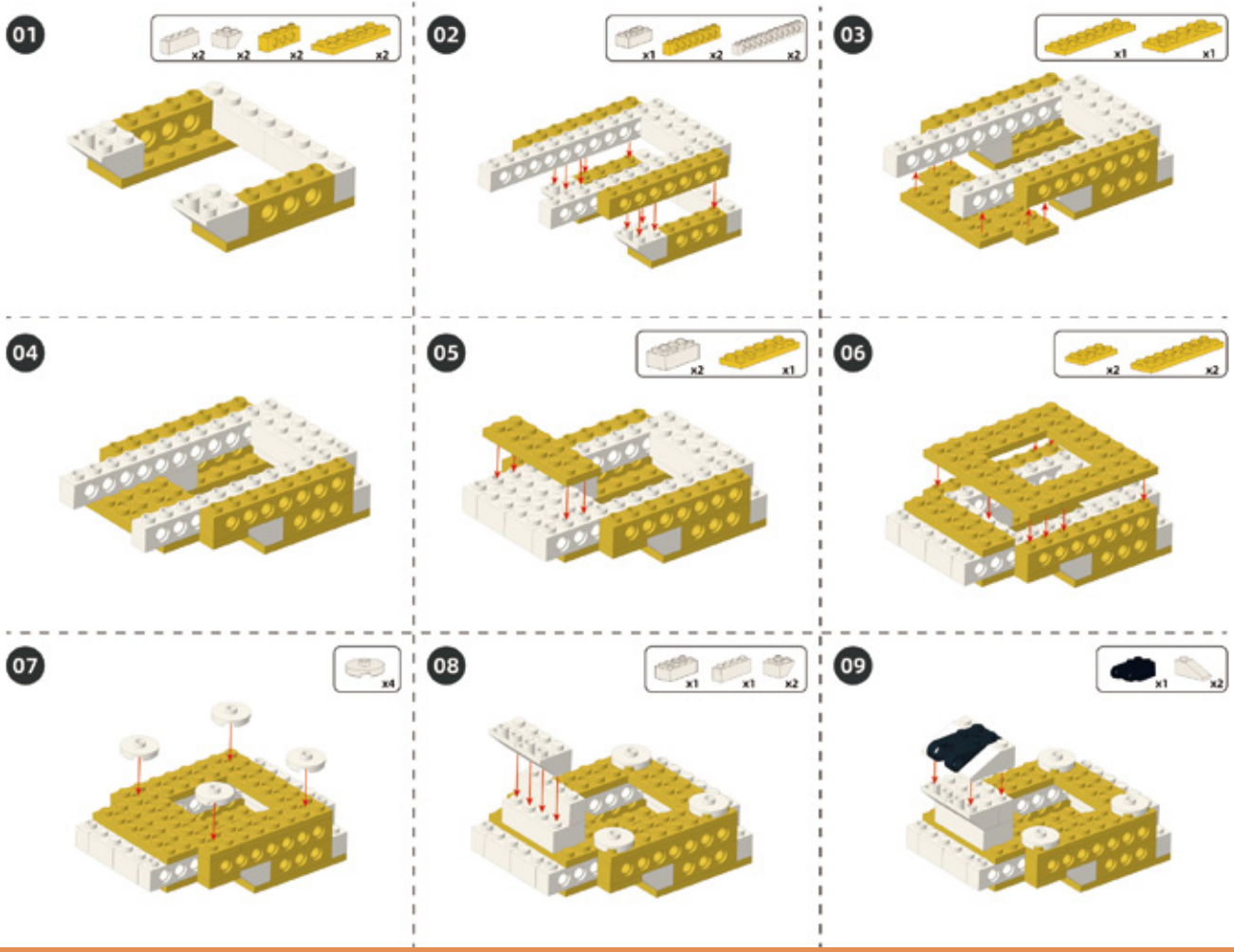
Signal input	Status
high level	vibration motor vibrates
low level	vibration motor stops vibrating

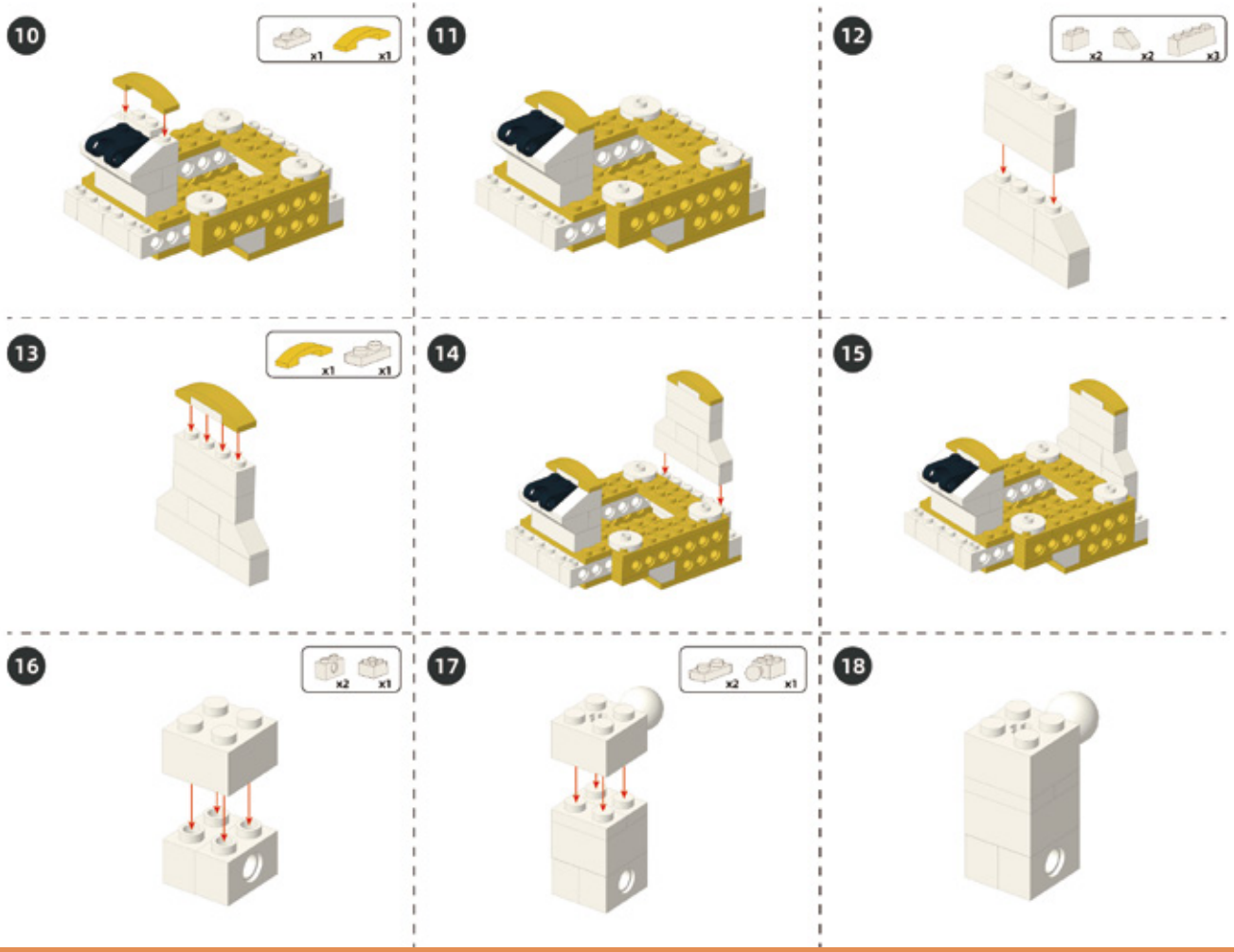
Next, we will use the following blocks:

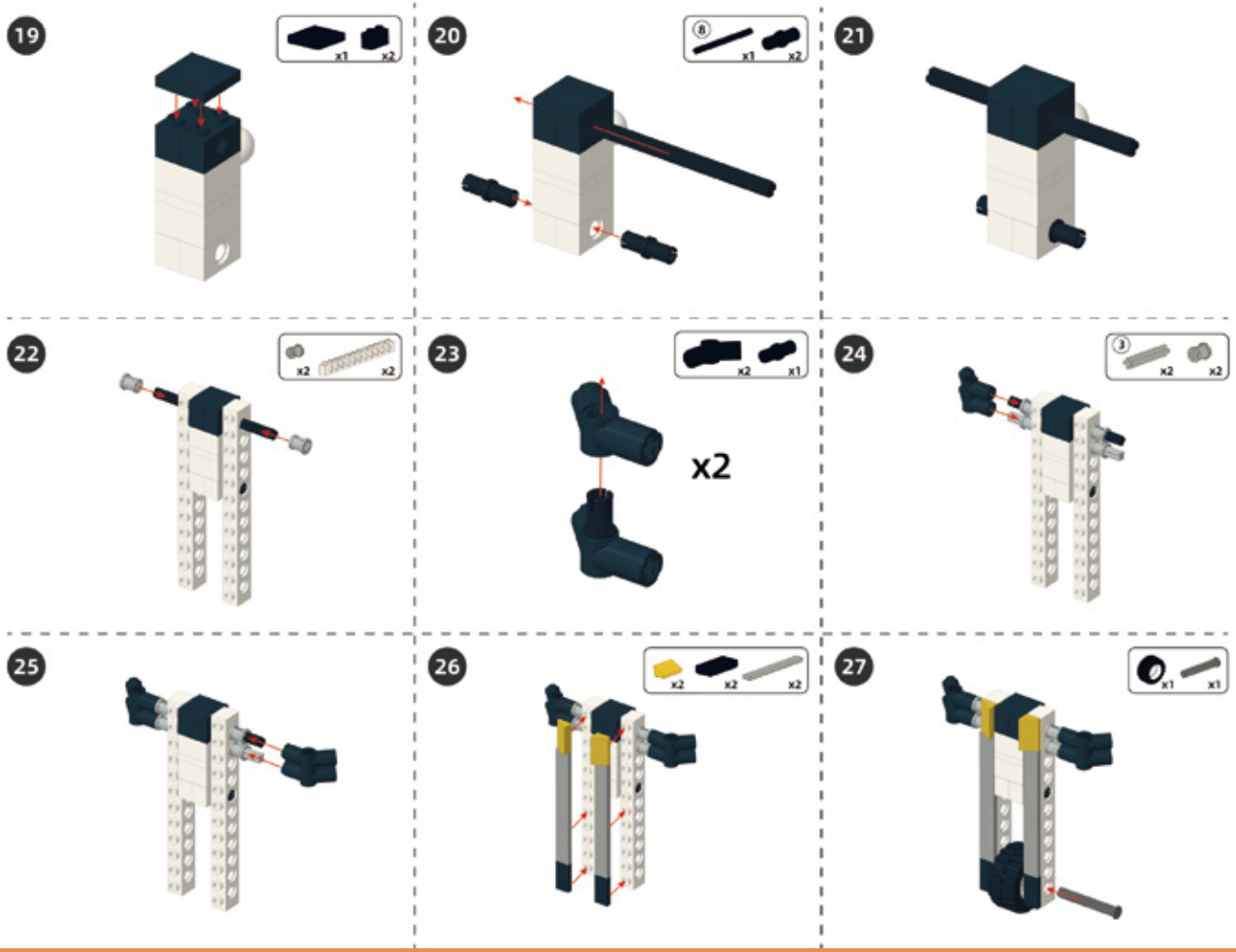
Block	Example
 Repeat: limited cycle logic code block, control the number of executions of the internal program through the set times	 Analysis: the Y-position is increased by 6 each time and repeated 10 times, that is, increased by 60, and the total time spent is 1 second
 Touch the mouse pointer or other sprites? Judge whether the sprite touches the mouse pointer, the return value is 1 when touched, and 0 is returned when not touched. The drop-down menu can be changed to other sprites	 Analysis: judge whether the sprite touch the mouse pointer, say "hello" if touch
 Set the pin level signal of the digital module: set different pins to output high level or low level	 Analysis: set pin 2 to output high level, and wait for 0.2 seconds; set pin 2 to output low level, and then wait 0.2 seconds. If pin 2 is connected to a vibration motor, it will vibrate intermittently

5. Model Building

- **Building Blocks**


01
x2 x2 x2 x2
02
x1 x2 x2
03
x1 x1
04
05
x2 x1
06
x2 x2
07
x4
08
x1 x1 x2
09
x1 x2

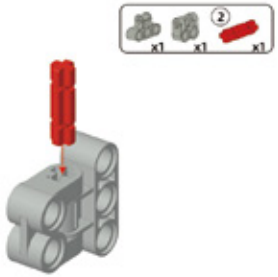

10
x1
x1
11
12
x2
x2
x3
13
x1
x1
14
15
16
x2
x1
17
x2
x1
18


19
x1
x2
20
8
x1
x2
21
22
x2
x2
23
x2
x1
x2
24
3
x2
x2
25
26
x2
x2
x2
27
x1
x1

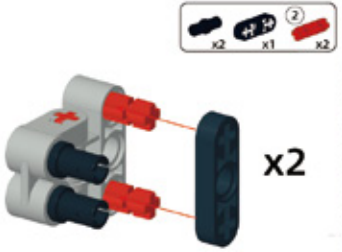
28

29

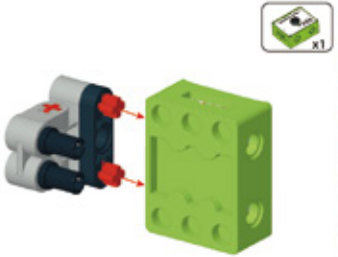
30



31



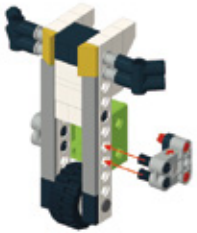
32



33

34

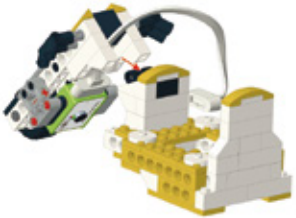
35



36

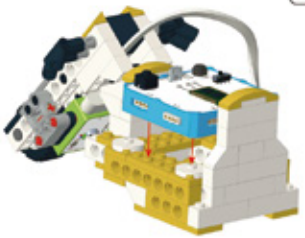
37

38



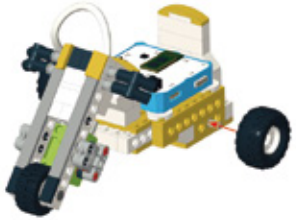
39

40



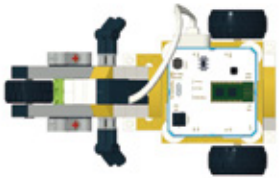
41

42



43

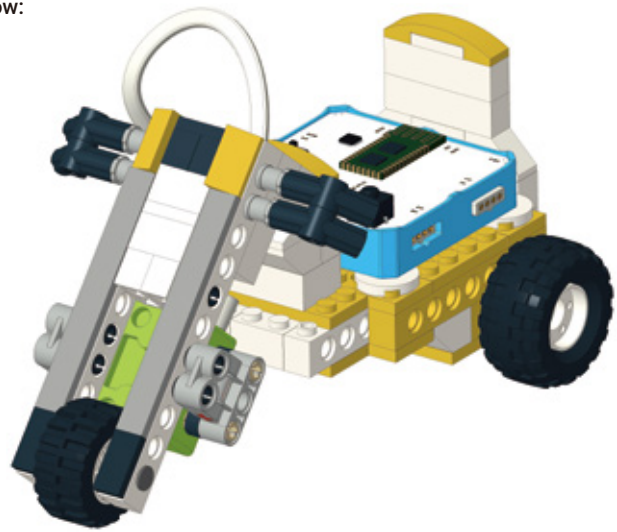
44



45

• Circuit Connection

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Control motorcycle. Control the motorcycle to jump when the main board is deflected upward, and the vibration motor will vibrate when the motorcycle falls freely.

Task Analysis:

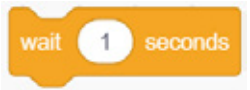
1. The program of vibration motor vibrates;
2. The motorcycle jumps when the main control board deflects upward;

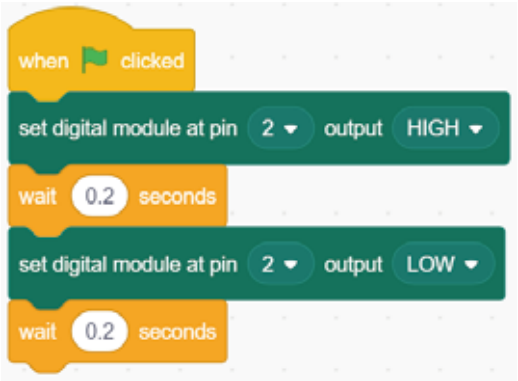
Task Steps:

The program of vibration motor vibrates:

1. Through this code block **[set digital module at pin "2" output "HIGH"]**, set the vibration motor pin and high level to control the vibration of the vibration motor;

2. Through this code block **[wait "1" seconds]**, adjust the time interval to experience the different vibration effects of the motor. The program for one vibration is as follows:



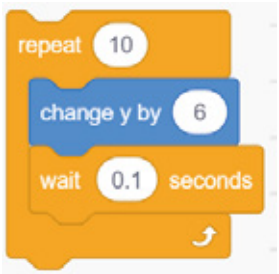


3. When the green flag is clicked, the vibration motor will vibrate for 0.2 seconds, and then stop. You can add cycle instructions to perform multiple tests and adjust the best time interval and vibration effect.

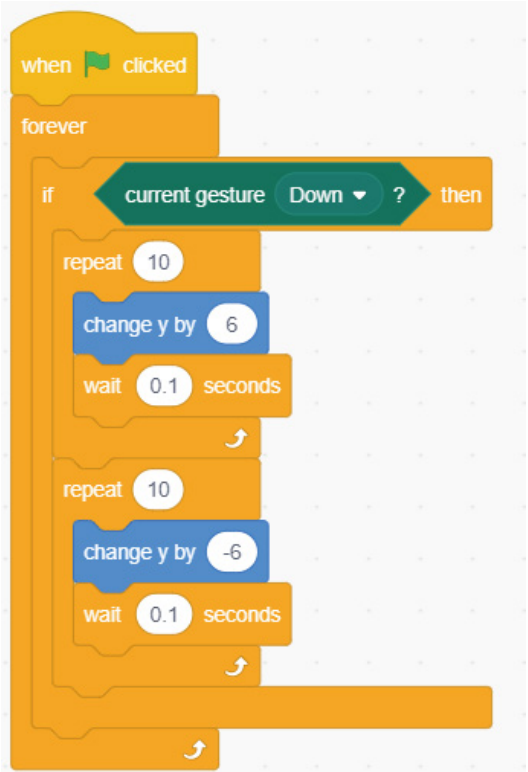
The motorcycle jumps when the main control board deflects upward:

(The motorcycle and the sprite are regarded as a whole here.)

1. In order to achieve the effect of the motorcycle jumping up, you need to use this **[change y by "10"]** code block. For the realistic jumping effect, you can execute this block multiple times to achieve the effect of slow jumping.

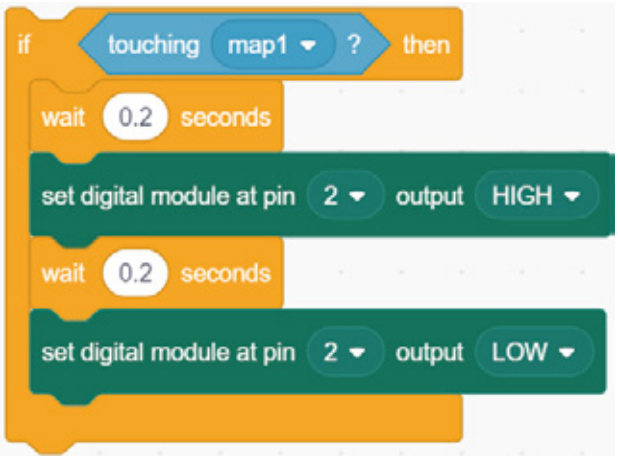


2. When the "motorcycle" jumps up, there will be a falling process before it falls back to the ground. If the sprite jumps up to a height of 60, then it has to fall to the same height before it will fall back to the jumping position. The entire jump and fall procedure needs to be repeated to determine whether the condition is triggered. The complete program is as follows:

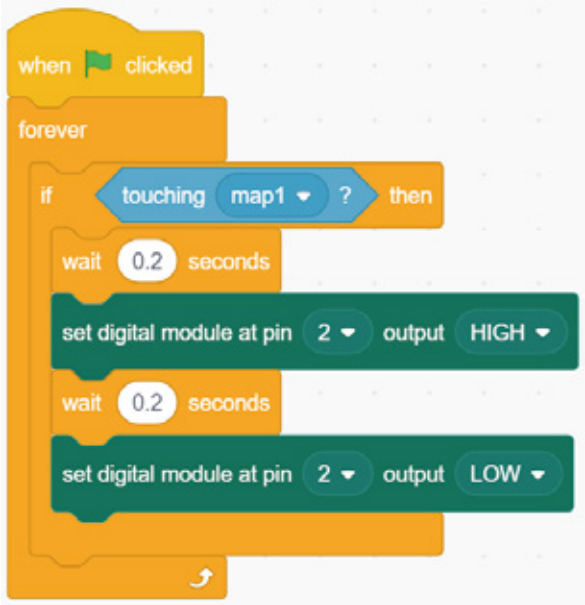


3. When the motorcycle sprite jumps up and falls back to the ground, the vibration motor vibrates to simulate the effect of wheels hitting the ground. The program is used to judge whether the wheels of the motorcycle touch the ground. If the wheels touch the ground, the vibration motor will vibrate.

4. Firstly, we need to use the code block **[touching "mouse-pointer"]** to determine whether the wheel touches the ground. Click the inverted triangle in the code block, and select the sprite need to be touched, then use the judgment code block to determine whether to trigger.



5. The explanation of the above program is that when the sprite touches the ground, the vibration motor will vibrate. Don't forget that you need to repeatedly judge whether the motorcycle touches the ground, so you need to add "forever" block.



Funny games may be very complicated. We have only learned the basic principles. Based on the knowledge mentioned above, you can try to write a program that triggers vibration after a motorcycle jumps and falls, and check what conditions you didn't expect that caused the game to run improperly.

We have provided a complete crazy motorcycle game in the software resources. You can download it and try it out. Let the motorcycle on the game side go crazy by controlling the motorcycle model! Other programs in the game require you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your friends to a crazy motorcycle competition and see who can get the highest score! You can modify the data in the game program according to your own ideas to increase the difficulty of the game. For example, increase the speed of a motorcycle or increase the frequency of obstacles to increase the difficulty of the game.

