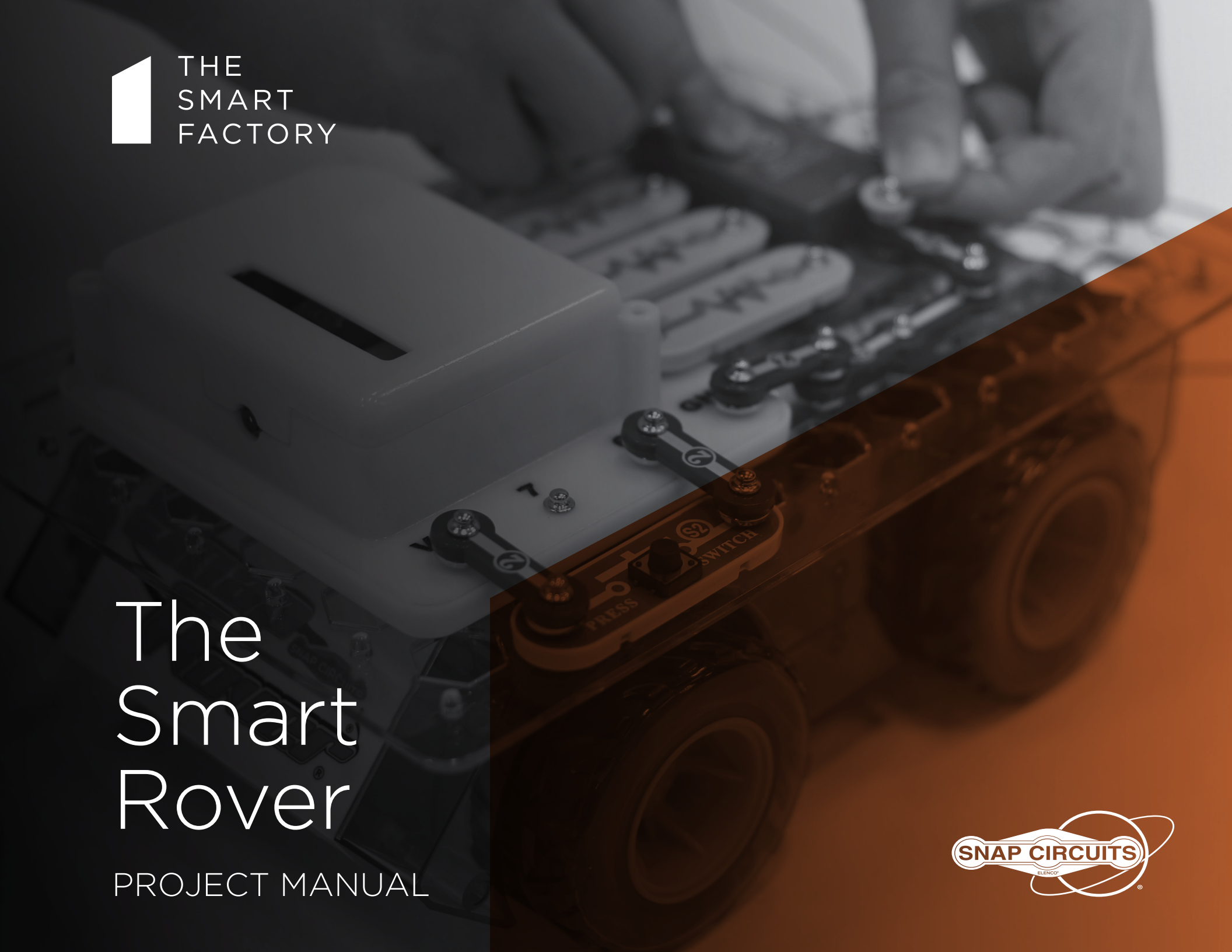


The Smart Rover

PROJECT MANUAL

CONFORMS TO ALL APPLICABLE U.S. GOVERNMENT REQUIREMENTS

WARNING: Always check your wiring before turning on a circuit. Never leave a circuit unattended while the batteries are installed. Never connect additional batteries or any other power sources to your circuits. Discard any cracked or broken parts.

ADULT SUPERVISION: Because children's abilities vary so much, even with age groups, adults should exercise discretion as to which experiments are suitable and safe (the instructions should enable supervising adults to establish the experiment's suitability for the child). Make sure your child reads and follows all of the relevant instructions and safety procedures and keeps them on hand for reference.

This product is intended for use by adults and children who have attained sufficient maturity to read and follow directions and warnings.

Never modify your parts, as doing so may disable important safety features in them and could put your child at risk of injury. The packaging has to be kept since it contains important information.

BATTERIES:

- Use only 1.5V AA type in the rover body.
- Insert batteries with correct polarity.
- Non-rechargeable batteries should not be recharged. Rechargable batteries should only be charged under adult supervision and should not be recharged while in the product.
- Remove batteries when they are used up.
- Do not mix alkaline, standard (carbon-zinc), or rechargeable (nickel–cadmium) batteries.
- Do not mix old and new batteries.
- Do not connect batteries in parallel.
- Do not short circuit the battery terminals.
- Never throw batteries in a fire or attempt to open their outer casing.
- Batteries are harmful if swallowed, so keep away from small children.

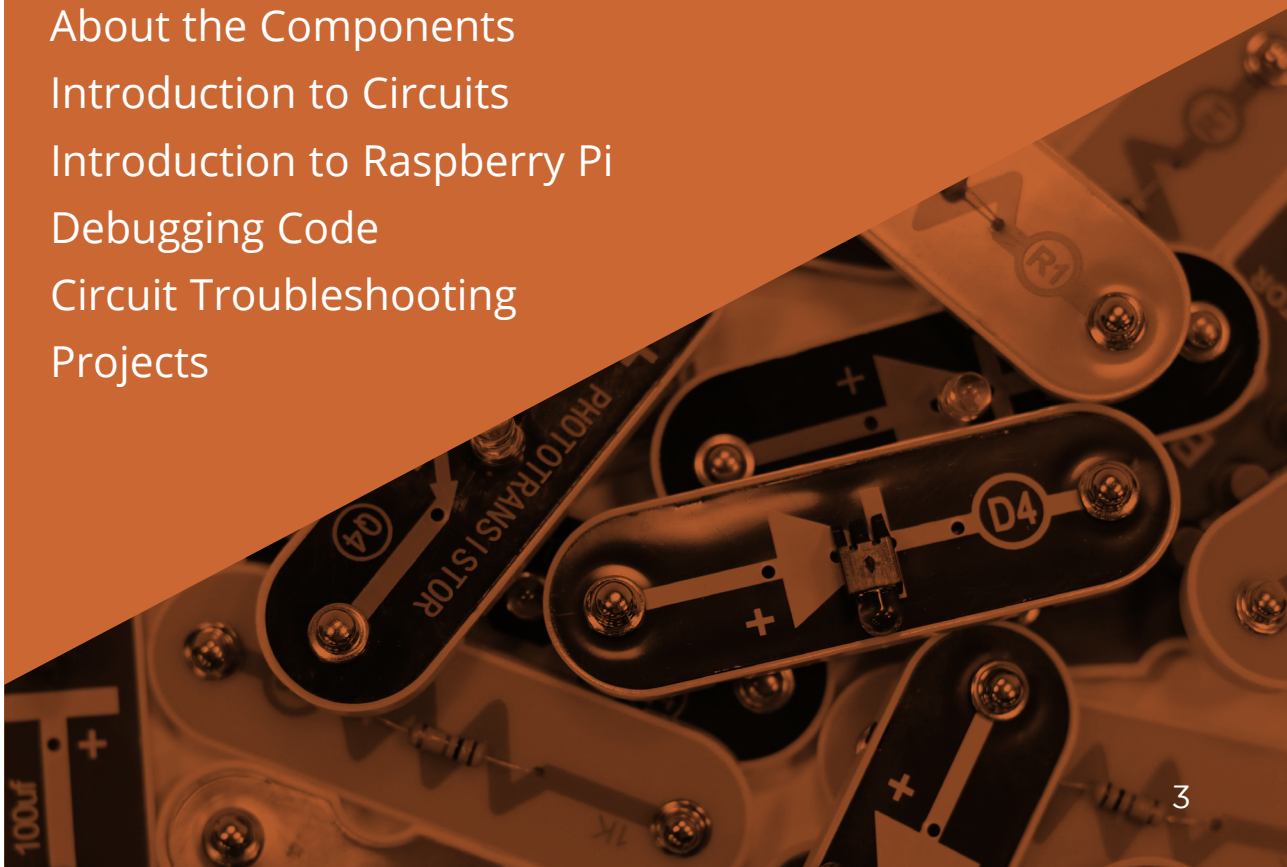
WARNING: the Smart Rover should ONLY be powered using AA batteries. The Smart Module should ONLY be powered using the power cord provided. Neither should ever be used with Snap Circuits® battery holders or other power sources!

WARNING: ELECTRIC TOY
The Smart Rover is rated for ages 10 and older, and is not recommended for children under 10 years of age.

WARNING: SHOCK HAZARD
The Smart Module should ONLY be powered using the power cord provided.

Warning to Snap Circuits® Owners
Do not use parts from other Snap Circuits® sets with this kit. The Smart Rover uses higher voltage, which could damage those parts.

Table of Contents



Parts List

Qty.	Part	Image	ID	Part #
1	Rover Body	NA	NA	6SCRB
1	Base Grid	NA	NA	6SCBG
2	1-Snap Wire		1	6SC01
6	2-Snap Wire		2	6SC02
2	3-Snap Wire		3	6SC03
1	4-Snap Wire		4	6SC04
1	5-Snap Wire		5	6SC05
1	6-Snap Wire		6	6SC06
1	7-Snap Wire		7	6SC07
1	0.02 microF Capacitor		C1	6SCC1
2	100 microF Capacitor		C4	6SCC4
1	White LED		D4	6SCD4
1	100 Ohm Resistor		R1	6SCR1
4	1K Ohm Resistor		R2	6SCR2
1	Slide Switch		S1	6SCS1
1	Motor Control		U8	6SCU8
1	Horn		W1	6SCW1
1	Smart Module		NA	NA

Qty.	Part	Image	ID	Part #
1	Selector		S8	6SCS8
1	Press Switch		S2	6SCS2
1	Phototransistor		Q4	6SCQ4
1	Jumper Wire (Orange)		NA	6SCJ3A
1	Jumper Wire (Yellow)		NA	6SCJ3B
1	Jumper Wire (Green)		NA	6SCJ3C
1	Jumper Wire (Blue)		NA	6SCJ3D
1	Jumper Wire (Gray)		NA	6SCJ3E
1	Jumper Wire (White)		NA	6SCJ3F

OPTIONAL COMPONENTS

Qty.	Part	Image	ID	Part #
1	Programmable Fan		M8	6SCM8
1	Speaker		SP	6SCSP
1	NPN Transistor		Q2	6SCQ2
1	Color Changing LED		D8	6SCD8
1	Slow Changing LED		D12	6SCD12
1	3D Snap	NA	NA	6SC3DSNAP

Each kit also includes a USB cable and wall power adapter. Optional components are dependent on the customizations requested from The Smart Factory @ Wichita.

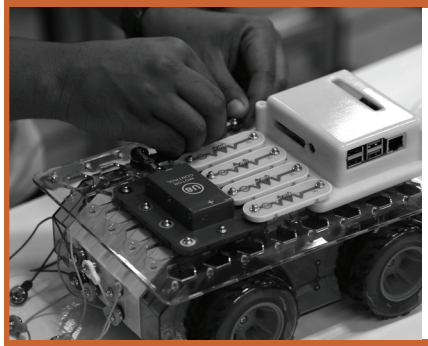
For issues with the Smart Module, please visit The Smart Factory @ Wichita website at **thesmartfactory.io**. For all other Snap Circuits® components, please contact Elenco® Electronics Customer Support via **elenco.com** or by emailing **support@elenco.com**.

About the Smart Rover

To foster the next generation of believers, The Smart Factory @ Wichita has partnered with Elenco® Electronics to produce a new STEM education product: the Smart Rover. The Smart Rover kit includes Elenco® Electronics' award-winning Snap Circuits® products and adds a new component, the Smart Module. Assembled in a custom-designed housing produced and assembled live at The Smart Factory @ Wichita, the module holds a Raspberry Pi microcomputer and camera.

The projects within the Smart Rover kit begin with teaching the basics of computer programming on the Raspberry Pi and explore all the way up to enabling the rover to self-drive and "see" using the embedded camera. But innovation does not end with the last page of this booklet. The Smart Rover kit seeks to empower all students, across all ages, with the enduring skills needed to design, code, and build for the future.

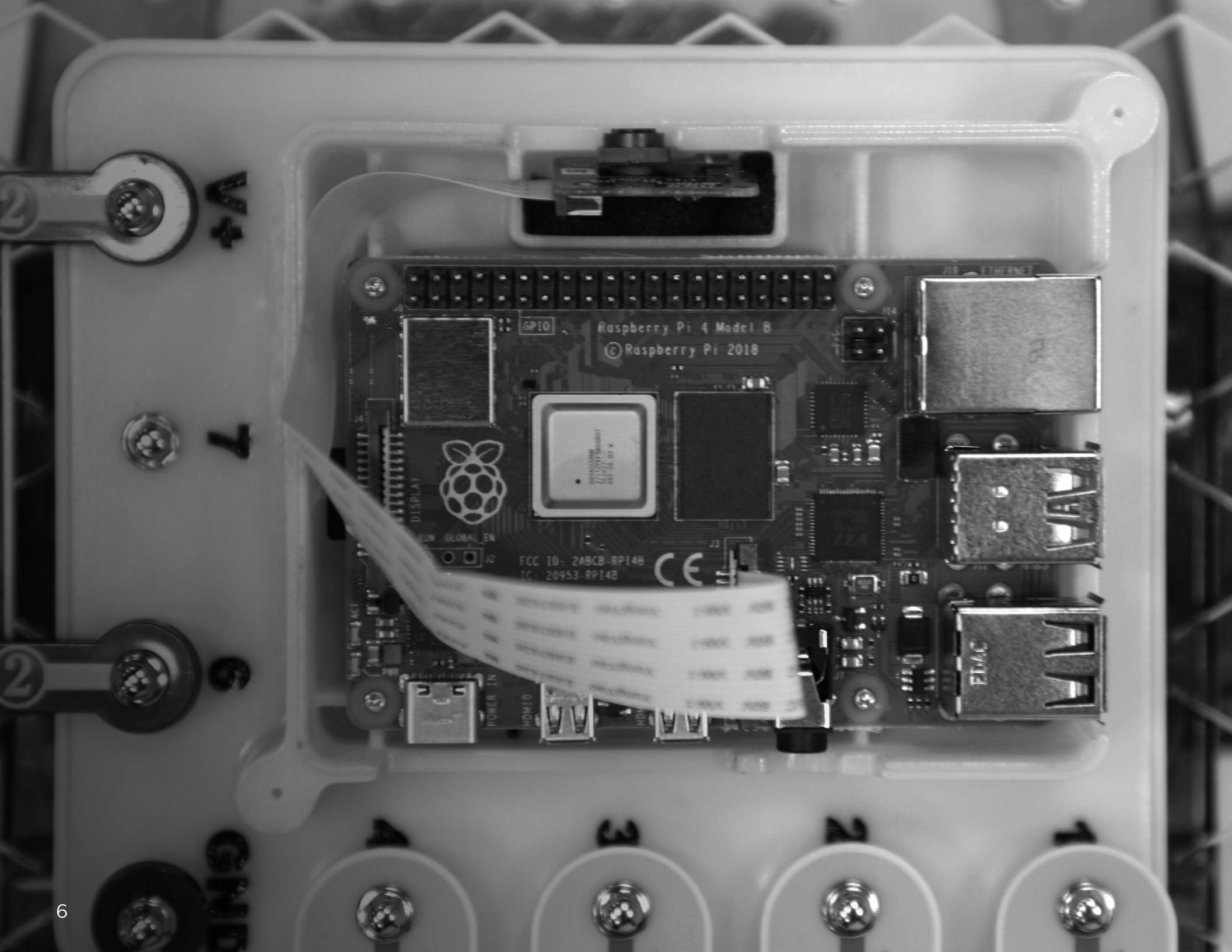
About Elenco® Electronics



Family owned and operated since 1972, Elenco® is located in the Chicago suburb of Wheeling, Illinois. Elenco® Electronics' pride is the award-winning Snap Circuits® brand. For nearly 20 years, Snap Circuits® has been the go-to toy for teaching electronics, engineering and more. Snap Circuits® has been endorsed by K–12 educators globally and used in schools, libraries, museums, after-school and homeschool programs, STEM and Maker programs, and at home. Many of today's rising leaders in science and technology learned the basics of engineering by creating and inventing with Snap Circuits® as a child.

