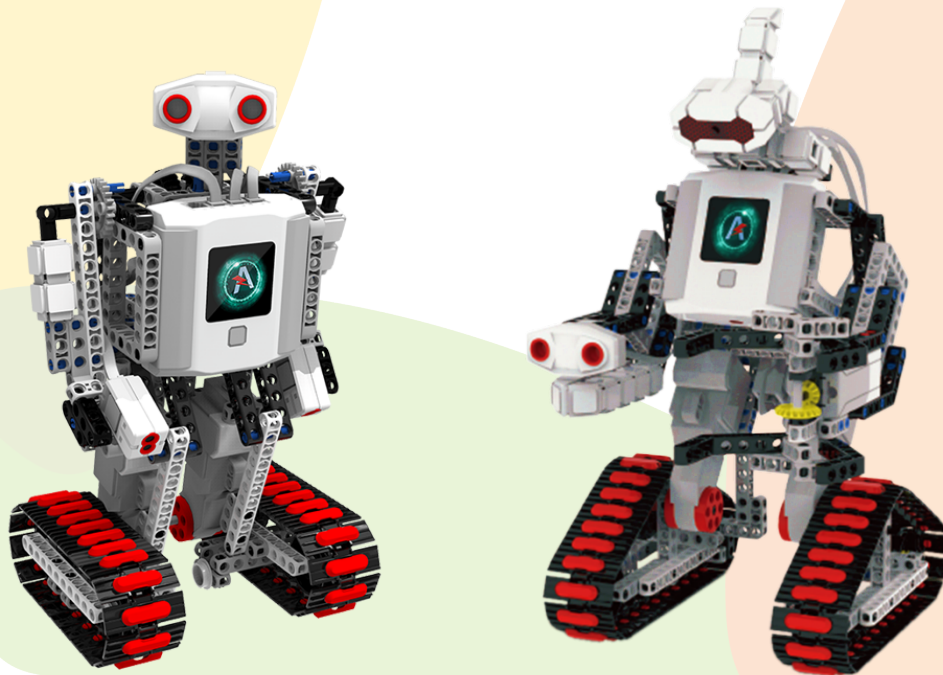




# Robotics Kit

## Krypton User Manual





# Table of Contents

1. Programmable Logic Controller (PLC) .....	4
1.1 Main interface .....	5
Program interface .....	6
Configuration interface .....	6
2. Using Abilix Krypton Software .....	7
2.1 Main menu .....	8
Options menu .....	9
QR code reader .....	9
2.2 Meet Krypton .....	10
Big motor .....	11
Small motor .....	11
Microphone .....	11
Compass .....	12
Gyroscope .....	12
Touch screen .....	12
LED indicator .....	13
Speaker .....	13
Ultrasonic sensor .....	13
Color sensor .....	15
Grayscale sensor .....	15
Touch sensor .....	15
2.3 Project Programming .....	16
Assembly of the project .....	17
Programming .....	18
Downloading the program .....	19
2.4 Abilix Chart .....	20
Tools .....	22
Actuators .....	23
Sensors .....	27
Control .....	29
Line tracker .....	31



Variables .....	37
2.5 Abilix Scratch.....	38
Actuators .....	40
Sensors .....	42
Integrated sensors.....	43
Control.....	44
Mathematical operators.....	45
3. Abilix Chart for Windows .....	46
3.1 Actuators .....	48
3.2 Sensors .....	51
3.3 Control.....	53
3.4 Line tracker.....	54



# 1. Programmable Logic Controller (PLC)



Processor: 4 Core Cortex-A7 1.3GHz

Connection: Wireless via Wi-Fi and Bluetooth

Color display: 2.3 Uni-touch

Memory Flash: 1.61 GB, 512 MB of RAM

Actuators: 2 Speakers, RGB LED

Sensors: Gyroscope, Compass, Microphone

Lithium Polymer Re-Chargeable Battery Environment:

*Model:* 21CP6/36/59

*Voltage:* 7.4V

*Power consumption:* 1500mAh

Power adapter:

*Model:* XSG0841000US

*Input:* 100-240V, 50/60Hz, 0.6A Max.

*Output:* 8.4V, 1A

Supports text, photos and videos.



It contains the following peripherals:

1. **Input ports:** Seven sensor ports and one camera port located on the controller sides. Each can handle analog and digital inputs.
2. **Output ports:** Four ports located on the top of the controller. They can control the rotation speed and direction of direct current motors.
3. **Charging port:** An input port to connect the charger to the battery while it is mounted on the controller.

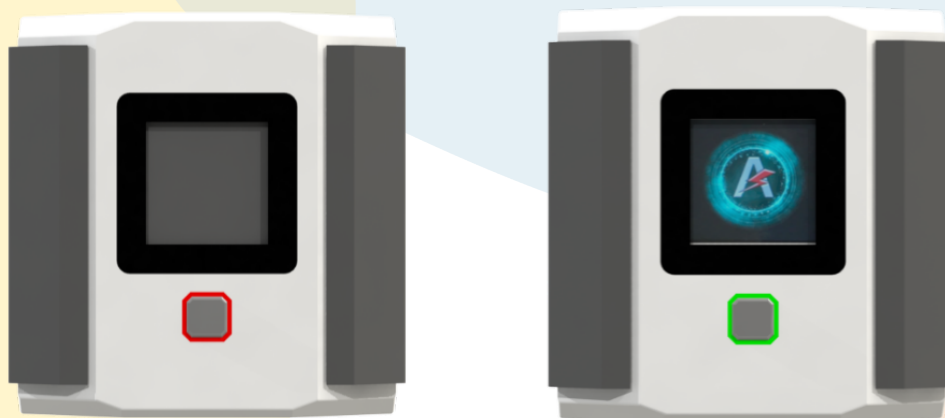









## 1.1 Main interface

To turn on the controller, press and hold the button below the screen until it turns on; you will notice that the area around the button lights up red, and when the controller starts to turn on, it turns green.

Once turned on, the display shows the navigation interface, where you can find the main menu of the controller. To turn off the controller, press and hold the button until the screen turns off.



The main menu includes four items. The one on the screen is the selected item. To access one of these items, swipe the screen to the desired bubble, and then click the bubble to select it. Each element and what it represents is listed in the following table:

Settings	Multimedia	Rest	Soul	QR Code
				
Access the configuration interface.	Access the memory media files such as photos.	Puts the controller on an energy saving mode.	Reproduces an introductory message from Krypton.	Used to establish communication between a mobile device and the controller.



## Program interface

By swiping over the screen, either left or right, you can navigate between the main menu items and the programs stored in the controller. Each program can be given a different name within the programming application.

The default program is called ProgramA. The selected program is the one displayed on the controller screen. To start or stop any program, click on the screen. To indicate that the selected program is running, the bubble will start to rotate; when the program stops running, the bubble will stop.

## Configuration interface

Configuration items:

- **Wi-Fi:** View available wireless networks or generate your own hotspot
- **Volume:** Regulate the intensity of sound
- **Language:** Change the language of the driver
- **Battery:** Display the battery charge
- **Update:** Check the availability of system updates
- **About me:** Display technical information about the controller

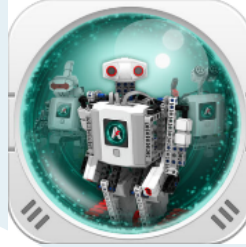


## 2. Using Abilix Krypton Software

To download the Krypton series programming software, search for 'Abilix Krypton' in the Play Store (Android) or App Store (iOS) and download the following application:

### Abilix Krypton - Today's Future

Shanghai PartnerX Robotics Co., Ltd.



**Abilix Krypton** is the main application for the Krypton series; it contains four applications corresponding to the assembly and programming of projects. These applications are described below.



**Meet Krypton:** This is located within the Abilix Krypton application and introduces the operation of the electronic components of the kit. This application allows you to learn about the elements contained in *Krypton* kits by enabling the reading of sensor values and controlling the speed at which the motors rotate.



**Krypton Project Programming:** This application includes assembly tutorials and project programming through block diagrams.



**Krypton Chart Programming:** This application allows you to program projects through the use of flowcharts.

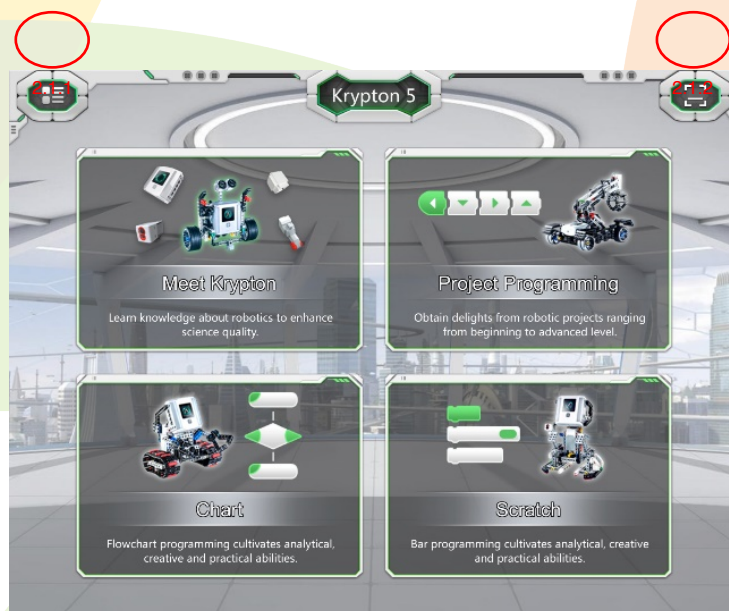


**Krypton Scratch Programming:** This application allows you to program projects through the use of Scratch language.

*NOTE: It is **not** necessary to download the Meet Krypton, Krypton Project Programming, Krypton Chart and Krypton Scratch applications separately, as they are already integrated into the Abilix Krypton application.*

## 2.1 Main menu

This menu allows you to read the QR code and launch the *Meet Krypton*, *Project Programming*, *Chart* and *Scratch* applications, in addition to a menu that displays other options.