8. Brainstorming

We used vibration motor and six-axis gyroscope as the trigger modules in this project. Vibration motors are often used in mobile phones or gamepads. You can try to use this electronic module, combined with software programming, to design an earthquake alarm.

Project 6 Tank Wars

1. Learning Goals

1. know military weapons: tanks;
2. Learn the operating principle of IR reflective sensor, six-axis gyroscope, button and them interactive programming control;
3. Complete the model building and game programming of joystick.

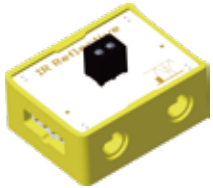
2. Application

Tanks first appeared in Britain and are one of the main weapons of modern land warfare. Tanks can suppress and destroy anti-tank weapons, fortifications and enemy land weapons.

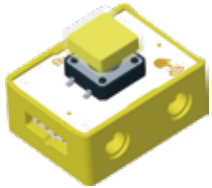
Next, let's build a joystick using blocks and electronic modules, to realize the interaction of tank wars through controlling the tanks in the game by using the joystick model.

3. Supplies List

• **Electronic Supplies:**

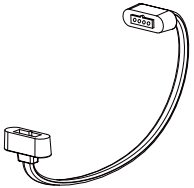


IR Reflective Sensor x1



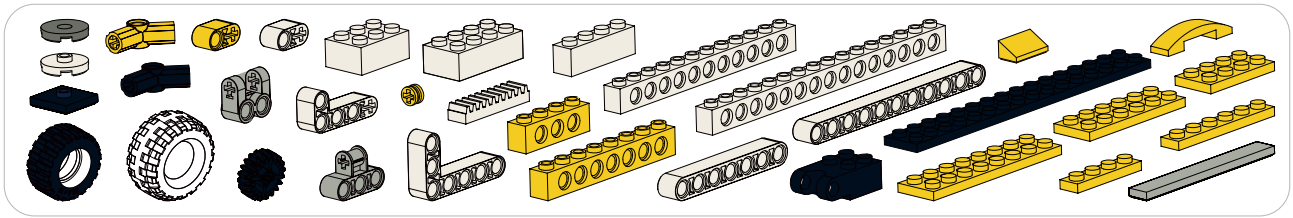
Button Module x1

Crowbits-UNO x1



Magnetic Cable x2

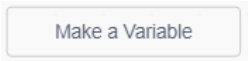
• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



4. Knowledge Analysis

• **What is a variable?**

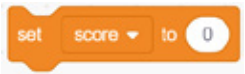
Do you still remember the game “catch the fruit”? The more fruit you catch, the more scores you get. We set the score to be 0 and make the score increased by one after receiving a fruit. The scores will constantly be accumulating until the game is over. The scores here are variable. The following blocks will be used when using variables.

<table>
<tr><th>Block</th><th>Example</th></tr>
<tr><td> 

Make a variable: Click to create a new variable</td><td> 

Analysis: you need to set the name and choose the available sprites for it when you create a variable</td></tr>
<tr><td> 

Variable name: Create a variable block. Click the check box next to block, then value of variable would display in the stage area</td><td rowspan="3"> 

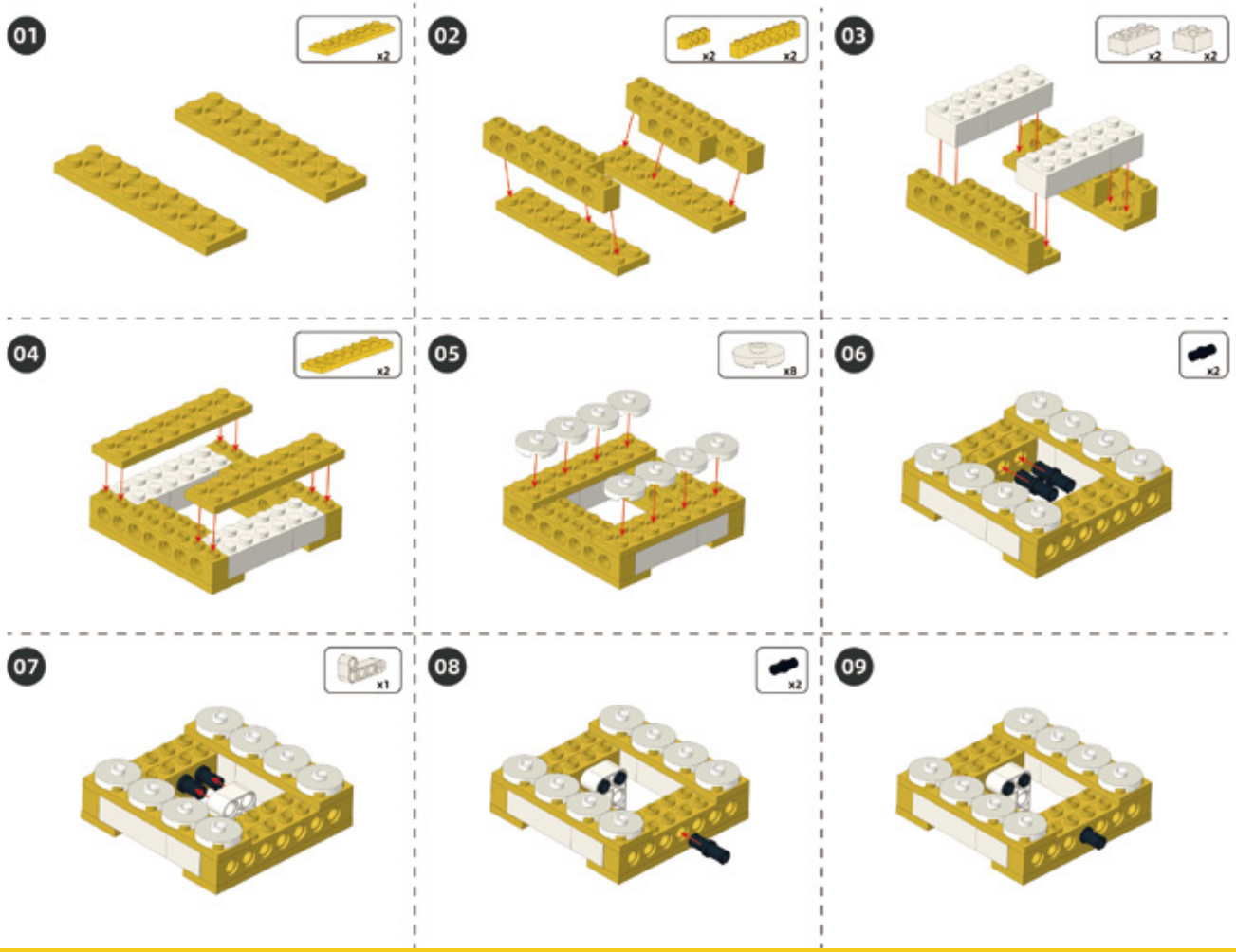
Analysis: The initial value of variable "score" is 0. If the sprite touches the fruit, then increase the variable “scores” by 1</td></tr>
<tr><td> 

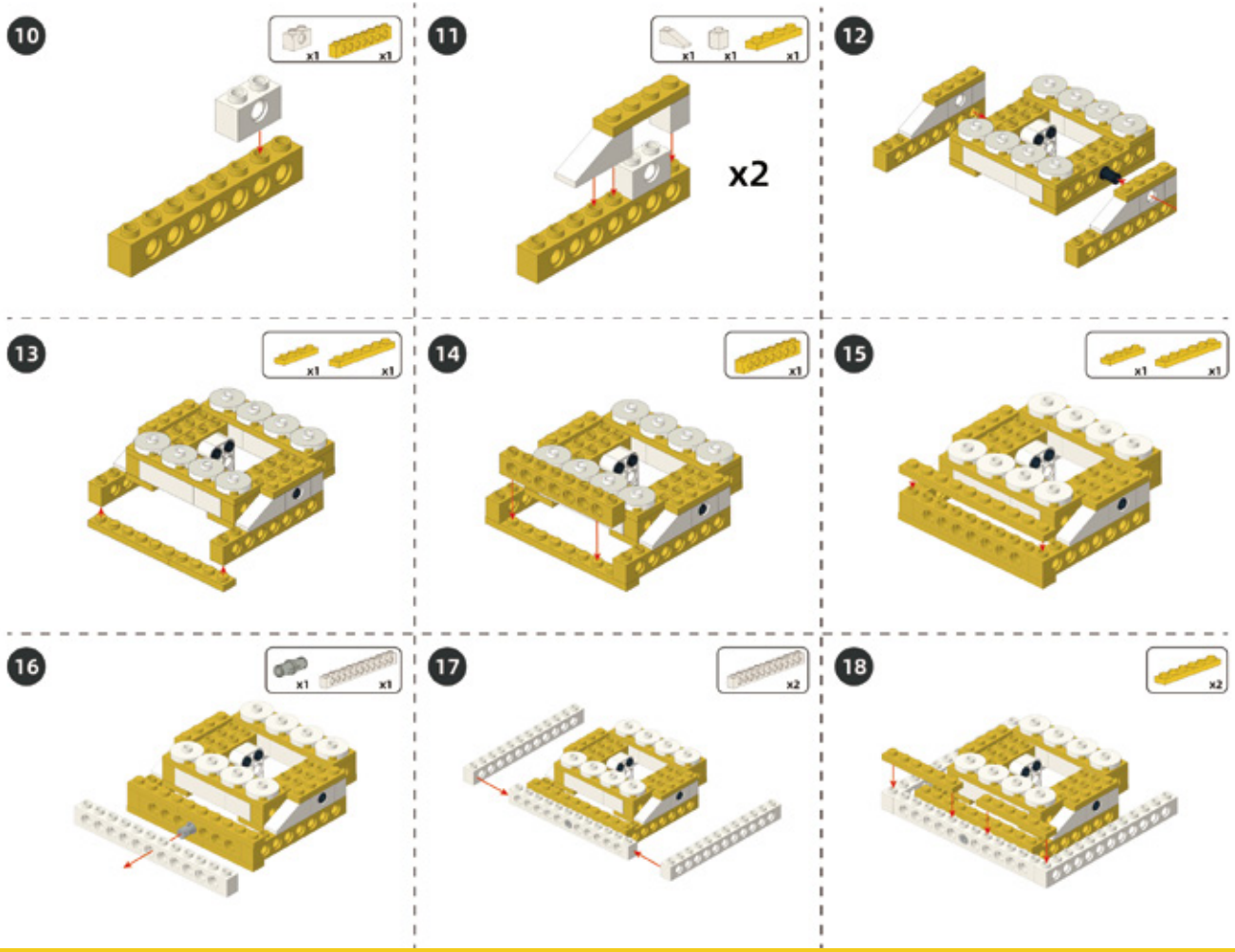
Set variable value: Set the variable to be a specified value, if there are many variables, you could select the one you want from the drop-down menu</td></tr>
<tr><td> 

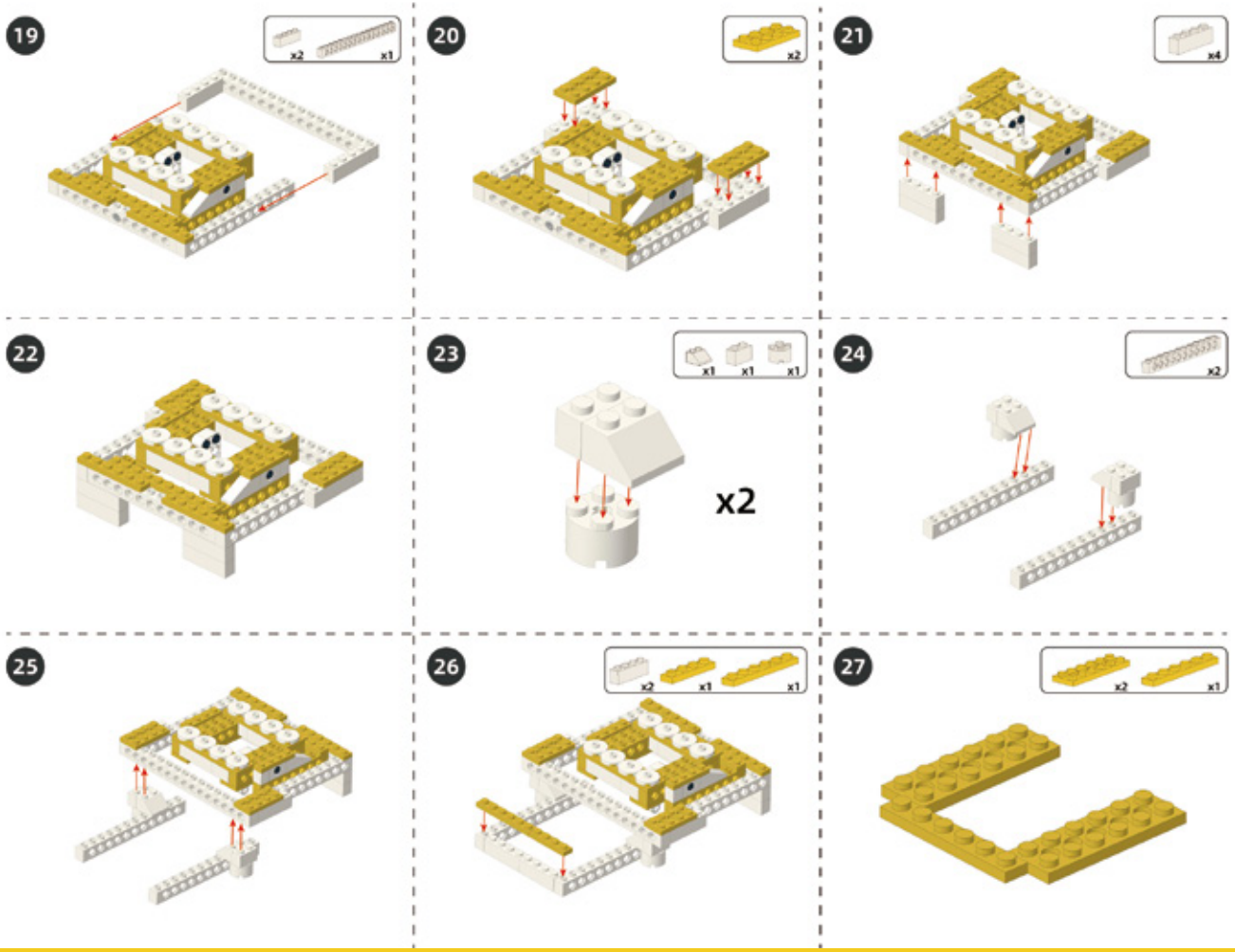
Change the variable’s value: Used to change the value of current variable</td></tr>
</table>