About The Smart Factory @ Wichita

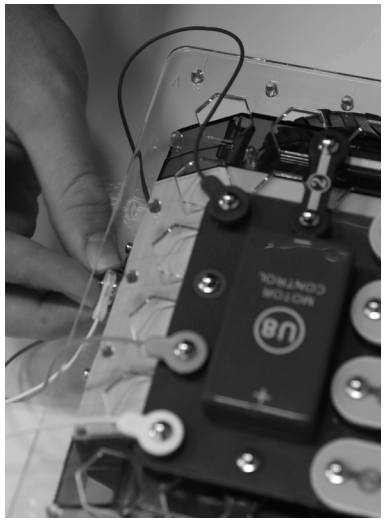
The Smart Factory @ Wichita is a live factory and experiential learning center based in Wichita, Kansas, designed to help organizations envision—and continually reimagine—the future of manufacturing. Convened by Deloitte with an ecosystem of world-renowned collaborators and a shared goal, the center showcases how today's innovative tools and technologies can solve the operations challenges of tomorrow. The Smart Factory @ Wichita is a one-of-a-kind, immersive, and sustainable Industry 4.0 center that spurs invention and growth—and a belief that anything is possible.


GPIO
Raspberry Pi 4 Model B
©Raspberry Pi 2018
DISPLAY
GLOBAL_EN
FCC ID: 2ABCB-RPI4B
IC: 20953-RPI4B
POWER IN
HDMI0
ETHERNET
EDAC

Introduction

Introduction to Electricity

Electricity is the movement of subatomic charged particles (called **electrons**) through a material due to electrical pressure across the material, such as from a battery. We can think of electricity similarly to how we think of the flow of water, and we can think of circuits like a plumbing system of pipes, valves, and faucets controlling the flow.



Power sources, such as batteries, push electricity through a circuit, like a pump pushes water through pipes. Wires carry electricity, like pipes carry water. Switches and transistors control the flow of electricity like valves and faucets control water. Resistors limit the flow of electricity, similar to how blockages can slow down the flow of water.

The electrical pressure exerted by a battery or other power source is called **voltage** and is measured in volts (V). Notice the "+" and "–" signs on a battery; these indicate which direction the battery will "pump" the electricity. This is similar to the pressure that pushes water through a pipe or hose.

The **electric current** is a measure of how fast electricity is flowing in a wire, just as the water current describes how fast water is flowing in a pipe. It is expressed in amperes.

The **power** of electricity is a measure of how fast energy is moving through a wire. Power is voltage multiplied by current and is expressed in watts.

The **resistance** of a component or circuit represents how much it resists the electrical pressure (voltage) and limits the flow of electric current. Voltage is current multiplied by resistance. When the resistance increases, less current flows, and vice versa. Resistance is measured in ohms.

Nearly all of the electricity used in our world is produced by enormous generators driven by steam or water pressure. Wires are used to efficiently transport this energy to homes and businesses where it is used. Motors can then convert the electricity into mechanical form to drive machinery and appliances. The most important aspect of electricity in our society is that it allows energy to be easily transported over distances, large and small. Power lines transport energy across miles, while tiny circuit boards transport energy to power our phones and computers.

Circuits are the basis of all electronics. There are two ways of arranging parts in a circuit: in series or in parallel. Placing components in series increases the resistance; highest value dominates. Placing components in parallel decreases the resistance; lowest value dominates. The parts within these series and parallel sub-circuits may be arranged in different ways without changing what the circuit does. Large circuits are made of combinations of smaller series and parallel circuits.

First, we will introduce the circuit components included in this kit and how to build circuits, then we will introduce the basics of programming, and finally, we will combine programming and circuitry together.

About the Components

The Smart Rover uses building blocks with snaps to build the different circuits for each project. Each component has a different function and is colored so you can identify it. Each component easily snaps together so you can build the circuit.

ABOUT YOUR COMPONENTS

Smart Module: The Smart Module is a Raspberry Pi and camera assembled in a housing. A Raspberry Pi is a full-functioning miniature computer.

The Smart Rover should only be powered with the AA batteries in the bottom. The Raspberry Pi should only be powered with the USB cable provided with the kit. Neither should ever be used with power sources from other Elenco® sets or external components.

Base Grid: The base grid functions like the printed circuit boards found in most electronic products. It is a platform for mounting parts and wires, though in most products the wires are "printed" on the board.

Snap Wires: The blue snap wires are used to connect components; they transport electricity and do not affect circuit performance. They come in different lengths to enable orderly arrangement of connections.

Jumper Wires: The jumper wires are used to make flexible connections where using snap wires would be difficult. Though they have different colors, they are interchangeable, but the color-coding helps when connecting them to the rover rear snaps.

Resistors: Resistors are used to control or limit the current in a circuit by "resisting" the electricity. The higher the resistance, measured in ohms, the more the resistor limits the current.

Slide Switch & Press Switch: These switches connect or disconnect the wires in the circuit, allowing you to turn it on or off, like water with a faucet. When on, they have no effect on the circuit.

LED: A light-emitting diode (LED) acts as a special one-way light bulb. In the "forward" direction (indicated by the "arrow" in the symbol), electricity flows if the voltage exceeds a turn-on threshold; brightness then increases. A high current will burn out an LED, so they have internal resistors to protect them. LEDs block electricity in the "reverse" direction. The color LEDs have a pre-programmed color sequence that is not programmable via code. In projects where the White LED (D4) is used, the Color Changing LED (D8) and the Slow Changing LED (D12) can be used instead.

Horn & Speaker: These convert energy to sound by making mechanical vibrations.

Phototransistor: The phototransistor uses light to control electric current; more light creates more current, and vice versa with less light.

Capacitors: Capacitors store electric charge. They are used to direct or block currents. The higher the capacitance, measured in farads, means more energy can be stored.

Programmable Fan: The fan is a motor with an LED circuit, which converts electricity into mechanical motion. The LEDs in the fan blade flash in a pattern based on the programmed phrase and are synchronized with the motor speed. The flashes are precisely timed and very brief, giving the illusion of words floating in space. For more details on using the Programmable Fan with the Selector, please see Project 15 in the Snap Circuits® Arcade (Model SCA200) manual, available online at **elenco.com/manuals**.

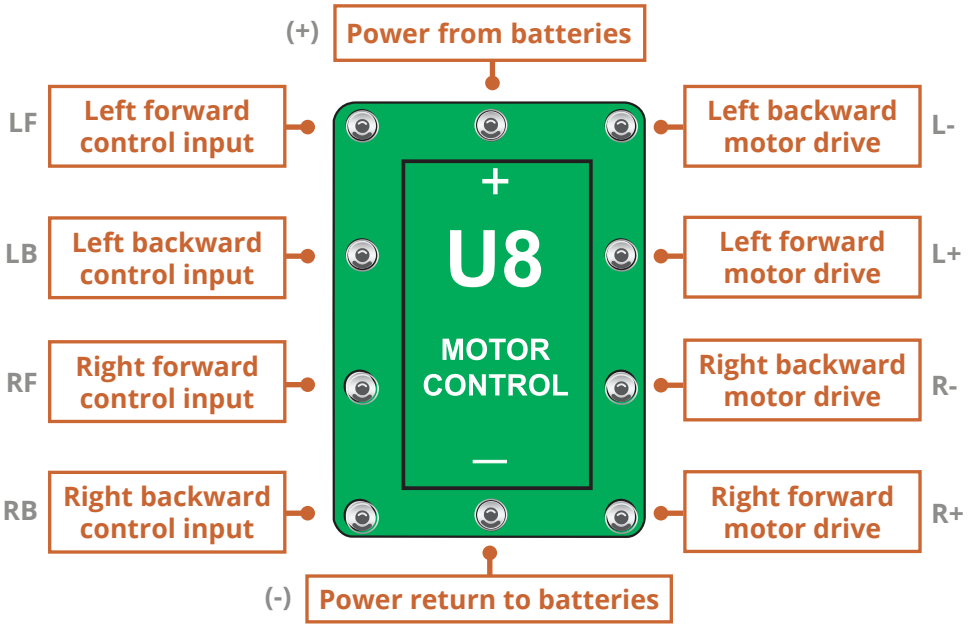
About the Components

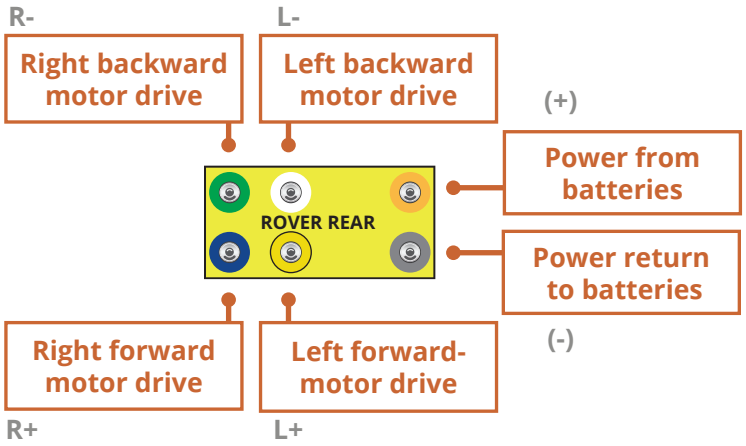
Motor Control: This module contains 16 transistors and resistors that are needed to control the motors of the rover.

NPN Transistor: NPN transistors are components that use a small electric current to control a large current and are used in current switching, amplifying, and buffering applications.

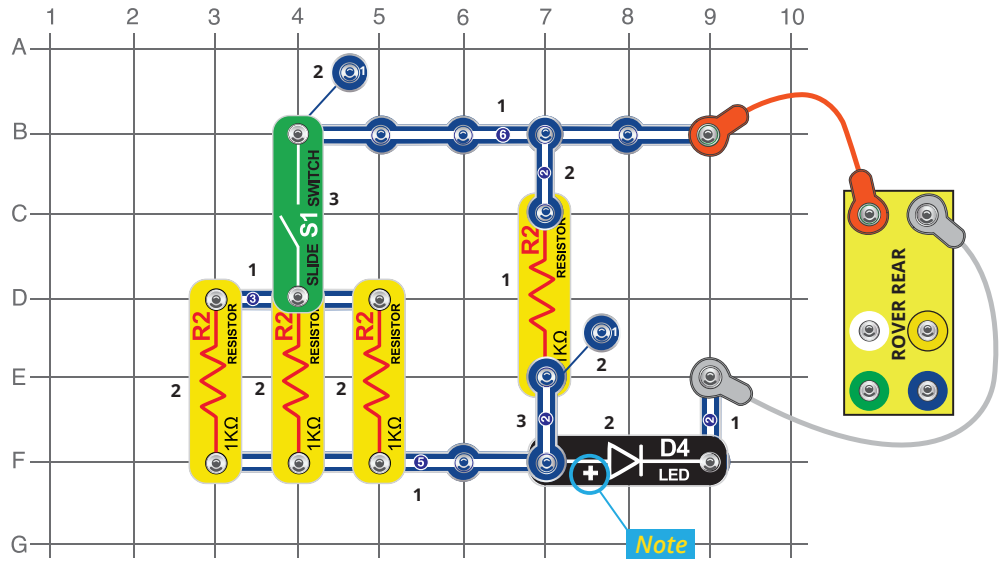
Selector: The selector is a more complex switch that allows for three inputs, rather than just one.

Rover Body: The rover body has both motors and a battery. We can build circuits on the rover body that are not connected to the motors, if we just want to use the batteries as a voltage supply. The **motors** in the rover body convert electrical energy into mechanical motion. Inside the motor is a coil of wire and metal plates. When a large electric current flows through the wire loops, it will turn ordinary metal into a magnet. When the current flows through the coil, it will magnetize the plates, which will repel from the magnet in the motor shell, causing the shaft to spin. The small gear on the shaft spins with it, which drives the larger gear system, moving the rover.

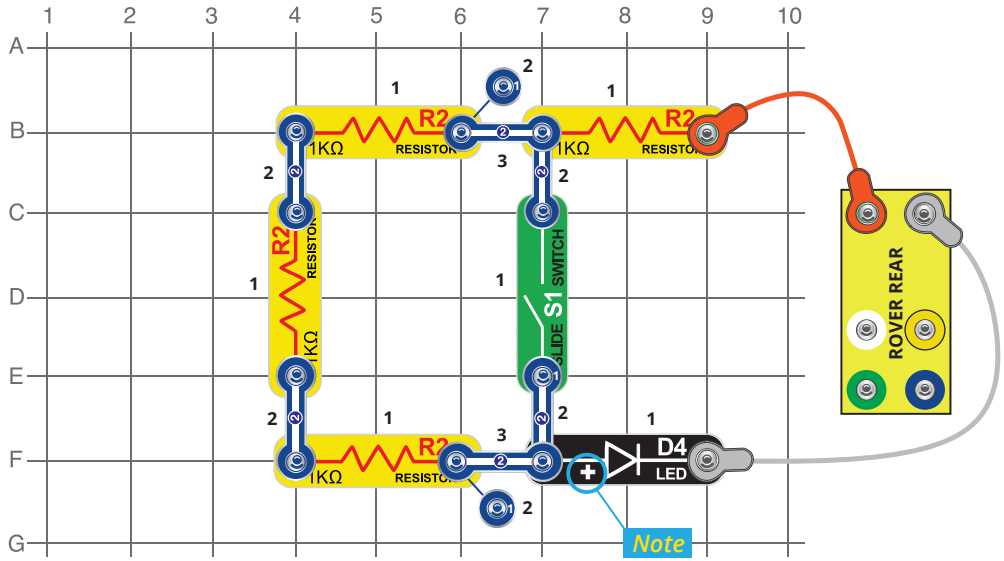




Introduction to Circuits



Numbers next to components indicate the order and level of placement.



Numbers next to components indicate the order and level of placement.

Resistors

OBJECTIVE: TO LEARN THE BASICS OF RESISTANCE AND BUILDING CIRCUITS

RESISTORS IN PARALLEL

Build the circuit and connect the jumper wires as shown. The LED will be on, but the resistor is limiting the electricity through it.