## Options menu

At the top left of the screen there is an options menu that displays the following options:

- *User center*: This option lets you log in with the 'Today's Future' account. To return to the main screen, click on 'Experience now.'
- *Type switch*: Allows you to choose the Krypton model you want to work with.
- *Version information*: Contains information about the application version.

## QR code reader

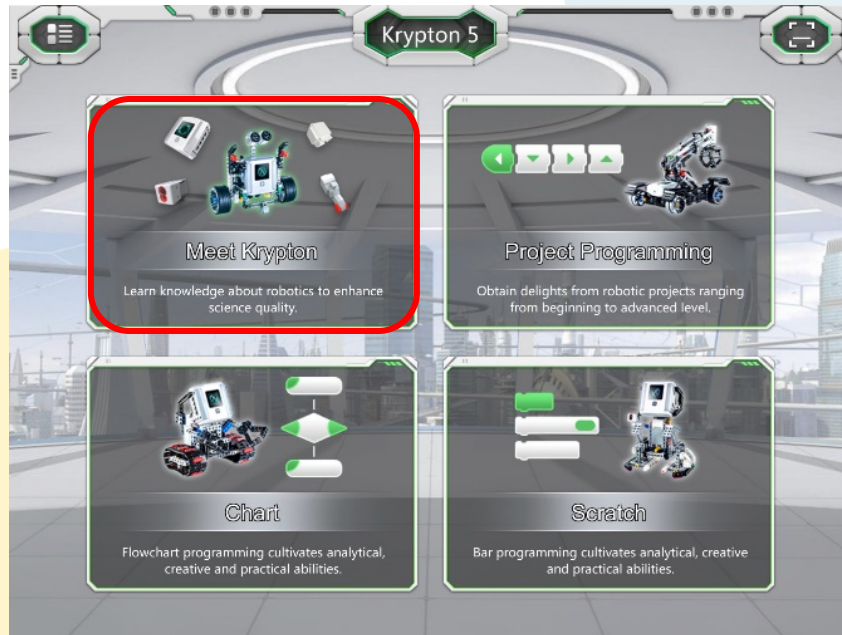
To connect the controller to a device, go to Settings, select Wi-Fi and select *Hotspot*. A message will appear confirming that the hotspot connection is enabled. On your mobile electronic device, go to Wi-Fi settings, select the controller network, and enter your password. Having successfully connected the mobile device to the controller's internet network, navigate to the controller's main menu and swipe to find the QR code.



Click on the button located on the top left of your device; another screen will pop up; in the lower right corner is the *Scan* button. Click it to bring up the camera and focus the QR code present on the controller screen in the center of the camera.



## 2.2 Meet Krypton



The *Meet Krypton* section serves as an introduction to getting to know every part of the kit, starting with the controller, battery, and charger and continuing with the motors and sensors. Follow a lesson to match each component to its name. Instructions for updating the software are included below.

The lessons for the above components are described below:





## Big motor

The speed and rotation direction of the large motor can be controlled using the following screen.



## Small motor

The speed and rotation direction of the small motor can be controlled using the following screen.



## Microphone

This sensor is located inside the controller. On the left side of the screen is a button for making recordings, and on the right side is a button for playing the recording.





## Compass

This sensor is located inside the controller. The red arrow indicates where the top of the controller is pointing.



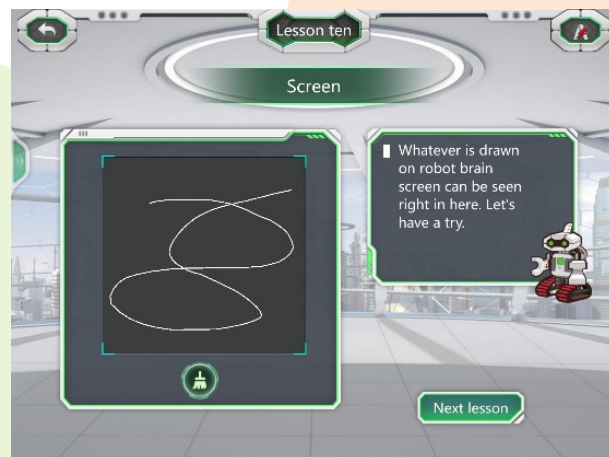
## Gyroscope

This is inside the controller. It grants values to the different axes to calculate the current state of the robot.



## Touch screen

This utility shows what is drawn on the controller's touch screen.

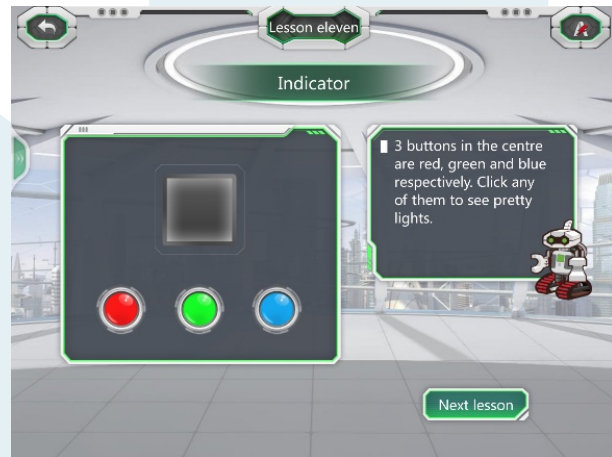






### LED indicator

Choose a color from the three available to turn the LED indicator of the controller on or off.



### Speaker

Plays a variety of sounds.



### Ultrasonic sensor

Measures and displays the distance between the sensor and some obstacle.







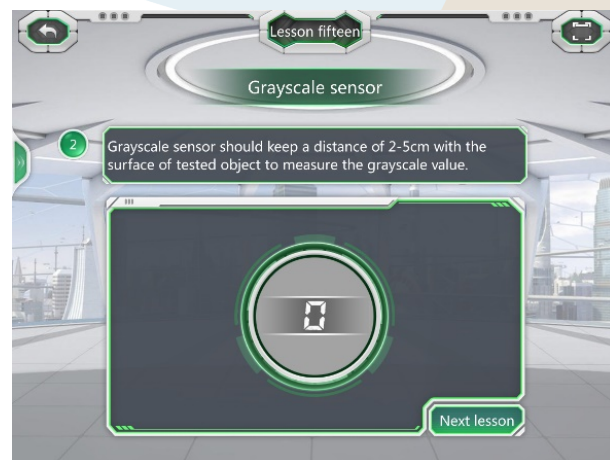
## Color sensor

You can identify the color of a surface and the intensity of light.



## Grayscale sensor

For the correct operation of this sensor, a distance between 2 and 5 cm must be maintained from the surface to be measured, compared and contrasted with shades of color between white and black. Values of approximately 0–1999 represent a white color while values of 2000–4000 represent a black color.



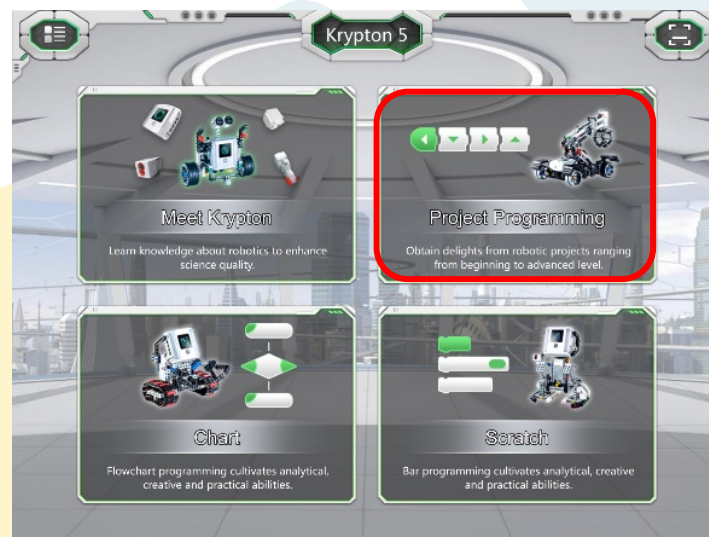
## Touch sensor

This sensor is activated when pressed. It is usually used to detect collisions.





