Blockly

User Guide (CR Robots)

Issue: V1.0

Date: 2021-04-13

Shenzhen Yuejiang Technology Co., Ltd

Disclaimer

To the maximum extent permitted by applicable law, the products described (including its hardware, software, and firmware, etc.) in this document are provided **AS IS**, which may have flaws, errors or faults. Yuejiang makes no warranties of any kind, express or implied, including but not limited to, merchantability, satisfaction of quality, fitness for a particular purpose and non-infringement of third party rights. In no event will Yuejiang be liable for any special, incidental, consequential or indirect damages resulting from the use of our products and documents.

Before using our product, please thoroughly read and understand the contents of this document and related technical documents that are published online, to ensure that the robot is used on the premise of fully understanding the robot and related knowledge. Please use this document with technical guidance from professionals. Even if follow this document or any other related instructions, Damages or losses will be happening in the using process, Dobot shall not be considered as a guarantee regarding all security information contained in this document.

The user has the responsibility to make sure following the relevant practical laws and regulations of the country, in order that there is no significant danger in the use of the robot.

Shenzhen Yuejiang Technology Co., Ltd

Address: Address: Floor 9-10, Building 2, Chongwen Garden, Nanshan iPark, Liuxian Blvd, Nanshan District, Shenzhen, Guangdong Province, China

Website: www.dobot.cc

Preface

Purpose

This manual introduces the functions and usage of Blockly, which is convenient for users to understand and use robot.

Intended Audience

This document is intended for:

- Customer
- Sales Engineer
- Installation and Commissioning Engineer
- Technical Support Engineer

Change History

Date	Change Description
2021/04/13	The first release

Symbol Conventions

The symbols that may be founded in this document are defined as follows.

Symbol	Description
⚠DANGER	Indicates a hazard with a high level of risk which, if not avoided, could result in death or serious injury
⚠WARNING	Indicates a hazard with a medium level or low level of risk which, if not avoided, could result in minor or moderate injury, robot damage
⚠NOTICE	Indicates a potentially hazardous situation which, if not avoided, can result in equipment damage, data loss, or unanticipated result
🕮NOTE	Provides additional information to emphasize or supplement important points in the main text

Contents

1. Overview

Blockly is a kind of building block programming. You can write programs by block to quickly and conveniently control the robot. Figure 1.1 shows the blockly panel, and Table 1.1 lists the description of blockly panel.

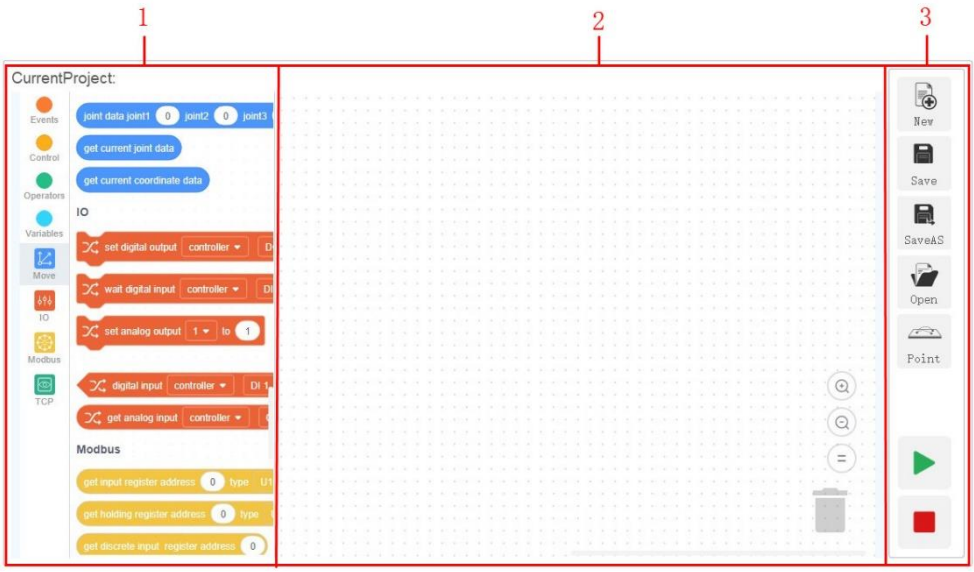


Figure 1.1 Blockly panel

Table 1.1 Description of blockly panel

No.	Description
1	Block area Provide all blocks
2	Code area Drag block to this page and edit it. Click the icon in the code area to zoom in, zoom out and restore the blocks, can be used to delete the selected block