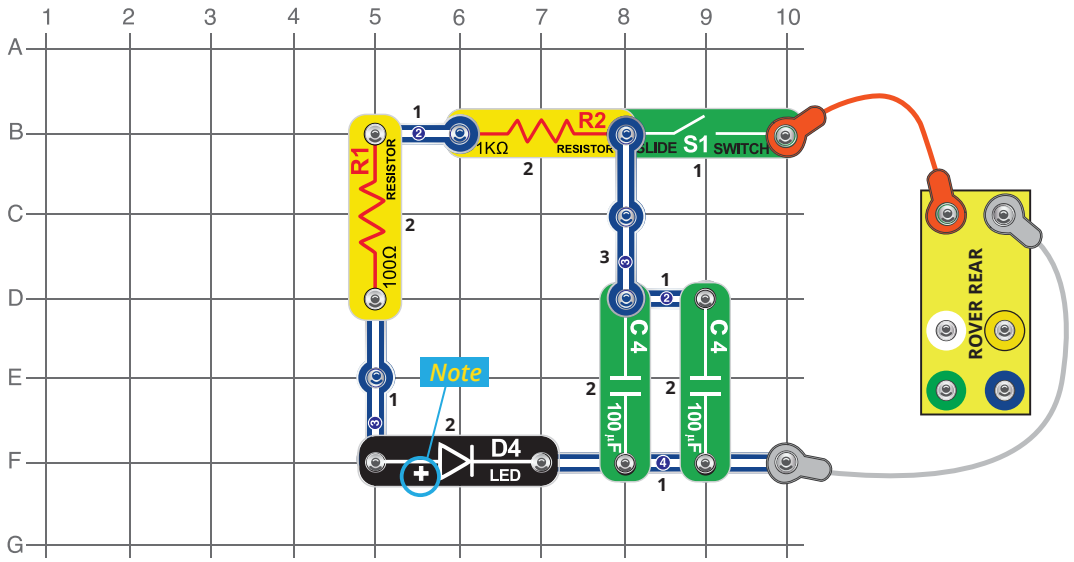
Turn on the switch to place three other resistors in parallel (notice how they make parallel lines) with the first one. This increases the flow of electricity to the LED and makes it brighter. Placing other resistors in parallel reduces the total resistance.

RESISTORS IN SERIES

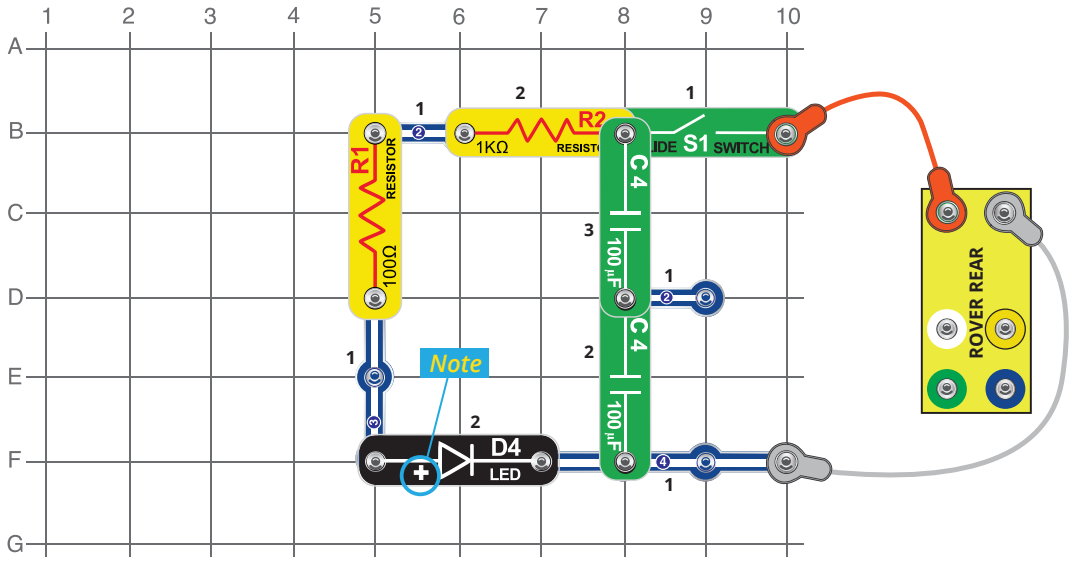
Build the circuit and connect the jumper wires as shown. The LED will be on, but the four resistors are limiting the electricity through it.

Turn on the switch to bypass three resistors that are in series with the first one (notice how they are lined up, one after the other). This increases the flow of electricity to the LED and makes it brighter. Placing other resistors in series increases the total resistance.

Introduction to Circuits



Numbers next to components indicate the order and level of placement.



Numbers next to components indicate the order and level of placement.

Capacitors

OBJECTIVE: TO LEARN THE BASICS OF CAPACITANCE AND BUILDING CIRCUITS

CAPACITORS IN PARALLEL

Build the circuit and connect the jumper wires as shown. Turn the switch on and the LED will turn on. Turn the switch off and the LED will go off slowly.

Electricity stored in the capacitors keeps the LED on after the batteries have been disconnected. If you remove one capacitor, then the LED will turn off faster since less electricity will be stored. If you remove both capacitors, the LED will turn off immediately.

CAPACITORS IN SERIES

Build the circuit and connect the jumper wires as shown. This is the same circuit as above, but with the capacitors connected differently. Turn the switch on and off, and watch how quickly the LED turns off.

The LED doesn't stay on as long with this circuit because capacitors connected in series store less electricity than when in parallel. This is the opposite of how we combined resistors. Capacitors connected in series cannot store as much electricty, but can be used with higher voltages.

Avoiding Short Circuits

After building the circuits in this manual, you may wish to build your own. Use the projects in this manual as a guide, as many important design concepts are taught throughout. Every circuit will include a power source, a resistance (which may be a resistor or another integrated part), and wiring paths between them and back.

Electricity flowing in a path in a circuit will always take the path of least resistance, and a circuit with no or low resistance is known as a short circuit. You must be careful not to create these short circuits, as this will damage your components and quickly drain your batteries.

BUILDING CIRCUITS

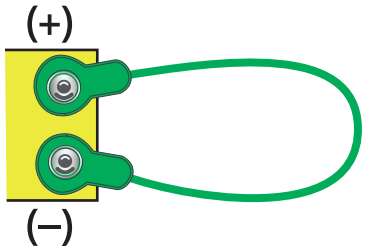
Remember to always...

- Use eye protection when experimenting on your own
- Include at least one component that will limit the current in the circuit, such as a resistor or a component with internal resistance, such as an LED
- Use switches and capacitors in conjunction with other components that will limit the current through them; failure to do this will create a short circuit
- Disconnect your power source immediately if something appears to get hot
- Check your wiring before turning on a circuit, and check the path the electricity will flow through

Never do the following...

- Connect the rover to an outlet in your home
- Leave a circuit unattended when turned on
- Connect other power sources to your rover or Smart Module

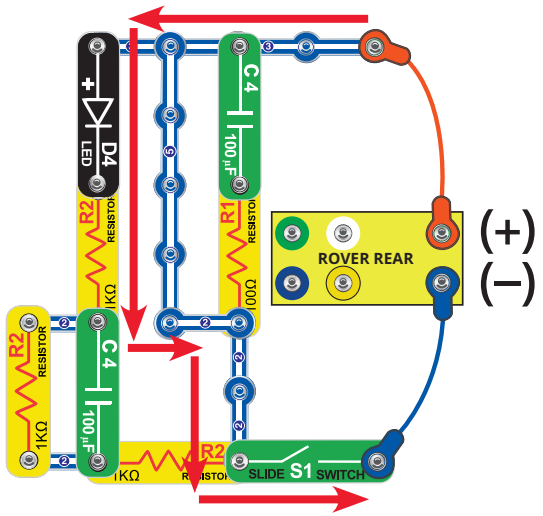
DO NOT BUILD: SHORT CIRCUITS



NEVER BUILD

Placing a jumper wire directly across the battery snaps is a short circuit.

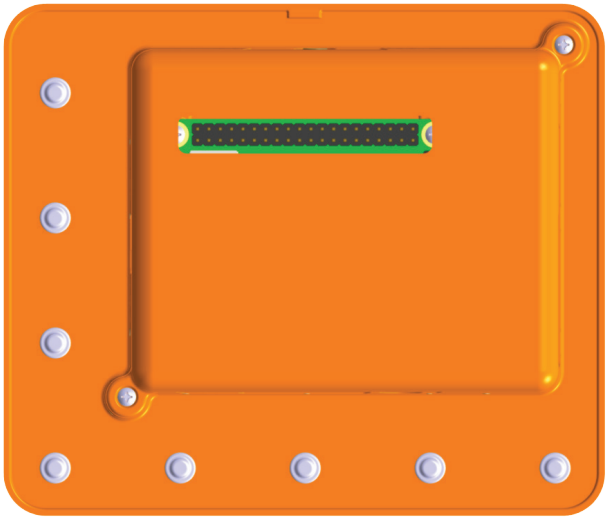
When the switch is turned on, this large circuit has a short circuit path (as shown by the arrows). The short circuit prevents any other portions of the circuit from ever working.

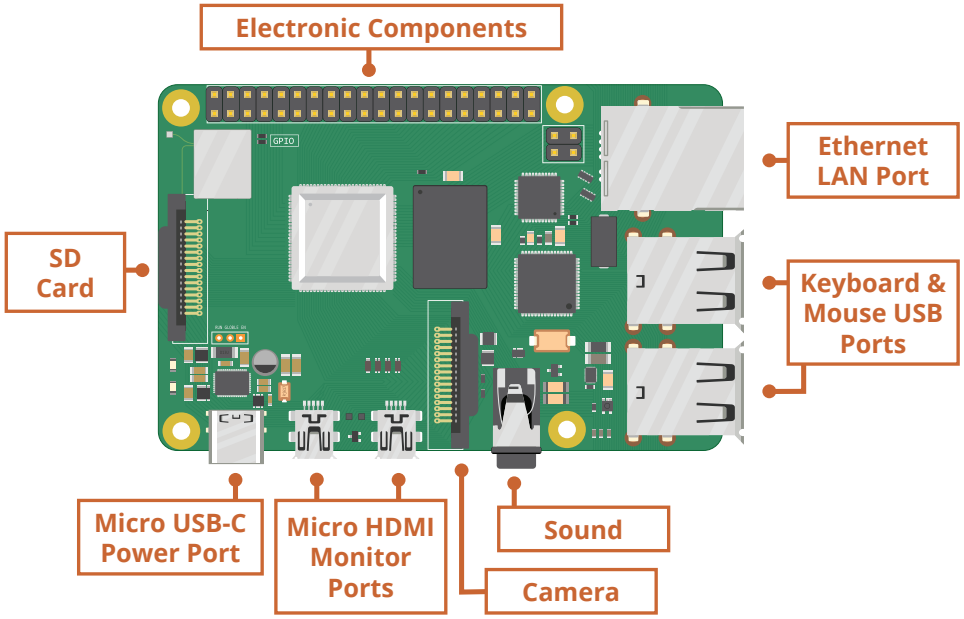


NEVER BUILD

Introduction to Raspberry Pi

Inside your **Smart Module** is a **Raspberry Pi**. The Raspberry Pi is a fully functioning microcomputer that plugs into a monitor and uses a standard **keyboard** and **mouse**. Like your computer, it runs on an operating system (OS) called the Raspberry Pi OS. It can do everything your desktop computer can and can also interact with a variety of different products—including the Smart Rover. By programming the Pi, we can enable the Rover and its components to do a variety of things.





The Smart Module houses a Raspberry Pi 4. This Pi comes pre-configured with an SD card installed with the Raspberry Pi OS. **To use the Raspberry Pi, you will need an external keyboard and monitor with USB connectors.**

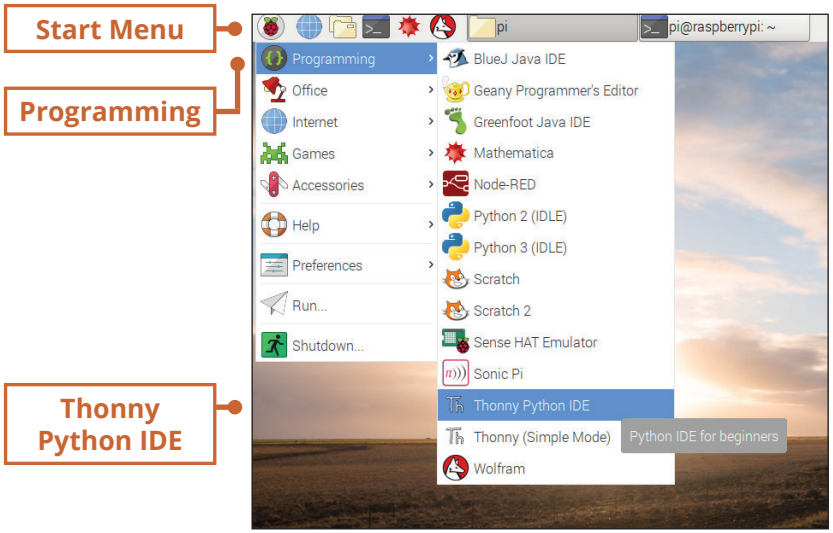
Connecting your keyboard, mouse, and monitor: To get started, you will need to connect your keyboard and mouse USB connectors to the two USB ports on the Pi (either port is fine). You will also need to connect the power cord to the Pi and to an outlet. Your monitor will be connected to the Pi through the left-hand HDMI port. An optional second screen can be connected to the right-hand port.

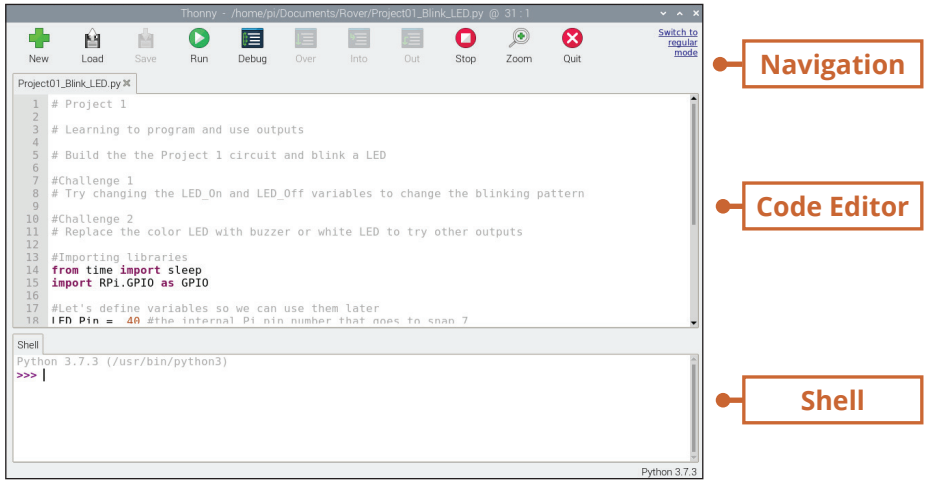
The camera will be used in later projects, while the network, sound, and electrical components ports will not be used in this manual. The Pi does not have a power switch; once the power port is connected, it will turn on. After a few moments, the Raspberry Pi OS desktop will appear.