## 2.3 Project Programming



To access programming, click on the logo shown in the image above. You'll see the following screen:

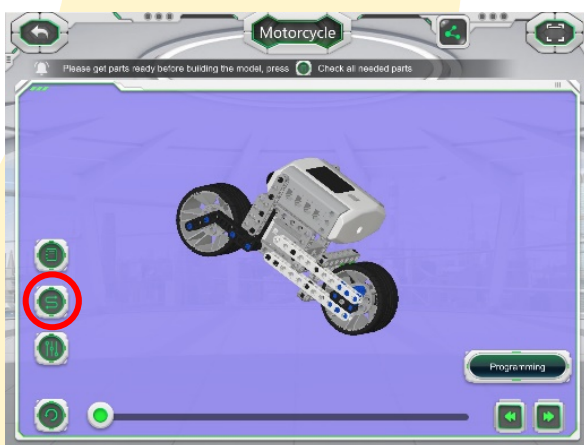
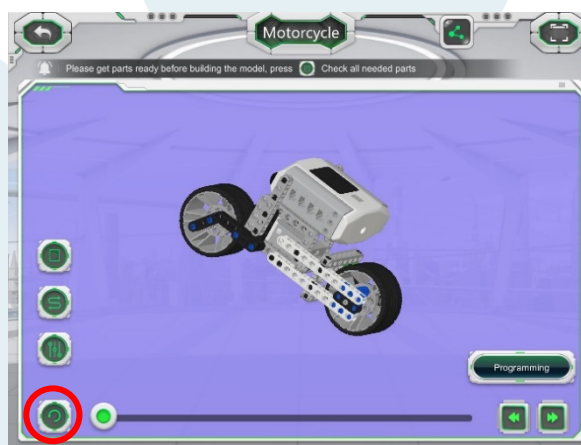


This screen shows the projects that can be carried out with the Krypton model that has been previously chosen, sorted by basic level, advanced level and extensions.

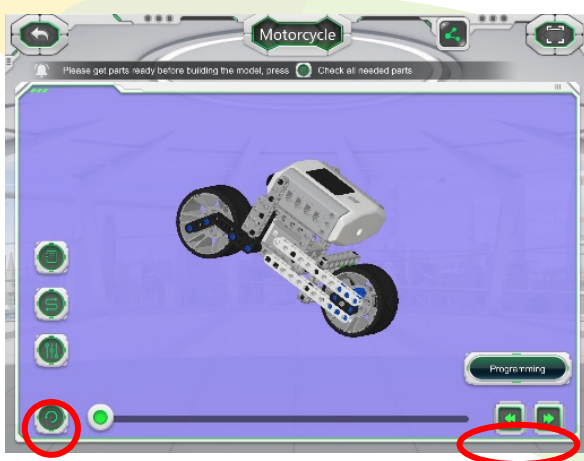


## Assembly of the project

This section gives you access to all existing projects of the selected Krypton model. When you select any project, its assembly manual appears. This manual is played as an interactive video, as it allows you to pause, advance and rotate the project being assembled. If you want to return the model to its original un-manipulated position, you can click on the icon indicated in the image.



At the end of the assembly manual, two buttons appear: one in the form of an 'S' whose function is to highlight the connections present within the project to the PLC, because the blocks of *Krypton Project Programming* already have previous programming and the motors are predefined.



The button with 3 lines is used to connect the PLC to the mobile device. [See Section 2.1.2: QR code reader]

At the end of the assembly, click on the box located in the bottom right to proceed with the programming.





## Programming

When you click the Programming button, the following screen appears.



The blocks at the bottom are designated for the programming of the project that has just been built. The center box gives you space to add the programming blocks and create a sequence.

To add the scheduling blocks, hold down the scheduling block and drag it to the center box. When you drag the programming boxes, a shadow will appear that denotes the place where the programming block can be placed.





To delete a schedule box, simply click on the box you want to delete and drag it out of the center box.



## Downloading the program

To run the program from a mobile device and download it to the controller, it is necessary to connect the PLC to the device that is being used to program. [Review Section 2.1.2: QR code reader]

The buttons and their functions are presented below.

Icon	Name	Function
	<b>RUN</b>	Run the program. The program is NOT saved or downloaded; it just runs.
	<b>DOWNLOAD</b>	Download the program and save it to the driver. The program is saved in the PLC as Project1. The user can also name the program when downloading it.
	<b>CLEAR ALL</b>	Delete the entire program from the screen.
	<b>RESET</b>	Restart the program as long as it is running.

To return to the assembly manual, click on the box at the top left. If you want to return to the main menu, click twice.

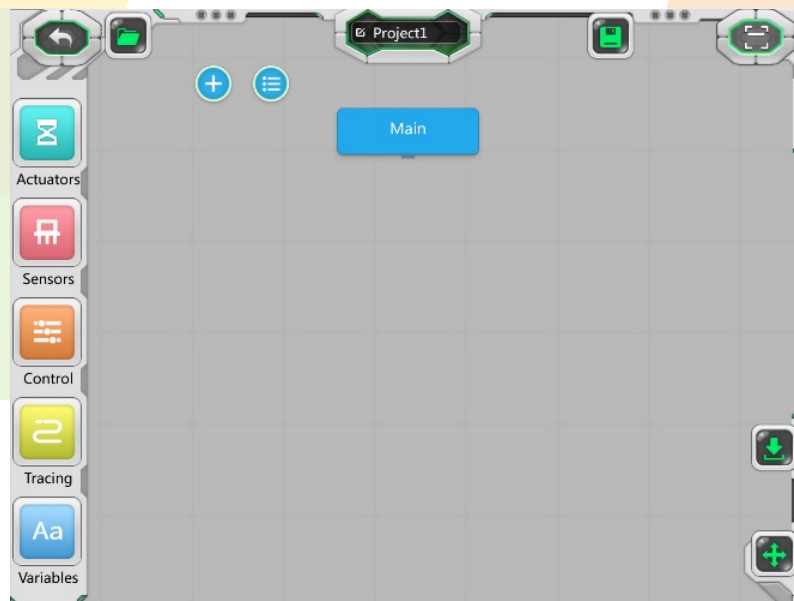


## 2.4 Abilix Chart

Abilix Chart is an application designed to create programming projects for the Krypton series using flowcharts. The concept of programming through diagrams arises from the need to simplify the way programming logic is approached.



This means that a beginner programmer, using Abilix Chart software, will be able to design simple or complex programs without the need for experience or knowledge of any programming language.















The software works through action blocks, time blocks and sensors. Each of these blocks has a specific function and can be configured in different ways; the true potential of this working mode lies in the large number of possible combinations of blocks and configurations.

To access this, click on the third box on the right with the legend *Chart*. When you access this option, you will be asked to choose which Krypton model you want to operate. The folders with already scheduled files and the folder to create a new file will appear at the bottom. To start programming, click on the folder with the '+' sign. A new screen will appear where the programming can take place.



## Tools

The tools on the main screen of the Abilix Chart are described below:

Icon	Name	Function
	<b>Back</b>	Allows you to return to create a new program or choose a different kit from the Krypton series.
	<b>Project1</b>	Displays the name given to the project; this is how the project name will appear once downloaded to the controller.
	<b>New Subprogram</b>	Creates a new subprogram.
	<b>List of Subprograms</b>	Allows you to view and change the different subprograms.
	<b>Main</b>	This is the initial block of the flowchart. All programming must be connected to this block.
	<b>QR Code</b>	Gives you access to the device's camera so you can scan the controller's QR code.
	<b>Save</b>	Saves the program to the device.
	<b>Download</b>	Downloads the program once the driver is connected to the device.

On the left side are libraries and programming blocks.



## Actuators

In this section you will find the programming blocks related to the different functions or options that exist for **actuators**, which in this case are two different types of direct current motors.

### Big Motor



Larger motors contained in the kit. They provide more power than smaller engines.

**Port:** Used to designate which port will be energizing the motor that is connected to the controller.

**Mode:** Variable of the motor to be controlled. You can choose between speed, angle, rotation or time.

**Value:** This is the amount assigned to the motor according to the variable that is activated.

**Speed:** Speed at which the motor will rotate.

**Validity:** Activates the motor.

The screenshot shows a 'Big motor setting' block with the following fields:

- Port:** A row of four buttons labeled A, B, C, and D. Button A is selected and highlighted in blue.
- Mode:** A dropdown menu currently showing 'Speed'.
- Value:** A text input field that is currently empty.
- Speed:** A text input field containing the value '-20'.
- Validity:** A checkbox that is checked, indicated by a small checkmark icon.



## Small Motor



Small motors provide less power than the big motors but are able to rotate at higher speeds.

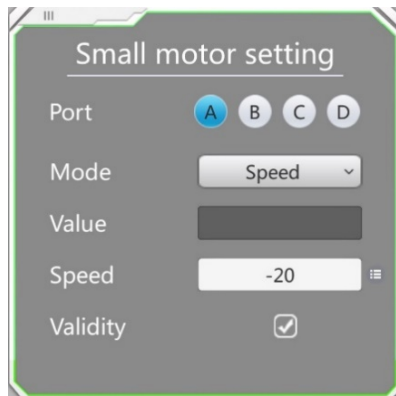
**Port:** Used to designate which port will be energizing the motor that is connected to the controller.