5. Model Building

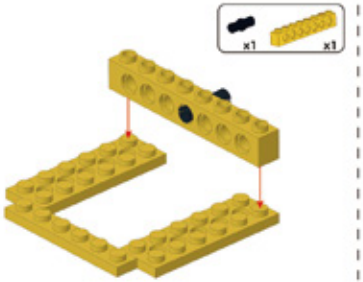
- **Building Blocks**


01
x2
02
x2
x2
03
x2
x2
04
x2
05
x8
06
x2
07
x1
08
x2
09

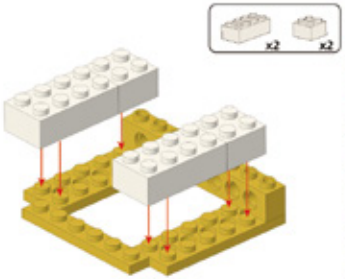

10
x1
x1
11
x1
x1
x1
x2
12
13
x1
x1
14
x1
15
x1
x1
16
x1
x1
17
x2
18
x2


19
x2
x1
20
x2
21
x4
22
23
x1
x1
x1
x2
24
x2
25
26
x2
x1
x1
27
x2
x1

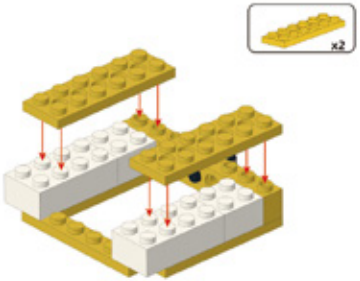
28



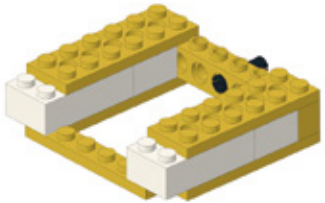
29



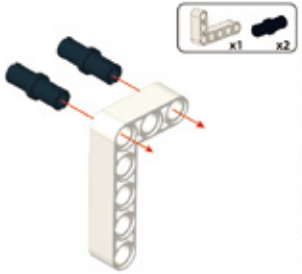
30



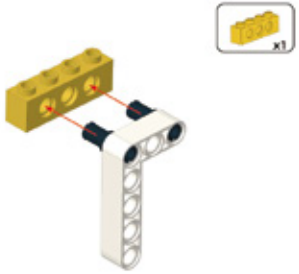
31



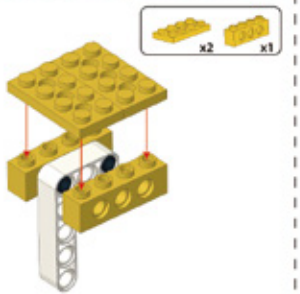
32



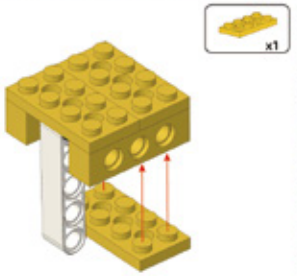
33



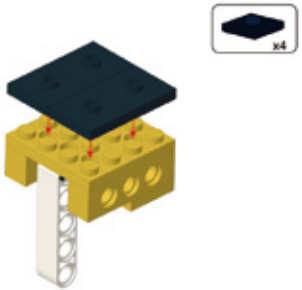
34

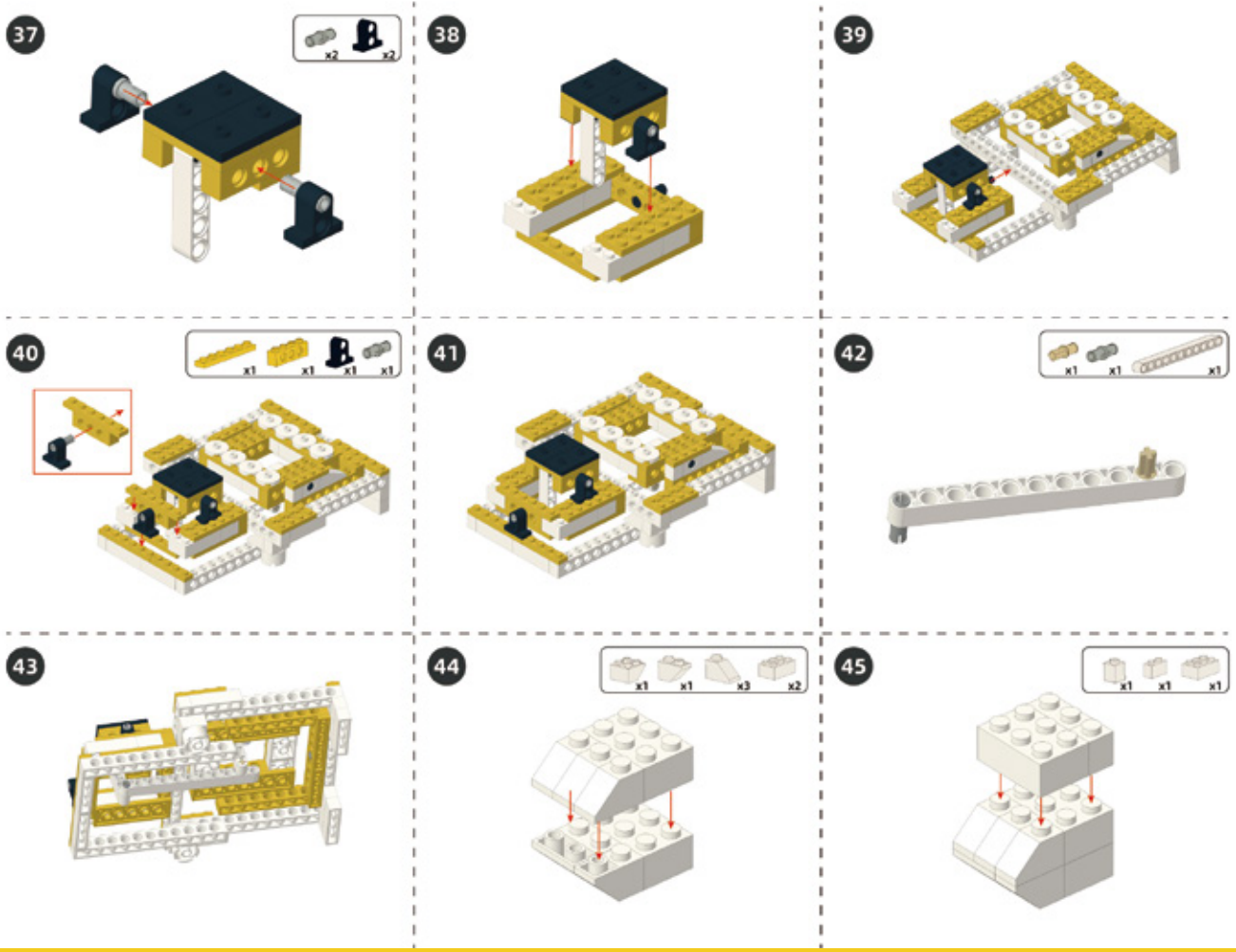


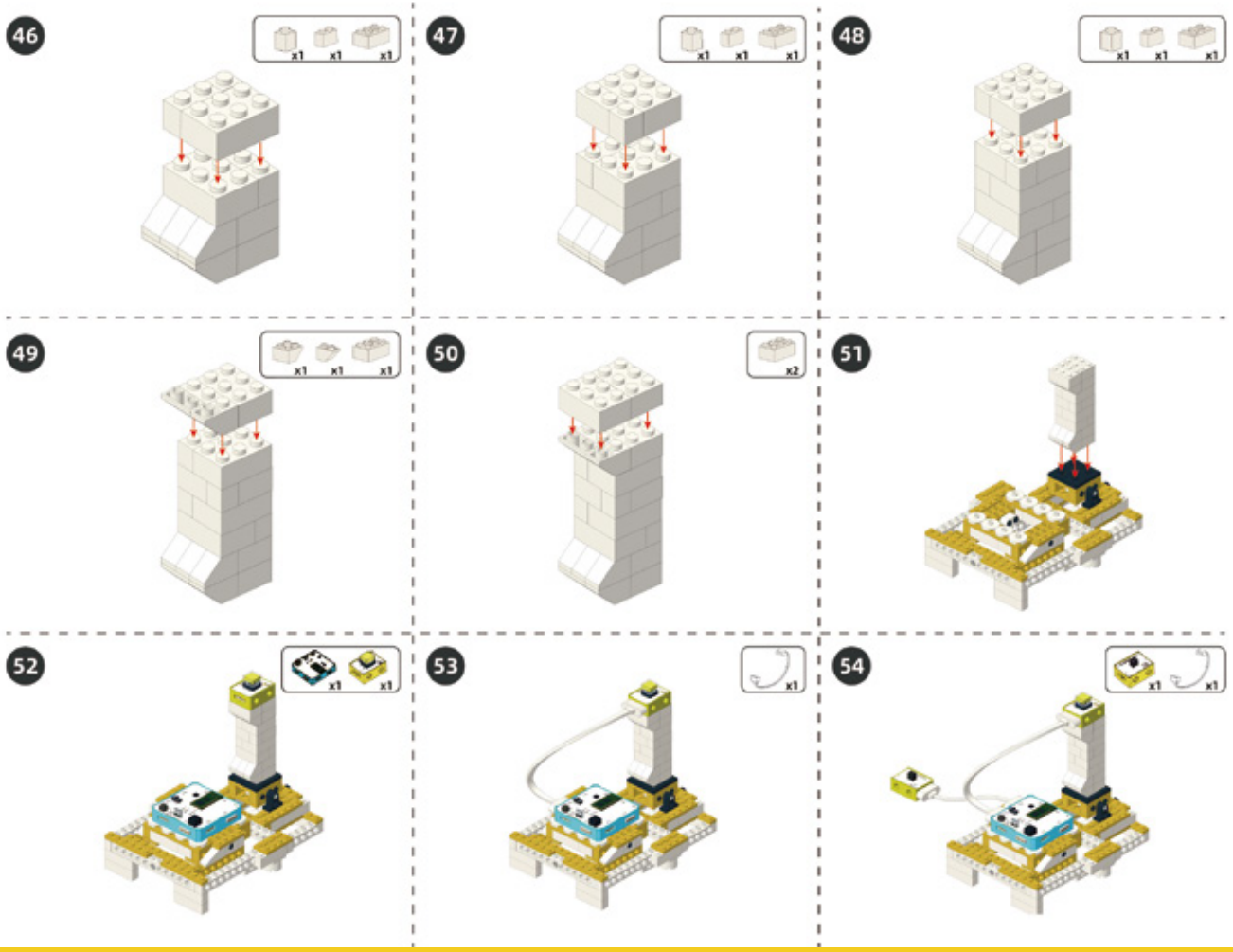
35



36

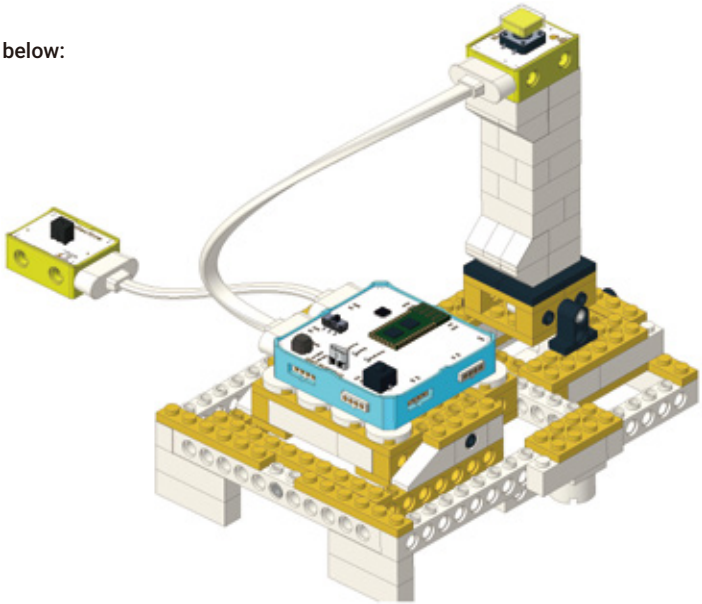



37
x2
x2
38
39
40
x1
x1
x1
x1
41
42
x1
x1
x1
43
44
x1
x1
x3
x2
45
x1
x1
x1


46
x1 x1 x1
47
x1 x1 x1
48
x1 x1 x1
49
x1 x1 x1
50
x2
51
52
x1 x1
53
x1
54
x1 x1

• **Circuit Connection**

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Control the tank movement and add health percentage(HP) for it.

Task Analysis:

1. Control the tank to move.
2. Create a new variable of the tank as HP, when the tank is bombarded, its HP will decrease.

Task Steps:

Control the tank movement:

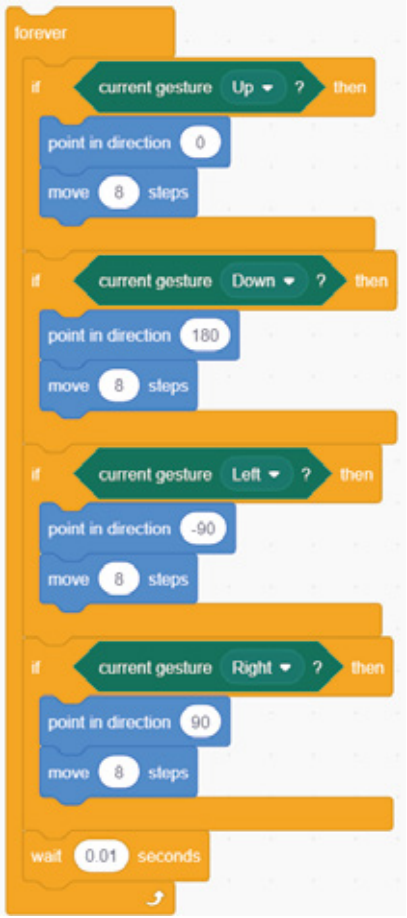
We have learned to detect the left and right deflection of the six-axis gyroscope to control the lateral movement of the pig. On this basis, the detection of the vertical position of the six-axis gyroscope is added to control the up and down movement, as shown in the right picture.

When the main board is deflecting upward, the tank moves upward (0° direction) 8 steps.

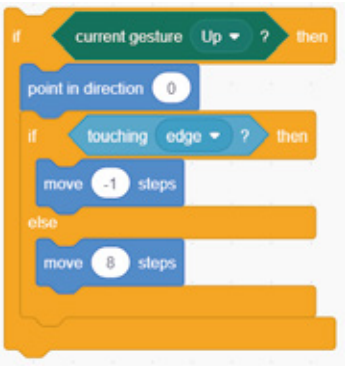
When the main board is deflecting downward, the tank moves downward (180° direction) 8 steps.

When the main board is deflecting to left, the tank moves to the left (-90°direction) 8 steps.

When the main board is deflecting to right, the tank moves to the right (90°direction) 8 steps.

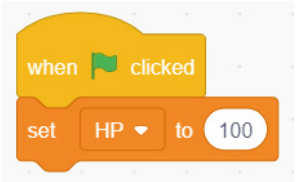


The tank might move out of the stage when it moves, so we need to limit the tank movement area into the stage area.



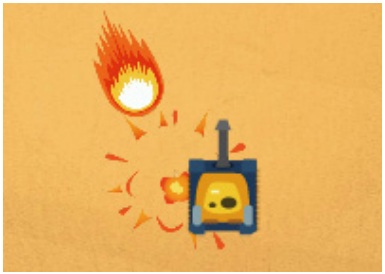
When the main board is deflecting upward, the tank will also move upward, when the tank moves to the edge of stage area, it will not move forward again, and on the contrary, it will move backward. In order to avoid the tank getting stuck in the edge and get in a loop when it moves to the edge, we could move -1 step. Similarly, we could program for downward, left and right direction in the same way.

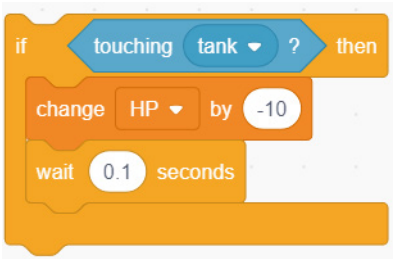
Create a new variable for the tank as HP, when the tank is bombarded, its HP will decrease.



We can create a new variable named HP and set it to 100, which means the tank's initial HP is 100.

When a cannonball hits the tank, the tank HP would increase by -10, in order to avoid repetitive judgment, we could add block "wait 0.1 seconds".





Tank's HP is 100. When it is hit once, its HP would reduce by 10, if the tank is hit 10 times, then its run out the HP, can you try to add a notification for game over? For instance, the tank will explode once its HP is 0.

We have provided a complete tank wars game in software resources, you can download it and try it out to see the difference between this one we did. The code of other sprites in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends together!

Invite your friends to a tank wars competition and see who can survive to the end. You can also modify the program according to your own ideas to increase the difficulty and experience of the game. For example, increase the speed of the bullets for each other, or increase the frequency of firing bullets to increase the difficulty.

8. Brainstorming

In the tank wars game, we used IR reflective sensor, six-axis gyroscope, button as trigger modules. Buttons are usually used in the electronic circuit, such as controlling switch. You would try to use these modules to make a game “Shooting a frisbee”: programming to make the frisbee parabola appear, when it passes the shooting area, use the button as a trigger module to shoot it down.

Project 7 Penalty Shootout

1. Learning Goals

1. Know the football term: penalty kick;
2. Extend the application of linear potentiometer and six-axis gyroscope;
3. Complete the model building and game programming of joystick and goalkeeper.

2. Application

If a team in the penalty area of its own team has violated one of the ten types of fouls that can be judged as a direct free kick, it will be awarded a free kick and a penalty kick shall be executed. In addition to penalties for fouls, most world and continental football matches have introduced "penalty shootout" since the 1970s. That is, after the 90-point regular game and the 30-minute overtime are over, the winner will be decided through rounds of penalties.

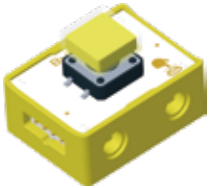
Next, we will use building blocks and electronic modules to build a joystick and goalkeeper, and control the players and goalkeepers on the game side through the model to achieve the interactive effect of the penalty shootout.

3. Supplies List

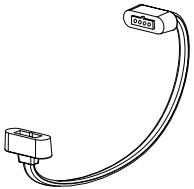
• **Electronic Supplies:**

Crowbits-UNO x1

Linear Potentiometer x1

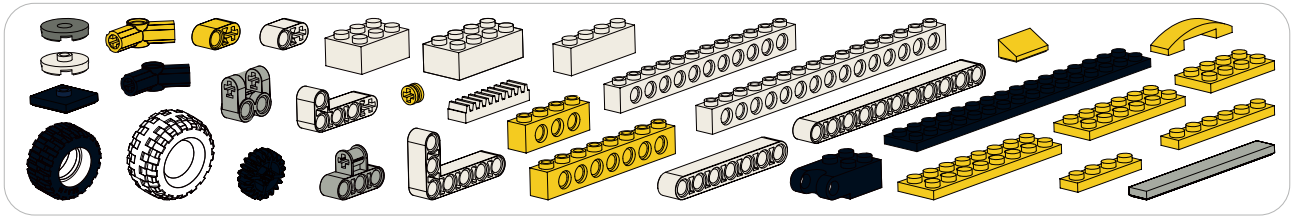


Button Module x1



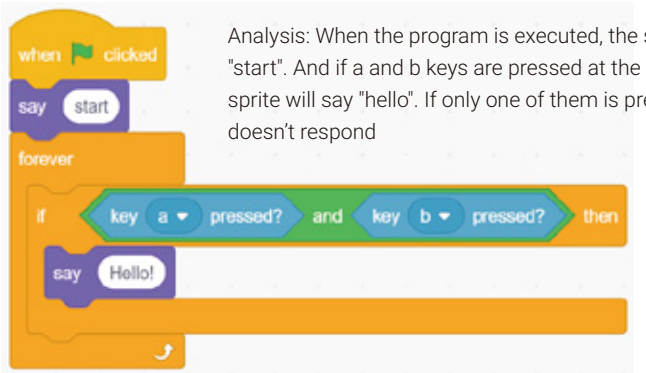
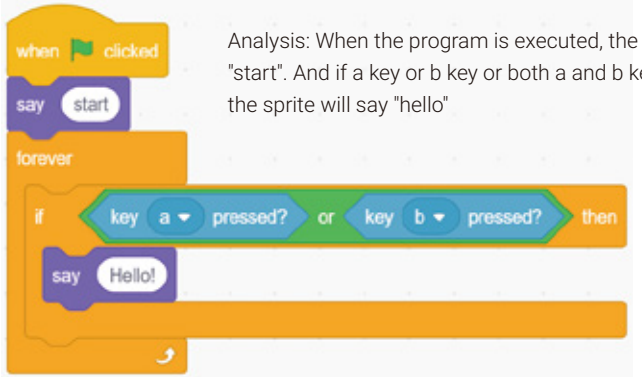
Magnetic Cable x2

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**

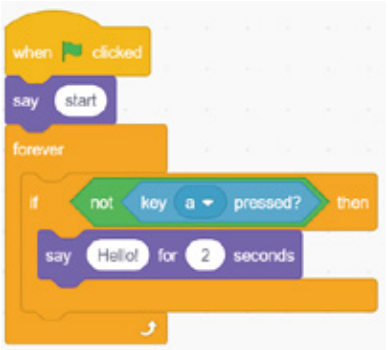
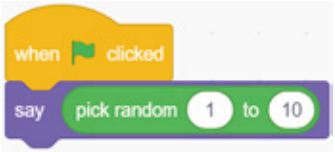


4. Knowledge Analysis

We will use the following blocks in this project:

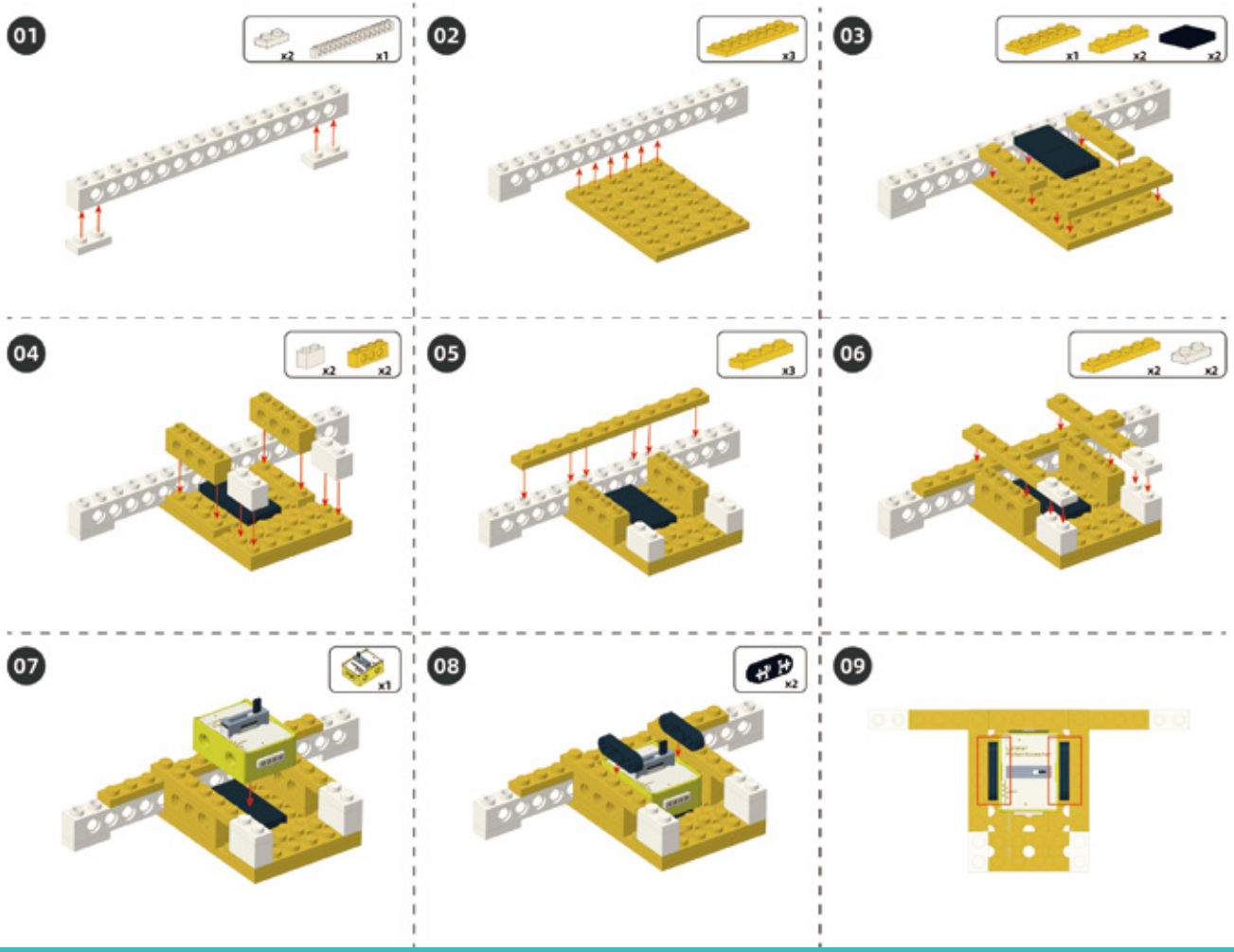
Block	Example
 And logic: Only when the expressions on both sides are valid at the same time, the logic is valid; otherwise, the logic is not valid	 Analysis: When the program is executed, the sprite will say "start". And if a and b keys are pressed at the same time, the sprite will say "hello". If only one of them is pressed, the sprite doesn't respond
 Or logic: As long as one of the expressions on both sides meets the condition, its logic is established	 Analysis: When the program is executed, the sprite will say "start". And if a key or b key or both a and b keys are pressed, the sprite will say "hello"

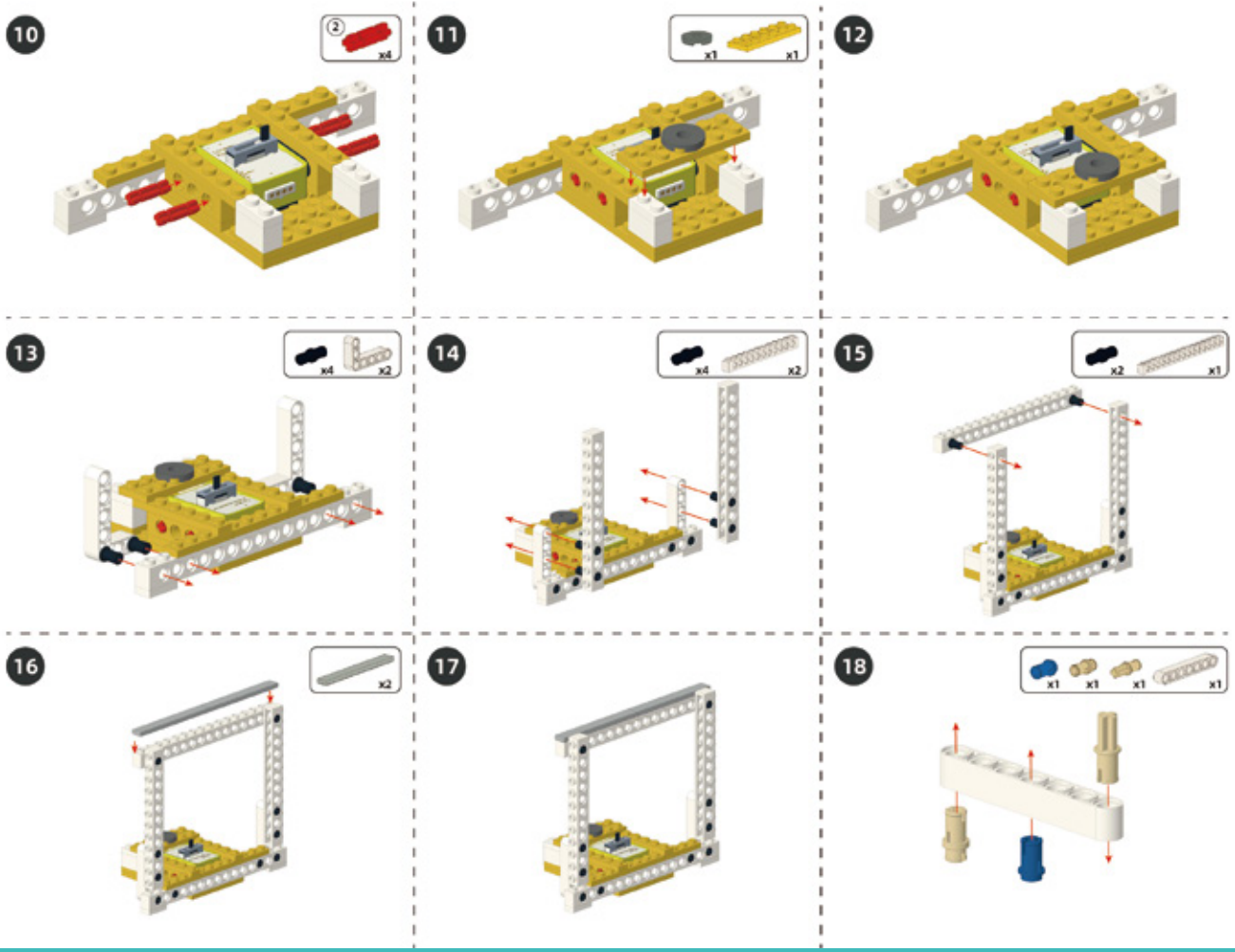
Continued

 Not logic: When the expression is not established, its logic is established	 Analysis: When the program is executed, if pressing key a, no statement is executed; if key a is released, the program saying "Hello" is executed
 Take a random number: Take out any random number between the two values	 Analysis: When the green flag is clicked, the sprite will say a number between 1 and 10
 Go to a random position: The sprite moves to a random coordinate position, can also move to the position of other sprites	 Analysis: when the green flag is clicked, the sprite will move to the position of the "goal" sprite