No.	Description
3	Menu bar  Create a new project  Save the project  Save the current project with a new name  Save teaching point that can be called when writing a program  Start running the program in the current code area.  Stop the running program

2. Introduction of Commands

2.1 Motion Commands

Table 2.1 Line move command

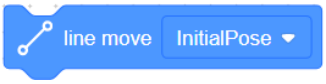
Instruction	
Description	Move from the current position to a target position in straight line mode
Parameter	InitialPose: Indicate target point, which is obtained from the TeachPoint page
Return	None

Table 2.2 Joint move command

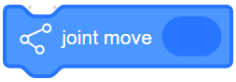
Instruction	
Description	Move from the current position to a target position in point to point mode
Parameter	Indicate the joint angle of the target position, the joint angles is set by Joint data command, please see Table 2.9.
Return	None

Table 2.3 Coordinate move command

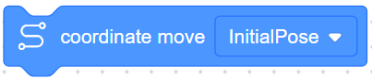
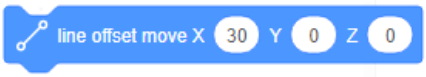
Instruction	
Description	Move from the current position to a target position in point to point mode
Parameter	InitialPose: Indicate target point, which is obtained from the TeachPoint page
Return	None

Table 2.4 Line offset move command

Instruction	
Description	Move the corresponding offset in X, Y and Z directions from the current position in straight line mode

Parameter	X: Indicate offset of X axis Y: Indicate offset of Y axis Z: Indicate offset of Z axis
Return	None

Table 2.5 Joint offset move command

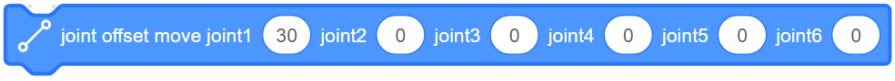
Instruction	
Description	Move the joint offset in each axis from the current position in the Joint coordinate system
Parameter	joint1~ joint 6: Indicate offset of J1 - J6 axes
Return	None

Table 2.6 Coordinate command

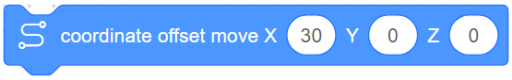
Instruction	
Description	Move the joint offset in each axis from the current position in point to point mode
Parameter	X: Indicate offset of X axis Y: Indicate offset of Y axis Z: Indicate offset of Z axis
Return	None

Table 2.7 Arc command

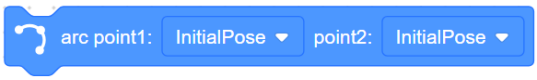
Instruction	
Description	Move from the current position to a target position in an arc interpolated mode
Parameter	point1: Indicate middle point, which is obtained from the TeachPoint page point2: Indicate end point, which is obtained from the TeachPoint page
Return	None

Table 2.8 Circle command

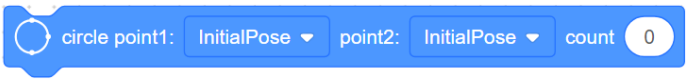
Instruction	
Description	Move from the current position to a target position in a circular interpolated mode
Parameter	point1: Indicate middle point, which is obtained from the TeachPoint page point2: Indicate end point, which is obtained from the TeachPoint page Count: number of whole circles, value range: 1 ~ 999
Return	None

Table 2.9 Joint data command

Instruction	
Description	Set the joint angle
Parameter	joint1~ joint 6: Indicate joint angle of J1 - J6 axes
Return	None

Table 2.10 Get current joint data

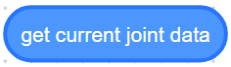
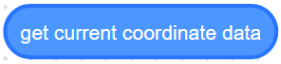
Instruction	
Description	Get the current pose of the robot under the Joint coordinate system
Parameter	None
Return	Joint angle of J1 - J6 axes

Table 2.11 Get current coordinate data

Instruction	
Description	Get the current pose of the robot
Parameter	None
Return	Cartesian coordinate of the current pose