**Mode:** Variable of the motor to be controlled. You can choose between speed, angle, rotation or time.

**Value:** This is the amount assigned to the motor according to the variable that is activated.

**Speed:** Speed at which the motor will rotate.

**Validity:** Activates the motor.



## Speaker

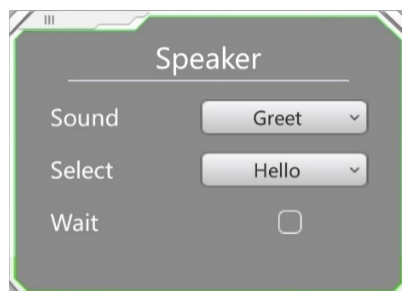


This feature enables playback of a variety of sounds.

**Sound:** Displays the three different sound galleries from which to choose.

**Select:** Displays the different sounds contained in the gallery.

**Wait:** Waits until the sound is completely played to continue to the next instruction.

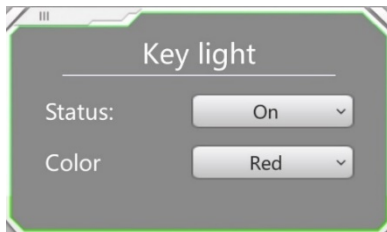




## LED



This block is used to configure and program the color of the driver LED.



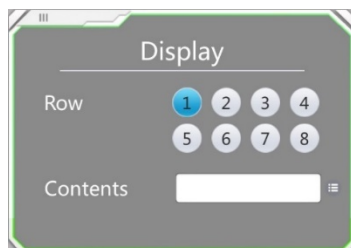
**Status:** Turns the driver LED on or off.

**Color:** Indicates the color of the LED.

## Display



This block has the function of displaying messages on the PLC screen, depending on what the user indicates.



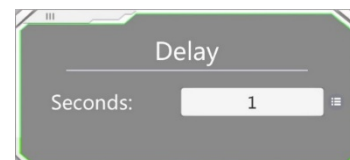
**Row:** Indicates the row in which the content will be displayed.

**Contents:** This is the content (text or values) that will be displayed in the selected row.

## Delay



This function generates a delay within the current program. Values are entered in seconds.

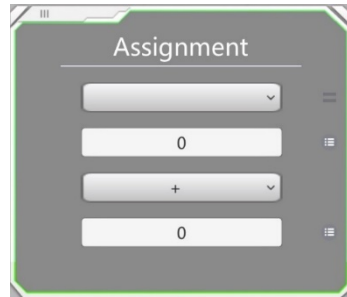




## Assignment



This function is used to make simple mathematical calculations (addition, subtraction, multiplication or division) and assign a value to a variable.



## Stop



Stops actions of LEDs, display and speaker.

## Calibrate Compass



Sets the value of the compass inside the controller.

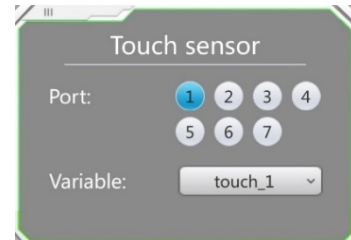


## Sensors

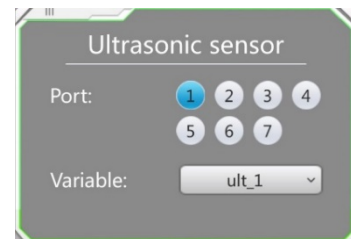
The **Sensor** section contains twelve different blocks. Some blocks are related to the introduction of the time variable and the rest to the introduction of different variables specific to each of the sensors available in the Krypton series.



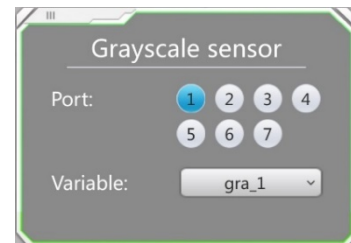
**Touch Sensor:** Sends a signal to the controller about its position (pressed or not pressed).



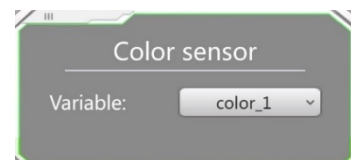
**Ultrasonic Sensor:** Measures the distance at which an obstacle is located. Thus, it can be programmed so that the robot avoids obstacles or measures distances.



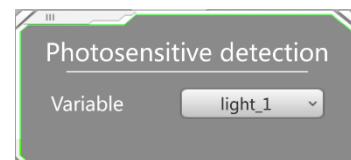
**Grayscale Sensor:** Differentiates between light and dark colors.



**Color Sensor:** Detects the frequency of light waves in a spectrum of 7 different colors.



**Photosensitive Sensor:** The color sensor is also capable of recording light intensity.





**Gyroscope:** Designed to measure, maintain or change the orientation of the controller in space. Information about the tilt of the controller can be obtained and used for a program.



**Compass:** Serves as a compass for the controller and provides information about the position of the Abilix Brain PLC with respect to the four cardinal points.



**Random Digit:** Allows you to define the name of the variable that stores the result and edit the range from 0 to 100.



**Time:** Allows you to select the name of the variable that stores the result and to restart the timer or run the counter from the beginning.



**Camera:** Captures light waves that are subsequently encoded to form an image (i.e., a photograph). The photographs taken will be stored in the controller's memory.





## Control

Complementary or related blocks to those belonging to the **Control** section are included here. These blocks serve to enhance the reach of other blocks through delays, comparators, cycles and so on.



**Condition Judgment:** Used to compare variables and/or constants with the help of some operator (greater than, lesser than, equal to).

The screenshot shows the 'Condition Judgment' block with the following settings: 'Select' is set to 'touch\_1', 'Contrast' is set to '<', and 'Ratio' is set to '0'. There is an 'Add conditions' button at the bottom.



**Multiple Loop:** Any sequence of actions or diagram steps within the Infinite Loop block will be repeated an infinite number of times.

The screenshot shows the 'Multiple Loop' block with 'Loop Time' set to '20' and the 'Infinite Loop' checkbox unchecked.



**Loop Judgment:** Functions as the 'while' command. Parameters are the conditions that users can set. As long as the cycle condition is true, the cycle will continue; at the time the condition is false, the cycle will be finished and the program will stop executing the instructions within it and continue with those outside the cycle.

The screenshot shows the 'Loop Judgment' block with the following settings: 'Select' is set to 'touch\_1', 'Contrast' is set to '<', and 'Ratio' is set to '0'. There is an 'Add conditions' button at the bottom.



**Break:** Used within a certain cycle. When the program reads this block, it stops executing the cycle that surrounds it and continues to the next instruction outside the cycle.



**Continuous Loop:** Used within a given cycle. When the program implements this block, it returns to the first part of the cycle, regardless of the programming that continues.



**Quit:** Used within a given cycle. When the program implements this block, it stops running all cycles and reaches the end of the programming.



## Line tracker

In this function tab, the robot is programmed to function as a **line follower**.

### Initialize



Used to assign ports and operation of the line follower vehicle.

**Motor type:** Allows you to choose between the small or large motor.

**L-Motor / R-Motor port:** Assigns the port to which the vehicle's motors are connected.

**L-Motor / R-Motor power:** Sets the direction and power with which the motors will rotate.

**Grayscale qty:** Allows you to choose how many grayscale sensors will be used: 5 or 7. The connection of the sensors in each controller port is defined from left to right.

**Venue type:** Defines the type of line to work with.

**Threshold value:** This value is adjusted to improve the capture of the black line values depending on the room lighting.

The screenshot shows the 'Initialize' configuration window for a line follower robot. It contains the following settings:

- Motor type:** Big Motor (dropdown)
- Left motor port:** A (dropdown)
- Left motor power:** 1 (text input)
- Right motor port:** B (dropdown)
- Right motor power:** 1 (text input)
- Grayscale qty:** 5 grayscale (dropdown)
- Sensor configuration:** A row of buttons labeled 1, 3, 4, 5, 7. Above the buttons are arrows pointing left and right, and a central button labeled 1. Dotted lines are between the arrows and the central button.
- Venue type:** Black line (dropdown)
- Threshold value deviation:** 0.5 (text input)



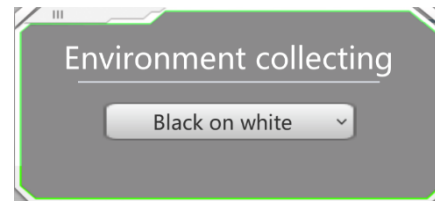
## Environment collecting



Sets the type of line and background with which to calibrate.

Allows the user to choose between:

- Black line on white background
- White line on black background
- Calibration of both





## Intersection tracing



Allows the controller to count each of the crossings along the way and program them from there.

**Intersection type:** Allows you to choose which intersection will have priority, left or right.

**Speed:** Sets the speed at which both vehicle engines will advance.

**L-turn / R-turn diff:** Assigns an adjustment speed for the motors in a given case if necessary.

**Loop time:** Specifies how many intersections will be counted by the controller.

**Cross time:** Allows the vehicle to advance for a defined amount of time after the last intersection counted.

**Stop after ending:** Programs the vehicle to stop or continue moving after the established instructions.

The screenshot shows a control panel titled "Intersection tracing" with the following settings:

Parameter	Value
Intersection type	Left side (dropdown)
Tracing speed	10
Left-turning differential	0
Right-turning differential	0
Loop time	1
Crossing time	0
Stop after ending	Yes (dropdown)



## Scheduled tracing



Programs the vehicle to work for a defined time.