5. Model Building

- **Building Blocks**


01
x2
x1
02
x3
03
x1
x2
x2
04
x2
x2
05
x3
06
x2
x2
07
x1
08
x2
09

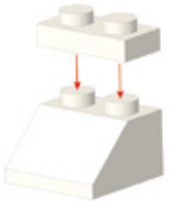

10
x4
11
x1
x1
12
13
x4
x2
14
x4
x2
15
x2
x1
16
x2
17
18
x1
x1
x1
x1

19

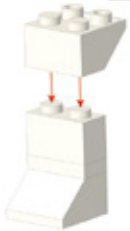
20

21

22



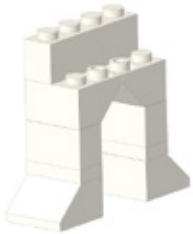
23



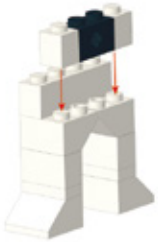
x2

24

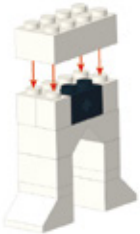
25



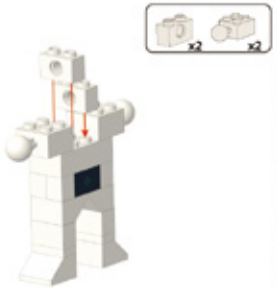
26



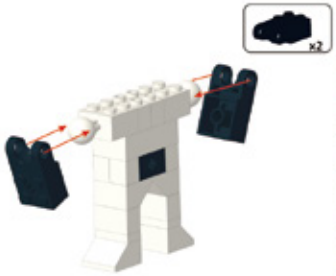
27



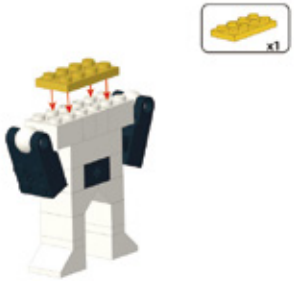
28



29

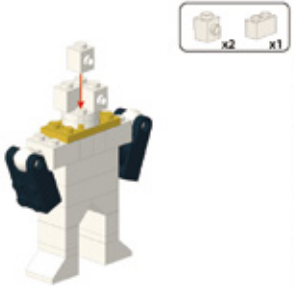


30

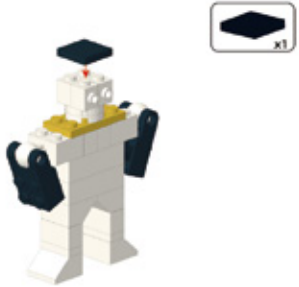


31

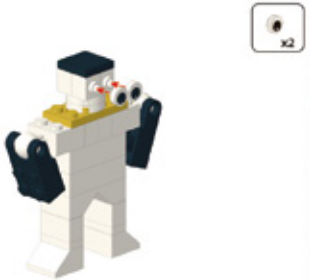
32



33



34



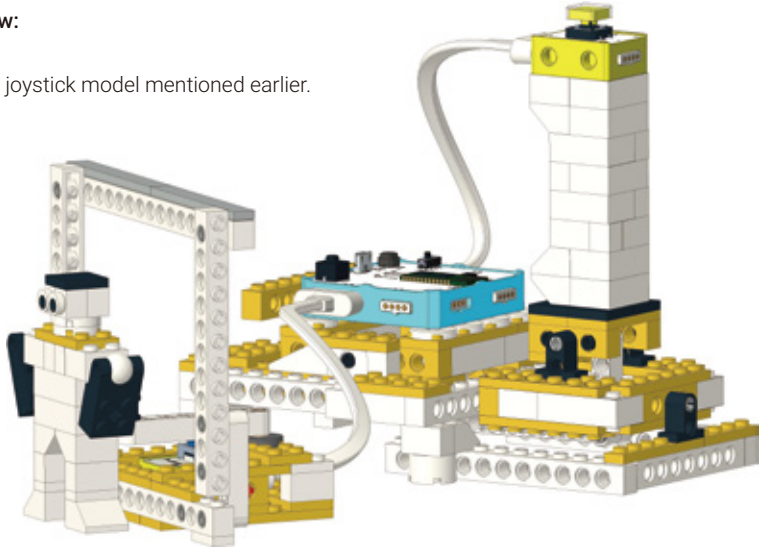
35

36

• **Circuit Connection**

Connect the circuits as shown below:

Need to make it work with the tank's joystick model mentioned earlier.



6. Task to Practice

Task Goal: Ball control training. Program to make the football fall and move with the player.

Task Analysis:

1. The football falls from a random position in midfield;
2. The football bounces off and down;
3. The ball moves with the player after the player touches the ball.

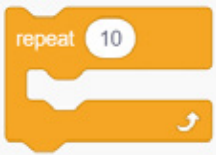
Task Steps:

The football falls from a random position in midfield:

Randomly generate the X and Y-position of the ball through this code block **[pick random “1” to “10”]**. To achieve the ball falling randomly from the midfield position, you also need to set the random number’s generation range;

The football bounces off and down:

1. When the ball is falling, the x-coordinate of the ball does not change, but the y-coordinate decreases.
Use the motion block **[change y to “10”]** to change the value to -5 to let the ball fall a certain distance. Here, the control module **[rapeat “10”]** needs to be used. Assuming the height of the ball from the ground.



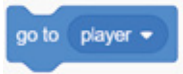
2. The ball will bounce a short distance after it hits the ground, and then it will hit the ground again, but the distance the ball bounces is not as long as the first time. You can use repeat and motion code block to write a program that bounces and falls after the ball hits the ground.

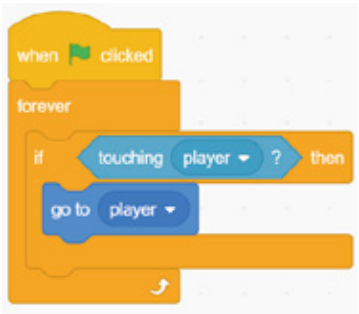
3. Combining the program in the first step with the program in the second step, you can write a program for the ball to appear from a random position, then fall, bounce, and then fall to the ground.

The ball moves with the player after the player touches the ball:

1. Firstly, use the detection block **[touching "mouse-pointer"]** and change the condition to "touching player?" to judge whether the ball has touched the player. After running the program, it is necessary to repeatedly check whether the ball touches the player.

2. When the ball touches the player, use the motion block to keep the ball moving with the player.





Awesome! You have learned how to make the ball fall and move with the player. Let's try to write the complete program, and then run the program to see if the effect achieved.

We have provided a complete penalty shootout game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprite in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a football competition and see who can get the most goals. You can also modify the data in the program according to your own ideas to increase the difficulty of the game. For example, increase the speed of the goalkeeper, or increase the speed of the shooter to increase the difficulty.

8. Brainstorming

We used a linear potentiometer in this project. Linear potentiometers are widely used in our life, such as adjusting the speed of fans, adjusting the brightness of lights, etc. You can try to use this electronic module to design a project to adjust the brightness of lights.

Project 8 Memory Challenge

1. Learning Goals

1. Know the game project: memory challenge;
2. Learn the operating principle of RGB matrix and its graphical programming control;
3. Complete the model building and game programming of memory challenge project.

2. Application

Memory challenge game refers to memorizing through 8 patterns displayed by RGB matrix. You're supposed to use your memory to mark the correct sequence of the cards displayed on the game after the countdown is over. And the game is cleared if all answers are correct.

Next, we will use building blocks and electronic modules to build the memory cards model, and use the model to select the card sequence on the game side to achieve the interactive effect of the memory challenge game.

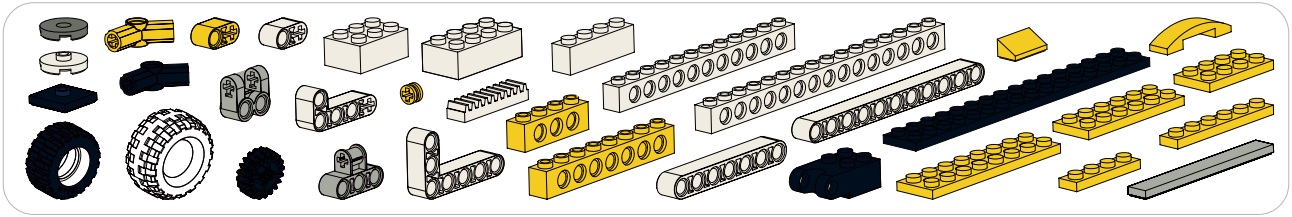
3. Supplies List

• **Electronic Supplies:**

RGB Matrix x1

Crowbits-UNO x1

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**

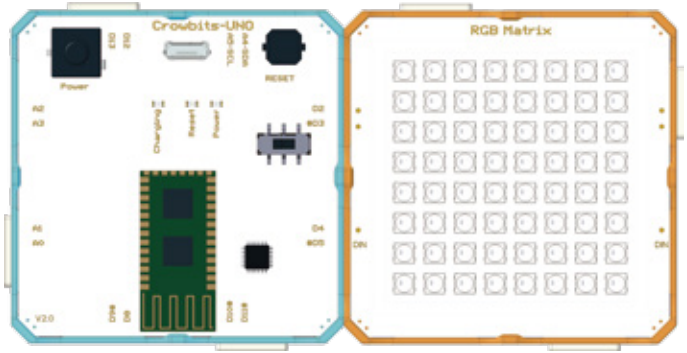


4. Knowledge Analysis

• **RGB Matrix**

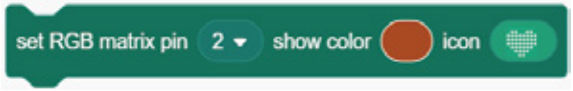
1. RGB matrix is an output module, which is composed of 8X8 matrix RGB lights. A total of 64 RGB lights are integrated with a single pixel control unit. The three primary colors (RED, GREEN, BLUE) of each pixel can achieve 256 levels of brightness display. Each lamp can be individually on and off, flashing, and gradual control. After receiving the signal, it will display the corresponding color, and turn on or off according to the signal, so that different patterns are formed according to the on and off of RGB lights.

2. Circuit Connection:



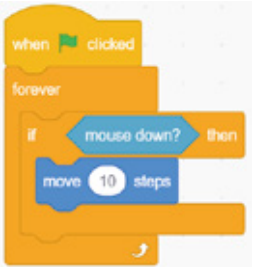
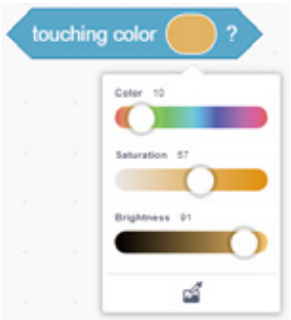
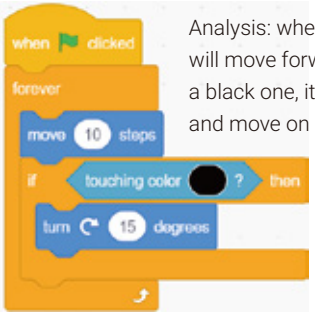
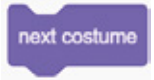
3. Program Driven:

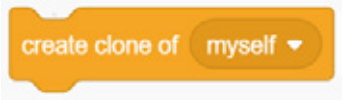
You can use this code block to control the pattern displayed on the RGB matrix.



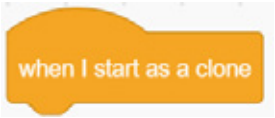
The module needs to fill in the correct pin number, choose the appropriate color, and the pattern can be drawn by yourself.

Next, we will use the following blocks:

Block	Example
 Mouse down? : used to detect whether the mouse is pressed in the stage area	 Analysis: run the program and when the mouse is clicked in the stage area, the sprite will move
 Touching color? : used to judge if the sprite is touching the set color	 Analysis: when the program is running, the sprite will move forward, and when the sprite touches a black one, it will rotate 90 degrees to the right and move on
 Next backdrop: used to switch the backdrop of the stage	 Analysis: if the stage has multiple backdrops, the backdrop will be replaced every 2 seconds after running the program



Create clone of myself: copy the "self" sprite, the cloned sprite has all the attributes of the original sprite



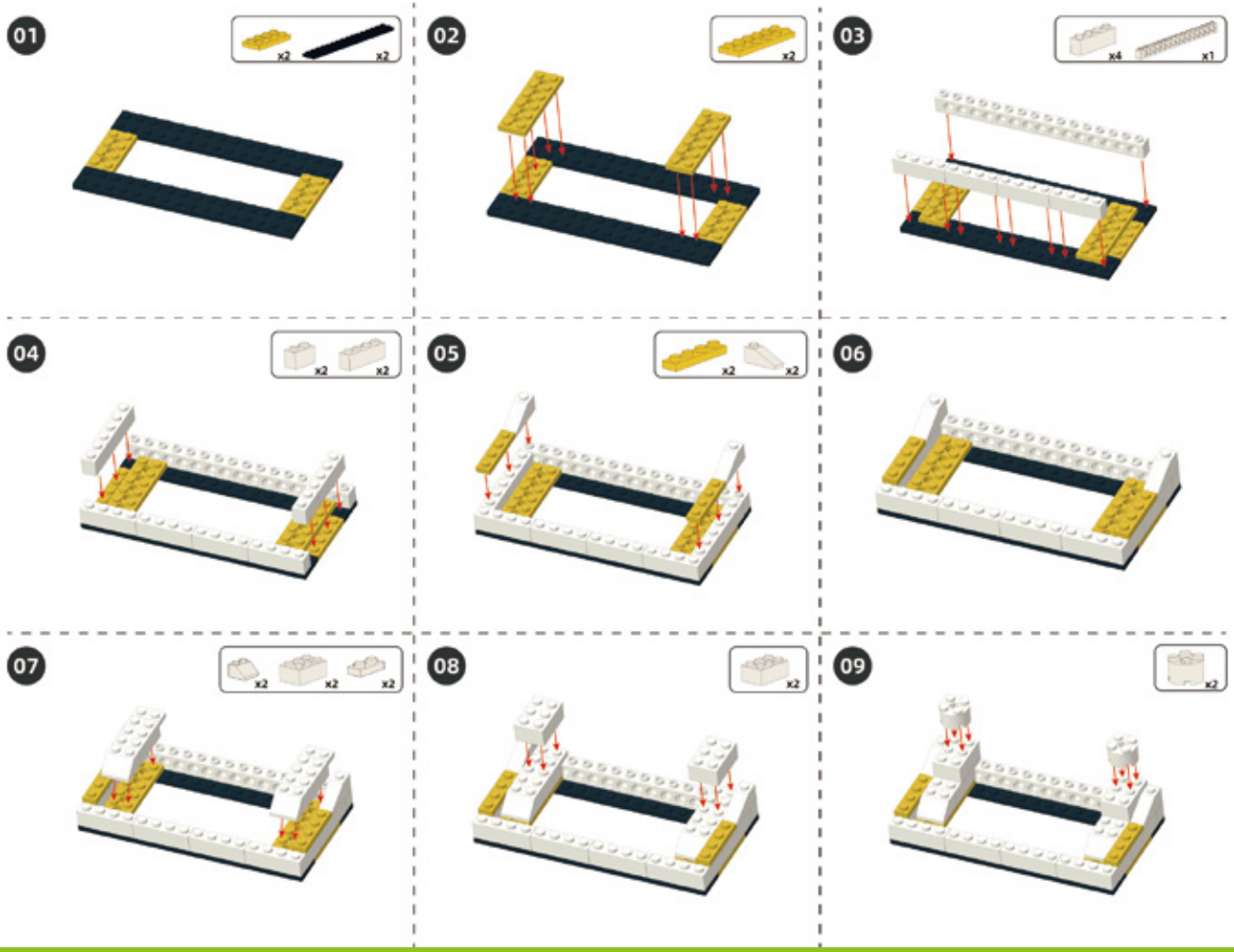
When I start as a clone: After successful cloning, you can write the corresponding program for the clone

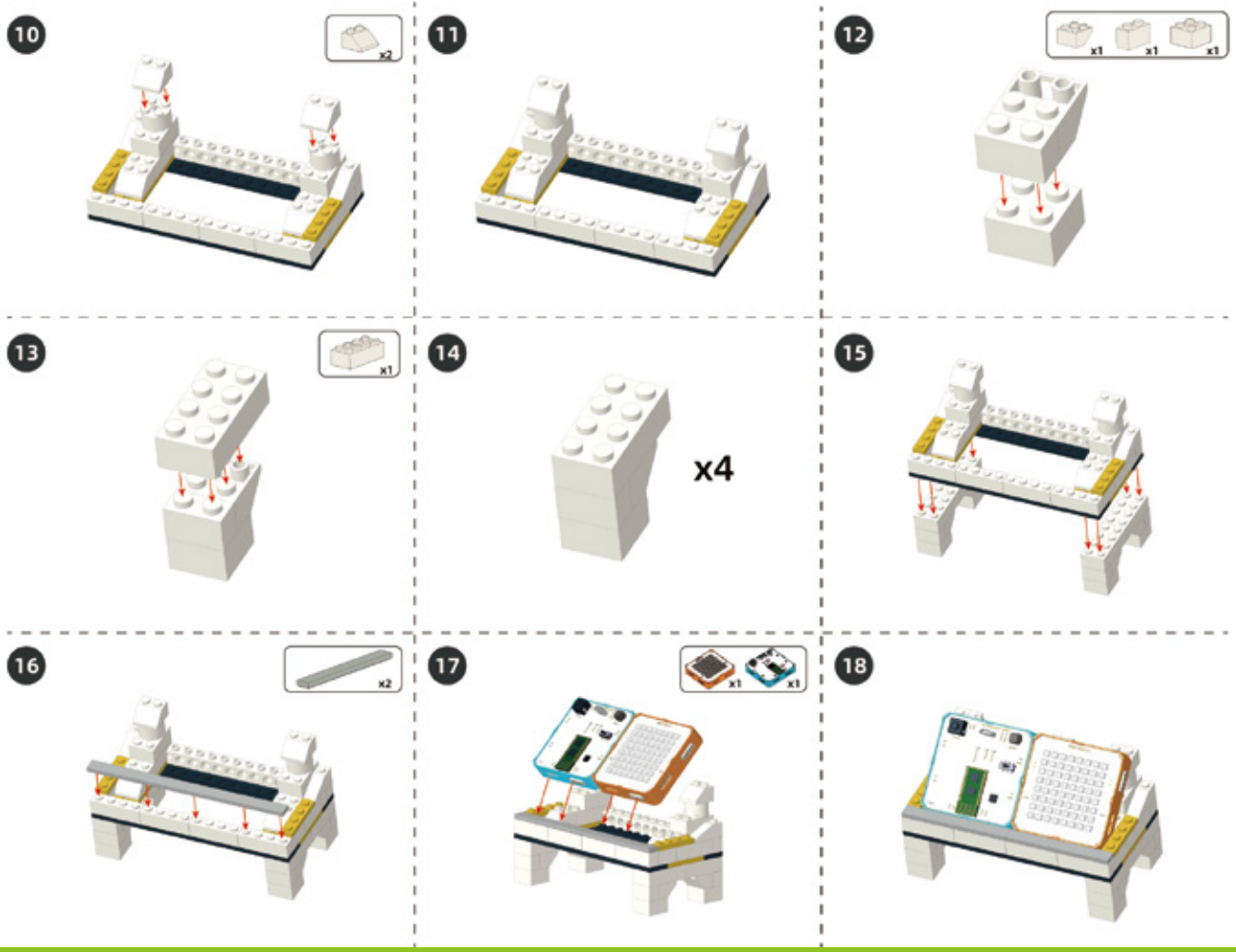
Delete this clone: delete clone

Analysis: after the green flag is clicked, "myself" is cloned, and the clone performs 10 times to turn right 15 degrees and waits for 1 second before deleting

5. Model Building

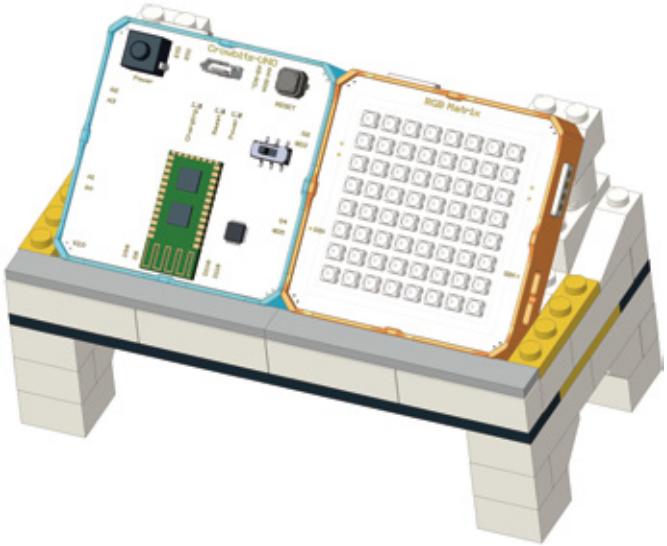
- **Building Blocks**


01
x2
x2
02
x2
03
x4
x1
04
x2
x2
05
x2
x2
06
07
x2
x2
x2
08
x2
09
x2


10
x2
11
12
x1
x1
x1
13
x1
14
x4
15
16
x2
17
x1
x1
18

• Circuit Connection

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Draw a pattern for the RGB matrix, and clone a sprite into 8 clones and evenly tile them on the stage area.

Task Analysis:

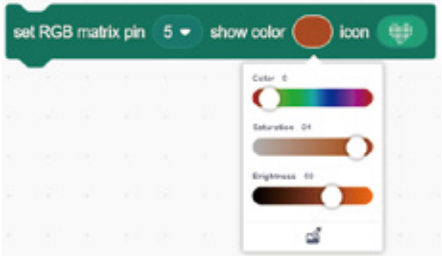
1. Programming to draw the patterns on RGB matrix;
2. Clone the sprite and spread it evenly on the stage area.

Task Steps:

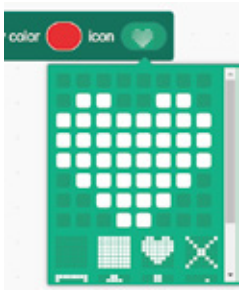
Programming to draw the patterns on RGB matrix:

1. Use this code block **["set RGB"]** to draw patterns on RGB matrix. Choose the pin you need to connect;

2. After the pin is set, select the color of RGB lights to be lit;

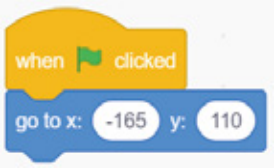


3. There are some patterns that come with the system. The white squares represent the RGB lights that to be lit. You can also draw patterns according to your preferences. When drawing a pattern, you only need to click the small square with the mouse. When the square becomes white, the RGB light will be on after running the program. The lit color is displayed according to the previously set color.



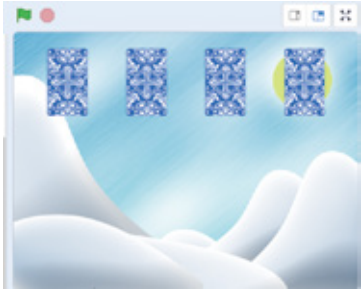
Clone the sprite and spread it evenly on the stage area:

1. You need to clone the sprite 8 times here, and each cloned position is in a different position of the stage area. Firstly, move the sprite to the upper left corner;



2. Start cloning at this position, and move to next position after cloning. Here the ordinate remains unchanged, and the abscissa plus 110. There are four in the first row, so repeat four times.

3. When the clone is started, we have to show the sprite. Of course, we can also do other actions.



According to the method above, can you place the other four cards evenly below?