2.2 I/O Commands

Table 2.12 Set digital output command

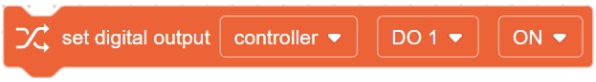
Instruction	
Description	Set the status of digital output port
Parameter	Control end: controller or tool DO: Digital output index. When you select controller, the value range is DO1~DO16; when you select tool, the value range is DO1~DO2 Status: set the DO to on or off
Return	None

Table 2.13 Wait digital input command

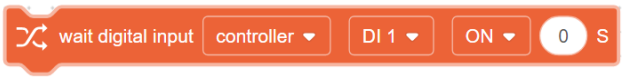
Instruction	
Description	If any of the following conditions are met, the program continues to execute: • The status of DI is the same as the status set by the instruction • The status of DI is different from the state set by the instruction, while the waiting time exceeds the preset time
Parameter	Control end: controller or tool DI: digital input index. When you select controller, the value range is DI1~DI32; when you select tool, the value range is DI1~DI2 Status: Indicate status of DI Time: Set the waiting time, if the waiting time value is 0, it will wait until the condition is met
Return	None

Table 2.14 Set analog output command

Instruction	
Description	Set the value of analog output port
Parameter	Port: analog output index Parameters: the value of the analog output
Return	None

Table 2.15 Digital input command

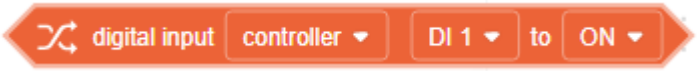
Instruction	
Description	Get the current I/O status and make judgments, which can be used as judgment conditions in a certain instruction
Parameter	Control end: controller or tool DI: digital input index. When you select controller, the value range is DI1~DI32; when you select tool, the value range is DI1~DI2 Status: indicate status of DI Time: set the waiting time, if the waiting time value is 0, it will wait until the condition is met
Return	Return the judgment result: the condition is true or not

Table 2.16 Get analog input command

Instruction	
Description	Get the value of analog input port
Parameter	Control end: controller or tool DI: analog input index
Return	Return the value of analog input port

2.3 Modbus Commands

Table 2.17 Get input register command

Instruction	
Description	Read the input register value with the specified data type from the Modbus slave
Parameter	Address: Starting address of the input registers. Value range: 0 – 4095 type: Data type • Empty: Read 16-bit unsigned integer (two bytes, occupy one register) • “U16”: Read 16-bit unsigned integer (two bytes, occupy one register) • “U32”: Read 32-bit unsigned integer (four bytes, occupy two registers) • “F32”: Read 32-bit single-precision floating-point number (four bytes, occupy two registers) • “F64”: Read 64-bit double-precision floating-point number (eight bytes, occupy four registers)

Return	Return the input register value

Table 2.18 Get holding register command

Instruction	
Description	Read the holding register value from the Modbus slave according to the specified data type
Parameter	Address: starting address of the holding registers. Value range: 0 - 4095 type: Data type • Empty: Read 16-bit unsigned integer (two bytes, occupy one register) • "U16": Read 16-bit unsigned integer (two bytes, occupy one register) • "U32": Read 32-bit unsigned integer (four bytes, occupy two registers) • "F32": Read 32-bit single-precision floating-point number (four bytes, occupy two registers) • "F64": Read 64-bit double-precision floating-point number (eight bytes, occupy four registers)
Return	Return the holding register value

Table 2.19 Get discrete input register command

Instruction	
Description	Read the discrete input register value from Modbus slave
Parameter	Address: starting address of the discrete inputs register. Value range: 0-4095
Return	Return the discrete input register value

Table 2.20 Get coil register command

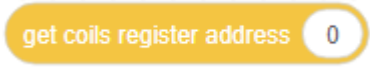
Instruction	
Description	Read the coil register value from the Modbus slave
Parameter	Address: starting address of the coils register. Value range: 0 - 4095
Return	Return the coil register value

Table 2.21 Set coil register command

Instruction	
Description	Set the coil register in the Modbus slave This command is not supported when the coil register address is from 0 to 5
Parameter	Address: Starting address of the coils register. Value range: 6 - 4095 Value: the value written into the coil register
Return	None