**Speed:** Sets the speed at which both vehicle engines will advance.

**L-turn / R-turn diff:** Assigns an adjustment speed for the motors in a given case if necessary.

**Tracing time:** Defines the amount of time that the programming of this block will work.

**Stop after ending:** Programs the vehicle to stop or continue moving after the established instructions.

The screenshot shows a configuration window titled "Scheduled tracing" with the following settings:

Parameter	Value
Tracing speed	10
Left-turning differential	0
Right-turning differential	0
Tracing time	20
Stop after ending	Yes



## Advanced tracing



This block allows the central grayscale sensors to keep tracking the line until another sensor (excluding grayscale ones) detects a value that validates the conditional statement.

**Speed:** Sets the speed at which both vehicle engines will advance.

**L-turn / R-turn diff:** Assigns an adjustment speed for the motors in a given case if necessary.

**Port:** Defines the port in which the sensor to be used is connected.

**Vs:** Allows logical comparison operations (less than, greater than, equal to, etc.)

**Default value:** Allows you to enter a reference numeric value to compare against the port value.

**Stop after ending:** Programs the vehicle to stop or continue moving after the established instructions.

The screenshot shows the 'Advanced tracing' block interface with the following settings:

- Tracing speed: 10
- Left-turning differential: 0
- Right-turning differential: 0
- Port: 1 (dropdown)
- vs: < (dropdown)
- Default value: 0
- Stop after ending: Yes (dropdown)



## Swerving



Allows the vehicle to make turns at intersections.

**Crossed lines:** Sets the number of lines the sensors will count when making the turn.

**L-motor / R-motor speed:** Assigns a speed for each of the motors.

**Ending position:** Sets the position in which the rotation of the vehicle will end.

**Stop after ending:** Programs the vehicle to stop or continue moving after the established instructions.

The screenshot shows a configuration window titled "Swerving". It contains the following settings:

Swerving	
Crossed lines	<input type="text" value="1"/>
Left motor speed	<input type="text" value="0"/>
Right motor speed	<input type="text" value="0"/>
Ending position	<input type="text" value="Centre"/>
Stop after ending	<input type="text" value="Yes"/>





## Start motor



Used to start the motors again after having turned them off in a previous block.

**Ending mark:** Defines by which method the motors will be turned on again (time or sensor).

**L-motor / R-motor speed:** Assigns a speed for each of the motors. If the delay option is used, the 'delay time' command will be enabled.

**Delay time:** Defines the time that the programming of this block will work. If the 'trigger sensor' option is used, the 'port,' 'contrast' and 'ref' commands will be enabled.

The screenshot shows the 'Start motor' configuration window. It has a title bar with a hamburger menu icon. The window contains the following fields:

Start motor	
Ending mark	wait ▼
Left motor speed	0
Right motor speed	0
Delay time	20

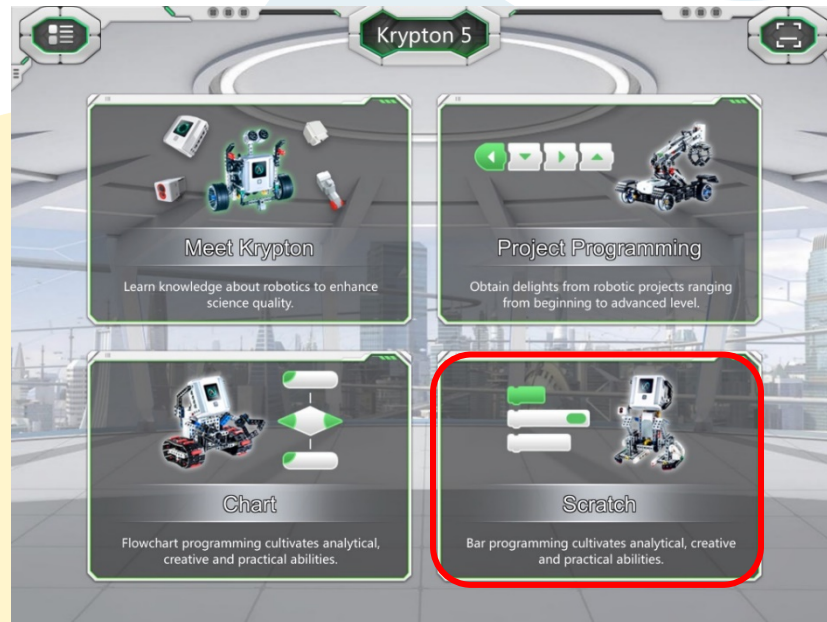
## Variables

This section contains all the variables involved in the flow chart, such as the sensors and/or actuators to be used in the project.

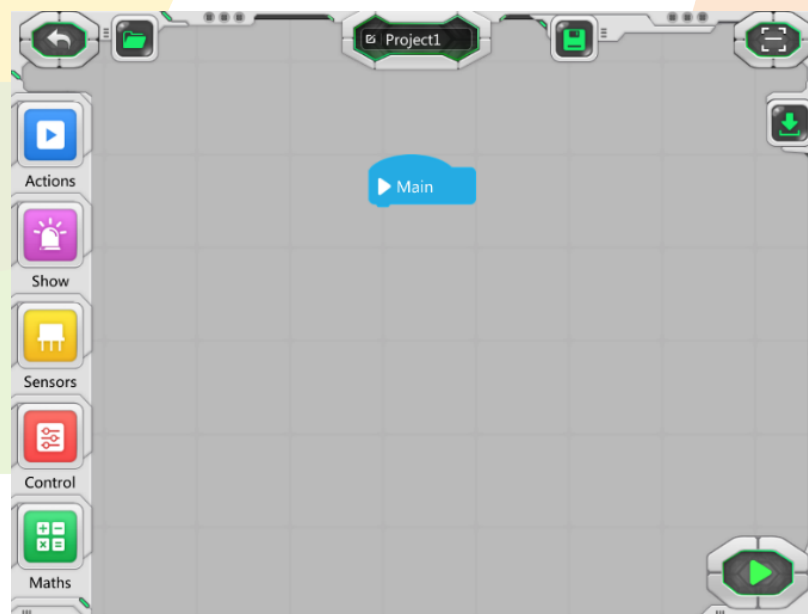


## 2.5 Abilix Scratch

Abilix Scratch is an application designed to create programming projects for the Krypton series using Scratch blocks.









This means that a beginning programmer, using Abilix Scratch software, will be able to design simpler programs without the need for experience or knowledge of any programming language.





The following table describes the functions of the icons contained in the main screen of Abilix Scratch.

Icon	Name	Function
	<b>Project List</b>	Displays the list of stored projects.
	<b>Save Project</b>	Saves the current project and allows you to name the project (or gives it the default <i>Project name</i> ).
	<b>Project1</b>	Displays the name given to the project; this is how the project name will appear once downloaded to the controller.
	<b>Download</b>	Downloads the program or project to the controller.
	<b>Stop</b>	Stops the running program.
	<b>GO</b>	Starts the current program.

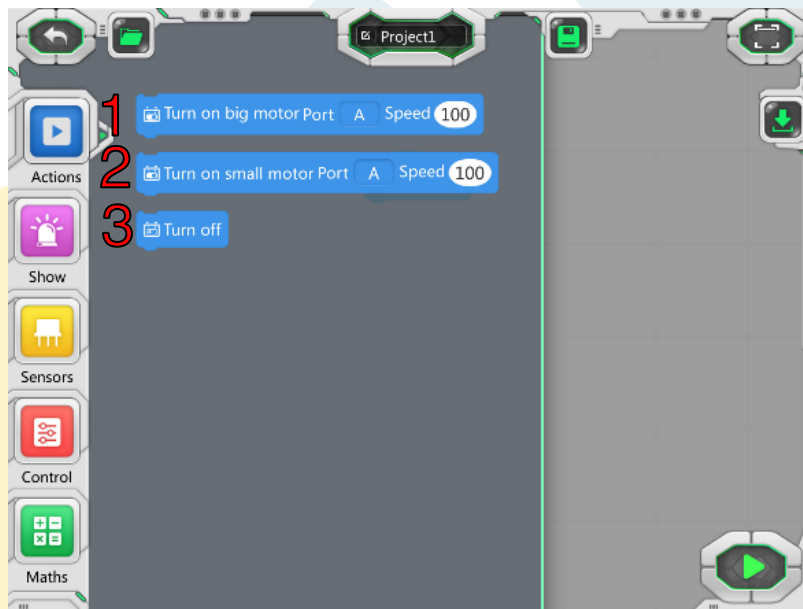
On the far left are the available programming blocks, including **Actions**, **Show**, **Sensors**, **Control** and **Math Operators**.