We have provided a complete memory challenge game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprite in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a memory challenge competition and see who can remember the most cards. You can modify the data in the game program to increase the difficulty of the game. For example, shorten the flop time to change the difficulty of the game.

8. Brainstorming

We used RGB matrix as the display module in this project. RGB matrix is often used in display devices, such as billboards. You can try to use this electronic module to design a robot dynamic meme.

Project 9 Running Racing

1. Learning Goals

1. Know the competition event: running racing;
2. Extend the application of IR reflective sensors and buttons;
3. Complete the model building and game programming of the runner.

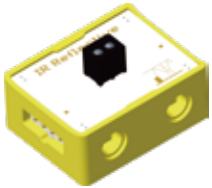
2. Application

Running is a convenient way of daily physical exercise and an effective exercise method for aerobic breathing. Meanwhile, running is also one of the most common track and field competitions. Running with friends occasionally, which can strengthen the body and enhance friendship.

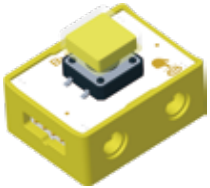
Next, we will use building blocks and electronic modules to build two runners, and control the runners on the game side through the runner model to achieve the interactive effect of the 100-meter dash game.

3. Supplies List

• **Electronic Supplies:**

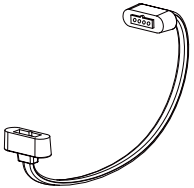


IR Reflective Sensor x1



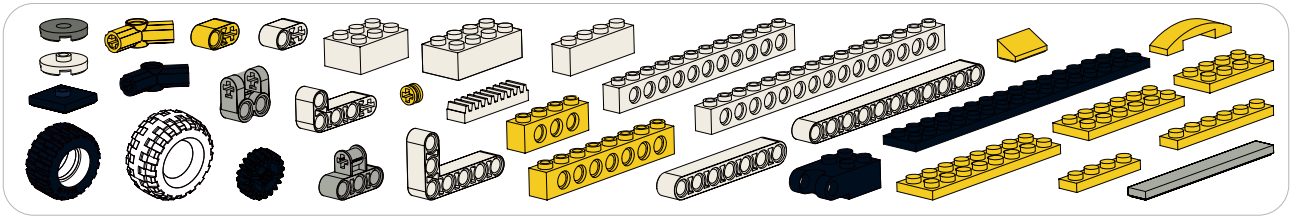
Button Module x1

Crowbits-UNO x1



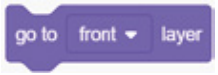
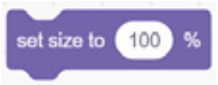
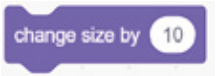
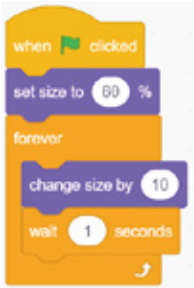
Magnetic Cable x2

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



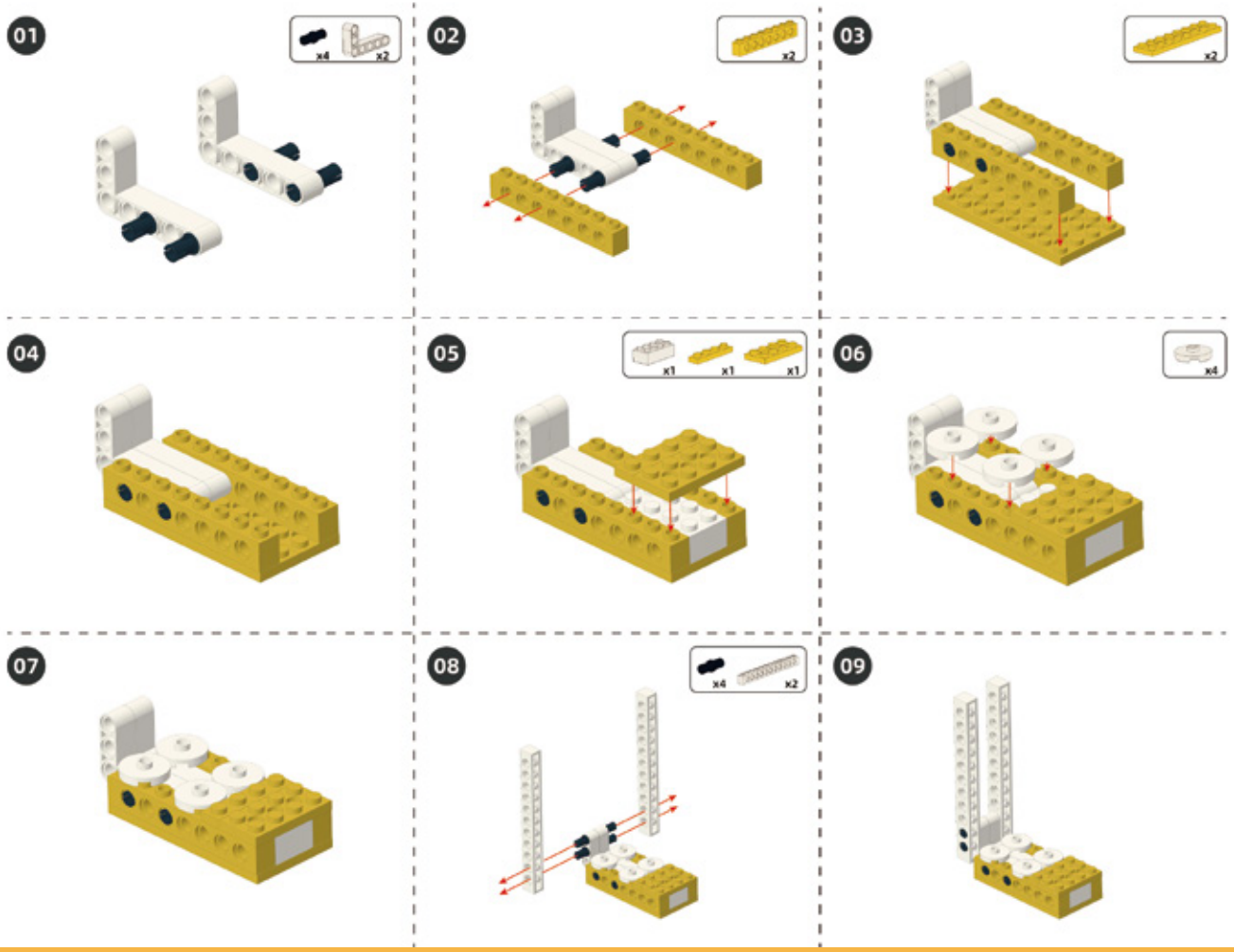
4. Knowledge Analysis

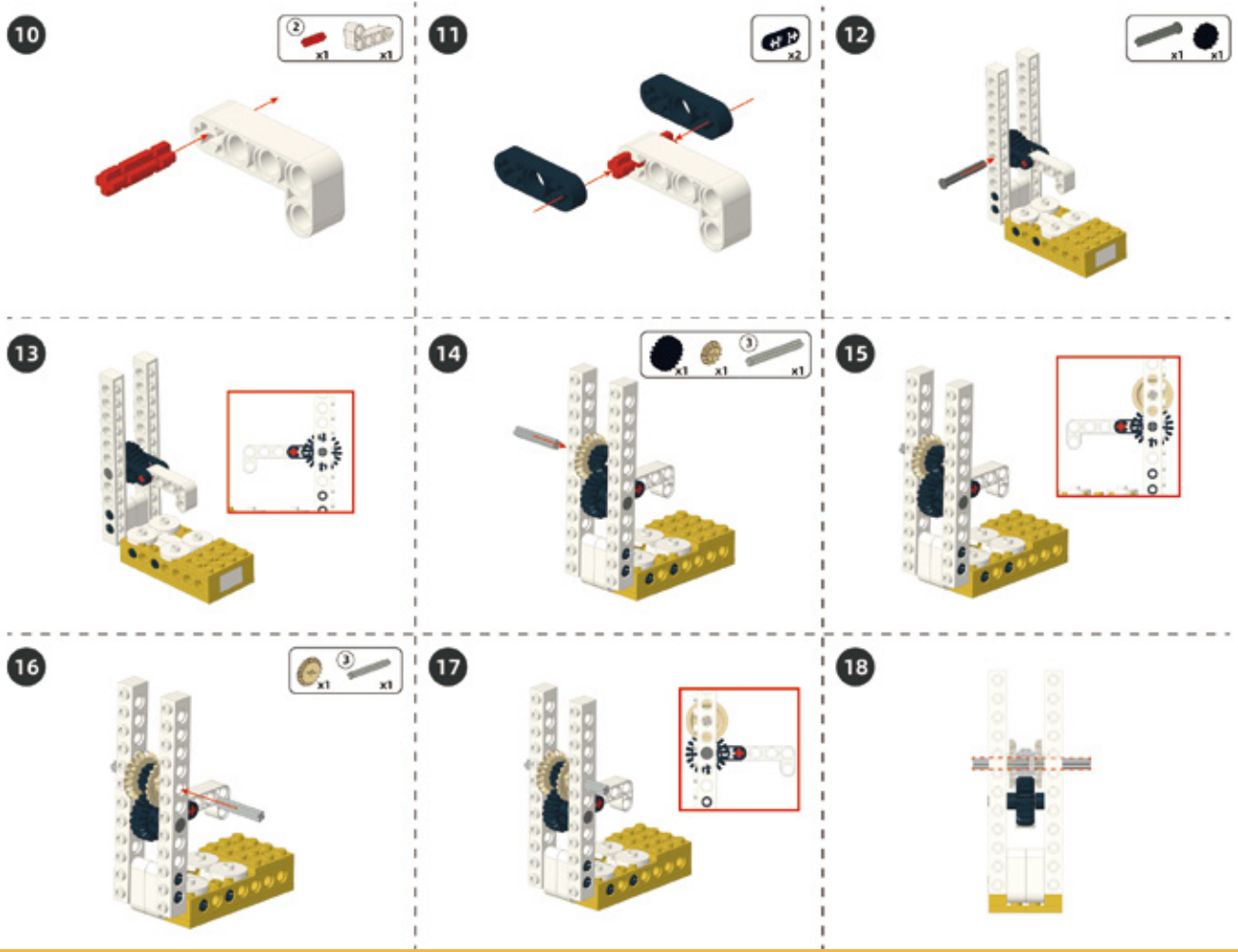
We will use the following blocks in this project:

Block	Example
 Go to front layer: There is a cascading relationship between the sprites on the stage. This command can make the sprites move to the front of other sprites without being covered	 Analysis: When the green flag is clicked, the sprite moves to the front and glides to a specific coordinate within 1 second
 Set size to: The larger the value is, the bigger the sprite becomes	 Analysis: When the green flag is clicked, the sprite will appear in the specified position with the set size.
 Set size to: The larger the value is, the bigger the sprite becomes	 Analysis: When the green flag is clicked, set the sprite's size to 60, and repeat the command "change size by 10" with an interval of 1 second

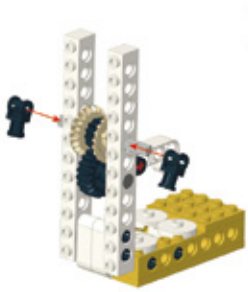
5. Model Building

- **Building Blocks**

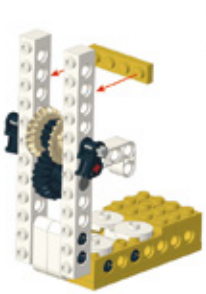

01
x4
x2
02
x2
03
x2
04
05
x1
x1
x1
06
x4
07
08
x4
x2
09


10
x1
x1
11
x2
12
x1
x1
13
14
x1
x1
x1
15
16
x1
x1
17
18

19

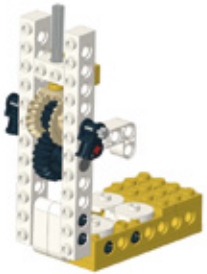


20



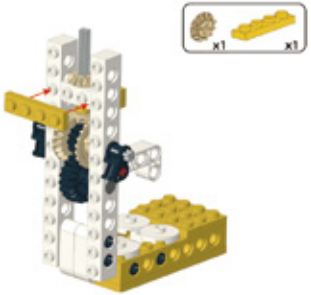
21

22

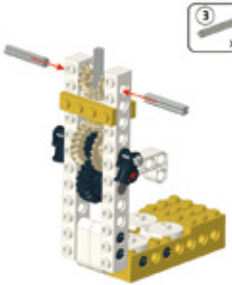


23

24

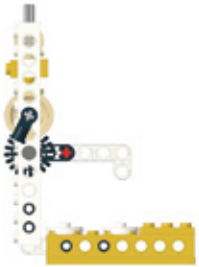


25



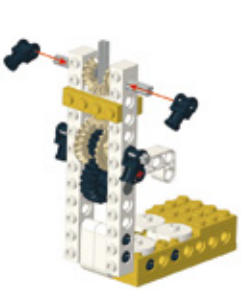
26

27



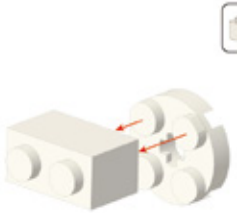
28

29

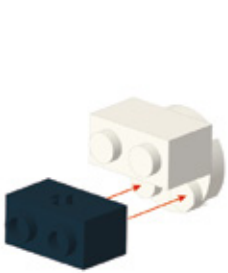


30

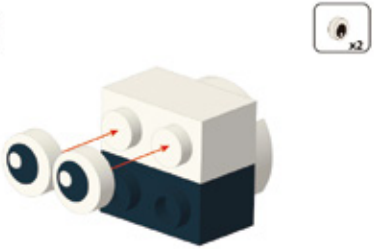
31



32

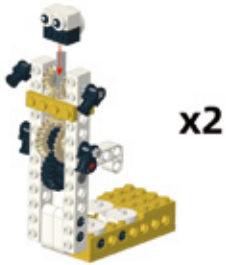


33

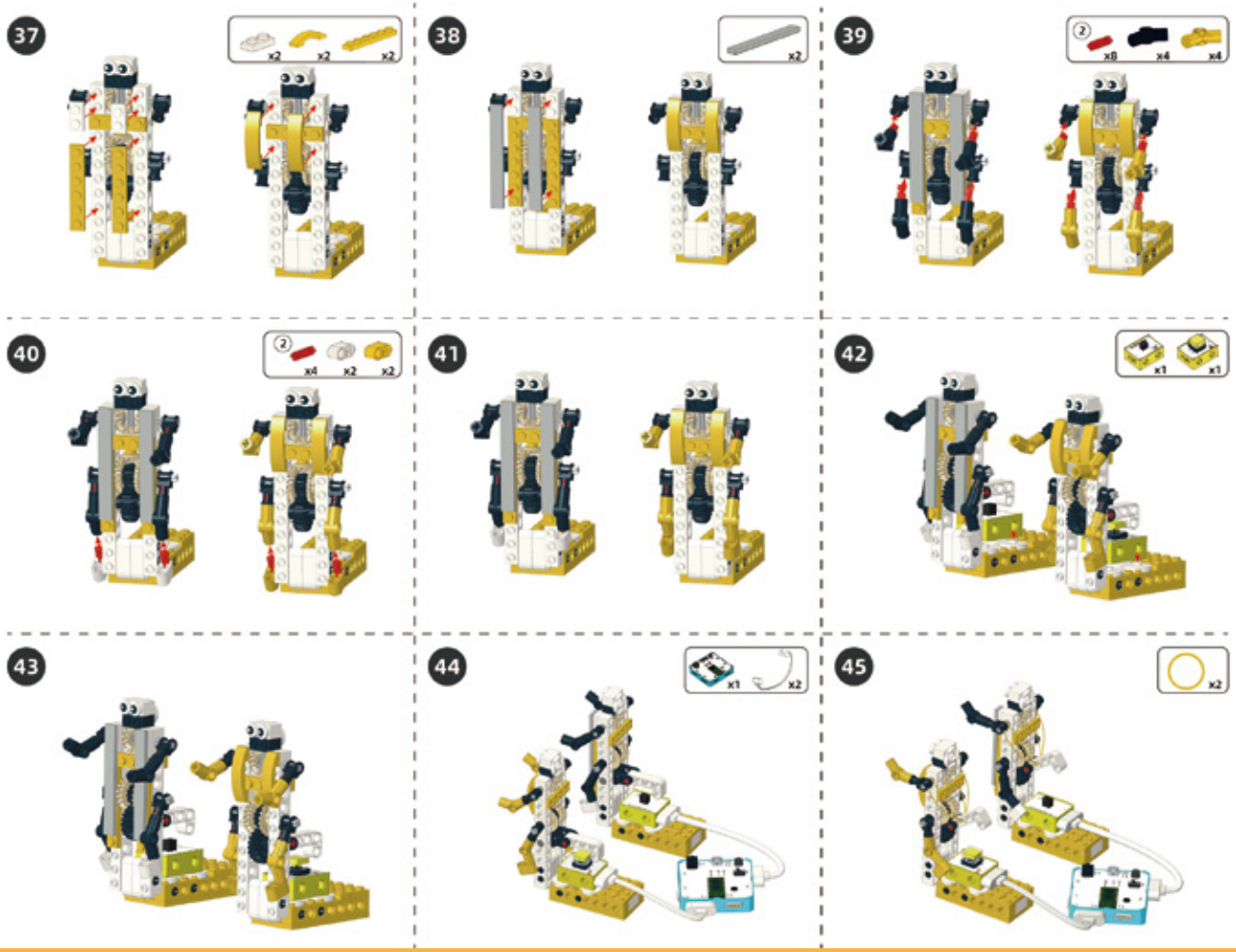


34

35

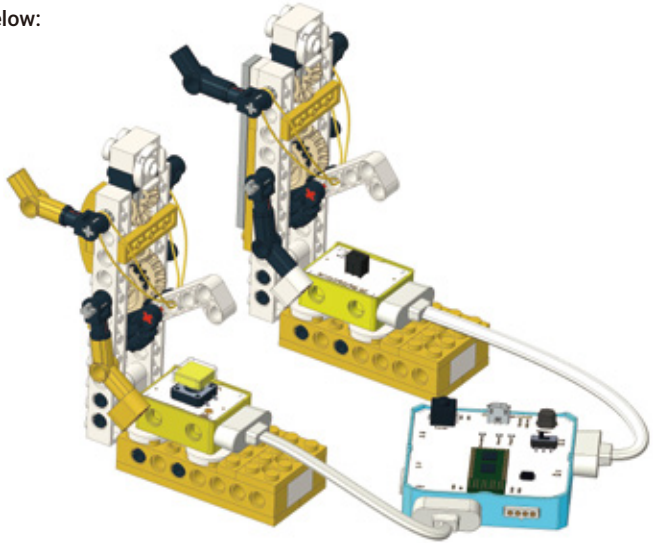


36


37
x2
x2
x2
38
x2
39
x8
x4
x4
40
x4
x2
x2
41
42
x1
x1
43
44
x1
x2
45
x2

• **Circuit Connection**

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Let the backdrop and sprites move backwards continuously (the negative direction of Y-position) to achieve the effect of "smaller when further away and bigger when closer";

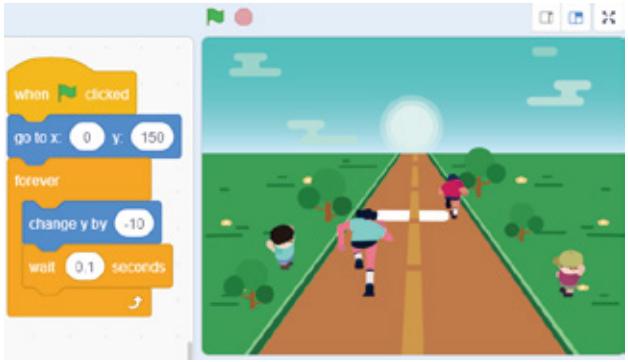
Task Analysis:

1. Sprites (tree and audience) on the backdrop move backward;
2. The runway moves backward.

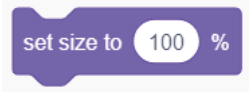
Task Steps:

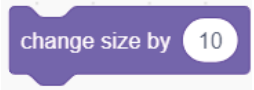
Sprites (tree and audience) move backward (take the tree as an example, and the audience program need to be completed by yourself):

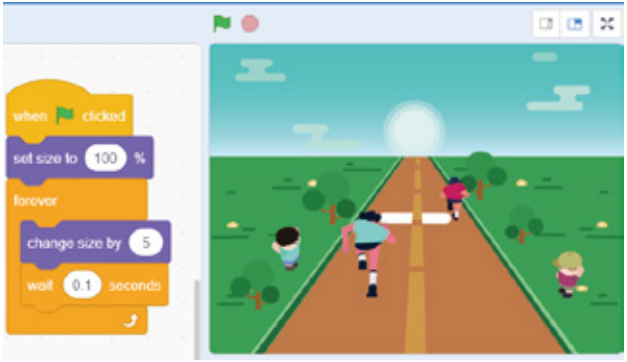
1. If you want to realize the function of "Runner" moving forward quickly in the game vision, you can make the reference object moving backward to achieve this effect. As long as you use this code block **[change y by "-10"]** for the sprites (tree and audience), the special effect can be realized.



2. When the object moves from far to near, it is visually seen to be larger and larger. And therefore, we need to use the code block **[set size to "100" %]** to set the initial size of the sprite. The size value is subject to the actual situation. Use the code block **[change size by "10"]** to change the sprite's size. The value of increment and interval time needs to be debugged according to the actual effect;

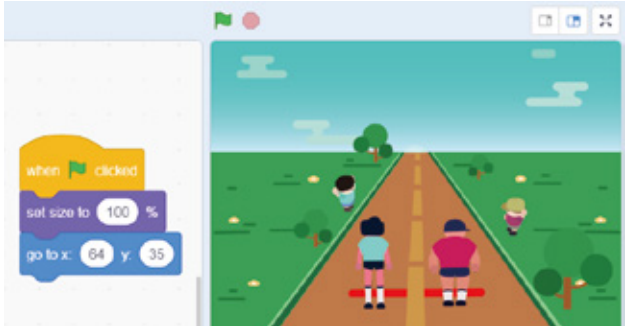




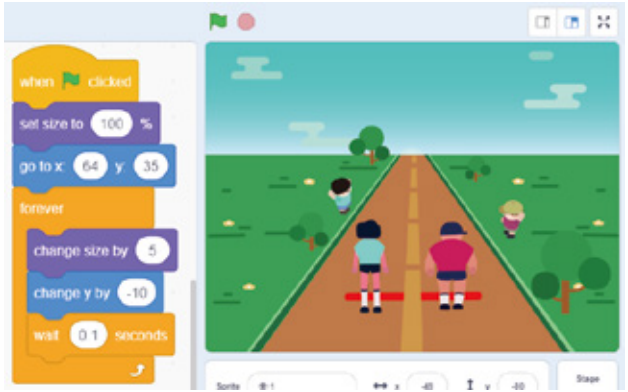


Having learned to move from far to near and from small to large, we can combine the two programs together.