When you start the Pi for the first time, the Welcome to Raspberry Pi application may pop up and guide you through setting up your country, language, time zone, password, and wireless network. There may also be software updates to be installed.

Programming the Raspberry Pi

Writing and running programs allows us to "speak" to computers and tell them the tasks we want them to do. Different programming languages are used for different types of tasks. Like spoken languages, programming languages each have their own types of “grammar” and punctuation, but learning one helps you understand the underlying logic of all of them. For the Smart Rover projects, we will be programming in **Python** using the **Thonny Python IDE** application.





Your Raspberry Pi comes with a variety of programming applications, which you can access under **Programming** in the menu, and click on **Thonny Python IDE.** IDE stands for Integrated Development Environment. IDE applications help us organize and manage our programs.

When you load up Thonny, across the top bar there are icons to create a **New** program, as well as **Save, Load, Run,** and **Stop** icons. Each project will be its own program, saved in its own file. In the middle of the page is the code editor, where you will write your code. At the bottom is the **Shell**, where you will see outputs from your code.

Once Thonny is open, click **Load** in the top left to select a program from the Projects folder. Once you have loaded a program and are ready to run it, click the green **Run** arrow button at the top. You’ll know your code was successfully loaded when it shows the **>>> %Run** command in the bottom in the **Shell** window. If you need to make changes to your code and want to run it again or run a different script, press the red **Stop** button at top. It’s important to stop running your code before you attempt to run something else so the Pi can reset itself.

Programming Basics

Learning to write code is a little like learning a new language, as each programming language has its own terminology and syntax. Python, the program we will be using, was designed for readability. The projects will walk you through how to learn writing Python step-by-step.

Loading & Saving Programs: For all programming languages, it is good to continously save your programs. When loading each project file in this manual, we recommend you save a new file immediately, in case you want to reference the original instructions or you accidentally delete a line of code.

Commands: All programming languages use commands. A command is a unique word to perform a specfic operation. For example, "print" is a command used to display text on the screen. Try writing **print "Hello World!"** in the Shell of Thonny IDE. Python uses new lines to complete a command. Other languages may require a command to end with a semicolon or parentheses to run.

Blocks: A Python program is constructed from blocks of code. A block is a piece of program text that is executed as a unit. A single command line is a block, as are multiple lines grouped under a function, loop, or conditional statement, all of which will be introduced throughout the projects.

Indents: Indentation refers to the spaces at the beginning of a code line, usually created using the tab key. While in other programming languages, indentation is used for readability only, the indentation in Python is very important. Python relies on indentation to indicate where a block of code starts and ends. You have to use the same number of spaces in the same block of code, otherwise Python will give an error.

Comments: A hashtag symbol (#) is used to create a comment in the program. Comments have no effect on your code but are helpful to explain programs or provide instructions. You can also 'comment out' parts of code if you don't want that part of the program to run but would like to have it for reference. For multi-line comments, you can use """ before and after your comment, to comment out multiple lines of code without starting each line with a #.

Keywords: Keywords are some predefined and specially reserved words that have specific meanings and purposes and can only be used for those specific purposes. Keywords are used to define the syntax of the coding, and will appear in a different color than the rest of the code text to signal their usage.

Examples of keywords include: and, if, for, while, return, import, else, and def. All the keywords in Python are written in lower case except True and False and are ready to be used without importing a library.

Punctuation: Parentheses, brackets, and braces are all utilized in Python but have different use cases and syntax to be aware of when writing your code. Similar to writing a normal sentence, proper punctuation matters in order to express what you mean. Parentheses, like (and), are most commonly used when passing arguments into functions, both custom and predefined. Square brackets, like [and], are used to define lists, arrays, and strings and can also index elements of those storage types to get a specific value at a certain location. Curly braces, like { and }, can separate a block of code and be used with classes but are not necessary for most Python applications.

Debugging Code

If your code isn't working as expected on the first try, don't worry! This is completely normal and to be expected whenever programming complex logic, which is why we have debugging tools to help us identify and fix errors. Next to the green **Run** button in Thonny IDE is **Debug Current Script**, which allows the code to run more slowly and present additional information about its performance. In debugging mode, there's a lot we can do to get back on track.

DEBUG CURRENT SCRIPT

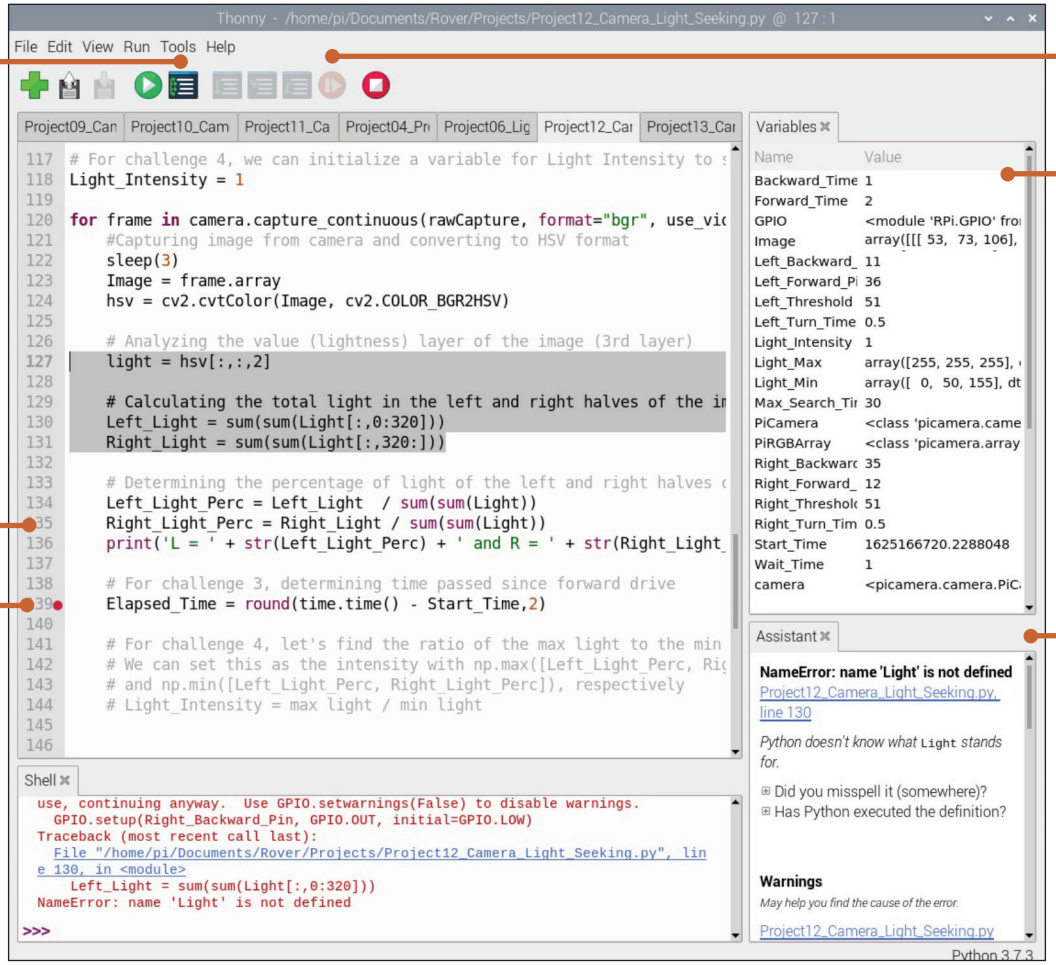
Runs your current code in debugging mode.

PRINT STATEMENT

To see how the code is working while running, use print statements for variables of interest. You can print strings alongside the variables for extra information.

BREAKPOINT

Click to the right of the number of a line of code to add a breakpoint, which appears as a small red circle. Now, when debugging, the code will stop here and allow you to view all variables.



RESUME

After hitting a breakpoint, press Resume to continue running the code.

VARIABLES

To see how your code is performing, looking at the variables is helpful. At a breakpoint, see if they are taking on the expected values and shapes/types. In the top menu bar, click on View, then select Variables.

ASSISTANT

Thonny has a built-in coding assistant that can help spot oddities within the code. Here it's clarifying the error the "Light" is not defined because it's "light" in line 127. It can also show warnings for stylistic or functional issues and always gives a line number. In the top menu bar, click on View, then select Assistant.

Circuit Troubleshooting

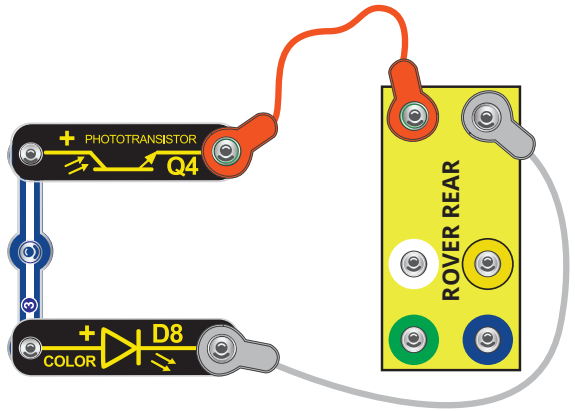
For the Rover Body, Jumper Wires, Slide Switch (S1), Snap Wires, Horn (W1), White LED (D4), Resistors (R1 and R2), Motor Control (U8) module, and Capacitors (C1 and C4): Please see "Advanced Troubleshooting" in the R/C Snap Rover® (Model 753131) manual, available online at **elenco.com/manuals**. All the necessary Snap Circuits® components to conduct troubleshooting via this manual are included in your Smart Rover kit.

For the Press Switch (S2): Please use the troubleshooting instructions for the Slide Switch (S1) above.

For the Color Changing LED (D8) and Slow Changing LED (D12): Please use the troubleshooting instructions for the White LED (D4) above.

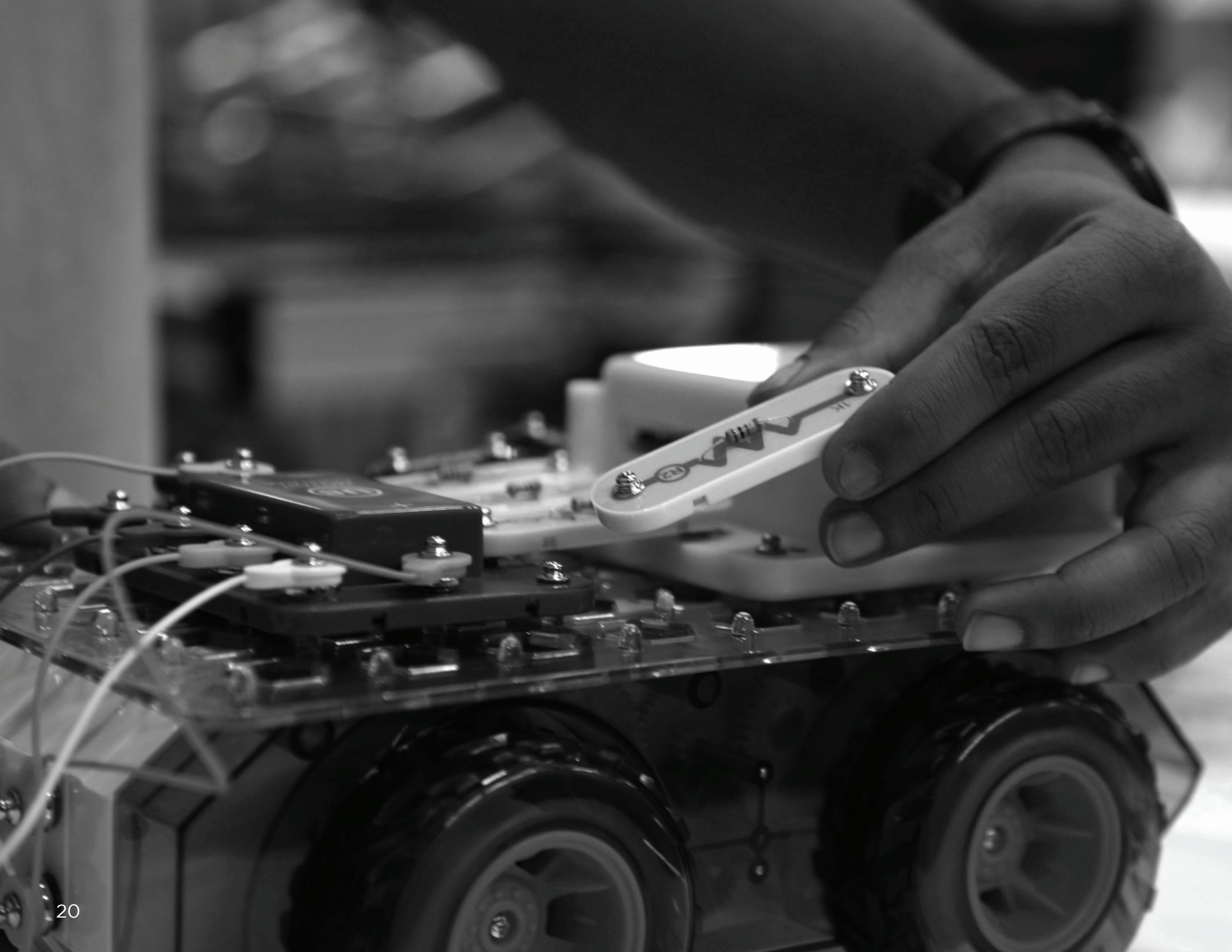
For the Programmable Fan (M8), NPN Transistor (Q2), and Selector (S8): Please see "Advanced Troubleshooting" in the Snap Circuits® Arcade (Model SCA200) manual, available online at **elenco.com/manuals**. All the necessary Snap Circuits® components to conduct troubleshooting via this manual are included in your Smart Rover kit. Instead of connecting the circuit to the Battery Holder (B3), you will connect it to the rover rear battery snaps, as indicated on page 11 of this Smart Rover manual. For the Q2 and S8 tests, use any of your D4, D8, or D12 LEDs in place of the LEDs shown in the SCA200 tests.

For the Phototransistor (Q4) and Speaker (S2): For the Phototransistor (Q4), use the mini circuit shown here, varying the amount of light shining on Q4 should change the brightness of D8. For the Speaker (SP), use the same mini circuit but replace Q4 with SP. You should hear sound from the speaker as D8 color changes, though the sound will not be very loud. If there is no sound, then the speaker is broken.



Smart Module: If you believe you have issues with your Smart Module, including the Raspberry Pi or the camera within the component, please visit The Smart Factory @ Wichita website at **thesmartfactory.io**.

For all other Snap Circuits® components, please contact Elenco® Electronics Customer Support via **elenco.com** or by emailing **support@elenco.com**.