*NOTE: For any program to work, it is necessary to connect the programming blocks under the initial block called Main program and it is necessary that the Programming function is selected in the controller.*



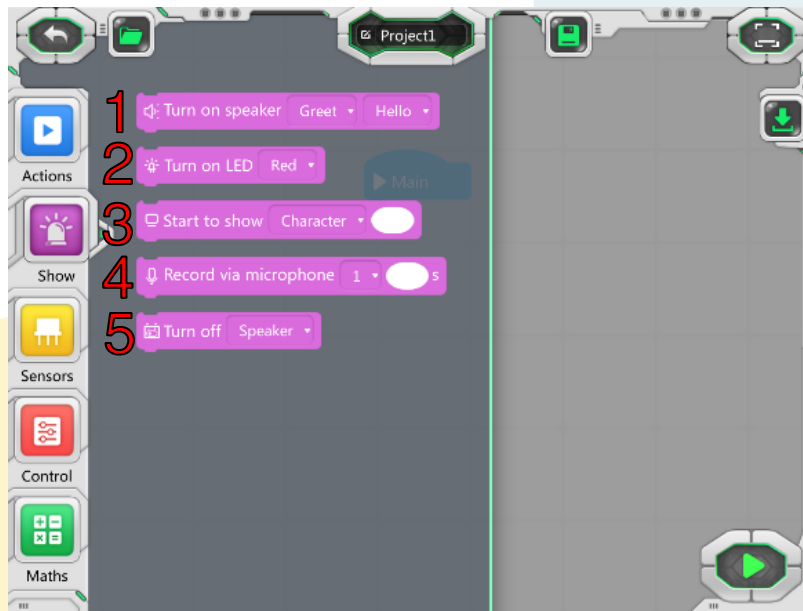
## Actuators

In the Scratch app, the actuators are divided into two sections: **Actions** and **Show**.



**Actions** block: Controls only the motors that connect to the controller with the following options:

1. **Turn on big motor:** Allows you to choose the port in which the motor to be driven is connected (A, B, C or D), the direction in which the motor rotates (counterclockwise or clockwise) and the rotation speed.
2. **Turn on small motor:** Allows you to choose the port in which the motor to be driven is connected (A, B, C or D), the direction in which the motor rotates (counterclockwise or clockwise) and the rotation speed.
3. **Turn off:** Stops any of the actions contained in **Actions**.



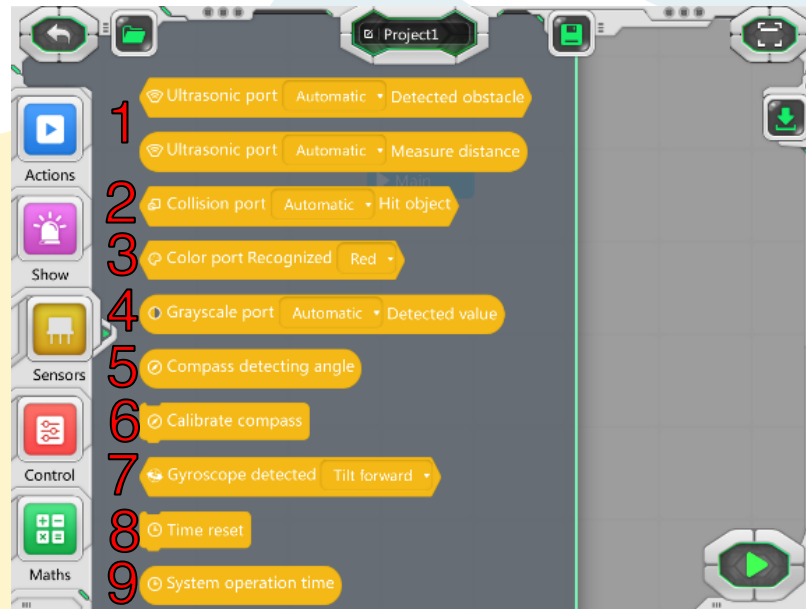
**Show** block: Controls the actuators that are integrated in the controller.

1. **Turn on speaker:** Allows you to choose the type of sound, the description of the sound and the moment when you want to play the sound.
2. **Turn on LED:** Allows you to turn on the LED and choose the color (red, green or blue).
3. **Start to show:** Allows you to choose text or a photo to display on the controller screen.
4. **Record via microphone:** Controls the recording duration via the microphone.
5. **Turn off:** Stops the actions of the speaker, LED or show (screen).



## Sensors

In the Scratch application, sensors are divided into two groups: sensors integrated into the controller and sensors connected to the controller. Both groups are described below.



**Sensors** block: Controls the sensors that are connected to the controller.

1. **Ultrasonic port:** You have two options. The first detects obstacles and the second detects distance. An object is considered an obstacle when it is within 200 mm (20 cm).
2. **Collision port:** Detects when an object is being touched. Gives you the option to choose the port in which it is connected (or this will be detected automatically).
3. **Color port recognized:** Allows you to choose which color to recognize (red / yellow / green / blue / white).
4. **Grayscale port:** Grants a detection value, depending on the perceived opacity. Gives you the option to choose the port in which it is connected (or this will be detected automatically).



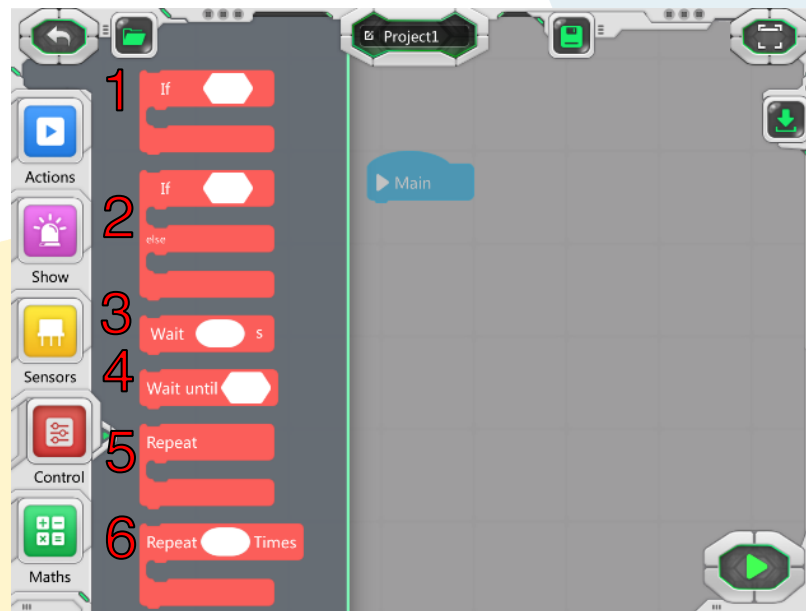
## Integrated sensors

**Sensors** block: Handles the sensors integrated with the controller with the following options:

5. **Compass detecting angle:** Obtains the angle granted by the robot compass ( $0^{\circ}$ – $360^{\circ}$ ).
6. **Calibrate compass:** Runs a pre-established routine to detect the direction of the northern magnetic pole.
7. **Gyroscope detected:** Detects when the controller is tilted forward, backward, left, or right.
8. **Time reset:** Gives the value of '0' to the robot *system time*.
9. **System operation time:** Displays the seconds elapsed since the operation was started.



## Control



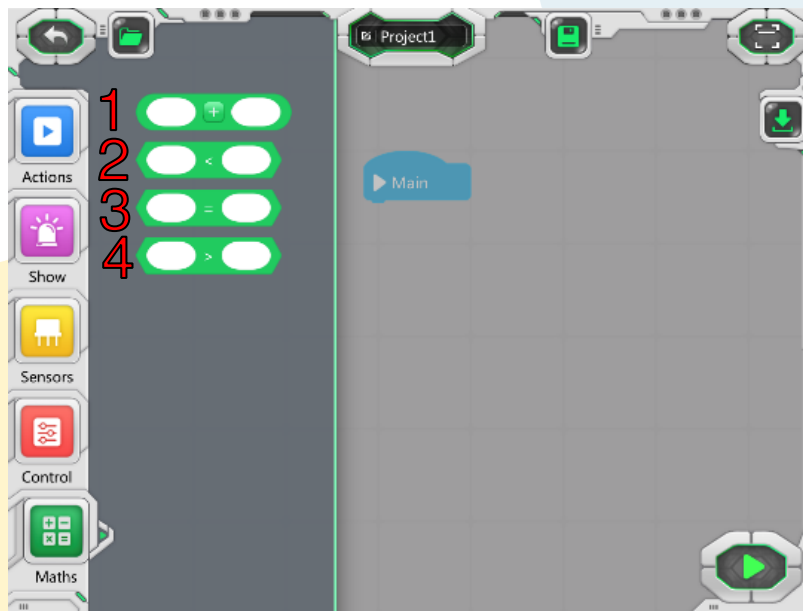
**Control** block: Operates the conditions and cycles of the programming with the following options:

1. **If**: If a condition is true, the action is performed; if not, the program continues.
2. **If else**: If one condition is true, one action is performed; otherwise, another is performed.
3. **Wait**: Sets a delay. Allows you to choose by how many seconds you want to delay the program.
4. **Wait until**: Creates a delay until a certain condition is true.
5. **Repeat**: Creates an infinite cycle. The program will continue to be repeated.
6. **Repeat times**: Repeats the program a certain number of times that the user chooses.
7. **Repeat until**: Repeats the program until a certain condition is true.