First of all, you need to determine the position and initial size of the sprite.

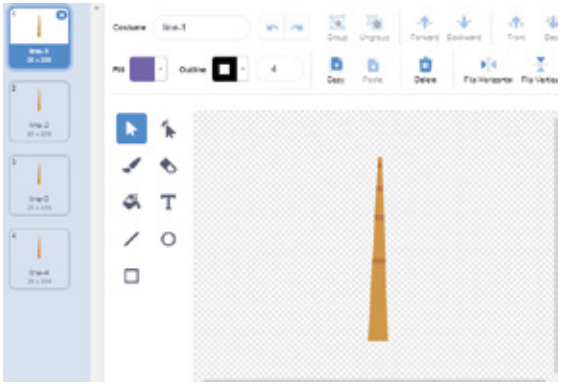


Then let the sprite move, and make the sprite bigger during the movement.

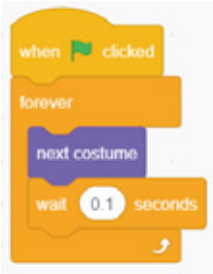


The runway moves backward:

Independent sprites in the backdrop, such as trees and audiences, can use coordinate and size switching to achieve a moving effect, but this is not applicable to the runway that spans half of the stage. We can achieve this by switching sprite's costume here.



Four costumes are drawn on the middle line of the road respectively, and the effect of movement is realized by programming the program to switch the costumes.



Awesome! Now you have learned how to program sprites from far to near. Let's try how to make the sprite change costumes while moving!

Running race program is very similar to the previous horse racing program. Compare it and see if there is any difference. We have provided a complete running racing game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprite in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a running racing competition and see who can get first place. You can also modify the program according to your own ideas to increase the difficulty of the game. For example, reduce the distance of each run to increase the difficulty of the game.

8. Brainstorming

We learned the use of IR reflective sensor and button again in this project. These two modules are widely used in our life, such as infrared remote control (TV, air conditioner), trigger switch, etc. Can you try to design a quiz answerer with existing equipment and software programming?

Project 10 Smart Home

1. Learning Goals

1. Know the project: smart home;
2. Learn the operating principle of PIR motion sensor, LED bar, DHT11 sensor and their graphical programming control;
3. Complete the model building and game programming of the door.

2. Application

Smart home is a residential platform, using integrated wiring technology, network communication technology, security technology, automatic control technology, audio and video technology to integrate facilities related to home life, and aims to improve home safety, convenience, comfort, artistry, and create an environmentally friendly and energy-saving living environment.

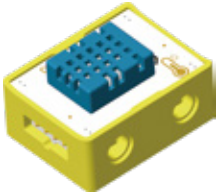
Next, we will use building blocks and electronic modules to build the door, and control the relevant equipment on the software side through the door model to achieve the interactive effect of smart home project.

3. Supplies List

• **Electronic Supplies:**

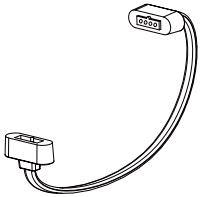
PIR Motion Sensor x1

LED Bar x1



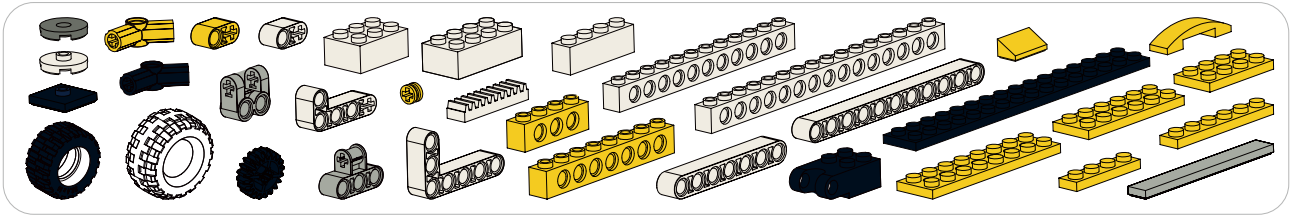
DHT11 x1

Crowbits-UNO x1



Magnetic Cable x1

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**

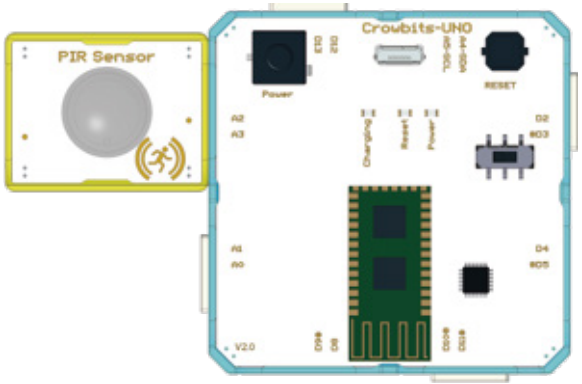


4. Knowledge Analysis

• PIR Motion Sensor

1. PIR motion sensor module is a digital input module. The pyroelectric sensor is known as PIR motion sensor. It is used for anti-theft alarm and visitor notification in daily life. The principle is to convert the released charge into a voltage output through an amplifier.

2. Circuit Connection:



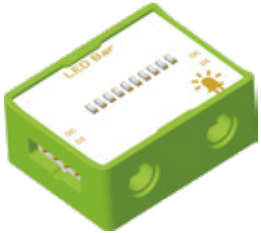
3. Program Driven:

You can program according to the situation listed in the following table:

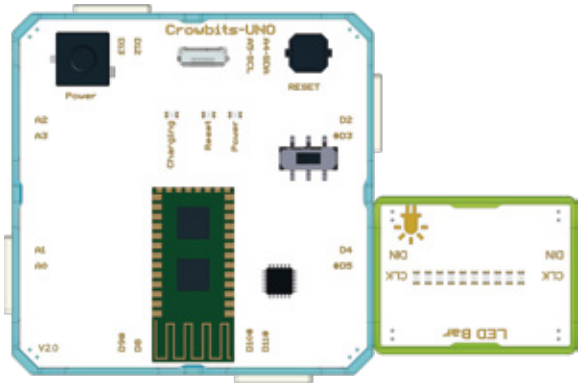
People or animals detected?	PIR motion sensor's signal
Yes	high level/1
No	low level/0

• LED Bar

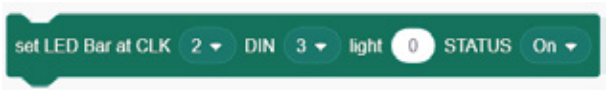
1. LED bar module is an output module composed of 10 LED lights of different colors, and each light can be controlled separately. It can be used to prompt battery power, voltage value, water depth, sound level and any other scenes that need to express gradient values.



2. Circuit Connection:



3. Program Driven:

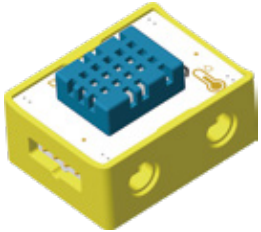


After setting the pins, you can control the on and off of any light on the module.

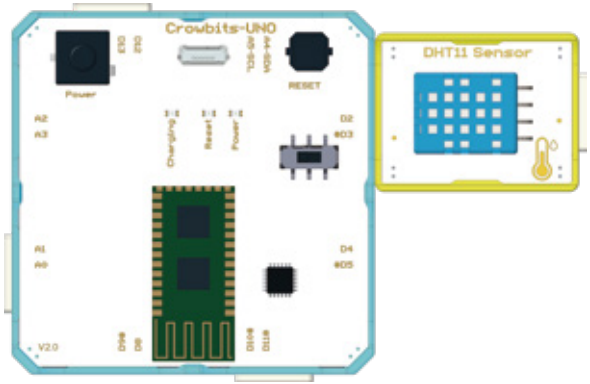
You can control multiple lights at the same time by numerical values.

• DHT11 Temperature And Humidity Sensor

1. DHT11 temperature and humidity sensor is an input module. DHT11 is a dedicated temperature and humidity sensor, which contains a temperature and humidity composite sensor with calibrated digital signal output. It can accurately detect the current temperature and humidity.



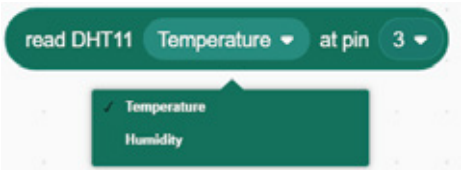
2. Circuit Connection:

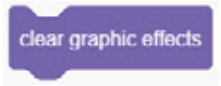
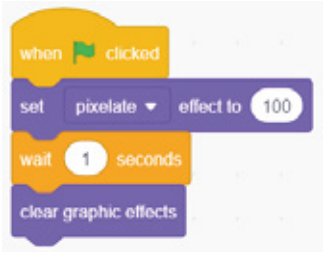
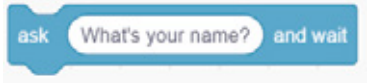
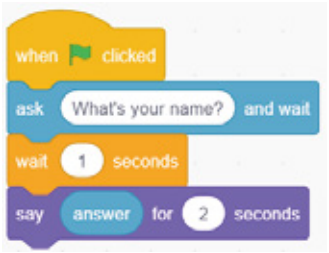


3. Program Driven:

Use the code block **[read DHT11 "Temperature" at pin "3"]**

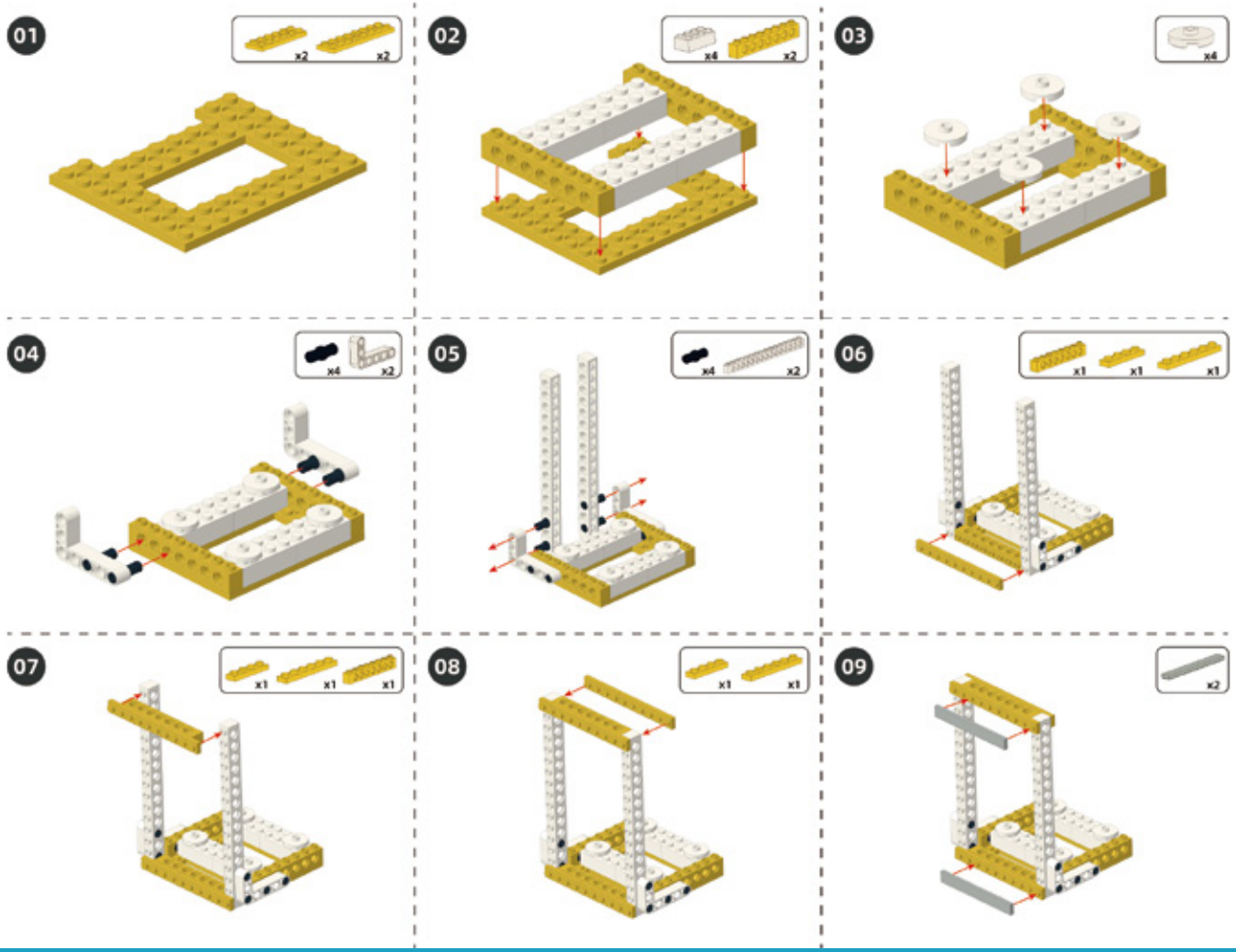
for reading the temperature and humidity, select the corresponding pin, and read the temperature and humidity values respectively by selecting the temperature or humidity options.

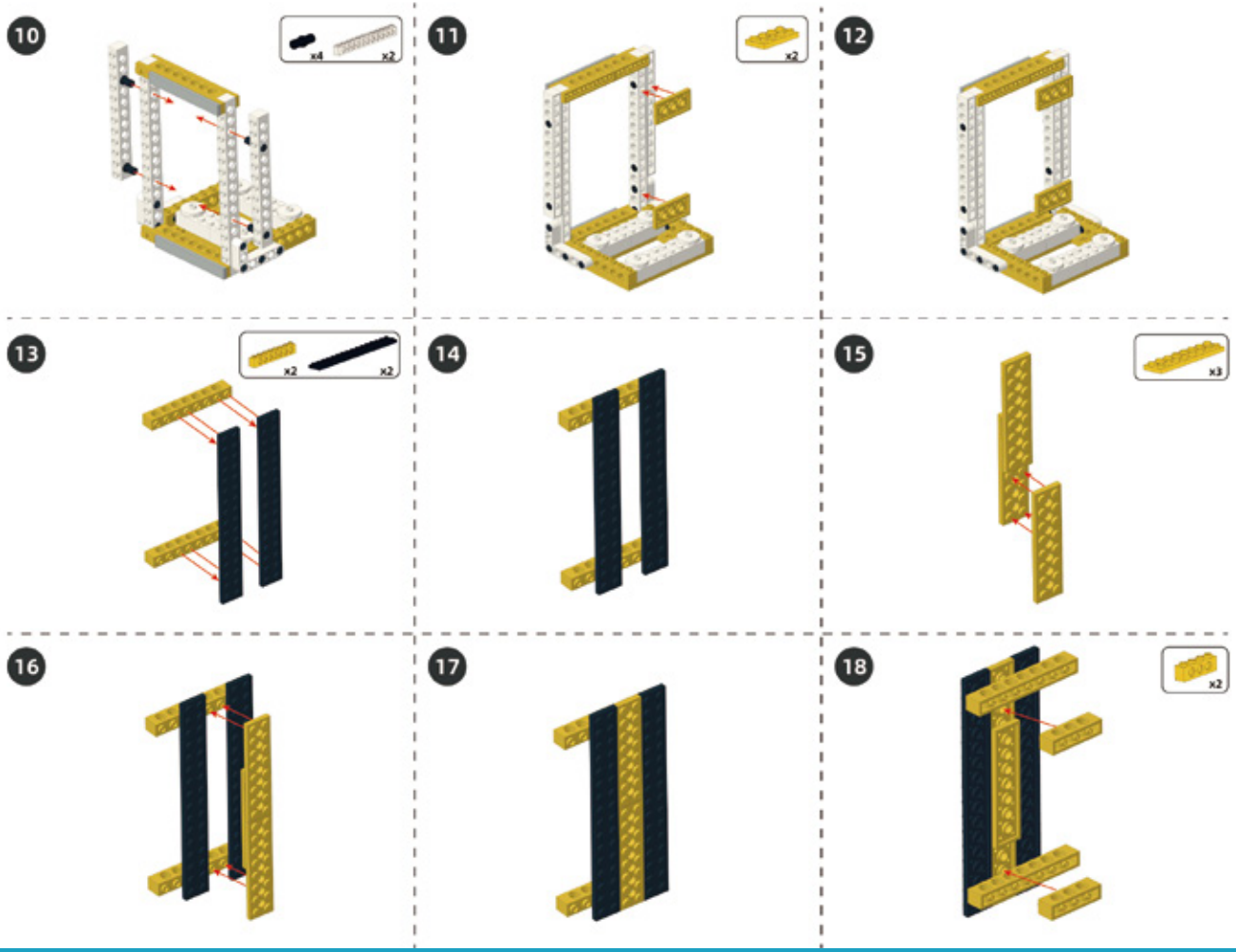


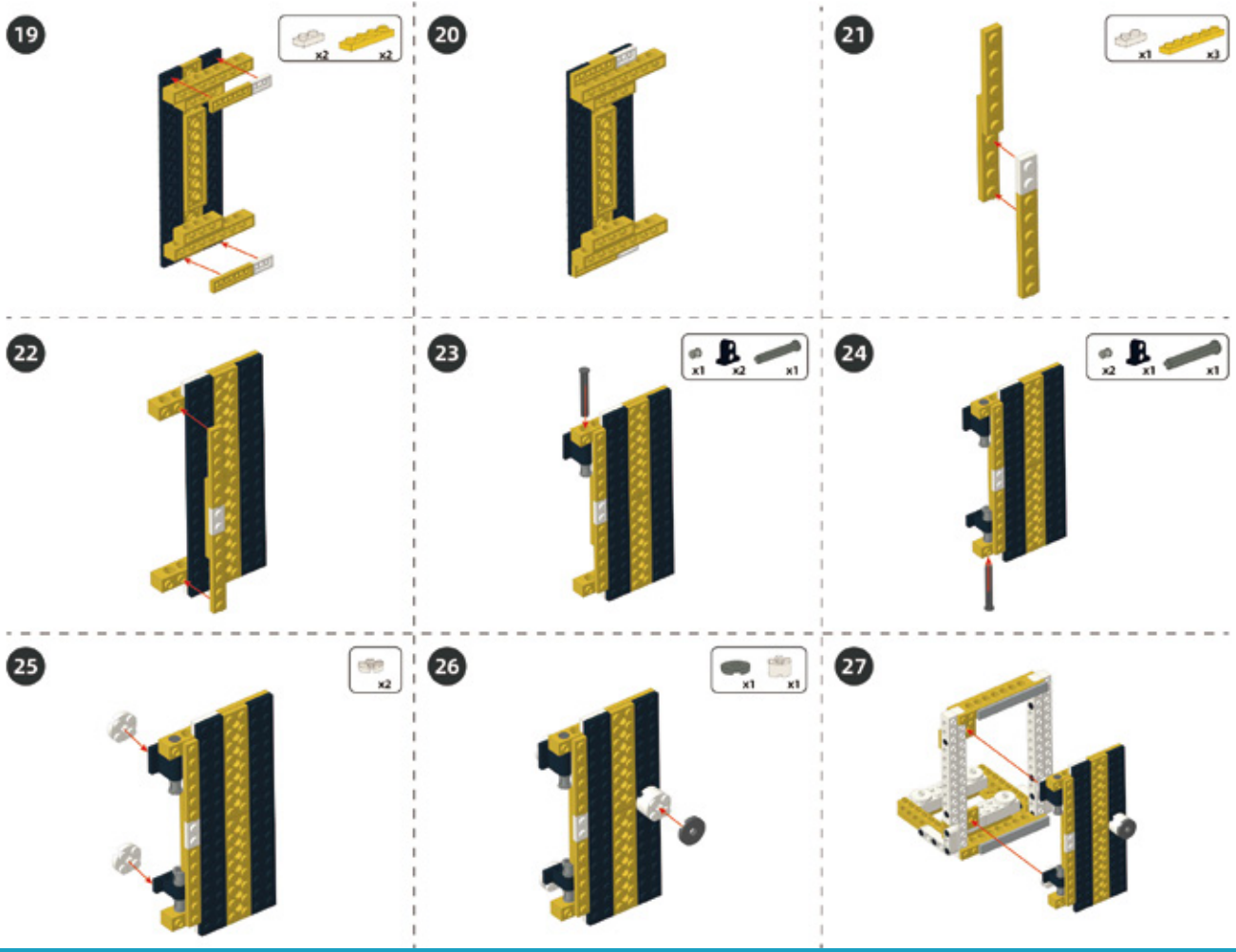
Block	Example
 Clear graphic effects: if you have set the sprite's graphic effects before, use this program to clear the graphic effects	 Analysis: after running the program, the sprite will become pixelated. After waiting for one second, the sprite will return to its original appearance
 Ask and wait: ask a question and wait for an answer	 Analysis: after running the program, ask a question and jump out of the answer box, then fill in the answer. The answer will be recorded in the "answer" block and let the sprite say it
 Answer: used with the ask command, the input answer will be stored in this block	