The Projects

Project Progression

The introductory projects will teach the basics of programming, as well as circuitry, using both the Raspberry Pi within the Smart Module and various components, including the LED light, the horn, the phototransistor, and the selector. Next, you will learn about motor controls and how to program the rover to drive. Finally, we will use the camera within the Smart Module to detect color and light, then incorporate those inputs into driving. **All the projects have a starting program file pre-loaded to the Raspberry Pi. A code reference guide for all projects is available on The Smart Factory @ Wichita website.**

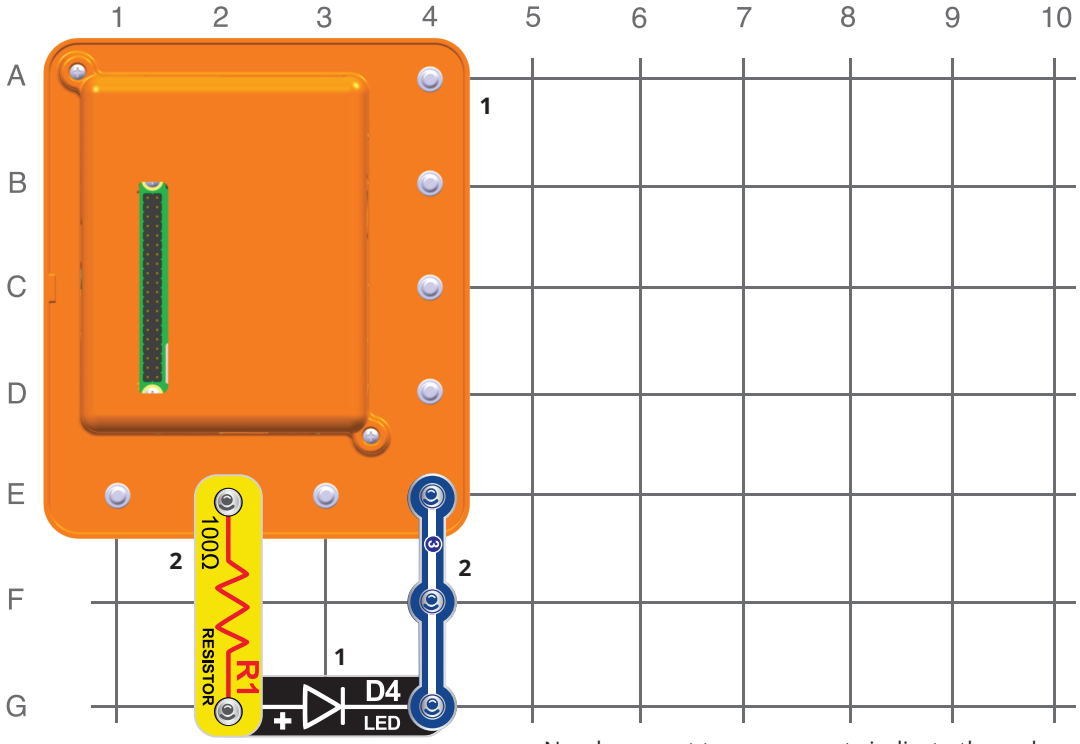
Please note that not all possible kit components are included in the projects. These include: the Programmable Fan, the Speaker, and the NPN Transistor. However, using the concepts introduced in the projects will allow you to incorporate all possible components.

Project	Title	Components	Objective
1	**Blinking LED**	Smart Module, LED	To learn programming basics and make the LED light blink
2	**Light-Activated LED**	Smart Module, Press Switch, LED	To control the LED light with a push button
3	**Selector Output**	Smart Module, Selector, LED, Horn	To program the selector to turn on the LED light in coded patterns
4	**Controlled Driving**	Smart Module, Motor Control	To program the rover to drive in a coded path
5	**Timed Driving**	Smart Module, Press Switch, Motor Control	To program the rover to drive in a coded path for timed button push
6	**Light-Activated Driving**	Smart Module, Phototransistor, Motor Control	To program the rover to detect light before driving forward
7	**Controlled Driving**	Smart Module, Selector, Motor Control	To program the rover to drive according to selected inputs
8	**Introduction to Camera**	Smart Module	To learn about the camera and different image settings
9	**Light Detection**	Smart Module, LED	To program the camera to detect light
10	**Color Detection**	Smart Module, Press Switch, Horn, LED	To program the rover to distinguish between various colors
11	**Color-Detection Driving**	Smart Module, Press Switch, Motor Control	To program the camera to detect sign colors and drive accordingly
12	**Light-Seeking Driving**	Smart Module, Press Switch, Motor Control	To program the rover to drive toward light it detects

Project #1

Blinking LED

OBJECTIVE: TO LEARN PROGRAMMING BASICS AND MAKE THE LED LIGHT BLINK



Numbers next to components indicate the order and level of placement.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, LED (D4, D8, or D12).

Once the circuit is created, open up Thonny and load Project01.py into the code editor. The pre-loaded programs are ready to run, but you will learn by reviewing the language and completing the challenges.

In this project, you will learn about **setting up your program, creating variables, and creating loops**.

Once you have completed the program, upload it the to the rover to blink the LED.

KEY CONCEPTS

Code Comments: Using a hashtag symbol (#) in front of a line in your program will turn the line into a "comment" instead of a line of program. Comments are essential to keep track of what different parts of the programs are doing.

Importing Libraries: Libraries are helpful modules with prewritten code blocks that we can use to save time and effort in controlling the Pi and beyond.

Defining Variables: Variables are words or phrases we can set as equal to specific values or locations so we use them later, reducing complexity.

Setting Up Pins: In order to connect the Pi with the circuit, we will need to tell it how to use its pins—we can set them to be either inputs or outputs depending on if we need to receive or send a signal.

Creating Loops: We will use loops to allow our code to run over and over again while we play with the circuit. A **For** loop will run for a fixed number of times, but a **While** loop will continue to run until the set condition changes.

PROJECT CHALLENGES

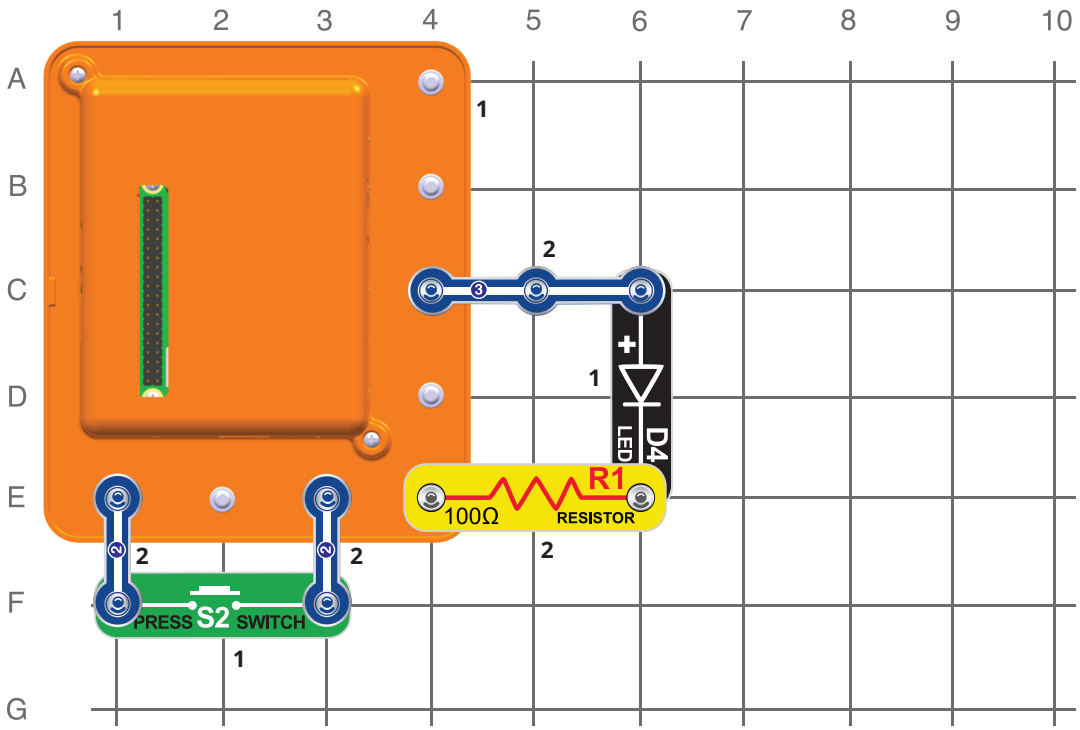
Challenge 1: Try changing the LED variables to change the blinking pattern.

Challenge 2: Replace the LED with the horn (W1) or a different LED to try different outputs.

Project #2

Light-Activated LED

OBJECTIVE: TO CONTROL THE LED LIGHT WITH A PUSH BUTTON



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

Inputs & Outputs: In Project 1, we learned to control the output of the Pi with code, but now we want to code an input to control the output. In this case, we will be coding a push button to activate the LED light.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Press Switch, LED (D4, D8, or D12).

Once the circuit is created, open up Thonny and load Project02.py into the code editor.

In this project, you will learn about **coding inputs and outputs**.

Once you have completed the program, upload it the to the rover and use the switch to blink the LED.

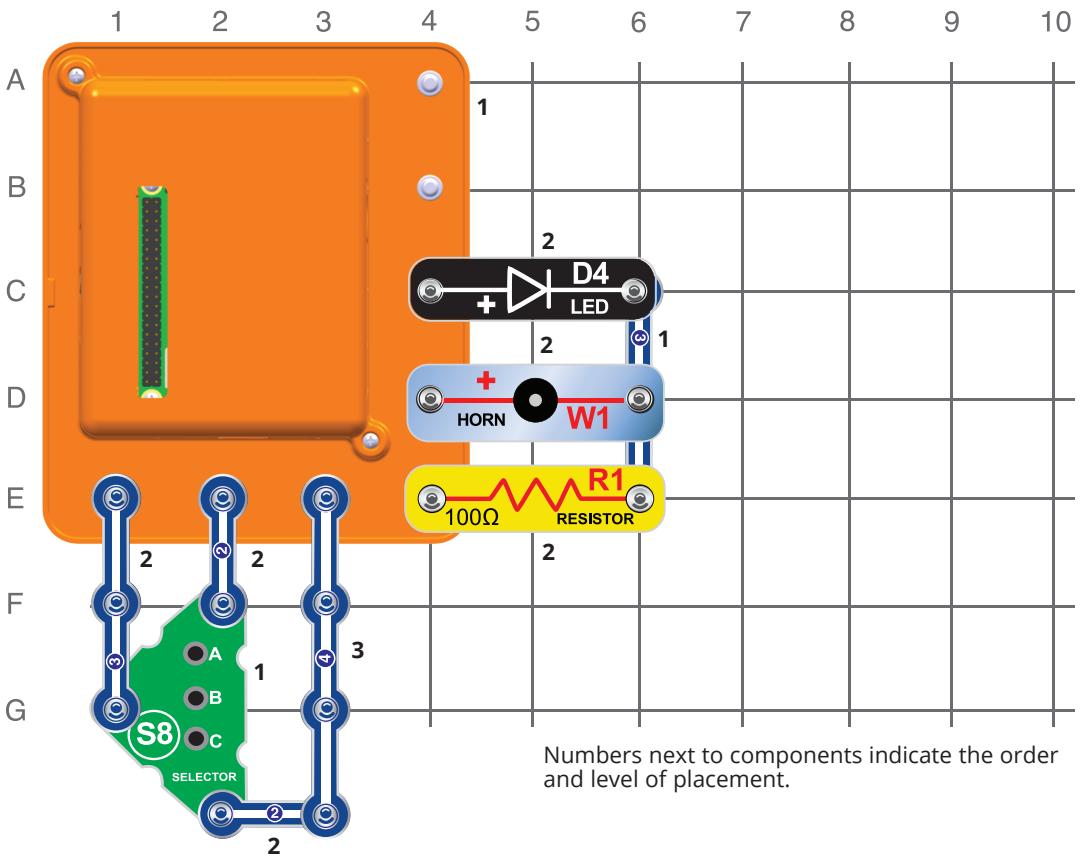
PROJECT CHALLENGES

Challenge 1: Try changing the LED variables to change the blinking pattern.

Challenge 2: Replace the LED with the horn (W1) or a different LED to try different outputs.

Challenge 3: Replace the push button with the phototransistor and cover it with your hand. What happens? Note that the phototransistor should be installed with the + on the left.

Project #3



Numbers next to components indicate the order and level of placement.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Selector, LED (D4, D8, or D12), Horn.

Once the circuit is created, open up Thonny and load Project03.py into the code editor. You will need to edit the question marks and remove the hashtag symbol (#) from the commented-out lines for the code to run.

In this project, you will learn about **writing functions.**

Once you have completed the program, upload it the to the rover and see how the LED light blinks in the programmed patterns. Please be sure to firmly push the selector buttons.

Selector Output

OBJECTIVE: TO PROGRAM THE SELECTOR TO TURN ON THE LED LIGHT IN PROGRAMMED PATTERNS

KEY CONCEPTS

Functions: Functions are a crucial tool in programming. You can write a function to perform a specific task or calculation, then, whenever you "call" upon the function, you can perform that task immediately. We will use functions going forward in different ways.

Comments: This project features "commented-out" code. You can use the hashtag symbol (#) parts of code if you don't want that part of the program to run but would like to have it for reference. You will need to remove the hashtag symbol (#) and update the code for it to run.

PROJECT CHALLENGES

Challenge 1: Try changing the LED variables to change the blinking pattern.

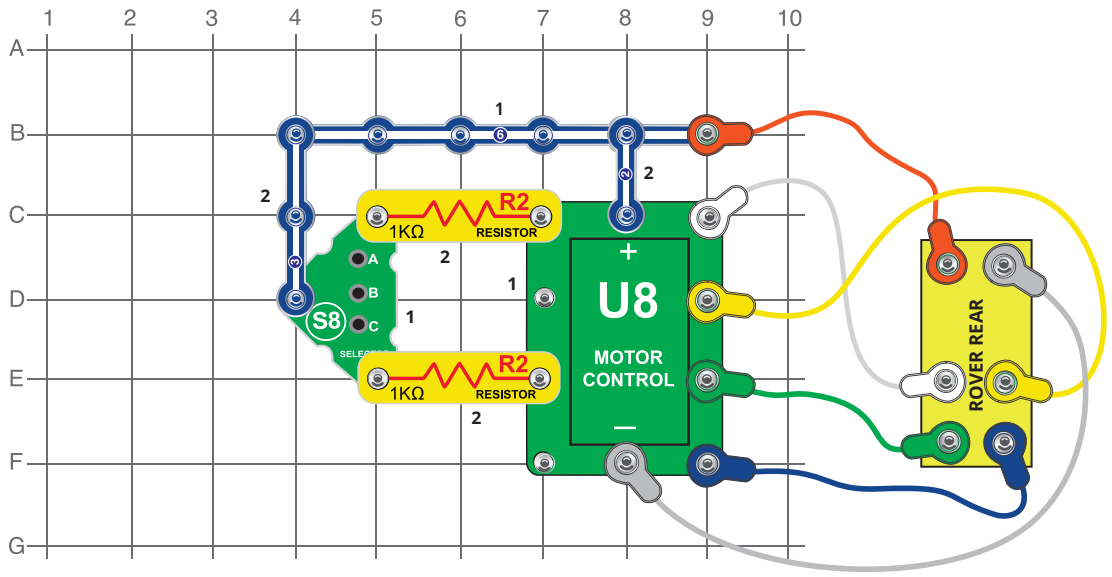
Challenge 2: Replace the LED with the horn (W1) or a different LED to try different outputs.

Challenge 3: Change input pins A and C in the While loop. What happens?