Table 2.22 Set holding register command

Instruction	
Description	Set the holding register value in the Modbus slave
Parameter	Address: Starting address of the holding registers to set. Value range: 0 - 4095 type: Data type • Empty: Read 16-bit unsigned integer (two bytes, occupy one register) • “U16”: Set 16-bit unsigned integer (two bytes, occupy one register) • “U32”: Set 32-bit unsigned integer (four bytes, occupy two registers) • “F32”: Set 32-bit single-precision floating-point number (four bytes, occupy two registers) • “F64”: Set 64-bit double-precision floating-point number (eight bytes, occupy four registers)
Return	None

2.4 TCP Commands

Table 2.23 Open socket command

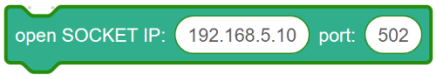
Instruction	
Description	Create a TCP network, robot as a client
Parameter	IP: IP address of the server port: port of the server
Return	None

Table 2.24 Get open socket command

Instruction	
Description	Get the connection result
Parameter	None
Return	0: TCP connection is successful 1: Input parameters are incorrect 2: Socket object is not found 3: Timeout setting is incorrect 4: If the robot is set as a client, it indicates that the connection is wrong. If the robot is set as a server, it indicates that receiving data is wrong

Table 2.25 Create socket command

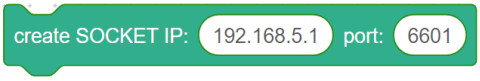
Instruction	
Description	Create a TCP network, robot as a server
Parameter	IP: address of the server port: Server port The port cannot be set to 502 and 8080. Otherwise, it will be in conflict with the Modbus default port or the port used in the conveyor tracking application, causing the creation to fail
Return	None

Table 2.26 Get create socket command

Instruction	
Description	Get the connection result
Parameter	None
Return	0: TCP network is created successfully 1: TCP network is created failed Socket: Socket object

Table 2.27 Socket send variable command

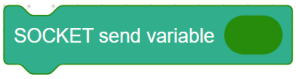
Instruction	
Description	Send data through socket communication
Parameter	Variable: data to be sent
Return	None

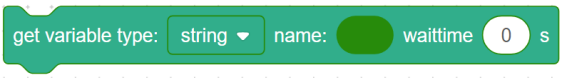
Table 2.28 Close socket command

Instruction	
Description	Release a TCP network
Parameter	None
Return	None

Table 2.29 Get socket send result command

Instruction	
Description	Get the result of the data communication through the Socket
Parameter	None
Return	0: Sending data is successful 1: Sending data is failed

Table 2.30 Get variable command

Instruction	
Description	Obtain data through Socket communication
Parameter	Type: string or number Name: Variable used to hold data waiting time: Set the waiting time, if the waiting time value is 0, it will wait until get data
Return	None

2.5 Variables Commands

Table 2.31 Make a variable command

Instruction	
Description	Make a variable
Parameter	Variable Name: You must start with a letter, and you can't use special characters like Spaces in variable names
Return	Returns a variable

Table 2.32 Set the value of a variable

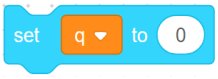
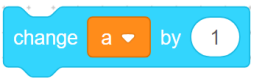
Instruction	
Description	Set the value of a variable
Parameter	Name: name of a variable parameter: value of a variable
Return	None

Table 2.33 Modify the value of a variable

Instruction	
Description	Modify the value of a variable
Parameter	Name: name of a variable parameter: The value of an increase or decrease
Return	None

3. Description of Programming

3.1 Basic operation

Prerequisites

The robot has been powered on.

Procedure

Step 1 Click  to enter the blockly page.

The system creates a new project by default and only supports only single thread running programs.

Step 2 Drag the blocks to the code area to start programming, as shown in Figure 3.1.

- Set the corresponding parameters of each block according to actual needs, for details see 2 Introduction.
- In the **point** page, you can save teaching point, when setting the parameters of the block, you can call the save point directly, for details see 3.2 Teaching points.