## Mathematical operators



**Maths** Block: Controls math operators/conditions the following options:

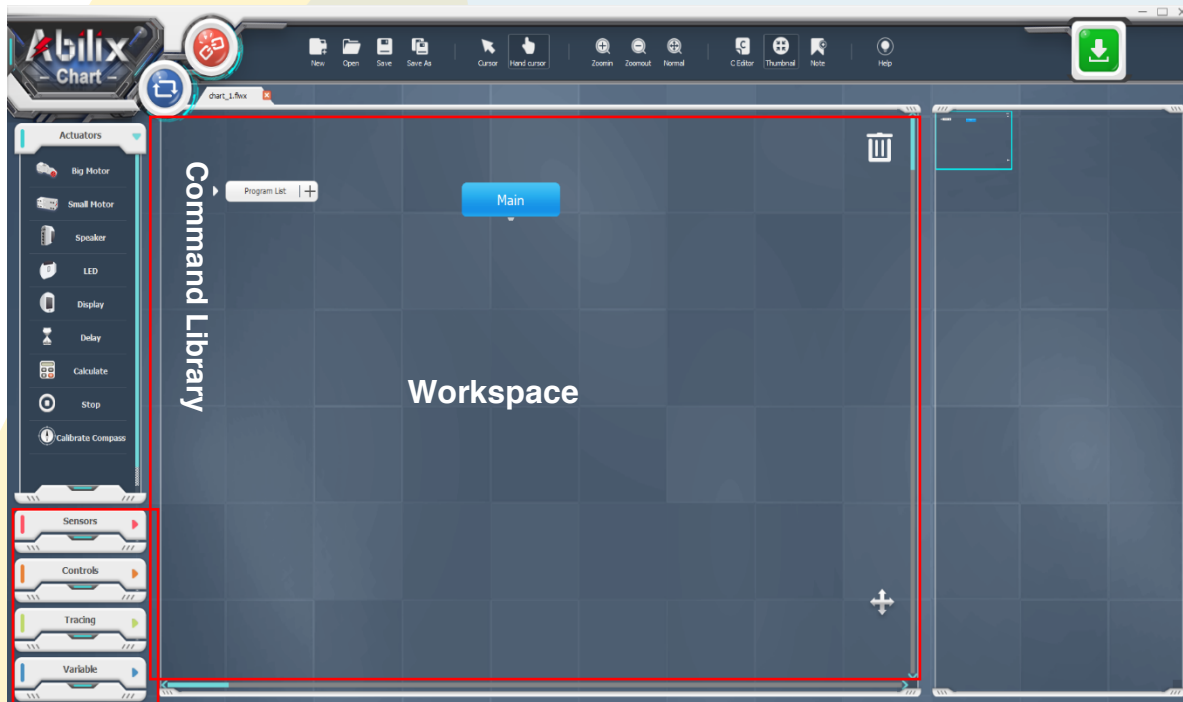
1. **Addition, subtraction, multiplication, division**
2. **Less than**
3. **Equal to**
4. **Greater than**


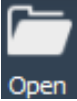




### 3. Abilix Chart for Windows

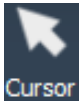



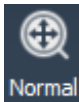
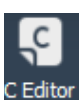





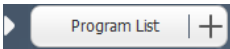


This program for Windows is equivalent to the Abilix Chart program used on iPad and tablet devices.

You can find the Windows software on the <https://abilixlms.com/> page at startup under the name Abilix Chart in the Windows section.



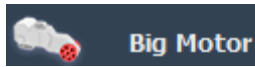
Icon	Name	Function
	<b>New</b>	Creates a new program.
	<b>Open</b>	Opens a previously saved document.
	<b>Save</b>	Saves changes made to the current design.
	<b>Save As</b>	Saves the program to the computer's memory.



	<b>Cursor</b>	Allows the simultaneous selection of different blocks.
	<b>Hand Cursor</b>	Allows you to manually move the work area.
	<b>Zoom In</b>	Allows you to zoom in on the work area.
	<b>Zoom Out</b>	Allows you to move away from the work area.
	<b>Normal</b>	Returns to the standard view of the work area.
	<b>Code Editor</b>	Allows you to visualize the program made in type C language.
	<b>Thumbnail</b>	Opens a window where you can view the complete program code.
	<b>Notes</b>	Adds a text box to the work area.
	<b>Selection</b>	Displays the different kits handled by the Abilix family.
	<b>Connect</b>	Starts searching for a driver to connect.
	<b>Download</b>	Downloads the program from the computer to the controller.
	<b>Program List</b>	Allows you to select a file from the list of saved programs.
	<b>Delete</b>	Removes any blocks that are placed above this icon.
	<b>Center</b>	Centers the view of the program.



## 3.1 Actuators



**Big Motor:** These motors have more power and generate less speed.

**Port:** Sets the port to which the motor is connected.

**Model:** Variable of the motor to be controlled. You can choose between speed, angle, revolutions and time.

**Value:** The quantity assigned to the motor according to the variable that is activated.

**Speed:** The speed at which the motor will turn.

**Effective:** Sets the engine operation.



**Small Motor:** These motors have less power and generate more speed.

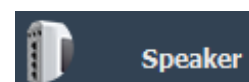
**Port:** Sets the port to which the motor is connected.

**Model:** Variable of the motor to be controlled. You can choose between speed, angle, revolutions and time.

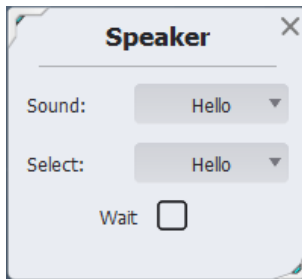
**Value:** The quantity assigned to the motor according to the variable that is activated.

**Speed:** The speed at which the motor will turn.

**Effective:** Sets the engine operation.



**Speaker:** Enables operation of the speakers built into the controller.

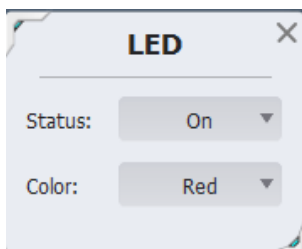


**Sound:** Displays the three different sound galleries.

**Select:** Shows different sounds by gallery.



**LED:** Enables the LED on the controller.

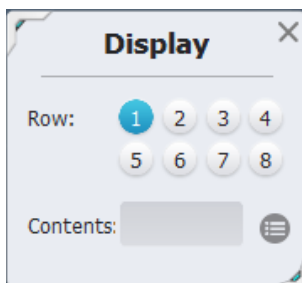


**Status:** Determines whether the LED is on or off.

**Color:** Indicates the color of the light.



**Display:** Enables the display of messages on the controller screen, depending on what the user indicates.

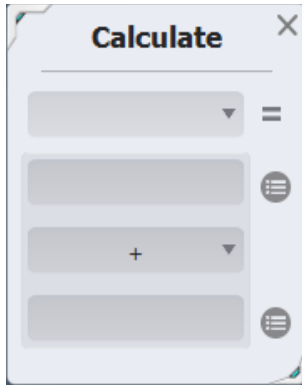


**Row:** Selects the row where the text will be displayed.

**Contents:** Text field to write what you want to show on the screen. It can be text or the value of a variable.



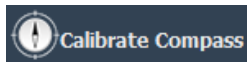
**Delay:** Sets a timeout in seconds between the previous action and the next one in the schedule.



**Calculate:** Allows you to do mathematical calculations (addition, subtraction, multiplication or division) or assign a value to a variable.



**Stop:** Stops certain actions of the program (LED & display).



**Calibrate Compass:** Adjusts the compass value found inside the controller.

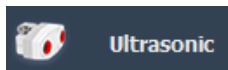
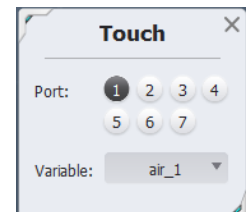


## 3.2 Sensors

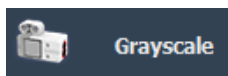
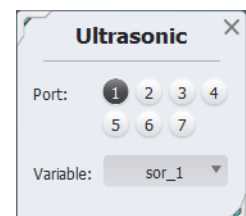
The Sensors section contains twelve different blocks. Some blocks are related to the introduction of the 'time' variable and the rest to the introduction of different specific variables for each of the sensors available in the Krypton series.



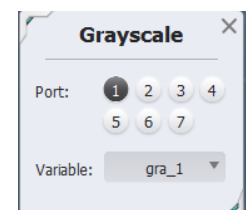
**Touch Sensor:** Sends a signal to the controller about its position (pressed or not pressed) and is saved in the assigned variable.



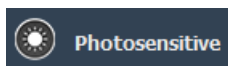
**Ultrasonic Sensor:** Measures the distance at which an obstacle is located; the value is saved in the assigned variable.



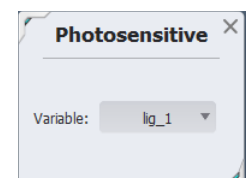
**Grayscale Sensor:** Differentiates between shades of black and white; its value is saved in the assigned variable.