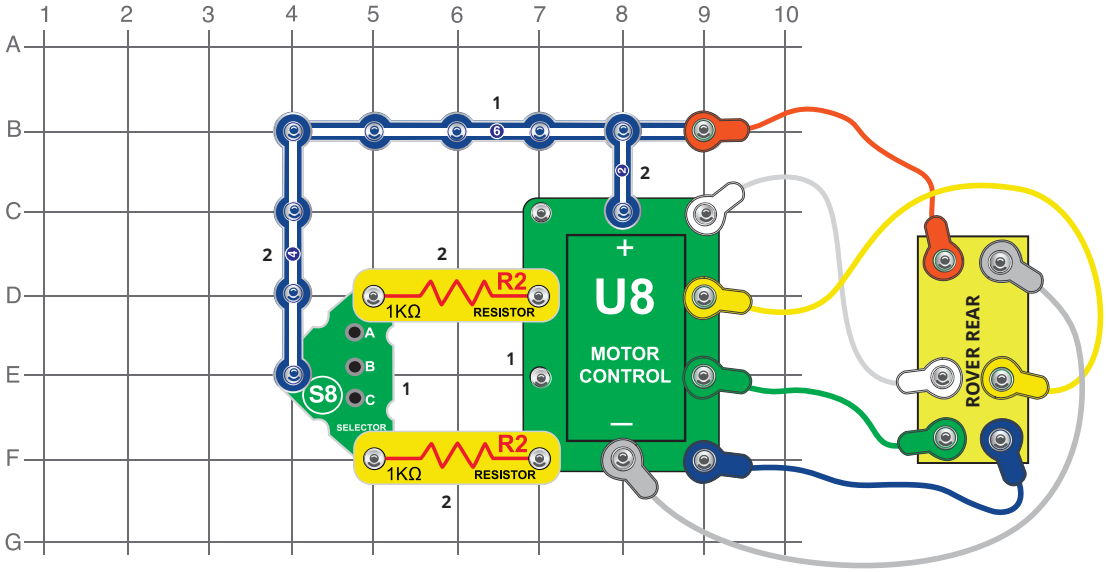
Challenge 4: Change output pins LED and horn (W1) in the While loop. What happens?

Challenge 5: Try to switch the order of the LED and horn (W1) function loops. What happens?

Introduction to Motor Control



Numbers next to components indicate the order and level of placement.



Numbers next to components indicate the order and level of placement.

Selector Driving

OBJECTIVE: TO USE THE SELECTOR TO DRIVE THE ROVER

KEY CONCEPTS

Each of the various pins on the Motor Control corresponds to a different driving function: turning left, turning right, or driving straight. See the **About Your Components** section for details.

Once we beginning programming the rover to drive, it will be important to remember which pins control which driving function, so we can accurately control our rover.

INSTRUCTIONS

This is not a coding project and thus, there is no programming involved. Instead, we will introduce the Motor Control.

Components: Selector, Motor Control.

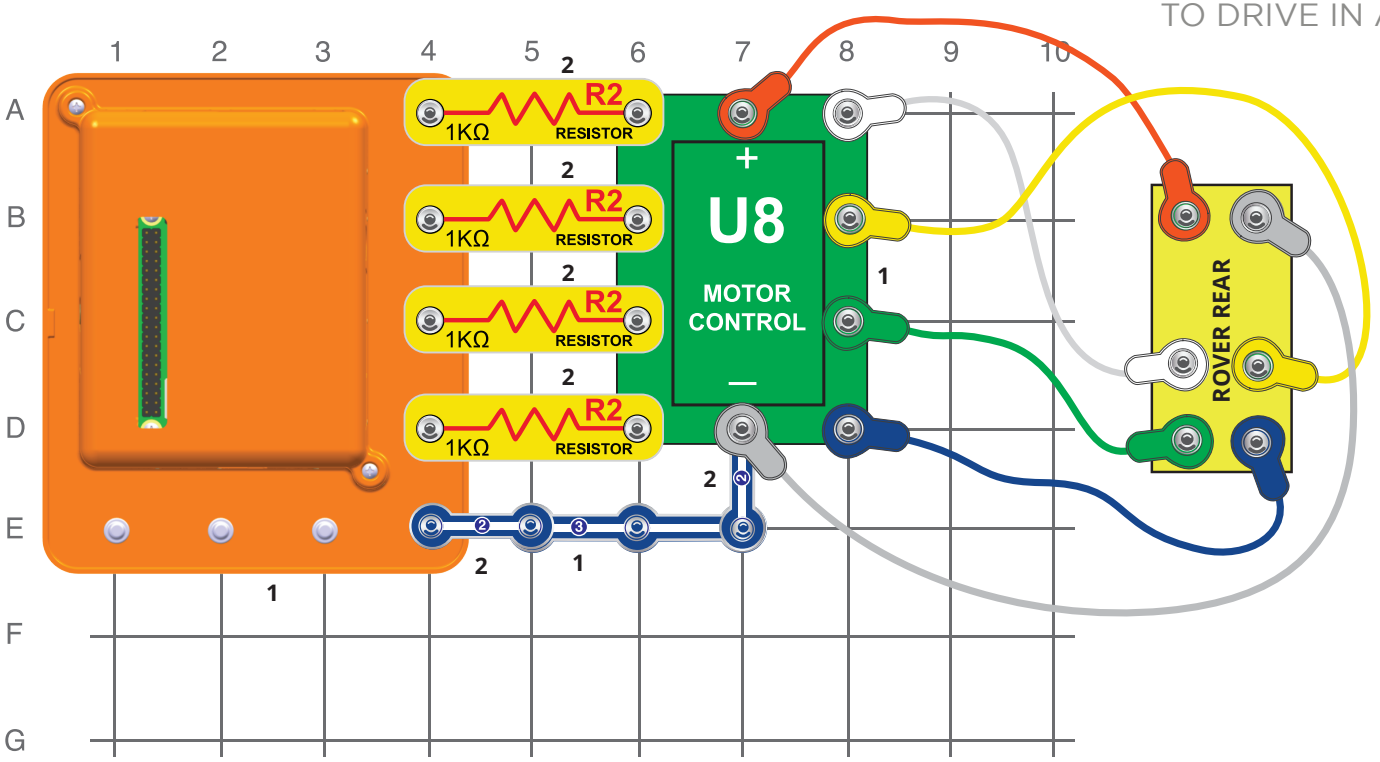
Firmly press the selector switch buttons in full to ensure responsiveness.

The diagrams shows two circuits, which change the inputs for the selector. Build the first circuit shown here and use the selector to turn 'right' with A or 'left' with C, and drive straight with B. Then build the second circuit, which flips the selector button controls.

Project #4

Programmed Driving

OBJECTIVE: TO PROGRAM THE ROVER TO DRIVE IN A CODED PATH



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

Motor Controls: We will use functions here to pulse the motor controller with pins 1-4 activating left forward, left backward, right forward, and right backward, respectively.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Motor Control.

Once the circuit is created, open up Thonny and load Project04.py into the code editor.

In this project, you will learn about **motor control outputs** and create **driving functions**.

Once you have completed the circuit, upload the program to drive it along your designated path.

PROJECT CHALLENGES

Challenge 1: Change the drive function time arguments to alter the driving path.

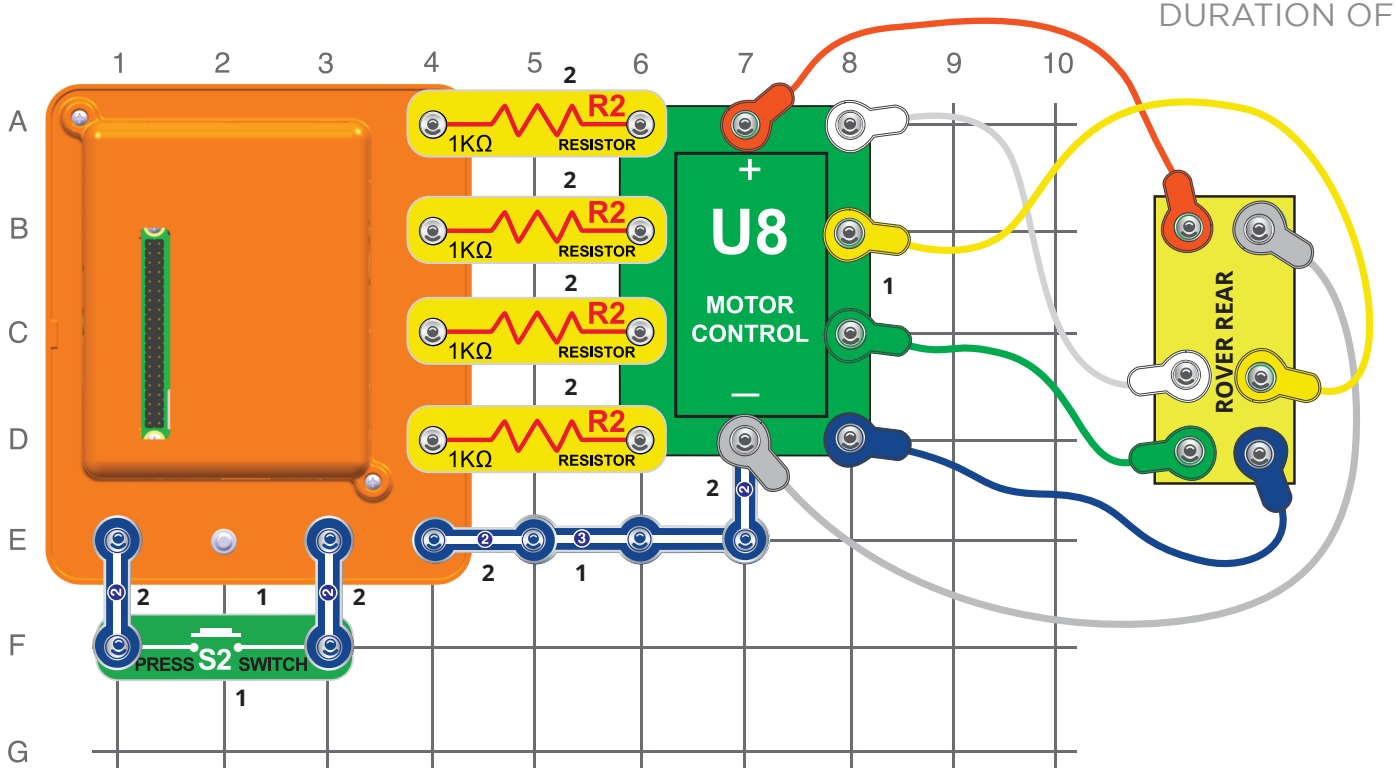
Challenge 2: Reorder the drive functions for a new driving path.

Challenge 3: Create your own custom driving path using the different drive functions and time arguments.

Project #5

Timed Driving

OBJECTIVE: TO PROGRAM THE ROVER TO DRIVE IN A CODED PATH FOR THE DURATION OF THE BUTTON PUSH



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

Time: The internal clock in the Pi can be used to time certain events, such as how long the push button is held for. We can then save this time as a variable to control for the duration of the rover driving.

Modulo Operator: This operator returns the remainder of a division equation for us to use. Because all even numbers are divisible by 2 with a remainder of 0, **modulo 2** can help us easily alternate between commands.

Callback Functions: In this project, you will learn about callback functions, where we'll connect an action function to a measurement function for the push button.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Press Switch, Motor Control.

Once the circuit is created, open up Thonny and load Project05.py into the code editor.

In this project, you will learn about **callback functions**.

Once you have completed the program, press and hold the switch to drive the rover as you hold the switch down.

PROJECT CHALLENGES

Challenge 1: Change the drive functions to switch the driving directions.

Challenge 2: Add new drive functions activated in the loop to create a new driving path.

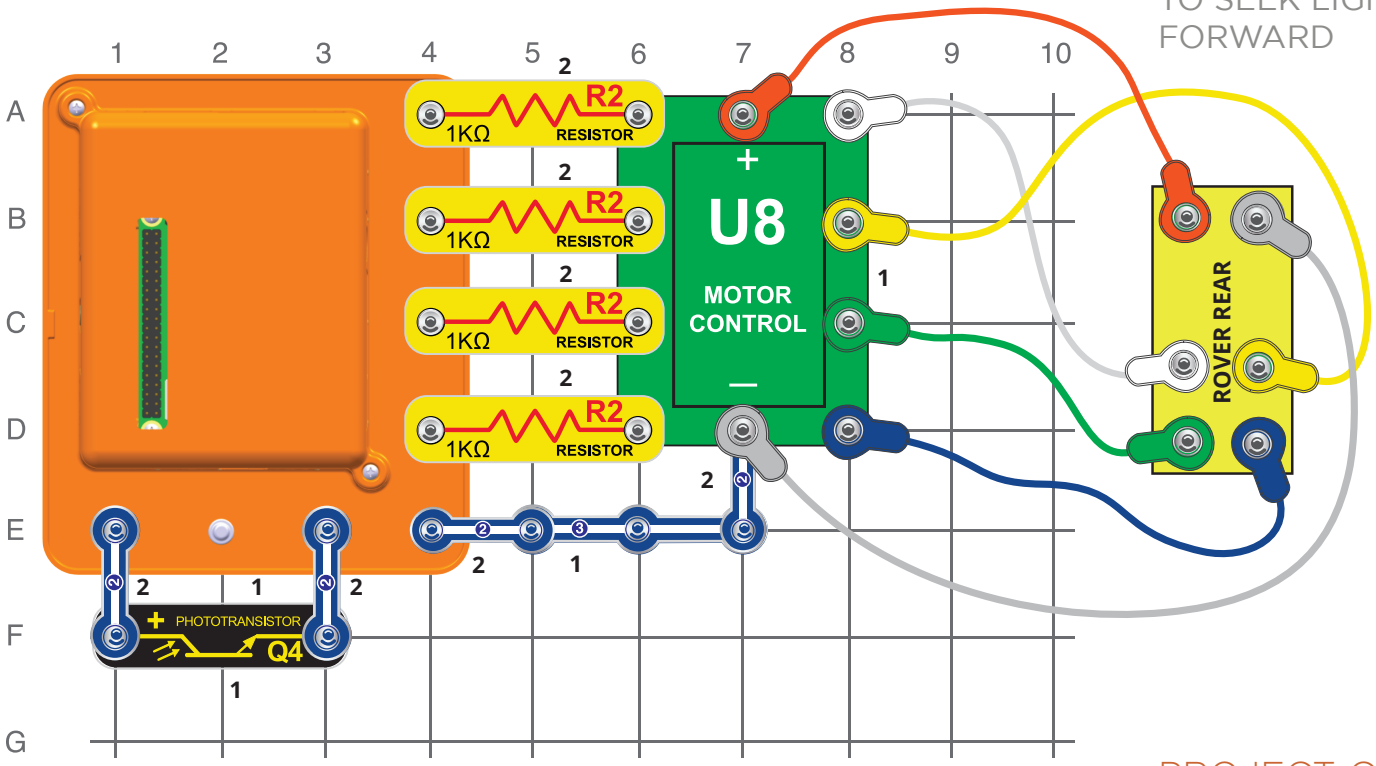
Challenge 3: Use the modulo operator to change the driving direction based on even- or odd-numbered presses.

Challenge 4: With the modulo, add new driving functions for different driving paths for even- or odd-numbered presses.

Project #6

Light-Activated Driving

OBJECTIVE: TO PROGRAM THE ROVER TO SEEK LIGHT BEFORE DRIVING FORWARD



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

If, Then: One of the most powerful logic statements is the **If, Then statement**, which allows for certain conditions to lead to certain actions. It reads as **if** this statement is true, **then** execute this command, otherwise execute this command. These statements can also be nested to account for multiple conditions and commands to be controlled.

Break: Break statements are used to stop a loop and restart the code from the beginning, helping us reset when conditions change.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Phototransistor, Motor Control.

Once the circuit is created, open up Thonny and load Project06.py into the code editor.