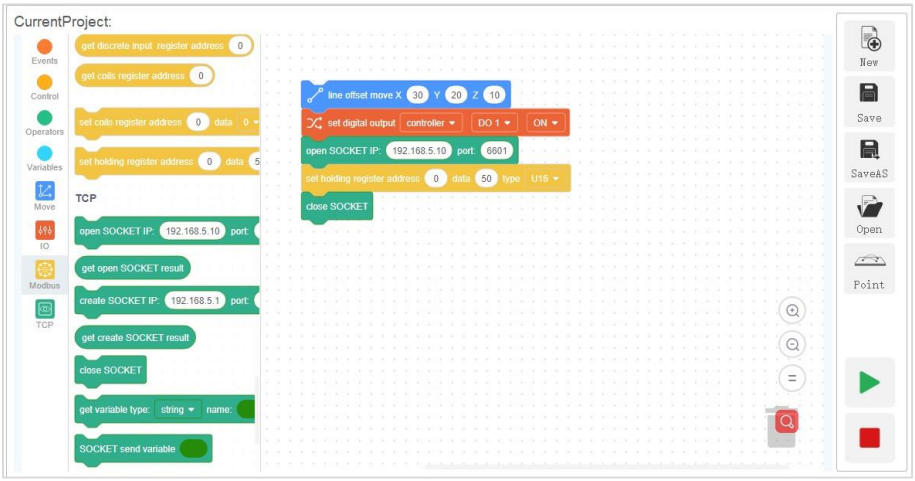


Figure 3.1 Writing a program

Step 3 Click  to save the current project

If it is the first time to save, you need to enter the project name.

Step 4 Click  to enable the robotic arm.

3.2 Teaching points

Prerequisites

The project has been created or imported.

Procedure

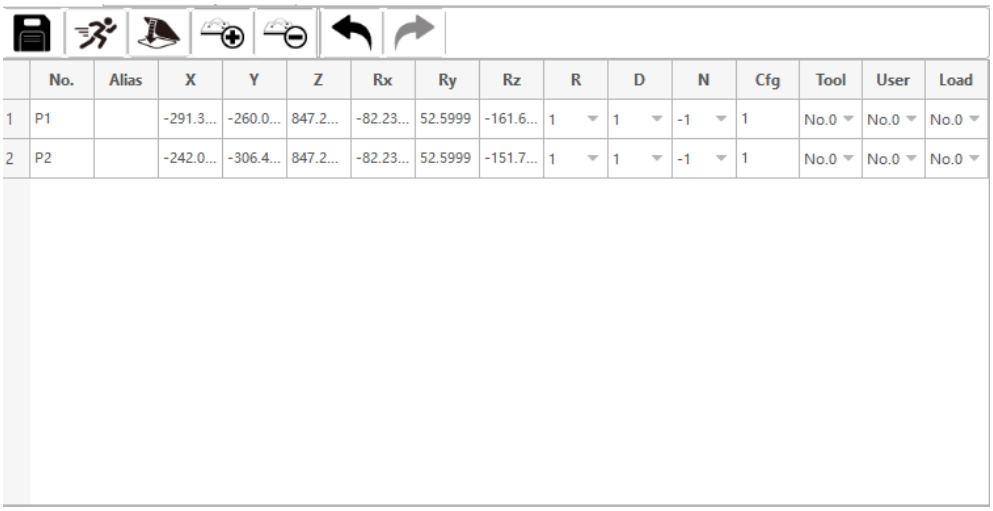
After creating a project, please teach positions on the **point** page for calling commands when programming a robot. If the existing taught positions list has been imported, this operation can be skipped.

Step 1 Enable the robot motor.

Step 2 Click **Jog** buttons to move the robot to a point.

Step 3 Double click **Point** to enter point page and click  to add a teaching point.

The teaching point information is displayed on the **point** page, as shown in Figure 3.2.



	No.	Alias	X	Y	Z	Rx	Ry	Rz	R	D	N	Cfg	Tool	User	Load
1	P1		-291.3...	-260.0...	847.2...	-82.23...	52.5999	-161.6...	1	1	-1	1	No.0	No.0	No.0
2	P2		-242.0...	-306.4...	847.2...	-82.23...	52.5999	-151.7...	1	1	-1	1	No.0	No.0	No.0

Figure 3.2 Teaching points list of SCARA robot

Table 3.1 Button description

Button	Description
	Add a point
	Delete a point
	Cover a point. Select a teaching point, after jogging the robot to a point, click the icon to cover the selected teaching point
	Run to a point, select a point, click the button to run the robot to this point
	Save teaching point
	Cancel

Button	Description
	Recover

- You can select a taught position and double-click the parameters on the line to modify the relevant information.