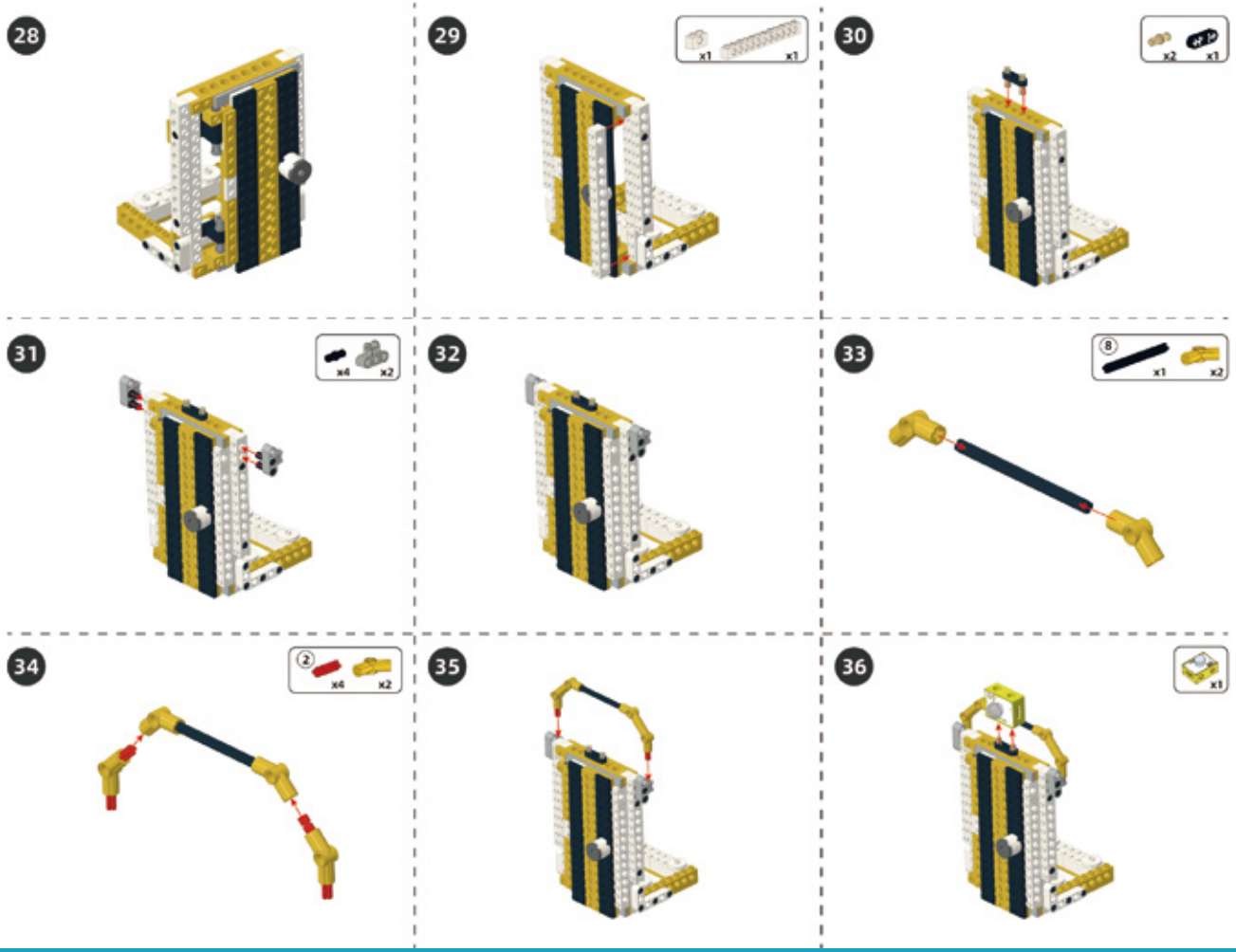
5. Model Building

- **Building Blocks**


01
x2
x2
02
x4
x2
03
x4
04
x4
x2
05
x4
x2
06
x1
x1
x1
07
x1
x1
x1
08
x1
x1
09
x2


10
x4
x2
11
x2
12
13
x2
x2
14
15
x3
16
17
18
x2


19
x2
x2
20
21
x1
x3
22
23
x1
x2
x1
24
x2
x1
x1
25
x2
26
x1
x1
27


28
29
x1
x1
30
x2
x1
31
x4
x2
32
33
8
x1
x2
34
2
x4
x2
35
36
x1

37

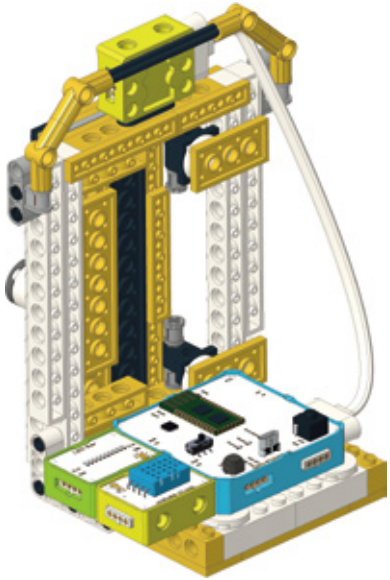
38

39

40

• Circuit Connection

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Judge whether someone is in front of the door, and ask for the password.

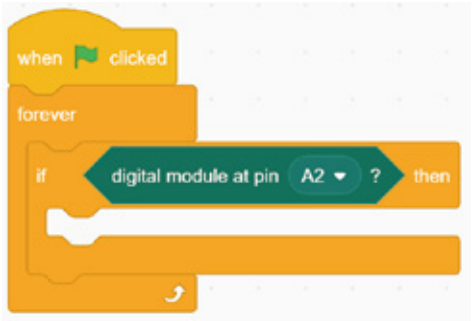
Task Analysis:

1. Use the PIR motion sensor to detect whether someone is in front of the door;
2. Ask for the password and judge whether the answer is correct, then the robot responds accordingly.

Task Steps:

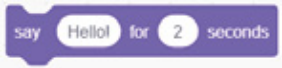
Use the PIR motion sensor to detect whether someone is in front of the door:

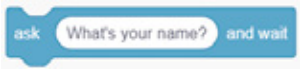
When someone appears in the detection range of the PIR motion sensor, the judgment sentence can be used to detect whether the output is high:



Ask for the password and judge whether the answer is correct, then the robot responds accordingly:

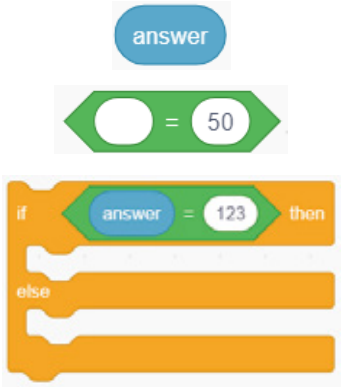
1. When someone appears in front of the door equipped with a PIR motion sensor, the robot will say a "Hello" greeting. So you need to use the code block **[say "Hello!" for "2" seconds]** in the looks module. After 2 seconds, the robot starts to ask for the password and waits for the answer from the visitor. You need to use the motion module **[ask "What's your name?" and wait].**



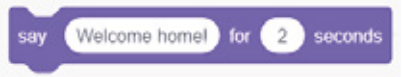


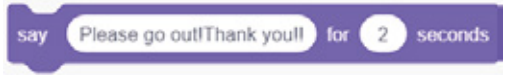
And fill in the blank space that you want the robot to say when asking for the password.

2. Set a password to open the door and judge whether the visitor's answer is correct. To set the password for opening the door, you need to use the motion module **[answer]** and the operators module **[" " = "50"]**. Assuming that the password for opening the door is set to 123, you need to determine whether the password answered by the visitor is correct. The program is as follows:

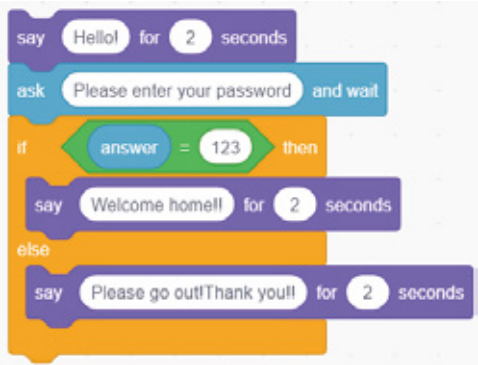


3. Base on the answer of the visitor, the robot makes the corresponding action. If the visitor answers correctly, the robot will say "Welcome home" and use the looks module **[say "Welcome home!" for "2" seconds]**. If the visitor answers incorrectly, the robot will ask him to leave **[say "Please go out! Thank you!!" for "2" seconds]**.

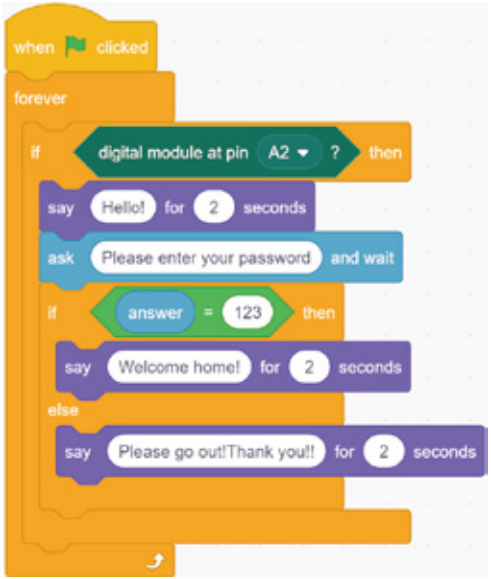




Combine the program for judging whether the password is correct with the program for the robot to take corresponding actions.



4. Finally, put the step 3 program and the program whether the PIR motion sensor has detected someone together, then it becomes the desired program.



Let's try to change what the robot says when there is a visitor!
There are other environmental detection modules in this project. Different environmental signals can control the work of different devices. Have a try!

We have provided a complete smart home game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprites in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a smart home game. You can also modify the data in the program according to your own ideas to make the home "smarter". For example: shorten the time of watering plants, or when the air humidity meets conditions to change the frequency of watering.

8. Brainstorming

We used a variety of sensors as trigger modules in this project. Multiple sensors cooperate with each other to achieve a smart home effect. You can try to use these electronic modules to design an intelligent control platform, that is, when the sensor detects a change, the robot will react accordingly.

Project 11 Aeroplane Chess

1. Learning Goals

1. Know the competitive game: Aeroplane Chess;
2. Extend the application of six-axis gyroscope and RGB matrix;
3. Complete the model building and game programming of dice.

2. Application

Aeroplane chess is a competitive game that consists of four colors, with planes drawn on it, and it allows up to four people to play together with different colors each. Aeroplane chess comes with a dice, as long as you roll the dice, you can move steps based on the dice number which in the front side. Sometimes it comes with traps, sometimes, it comes with surprises.

Next, let's build a dice with the building block and electronic modules, then control the dice of the game by dice model to realize the interaction of Aeroplane chess.

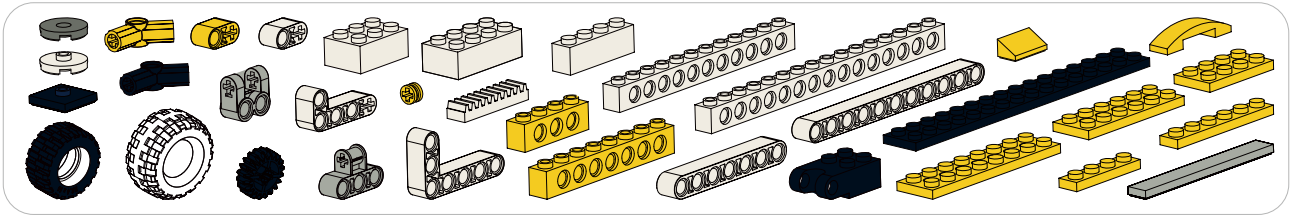
3. Supplies List

• **Electronic Supplies:**

RGB Matrix x1

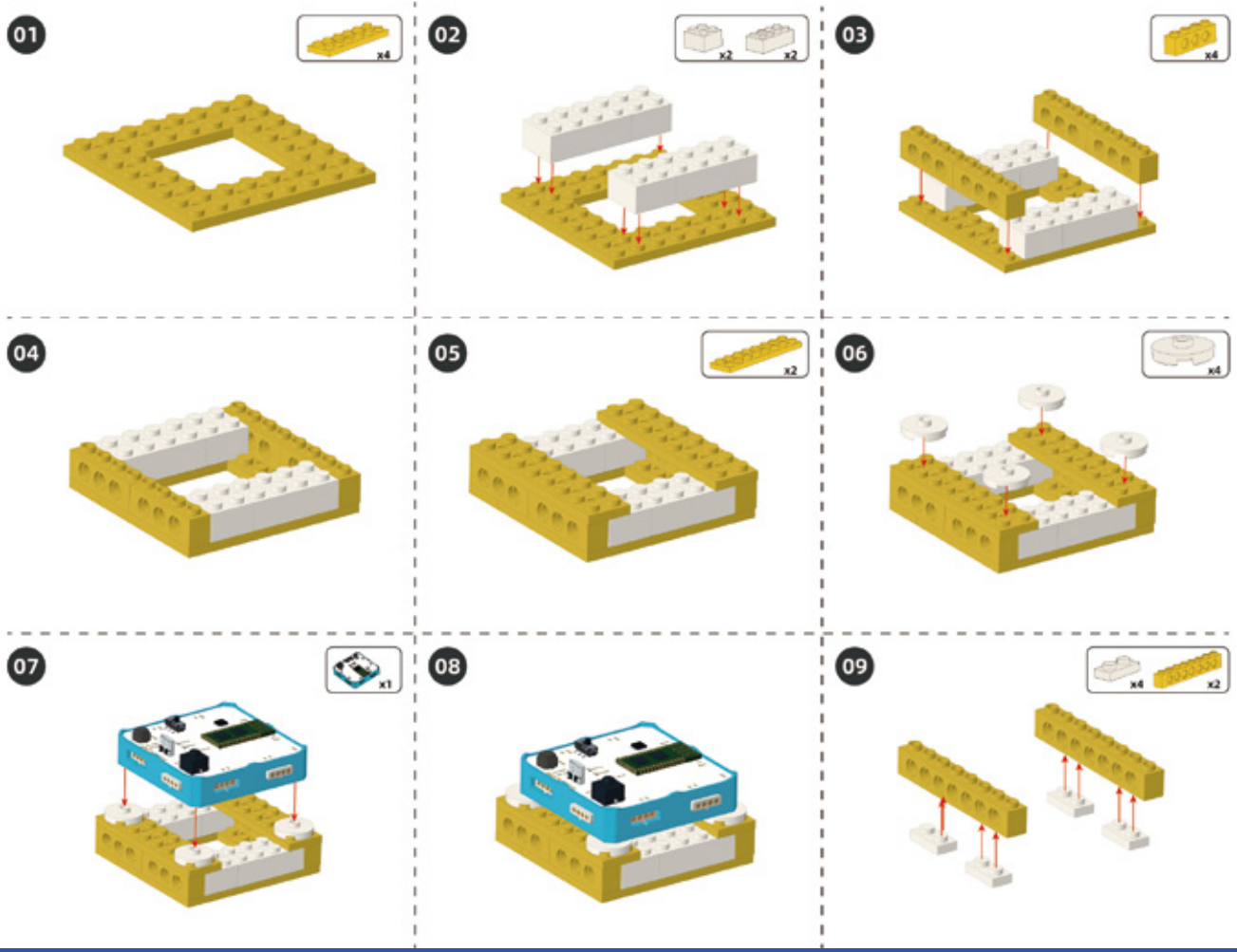
Crowbits-UNO x1

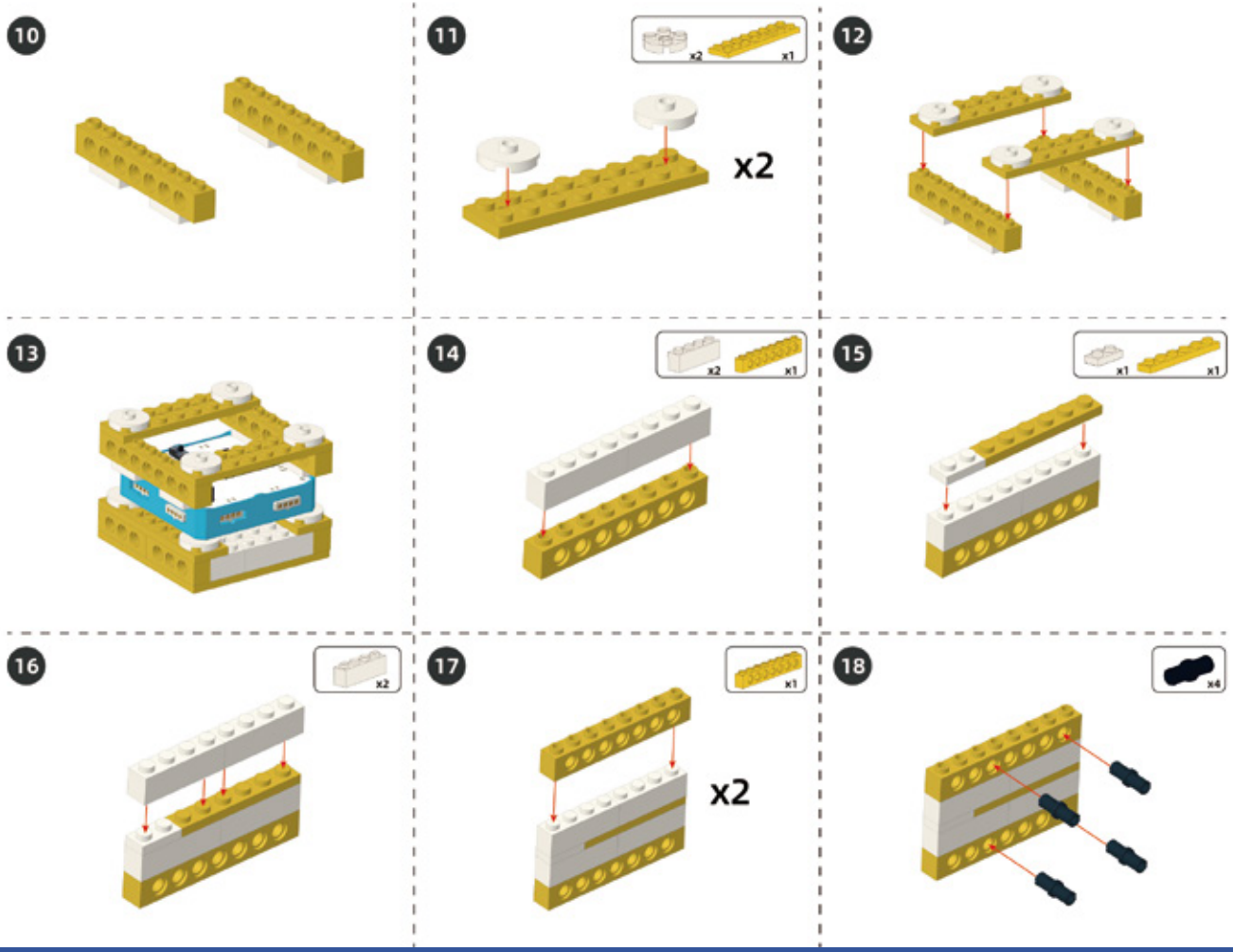
• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



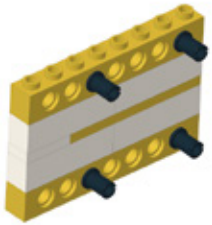
4. Model Building

• **Building Blocks**

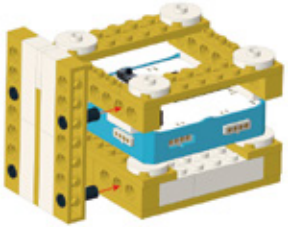

01
x4
02
x2
x2
03
x4
04
05
x2
06
x4
07
x1
08
09
x4
x2


10
11
x2
x1
x2
12
13
14
x2
x1
15
x1
x1
16
x2
17
x1
x2
18
x4

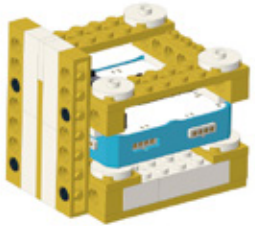
19



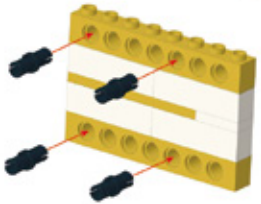
20



21

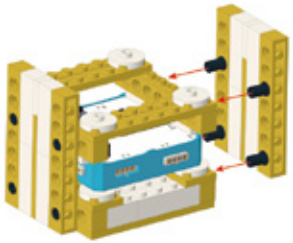


22



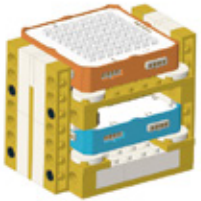
23

24



25

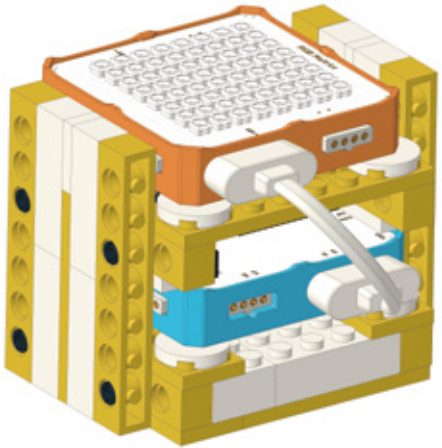
26



27

• Circuit Connection

Connect the circuits as shown below:



5. Task to Practice

Task Goal: Roll the dice and let the plane move forward based on the number of dice.

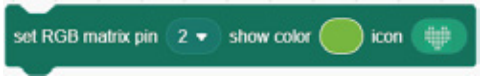
Task Analysis:

1. Make RGB matrix shows the number of dice;
2. Use hardware as the trigger module to roll the dice.

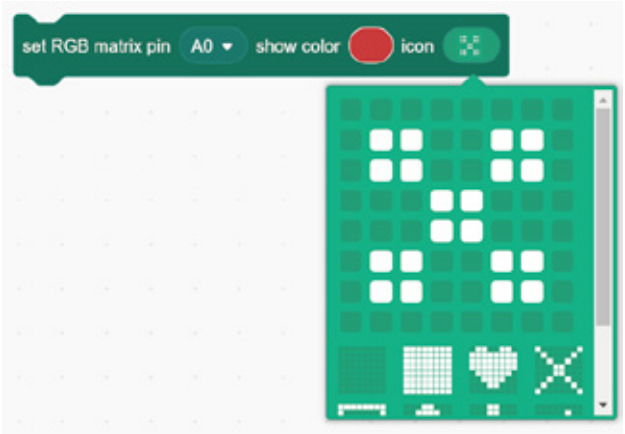
Task Steps:

Make RGB matrix shows the number of dice:

1. Draw the 6 points pattern of the dice on the RGB matrix with the code block;



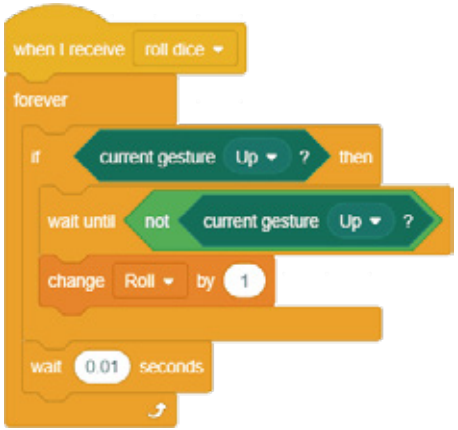
2. Choose the pin number, draw pattern of the number "A0";

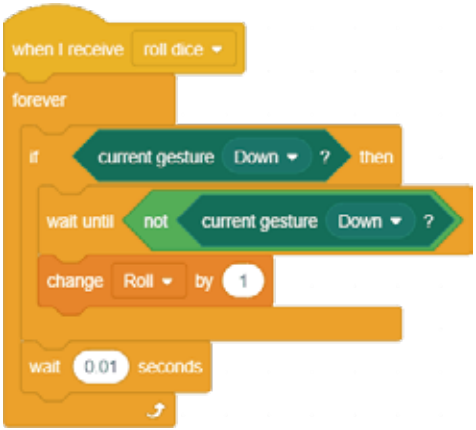


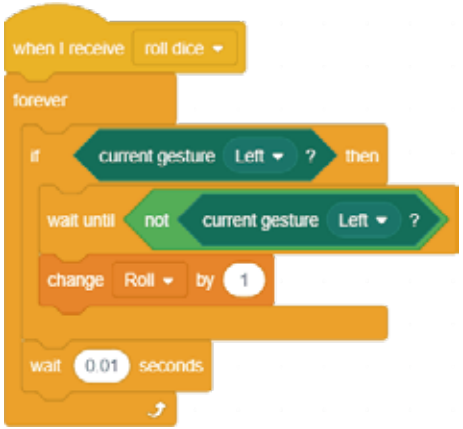
Use hardware as the trigger module to roll the dice:

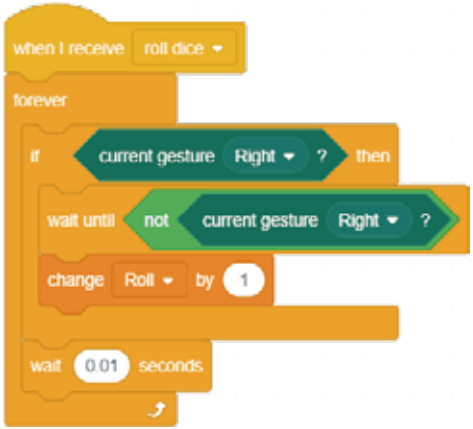
1. We can shake the main board to achieve the effect of rolling dice according to the gyroscopes' features. The gyroscope would de deflected in different directions during the shaking, the deflection gesture would also change, then we count the deflection times of the gyroscope in four directions to determine whether the main board is shaking. When it deflects to one direction until recover, it can be regarded as a shake.