In this project, you will learn about **adding loop complexity**.

Once you have completed the program, use a flashlight in a dark room to have the rover drive to follow the light source.

PROJECT CHALLENGES

Challenge 1: Add new drive functions to change the light-seeking spin pattern.

Challenge 2: Add the 100 Ohm resistor in series with the phototransistor to increase light sensitivity.

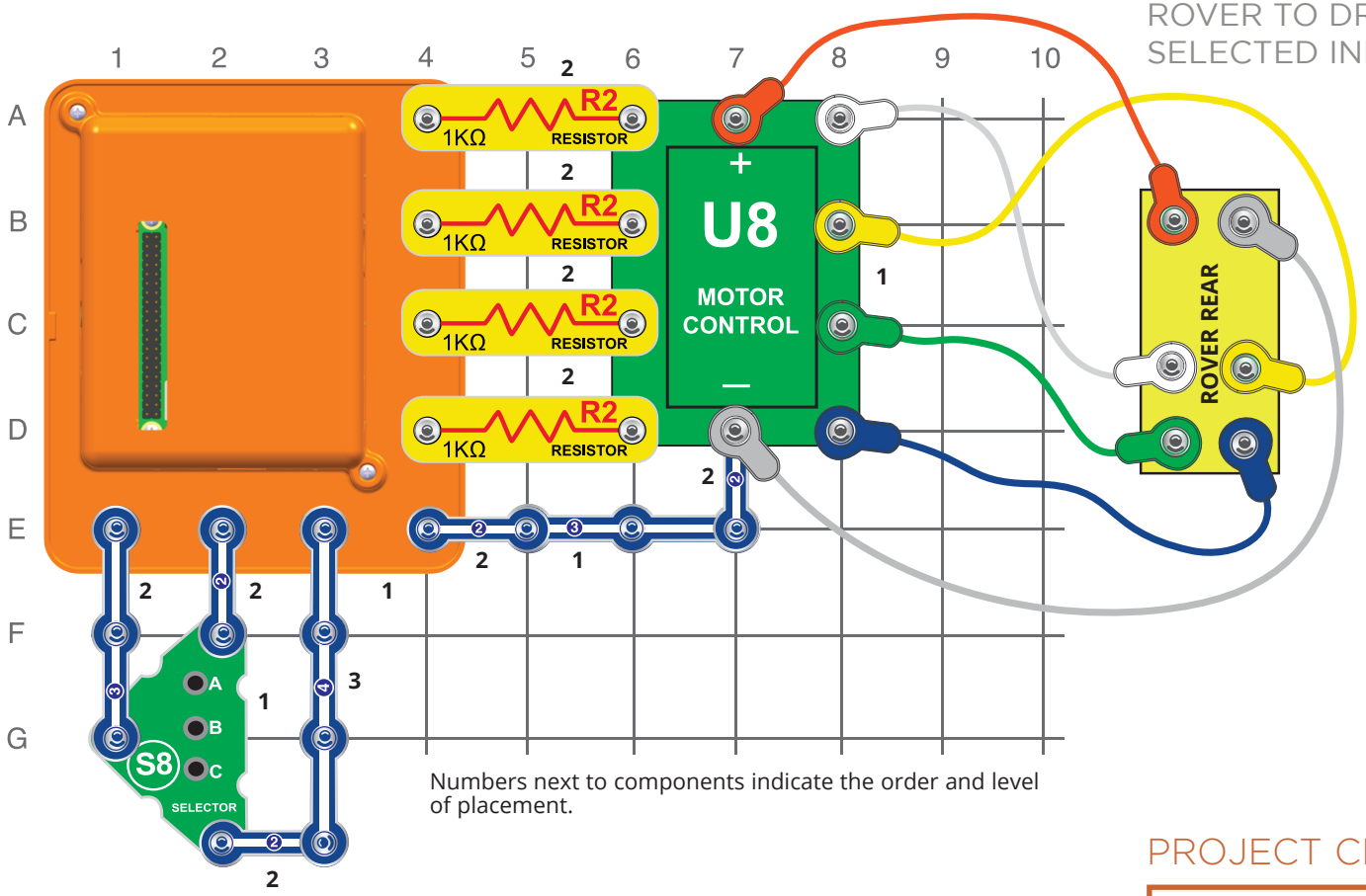
Challenge 3: After a certain amount of time, have the rover spin to look for light.

Challenge 4: With the modulo operator, have the rover alternate left or right when spinning to search for light.

Project #7

Controlled Driving

OBJECTIVE: TO PROGRAM THE ROVER TO DRIVE ACCORDING TO SELECTED INPUTS



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

True-False: Here we will use true-false statements to enable unique driving controls based on the A, B, and C buttons. Certain conditions can be evaluated as true or false, enabling different results. An **And statement** requires several conditions to be evaluated as True, while a **Not statement** can flip a true to a false and vice versa.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Selector, Motor Control.

Once the circuit is created, open up Thonny and load Project07.py into the code editor. Firmly press the selector switch buttons in full to ensure responsiveness.

In this project, you will learn about **adding complex logic**. Once you have completed the program, control the rover's driving using the commands you coded.

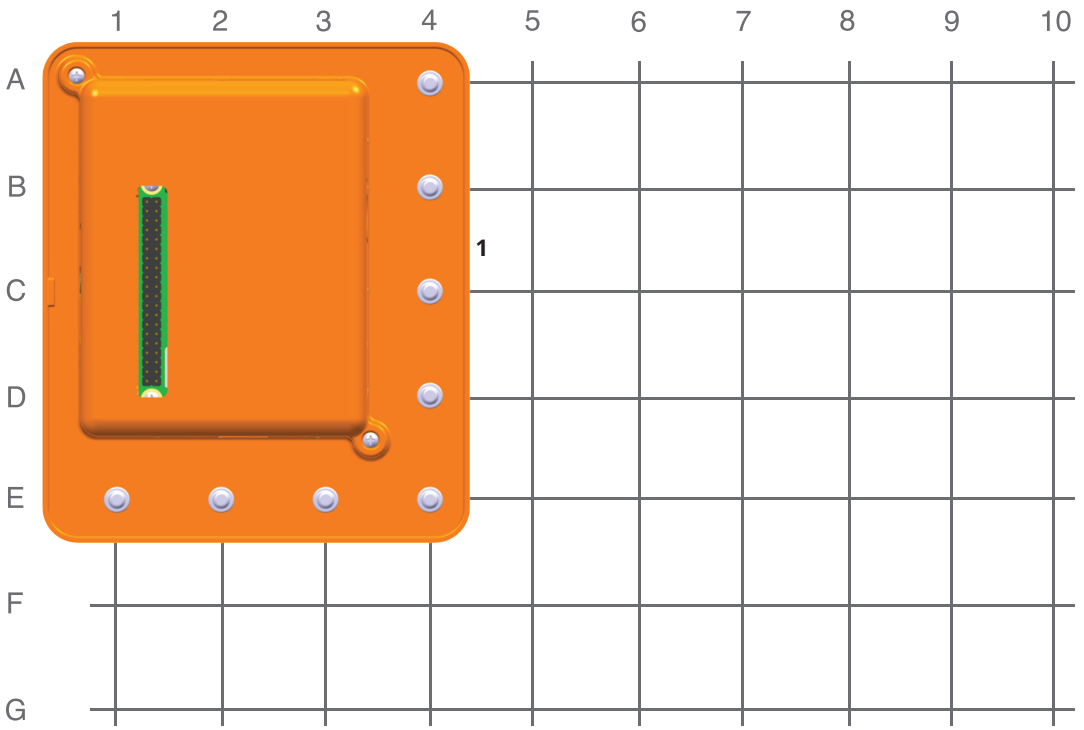
PROJECT CHALLENGES

Challenge 1: Incorporate the button press timer from Project 5 to add Simon Says to driving functions.

Challenge 2: The B pin uses a double If statement. Can you apply the same for the A and C pins?

Challenge 3: Replace the 3-length snap connector with a phototransistor; now all 3 buttons are light-dependent. Can you add an outer loop for new driving instructions when there is no light or button presses for more than 5 seconds?

Project #8



Numbers next to components indicate the order and level of placement.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. There is no circuit for this project, as we will be introducing the camera within the Smart Module.

Components: Smart Module.

Once connected, open up Thonny and load Project08.py into the code editor. In this project, you will learn about different image settings.

Once you have completed the program, upload it to the rover to experiment with the camera.

Introduction to Camera

OBJECTIVE: TO LEARN ABOUT THE CAMERA AND DIFFERENT IMAGE SETTINGS

KEY CONCEPTS

The PiCamera module allows the Pi to see the world and interpret its surroundings. The images it produces can be analyzed in the code and interpreted to make decisions.

The camera has various settings that can be tuned to better perform functions like image recognition including Resolution (clarity of an image), Frame Rate (recording speed), Contrast (luminance between objects), and Brightness (lightness).

When programming to use the camera, you may have to adjust the orientation from 180 to 90 degrees, depending on your module.

PROJECT CHALLENGES

Challenges 1 & 2: Try changing the camera resolution to the minimum (64,64) or the maxmium (2592,1944 and framerate = 15) and see what the image looks like.

Challenge 3: Try changing the camera rotation to flip it.

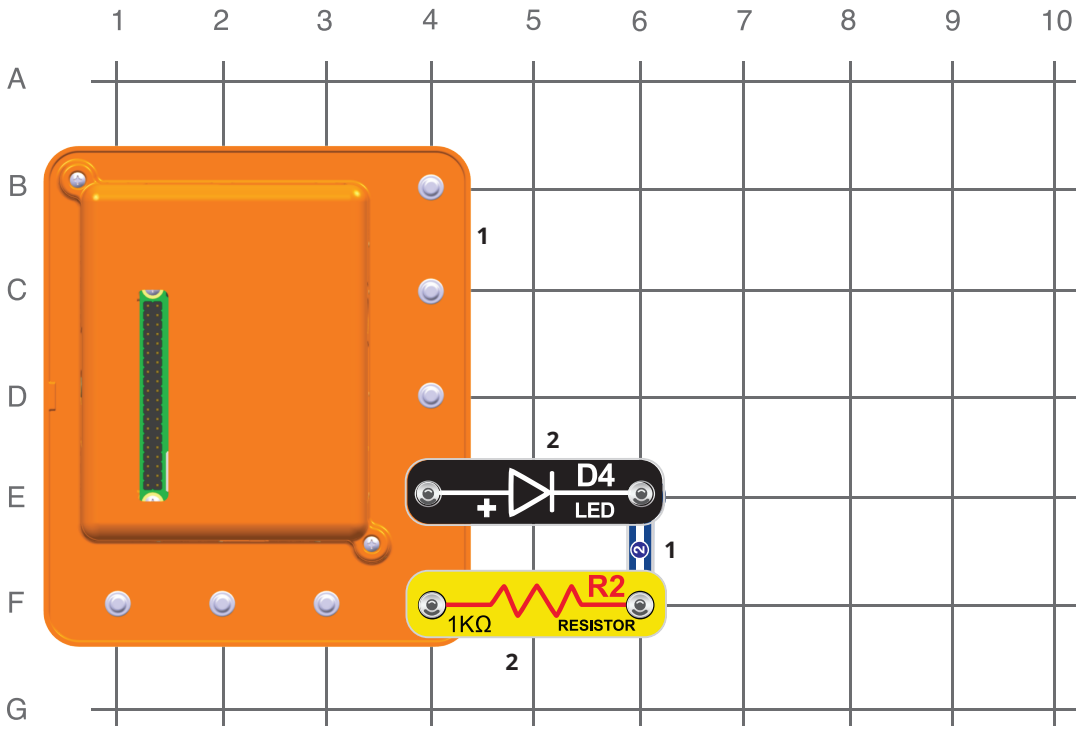
Challenge 4: Try adding text on top of the image.

Challenge 5: Try looping through all the contrast and brightness options, with annotations on the image of the current level.

Project #9

Light Detection

OBJECTIVE: TO PROGRAM THE CAMERA TO DETECT LIGHT



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

Image processing is a key component of computer vision, which analyzes each layer of an image. While you may be familiar with the RGB scale, where an image is defined as a ratio of red, green, and blue intensities, computers prefer HSV. H (Hue) is a color scale through the rainbow, S (Saturation) is the greyness added, and V (Value) is lightness. This allows for differentiation of colors on one axis and brightness on another, which allows for object identification.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, LED (D4, D8, or D12).

Once the circuit is created, open up Thonny and load Project09.py into the code editor. In this project, you will learn about HSV image processing and light detection.

Once you have completed the program, see how the light flashes when different light thresholds are met.

PROJECT CHALLENGES

Challenge 1: Try changing the light threshold value to keep the LED on at all times.

Challenge 2: Try changing the light threshold value to keep the LED off at all times.

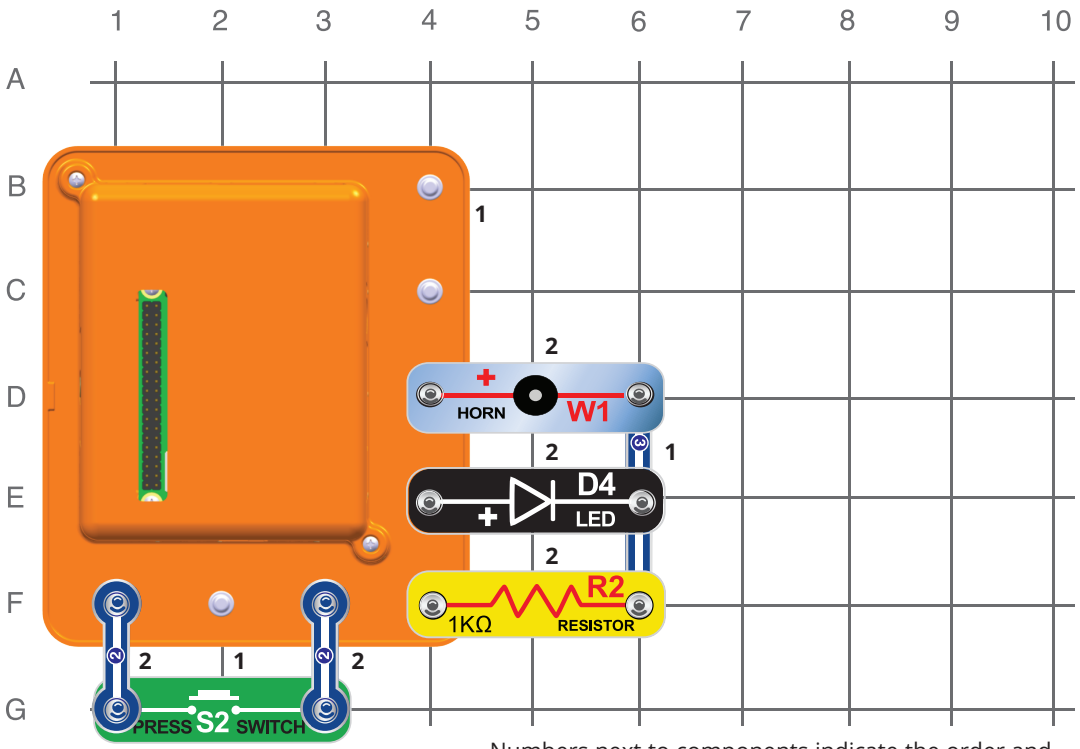
Challenge 3: Can you add another pin for the horn to sound when the light is too low?

Challenge 4: How can you activate the LED in darkness?