- Also, you can select a taught position and click to cover the current taught position.

Step 4 Add points by referring to Step 2 and Step 3.

Step 5 Click to save the teaching points.

3.3 Quick Start

This section gives examples of Blockly for Motion commands, I/O commands, Modbus commands and TCP commands, for user reference only.

3.3.1 Robot movement

Running the Motion commands can control the movement of robot in the joint coordinate system and the Cartesian coordinate system. The detailed description of the Motion commands, please see 2.1 Motion Commands. Figure 3.3 shows a programming program that includes Motion commands.

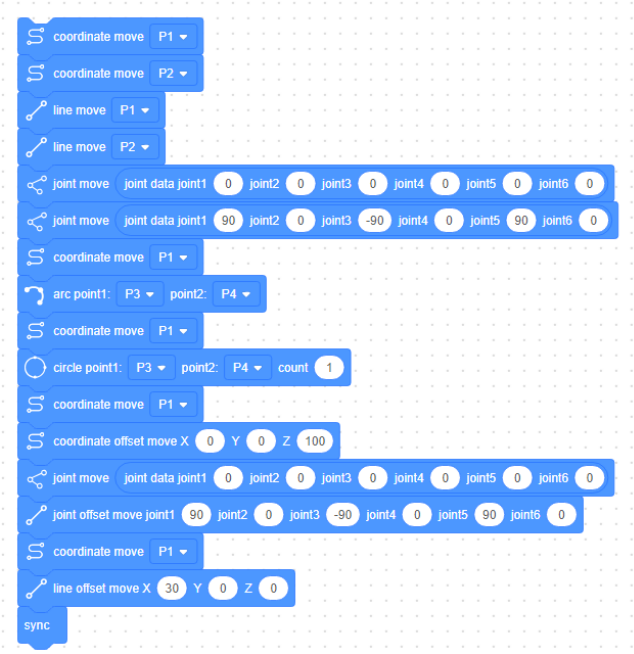


Figure 3.3 Robot movement

3.3.2 I/O Setting

Running I/O commands to set or get each I/O statue. The detailed description of the I/O commands, please see 2.2 I/O Commands. Figure 3.4 shows a programming program that includes I/O commands.

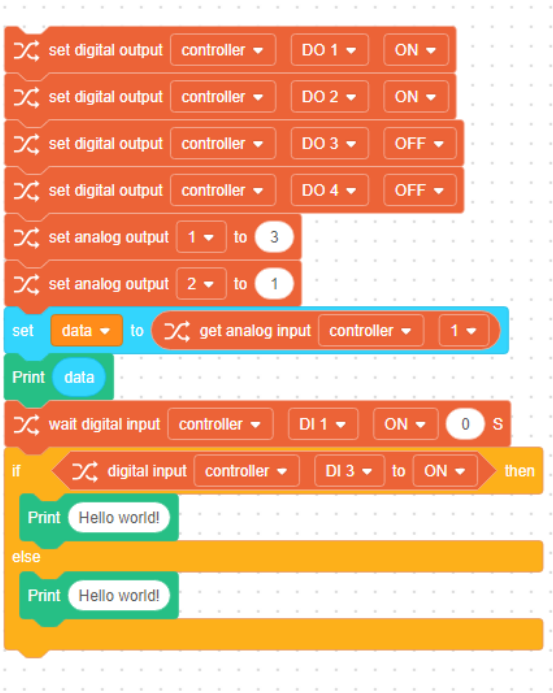


Figure 3.4 I/O Setting

3.3.3 Register setting and reading

By running the Modbus commands to set or read the value of each register address. The detailed description of the Modbus commands, please see 2.3 Modbus Commands. Figure 3.5 shows a programming program that includes Modbus commands.