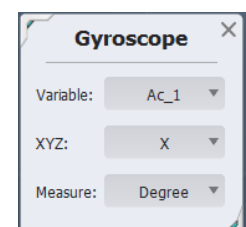
**Color Sensor:** Differentiates between the colors of objects; the value is saved in the assigned variable.



**Photosensitive Sensor:** Measures the intensity of light generated by a light source. Its value is saved in the assigned variable.



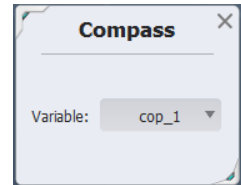
**Variable:** Stores the values obtained by the sensor.  
**XYZ:** Determines the axis to measure.  
**Measure:** Measures at either angles or radians.





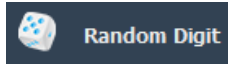
Compass

**Compass:** Provides information regarding the controller's position with respect to the North cardinal point.



Camera

**Camera:** Connects to the port assigned for the camera. Allows you to capture an image.



Random Digit

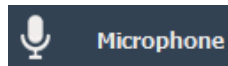
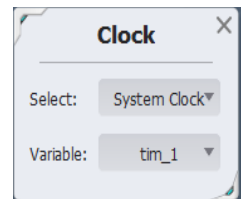
**Random Digit:** Stores the values obtained by the instruction.



Clock

**System Time:** Measures the time it takes to perform an action. The 'Time system' block allows you to enter time as a program variable.

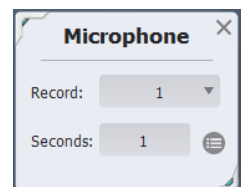
**Reset Time:** Resets the stopwatch that was started with the introduction of the time system. Said restart can be conditioned to different situations.



Microphone

**Record:** Defines the memory pool to save the sound being recorded.

**Seconds:** Sets the recording duration.

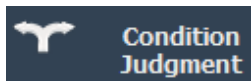




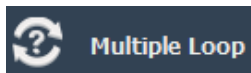
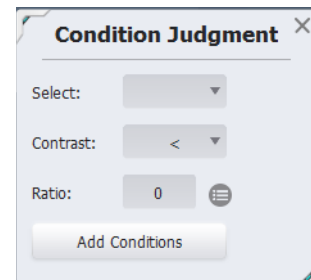


### 3.3 Control

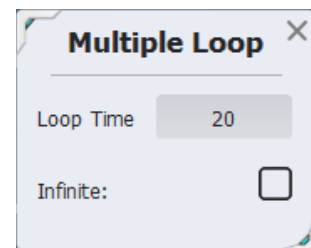
These blocks are used to enhance the scope of other blocks through delays, comparators, 'loops' (cycles), etc.



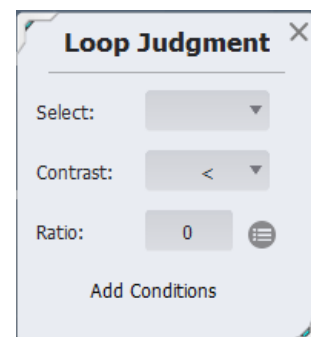
**Condition Judgment:** Allows you to compare variables and/or constants with the help of some operator (greater than, lesser than, equal to).



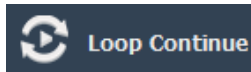
**Multiple Loop:** Any sequence of diagram actions within the block will be repeated a finite number of times. Additionally, it is possible to repeat it infinitely by selecting the Infinite checkbox.



**Loop Judgment:** Parameters are the conditions that users can set. As long as the condition of the cycle is fulfilled, it will continue; when the condition is not fulfilled, the cycle will be terminated and the program will stop executing the instructions within it and will continue with those that are outside the cycle.



**Break:** This block works as a phrase 'Break' in C language. It is used within a certain cycle. When the program reads this block, it stops executing the loop around it and continues to the next statement out of the loop.



**Loop Continue:** This is used within a certain cycle. When the program implements this block, it returns to the first part of the cycle, regardless of the programming that follows it.



**Quit:** This is used within a certain cycle. When the program implements this block, it stops running all cycles and reaches the end of the programming.

### 3.4 Line Tracker

**Initialize:** Used to assign the ports and operation of the line follower vehicle.

**Motor Type:** Selects the small or the big motor.

**L-Motor / R-Motor Port:** Assigns the port where the vehicle's motors are connected.

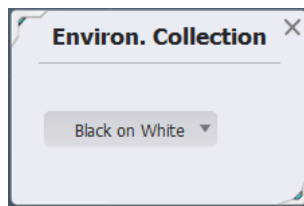
**Power:** Sets the direction of the motors and the power with which they will rotate.

**Ground Gray:** Selects the grayscale sensor: 5 or 7.

The connection of the sensors in each port of the controller is defined from left to right.

**Site Type:** Defines the type of line in which the robot will work.

**Threshold Offset:** This value is adjusted to improve the capture of site type values depending on the lighting in the room.



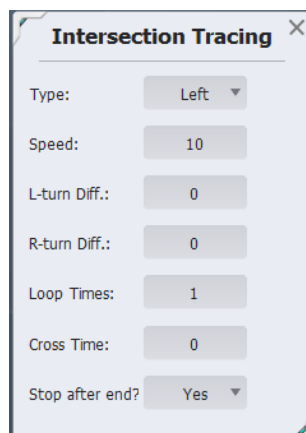
**Environment Collection:** Sets the type of line and background in which it will be calibrated.

Allows users to choose between:

- Black line on white background
- White line on black background
- Calibrate both

**Intersection Tracing:** Allows the controller to count each of the junctions found on the road and to carry out its programming based on it.

**Type:** Allows you to choose which of the intersections will have priority, the left or the right.



**Speed:** Sets the speed at which the motor will advance.

**L-turn / R-turn Diff:** Assigns an adjustment speed for each of the motors if necessary.

**Loop Times:** Lists how many intersections will be counted by the sensor selected in Type before passing to the next block.

**Cross Time:** Allows the vehicle to advance for a defined amount of time after the last intersection counted.

**Stop After End?:** Programs the vehicle to stop or continue moving after the established instructions.



The screenshot shows the 'Timed Tracing' configuration window. It has a title bar with a close button. Inside, there are five settings: 'Tracing Speed' with a value of 10, 'L-turn Diff.' with a value of 0, another 'L-turn Diff.' with a value of 0, 'Time' with a value of 20, and 'Stop after end?' with a dropdown menu set to 'Yes'.

**Timed Tracing:** Programs the vehicle to work within a defined time.

**Speed:** Sets the speed at which both vehicle motors will advance.

**L-turn / R-turn Diff:** Assigns an adjustment speed for each of the motors if necessary.

**Time:** Defines the amount of time that the programming of this block will work.

**Advanced Tracing:** This block works until a sensor (excluding the grayscale sensor) is activated.

**Tracing Speed:** Sets the speed at which both vehicle engines will advance.

**L-turn / R-turn Diff:** Assigns an adjustment speed for each of the motors if necessary.

**Port:** Defines the port in which the sensor to be used is connected.

**Contrast:** Allows logical operations (<, >, =).

**Ref. (Reference):** Allows you to enter numeric values to compare against the port.

**Stop After End?:** Programs the vehicle to stop or continue moving after the established instructions.

The screenshot shows the 'Advanced Tracing' configuration window. It has a title bar with a close button. Inside, there are five settings: 'Tracing Speed' with a value of 10, 'L-turn Diff.' with a value of 0, another 'L-turn Diff.' with a value of 0, a section with 'Port' (dropdown set to 1), 'Contrast' (dropdown set to <), and 'Ref.' (value 0), and 'Stop after end?' with a dropdown menu set to 'Yes'.



**Turn:** Allows the vehicle to make specific turns at intersections.

The 'Turn' block configuration window has a title bar with a close button. It contains five settings: 'No. of Lines' with a numeric input set to 1; 'L-motor Spd' with a numeric input set to 0; 'R-motor Spd' with a numeric input set to 0; 'End Position' with a dropdown menu set to 'Middle'; and 'Stop after end?' with a dropdown menu set to 'Yes'.

**No. of Lines:** Establishes the number of lines which the sensors will count to make their turn.

**L-turn / R-turn Speed:** Assigns a speed for each of the motors.

**End Position:** Defines which sensor will see the black line to follow.

**Stop After End?:** Programs the vehicle to stop or continue moving after the established instructions.

**Start Motor:** Starts the engines again after having turned them off in a previous block.

**Ending Mark:** Defines by which method the motors will be turned on again (time or sensor).

The 'Start Motor' block configuration window has a title bar with a close button. It contains four settings: 'Ending Mark' with a dropdown menu set to 'Delay'; 'L-motor Speed' with a numeric input set to 0; 'R-motor Speed' with a numeric input set to 0; and 'Delay Time' with a numeric input set to 20.

**L-turn / R-turn Speed:** Assigns a speed for each of the motors. If the delay option is used, the 'delay time' command will be enabled.

**Delay Time:** Defines the amount of time that the programming of this block will work.

If the 'trigger sensor' option is used, the 'port,' 'contrast' and 'ref' commands will be enabled.

**Port:** Defines the port in which the sensor to be used is connected.

**Contrast:** Allows logical operations (<, >, =).



**Reference:** Allows you to enter numeric values to compare against the port.