Project #10

Color Detection

OBJECTIVE: TO PROGRAM THE ROVER TO DISTINGUISH BETWEEN VARIOUS COLORS



Numbers next to components indicate the order and level of placement.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Press Switch, Horn, LED (D4, D8, or D12), Motor Control.

Once the circuit is created, open up Thonny and load Project10.py into the code editor. In this project, you will learn about primary color identification and object detection.

Once you have completed the program, use the camera to capture and analyze the color profile of cutout signs in the back of the manual.

KEY CONCEPTS

Although HSV was helpful before, **RGB** is better served here when it comes to differentiating primary colors. The camera images are stored as 3D arrays with a different layer of pixels for red, green, and blue that together form the full picture when added up. By subtracting the background noisy colors from the capture image of the object, we can see which of the three color layers is most prominent. This becomes the code's best guess for the object's color, and now we can ascribe outputs.

The camera takes some time to settle, so please be patient and try taking another image if a sign was misidentified.

PROJECT CHALLENGES

Challenge 1: Try changing the LED and horn outputs both in the code and on the rover.

Challenge 2: Try writing a function to handle the LED and horn so it can be called after each color.

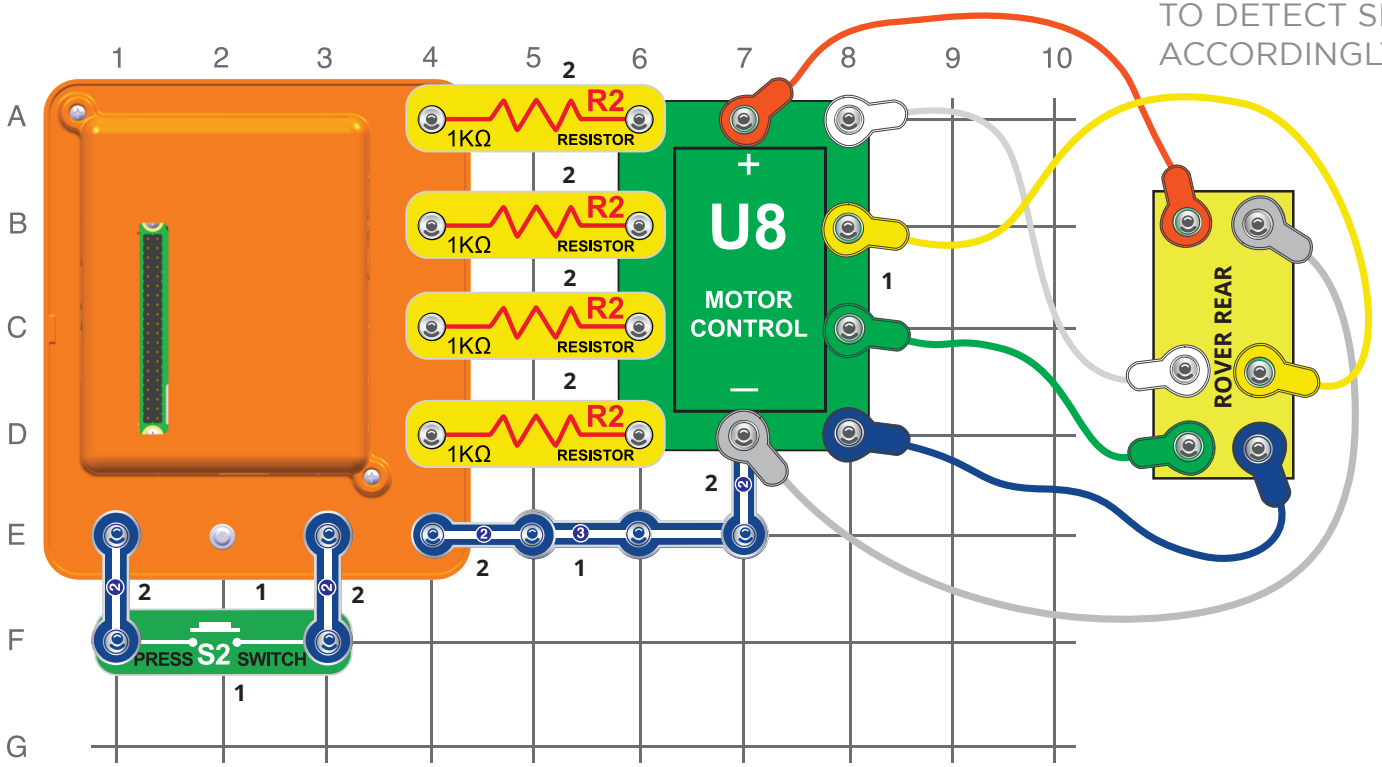
Challenge 3: Try adding a margin that the argmax for color must be exceeded to be considered a certain color.

Challenge 4: Try adding a memory array for the last two colors identified and output flashes or buzzes based on the pattern.

Project #11

Color-Detection Driving

OBJECTIVE: TO PROGRAM THE CAMERA TO DETECT SIGN COLORS AND DRIVE ACCORDINGLY



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

Using the same object color identification processes as before, incorporating driving outputs takes us one step closer to **autonomous driving**. Remembering the rules of the road, the rover can navigate an environment by identifying signs and taking action accordingly. See if you can set up the signs and add the driving commands to get the rover through the course.

The camera takes some time to settle, so please be patient and try taking another image if a sign was misidentified.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Press Switch, Motor Control.

Once the circuit is created, open up Thonny and load Project11.py into the code editor. In this project, you will learn about intrductory autonomous driving.

Once you have completed the program, use the rover and the cut-out signs in the back of the manual to have it drive according to the signs.

PROJECT CHALLENGES

Challenge 1: Try changing the colors associated with the driving commands.

Challenge 2: Try adding a modulo operator to alternate between left and right turns on blue signs.

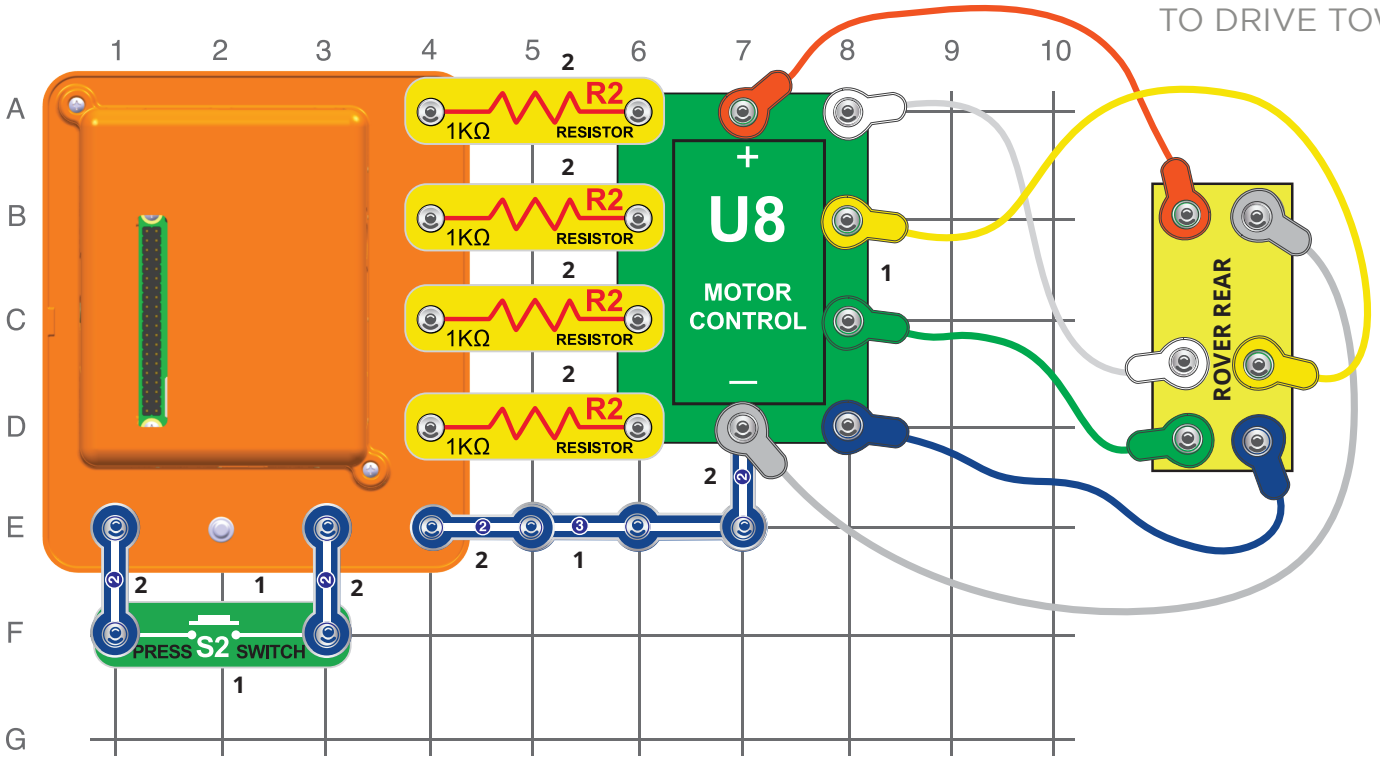
Challenge 3: Try setting the drive_time variable based on the prominence of color from the argmax.

Challenge 4: Try adding a memory variable for the last color identified and dictate driving based on the pattern.

Project #12

Light-Seeking Driving

OBJECTIVE: TO PROGRAM THE ROVER TO DRIVE TOWARD LIGHT IT DETECTS



Numbers next to components indicate the order and level of placement.

KEY CONCEPTS

Having used the camera to assess light levels, layering in image processing allows for **light-seeking autonomous driving**. By dividing the image into left and right components, the rover can determine which direction is brightest and rotate until it is properly aligned, and then drive toward the target. By adjusting the threshold variables and driving durations, the rover can more efficiently seek out the lightest regions of its environment. Try raising and dimming your room's lights to see how the rover responds.

INSTRUCTIONS

First, connect your Smart Module to your laptop or monitor. Then build the circuit shown here.

Components: Smart Module, Press Switch, Motor Control.

Once the circuit is created, open up Thonny and load Project12.py into the code editor. In this project, you will learn about intermediate autonomous driving.

Once you have completed the program, use a flashlight to guide the rover as it drives and detects light.

PROJECT CHALLENGES

Challenge 1: Try changing the 'left' and 'right' thresholds to force different turning patterns.

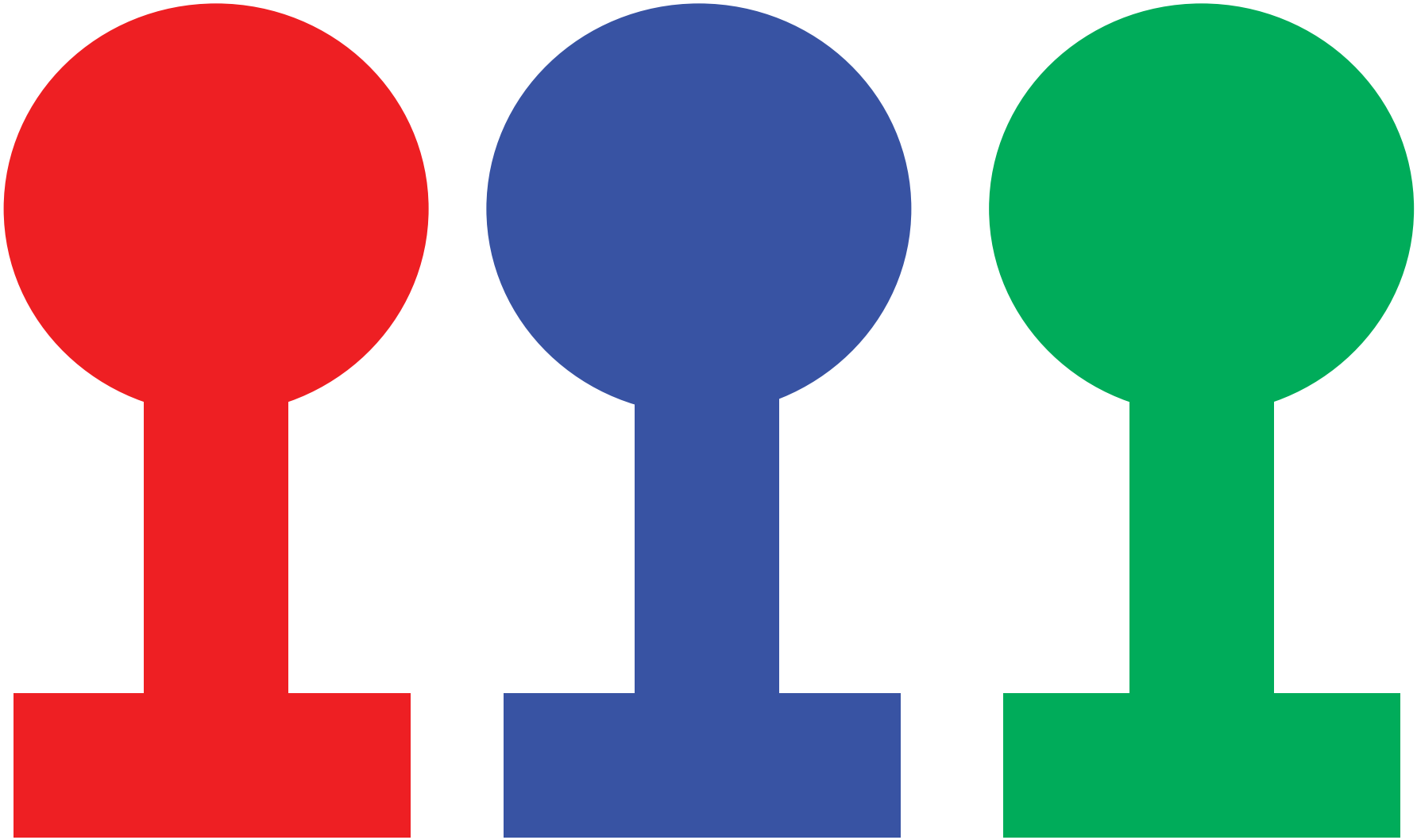
Challenge 2: Try using the modulo function and loop counter to go from 'forward' to 'reverse' every cycle.

Challenge 3: Can you add a timer to the loop to have the rover do a spin after thirty seconds of searching?

Challenge 4: Can you set the drive_time duration based on the ratio of left-to-right light?

Cutout Color Signs

Cut these signs out of your manual and use them for the color-detection projects (Project #10 and Project #11) or with your own projects with the camera. The signs can be folded to stand up on their own.



THE
SMART
FACTORY

**Scan to visit
The Smart Factory
@ Wichita website
thesmartfactory.io**

Rated for ages 10 and older. Caution: Electric toy, not recommended for children under 10 years of age.

The Smart Rover is not intended for retail sale.

For issues with the Smart Module, please visit The Smart Factory @ Wichita website at **thesmartfactory.io**.
For all other components, please contact Elenco® Electronics Customer Support via **elenco.com** or by emailing **support@elenco.com**.

This product is manufactured at The Smart Factory @ Wichita, located at 1960 Innovation Blvd, Wichita KS, 67208.

The Smart Factory @ Wichita is one of Deloitte's several global immersive experiences designed in collaboration with ecosystem participants to accelerate digital transformation. It demonstrates first-hand how manufacturers can evolve for the future and up-level their capabilities and processes through technological innovation.