2. Four directions need to be detected simultaneously by four broadcasts, and the number of shaking is counted.









3. Because gyroscope can easily trigger the deflection, it needs to count multiple deflections, when the number of shaking is greater than 10 times, a dice rolling operation is triggered.

4. When the dice stops, it displays a number, then the random code block **[pick random “1” to “6”]** will be needed, create a new variable “dice number”, **[set “dice number” to “0”]** then we correspond the generated random number to the corresponding dice costume.

We have provided a complete Aeroplane chess game in the software sources, you can download and try it out to see what is the difference between this one we did. The code of other sprites in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

6. Game Interactions

Let's experience your projects with your friends together.

Invite your friend to an Aeroplane chess competition and see who can reach the finish line first. You can modify the program according to your own ideas to increase the difficulty and experience of the game. For instance, change the game rules, increase the number of steps forward or backward to improve the fun of the game.

7. Brainstorming

In the Aeroplane chess game project, we regard the six-axis gyroscope as a trigger module. Six-axis gyroscopes are usually be used in smart devices to determine position and control direction, such as phone map navigation, game control, etc. You can try to use this module to design a gravity remote control sensor and control the car in the game.

Project 12 Boxing

1. Learning Goals

1. Know the sports event: boxing;
2. Learn the operating principle of accelerometer and its interactive programming control;
3. Complete the model building and game programming of boxing project.

2. Application

Boxing is a sport for fighting with boxing gloves. The goal of the game is to get more points than the opponent in order to beat or knock the opponent down. At the same time, the players must try to avoid the opponent's blow. Boxing is called "the sport of the brave". There are many records about boxing as early as ancient Greece and Rome. In the ancient Olympics, boxing was already one of the competition events. By the third Modern Summer Olympics in St. Louis, men's boxing was officially included in the competition.

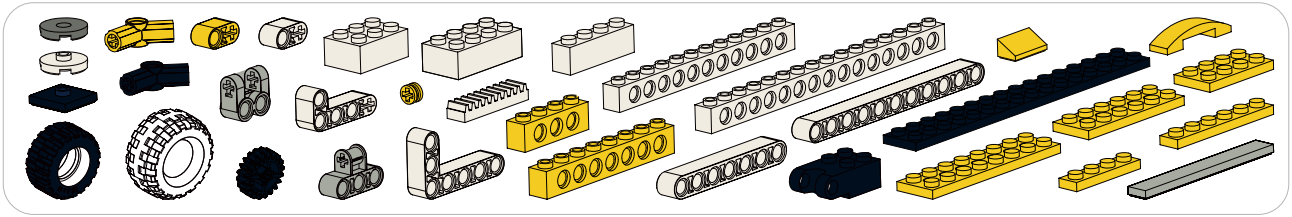
Next, we will use building blocks and electronic modules to build a boxing device, and control the boxer on the game side through the boxing model to achieve the interactive effect of boxing game.

3. Supplies List

• **Electronic Supplies:**

Crowbits-UNO x1

• **Structure Supplies: LEGO blocks. Choose the blocks according to the assembly diagram.**



4. Knowledge Analysis

• **Accelerometer**

The accelerometer is a sensor that can measure acceleration. It is usually composed of masses, dampers, elastic components, sensitive components and adaptive circuits. In the process of acceleration, the sensor obtains the acceleration value by measuring the inertial force of the mass block.

Note:

1. Do not drop, hit or apply acceleration beyond the shock or vibration limit of the sensor.

2. The accelerometer should be firmly fixed to avoid accidents when testing the acceleration of the object.

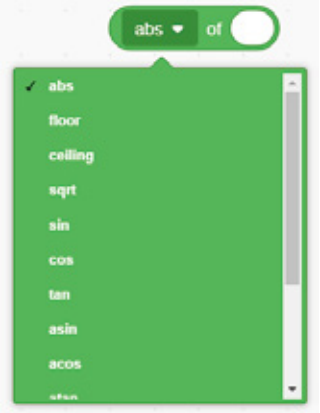
Program Driven:

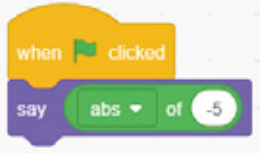
You can program according to the situation listed in the following table:

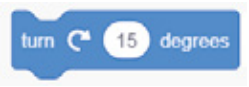
move to generate acceleration?	accelerometer output value
Yes	value change
No	no value

Next, we will use the following blocks:

Block	Example
 Reset timer: Reset the timer count to zero and restart counting	 Analysis: when the running time is greater than 2 seconds, jump out the program of rotating to the right
 Timer: After the game starts, get the program execution time continuously; when the timer returns to zero, start timing again	

<table>
<tr>
<td> 

Mathematical operation: used to conduct a variety of mathematical operations on the data</td>
<td> 

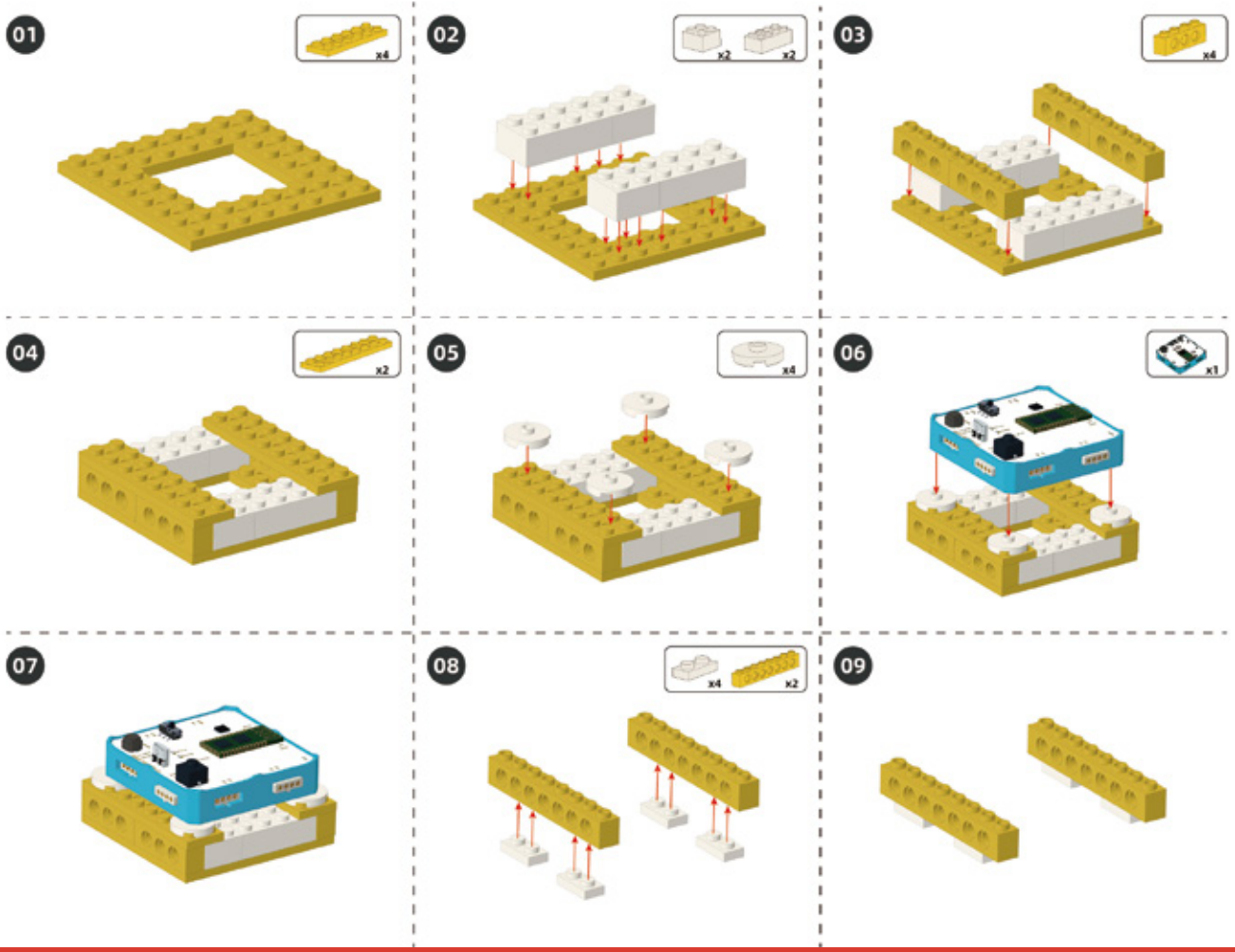
Analysis: when the green flag is clicked, the sprite says 5 which is the absolute value of -5</td>
</tr>
<tr>
<td> 

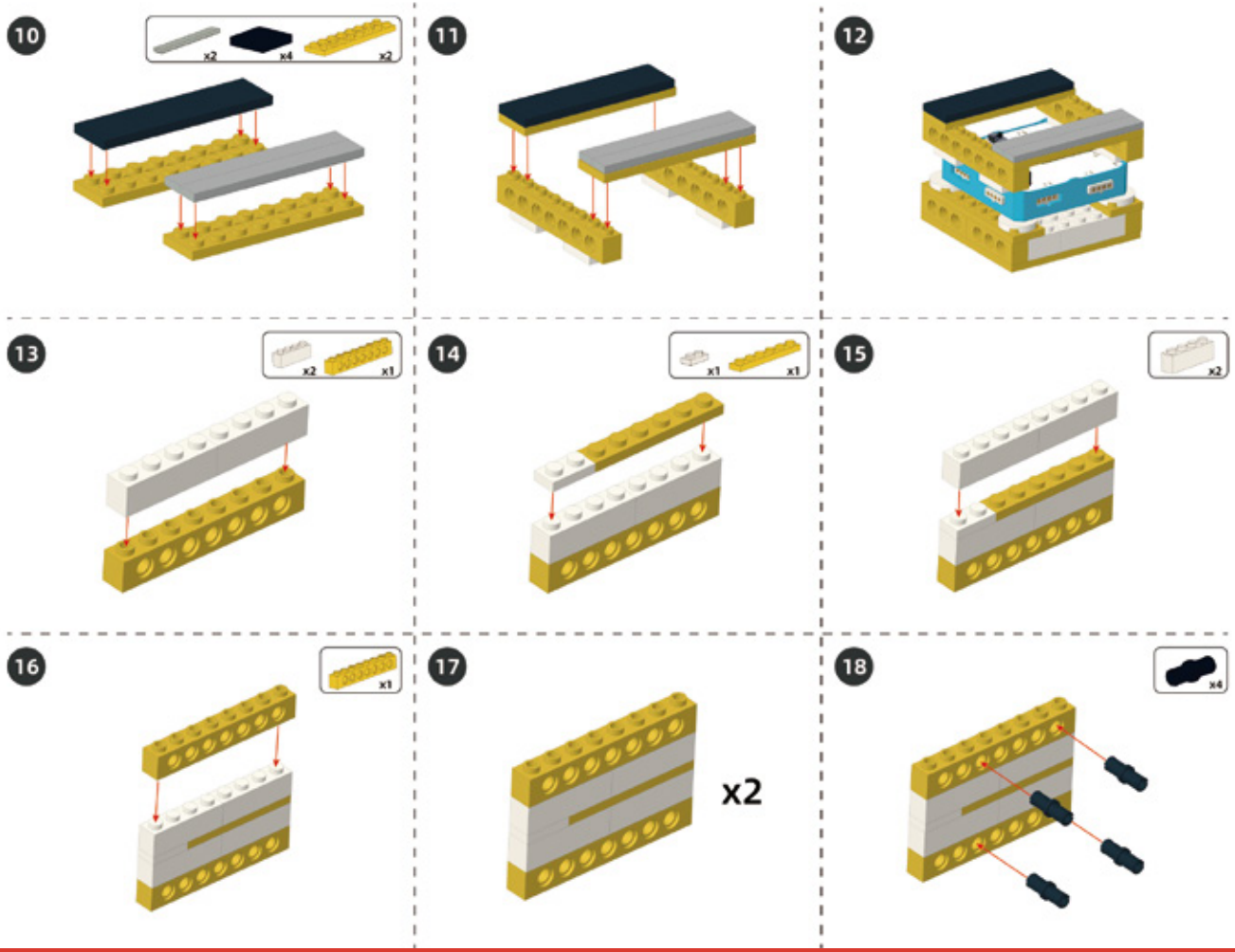
Turn to the right: controls the number of degrees the sprite rotates to the right</td>
<td> 

Analysis: when the green flag is clicked, the sprite will rotate 15 degrees to the right every 1 second</td>
</tr>
</table>

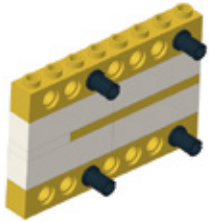
5. Model Building

- **Building Blocks**


01
x4
02
x2
x2
03
x4
04
x2
05
x4
06
x1
07
08
x4
x2
09

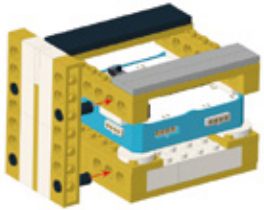

10
x2
x4
x2
11
12
13
x2
x1
14
x1
x1
15
x2
16
x1
17
x2
18
x4

19

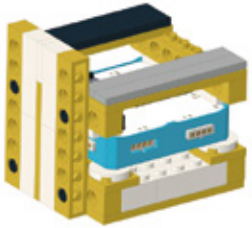


20

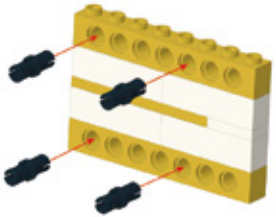
21



22



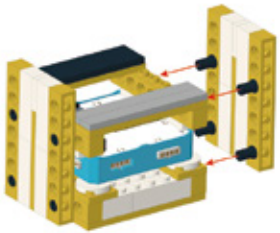
23



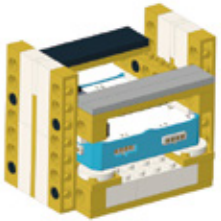
24

25

26

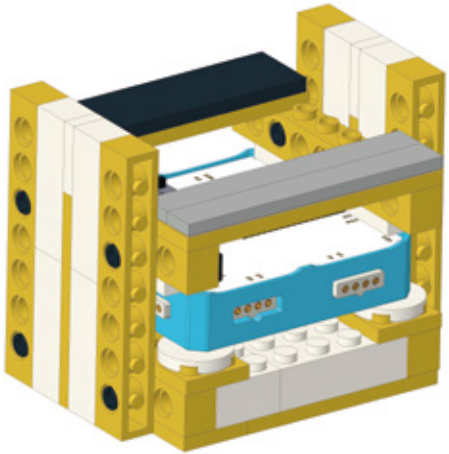


27



• Circuit Connection

Connect the circuits as shown below:



6. Task to Practice

Task Goal: Use the accelerometer to test the value of the punch.

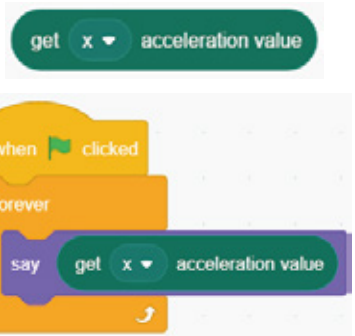
Task Analysis:

1. Get the acceleration;
2. Display the acceleration value when punching;
3. Show the score of each punch.

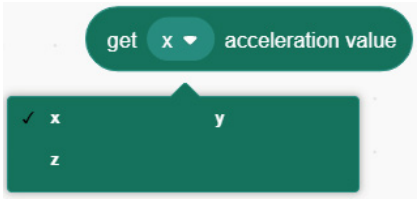
Task steps:

Get the acceleration:

1. Put **[get "x" acceleration value]** into the "say" code block and add a forever code block outside;



2. Click on the program with the mouse and shake the main board, and you can see the change of the acceleration value. The accelerometer can measure acceleration in six directions, namely left and right (X-axis), front and back (Y-axis), and up and down (Z-axis). Choose different directions and measure the change value of acceleration with code block.

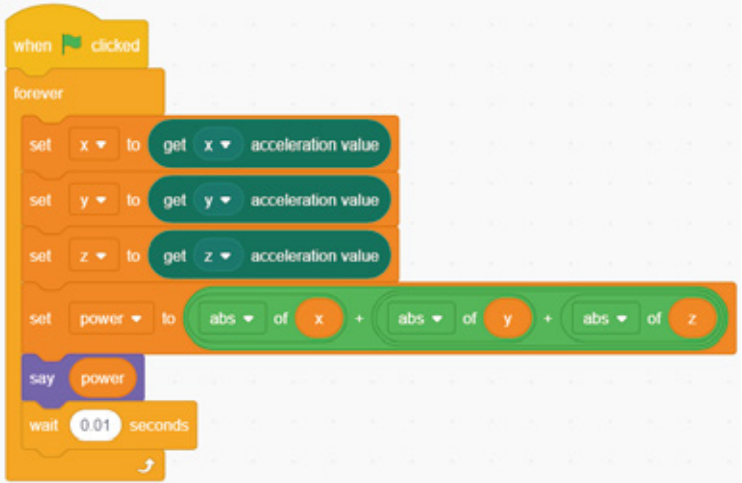


Display the acceleration value when punching:

1. The force of each punch is different, and its acceleration will also be different. Firstly, you need to create three variables to store the values of the accelerometer in the x, y, and z directions of each punch.

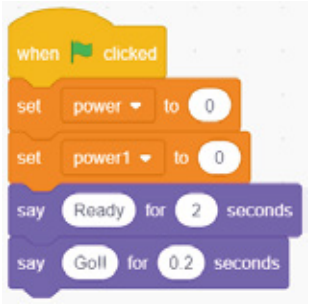
2. Tick the newly created x, y, and z variables respectively, so that the values of these three variables will appear in the upper left corner of the stage area.

3. Hold the main board with your hand, the direction is random. So it is not easy to measure the acceleration in one direction alone. We should add the absolute value of the acceleration of the three axes to reflect the force of the punch. Click the green flag and hold the main board to make a punch to see the change in acceleration.



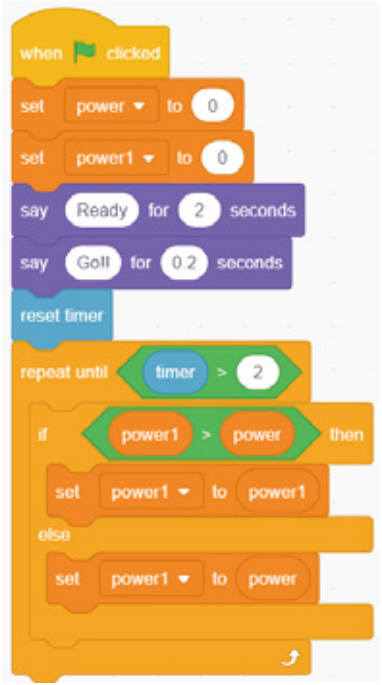
Show the score for each punch:

1. We also need to consider that we must have time for preparing after the program runs. Therefore, we need to add a prompt before punching. In this way, every measurement result is fair and just when we are fully prepared. Here another variable "power1" is set to store the maximum value of punches. And then, every time you run the program, you can punch according to the prompts of the sprite in the stage area.



2. In the process of punching, the acceleration changes all the time. We take the biggest value within two seconds of punching as the punch strength here. Start the timer when you start to punch and execute it for 2 seconds.

3. We store the maximum value measured each time in a variable, and finally get the maximum value from that variable.



Awesome! You have learned how to use accelerometer to get the strength of each punch. Can you try to make the sprite in the game follow your punches and make corresponding punch movements?

We have provided a complete boxing game in the software resources. You can download it and try it out to see the difference between this one we did. The code of other sprites in the game requires you to explore and learn by yourself.

For the complete program, please visit the official website to download: **forum.elecrow.com**.

7. Game Interactions

Come and experience your work with your friends!

Let's invite your good friends to a boxing competition and see who can get the highest score. You can modify the data in the game program to change the difficulty of the game. For example, adjust the acceleration value to use more (less) strength to get a higher score and increase the difficulty of the game.

8. Brainstorming

We used an accelerometer as the trigger module in this project. Accelerometers are widely used in our life, such as checking seismic waves, car accident alarms, and pedometers. Can you try to make a somatosensory game with existing equipment? Use accelerometers to control sprites in the game to move in different directions.

Parts List


x6
x4
x1
x1
x4
x5
x16
x2
x14
x2
x6
x2
x8
x2
x8
x4
x2
x4
x3
x2
x2
x4
x1
x4
x2
x3
x12
x4
x4
x4
x2
x8
x4
x3
x2
3
x8
4
x1
x2
5
x2
8
x2

Parts List


x2
x8
x2
x2
x4
x4
x4
x2
x2
x10
x7
x7
x9
x8
x4
x2
x2
x9
x8
x6
x2
x2
x4
x2
x2
x2
x1
x6
x7
x1
x2
x3
x2