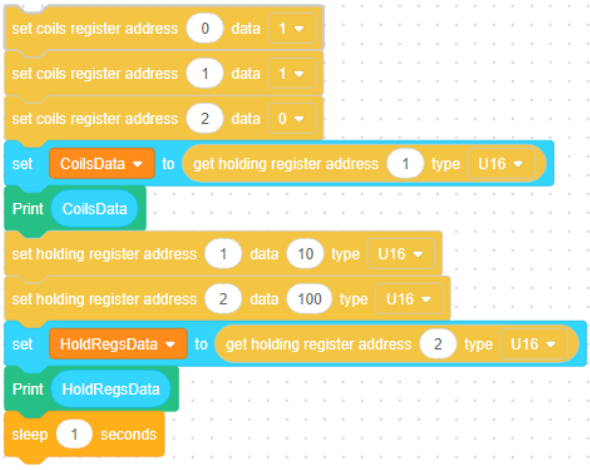
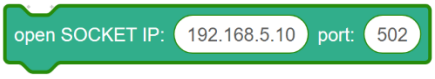


Figure 3.5 Register setting and reading

3.3.4 Create TCP Client

Run 

to establish communication with the TCP server,

the robot as the TCP client. Running TCP commands can send and read communication data, the detailed description of TCP commands, please see 2.4 TCP Commands. Figure 3.6 shows a programming program that includes TCP commands.

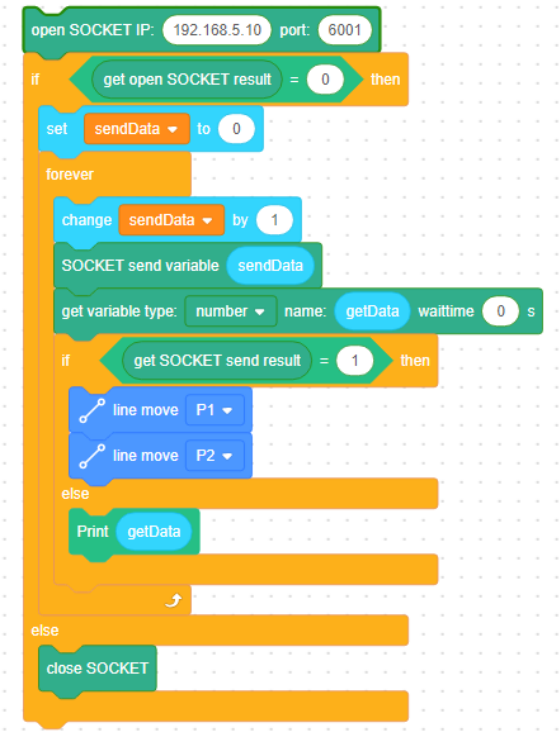
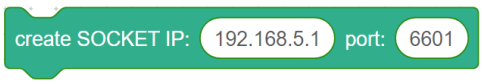


Figure 3.6 Create TCP Client

3.3.5 Create TCP Server

Run 

to set the robot as the server, waiting for the TCP client to connect. Running TCP instruction can send and read communication data, the detailed description of TCP commands, please see 2.4TCP Commands. Figure 3.7 shows a programming program that includes TCP commands.

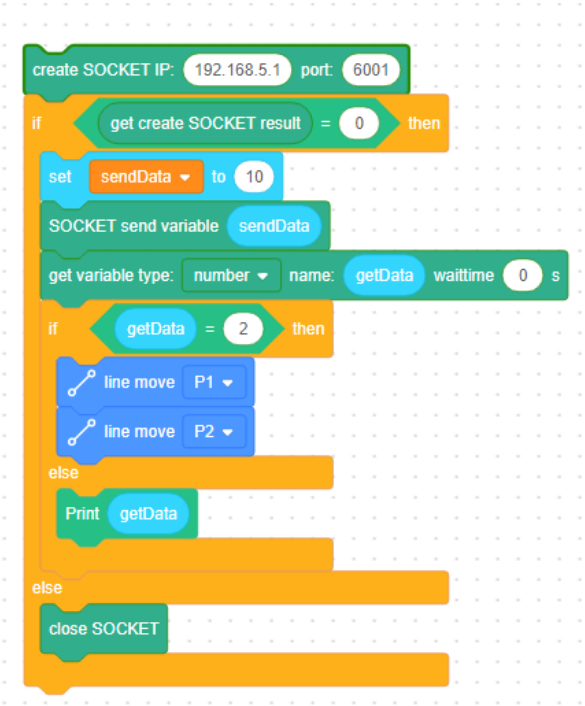


Figure 3.7 Create TCP Server

